Deep Convolutional Neural Network for Cattle Disease Identification

Hashim Siraj Usmael

A Thesis Submitted to the department of Computer Science and Engineering

School of Electrical Engineering and Computing

Presented in Partial Fulfillment of the Requirements for the Degree of Master of science in Computer Science and Engineering

Office of Graduate Studies

Adama Science and Technology University

June, 2023

Adama, Ethiopia

Deep Convolutional Neural Network for Cattle Disease Identification

Hashim Siraj Usmael

Advisor: Getinet Yilma (PhD)

A Thesis Submitted to the department of Computer Science and Engineering

School of Electrical Engineering and Computing

Presented in Partial Fulfillment of the Requirements for the Degree of Master of science in Computer Science and Engineering

Office of Graduate Studies

Adama Science and Technology University

June, 2023

Adama, Ethiopia

DECLARATION

I hereby declare that this Master Thesis entitled "**Deep Convolutional Neural Network for Cattle Disease Identification**" is my original work. That is, it has not been submitted for the award of any academic degree, diploma or certificate in any other university. All sources of materials that are used for this thesis have been duly acknowledged through citation

Hashim Siraj		15/07/2023
Name of the student	Signature	Date

RECOMMENDATIONS

I/we, the advisor(s) of this thesis, hereby certify that I/we have read the revised version of the thesis entitled "**Deep Convolutional Neural Network for Cattle Disease Identification**" prepared under my/our guidance by Hashim Siraj Usmael. Submitted in partial fulfillment of the requirements for the degree of Mater's of Science in Computer Science and Engineering. Therefore, I/we recommend the submission of revised version of the thesis to the department following the applicable procedures.

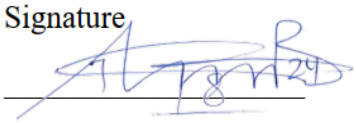
Getinet Yilma (PhD)		15/07/2023
Major Advisor	Signature	Date

APPROVAL SHEET

I/we, the advisors of the thesis entitled "**Deep Convolutional Neural Network for Cattle Disease Identification**" developed by Hashim Siraj Usmael, hereby certify that the recommendation and suggestions made by the board of examiners are appropriately incorporated into the final version of the thesis.

Getinet Yilma (PhD)		15/07/2023
Major Advisor/Supervisor	Signature	Date

We, the undersigned, members of the Board of Examiners of the thesis by Hashim Siraj Usmael have read and evaluated the thesis entitled "**Deep Convolutional Neural Network for Cattle Disease Identification**" and examined the candidate during open defense. This is, therefore, to certify that the thesis is accepted for partial fulfillment of the requirement of the degree of Master of Science in Computer Science and Engineering.

Chairperson	Signature	Date
Internal Examiner	Signature	Date
Asrat Mulatu (Ph.D.)		05/07/2023
External Examiner	Signature	Date

Final approval and acceptance of the thesis is contingent upon submission of its final copy to the Office of Postgraduate Studies (OPGS) through the Department Graduate Council (DGC) and School Graduate Committee (SGC).

Department Head	Signature	Date
School Dean	Signature	Date
Office of Postgraduate Studies, Dean	Signature	Date

ACKNOWLEDGMENTS

First, I would like to thank the Almighty **Allah**, who made everything possible. Next, I would like to express my deepest gratitude and appreciation for my Advisor, **Getinet Yilma (PhD).** I am deeply thankful to him for his insights and constructive comments. One day I expect, I would become as good an advisor as he becomes and guide my students as he has been to me.

Next, West Hararghe Zone Agriculture Office Animal Health Team Director Dr. Mohammed A, Chiro woreda veterinary Clinic Director Dr. Haji B, Haramaya University Veterinary specialist student Dr. Mitiku Sh, Haramaya University College of Veterinary Medicine Dean Prof. Dr. Adem Hiko and also all staff members of Haramaya University Veterinary Medicine, Chiro Woreda Veterinary Clinic, and West Hararghe Zone Veterinary Office. I would like to thanks all of you for helping me in capturing cattle disease images and correctly identify the diseases.

I would like to thanks my Mother, My Father, and my wife, Lubaba Yasin, who never stopped helping me financially and mentally from beginning to the end. I love you so much. May Allah Grant you long life and Health.

Finally, I would like to express my sincere thanks to Nasiro M, Ahmed A, Hassan S, Nuru B, Lencho A, Mohammed, Ahmednajesh A, Ziyad M, Hadi H, and all of my friends for your support. For those of you whose names are not listed here, I would like to apologize for not mentioning your names here, and I would like to tell you that you are in my heart even if you are not listed here. You are all the reason I was so diligent and committed to my studies, and I want to express that you will always have a special place in my heart and life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

SAMPLE CODE

LIST OF ACRONYMS AND ABBREVIATIONS

CNN: - Convolutional Neural Network

DCNN: Deep Convolutional Neural Network

LSD/LD: - Lumpy Skin Disease

IBK/KCD: - Infectious Bovine Kerato-Conjunctivitis Disease

FMD: - Foot and Mouth Disease

RWD: - Ringworm Disease

GAN: - Generative Adversarial Network

VAE: - Variation Auto Encoders

ReLU: - Rectified Linear Unit

R-CNN: - Region-Based Convolutional Neural Network

R-DCNN: - Region-based Deep Convolutional Neural Network

RNN: Recurrent Neural Networks

CCSAM: - Combination of Channel and Spatial Attention Mechanism

KNN: - K-Nearest Neighbors

SVM: - Supportive Vector Machine

ANN: - Artificial Neural Network

ISIC: - International skin Image Collaboration

RAM: - Random Access Memory

GPU: - Graphical Processing Unit

RGB: - Red-Green-Blue color

CPU: - Central Processing Unit

GB: - Gigabytes

CONV: - Convolutional Layers

MaxPool: - Max Pooling Layers

AvgPool: - Average Pooling Layers

FC: - Fully-Connected Layers

SGD: - Stochastic gradient Descent

MLP: - multi-layer perceptron

ABSTRACT

Most of the Ethiopian population resides in rural regions, and the foundation of its economy relies on agriculture. Ethiopia is famous in Africa for being a prominent producer of cattle. Despite this fact, cattle's economic contribution is not satisfactory due to several factors, among which animal diseases trouble the cattle business productivity. Therefore, Early and accurate identification of cattle diseases could be among the key measures required for preventing disease in its early stages. Several efforts have been made to advance cattle disease identification; In this thesis, a deep residual CNN model, ResNet18, with channel and spatial attention mechanisms (CCSAM) added at each residual network basic block is proposed. The CCSAM is integrated into the basic blocks of the ResNet18 to extract disease-related features from spatial and channel dimensions. These features are combined via elementwise multiplication to amplify the disease feature and again added to the original cattle disease image feature map via skip connection to minimize the impact of model overfitting. Apart from the aforementioned model development, the proposed method also applied several geometrical and photometric augmentation techniques for improving the diversity of the dataset, increasing class distribution, and minimizing model overfitting. The proposed method was extensively trained on two datasets. The first dataset was collected from Haramaya University Veterinary Medicine, Chiro Woreda Veterinary Clinic, and West Hararghe Zone Veterinary Office. The second was extracted from the Kaggle open cattle disease dataset. To measure the overall performance of the proposed method, the mean aggregate was calculated for values of precision, recall, F1 score, and accuracy for each class. To further strengthen our evaluation, the confusion matrix and classification report metrics were used. In general, the fine-tuned proposed ResNet18-CCSAM method on a geometrically augmented dataset has obtained a precision of 99%, a recall of 98%, F1 score of 98%, and an accuracy of 98.97%. Compared to the other related research works, our proposed method has demonstrated better cattle disease identification performance.

Keywords: Deep Convolutional Neural Network, Spatial and Channel Attention, Cattle Disease Identification

CHAPTER ONE

INTRODUCTION

1.1. Background of the study

Ethiopia has the highest number of draft animals, ranking first in Africa and ninth worldwide. (Birhanu, 2014). In Ethiopia, there is a population of approximately 59.5 million cattle. These animals play a significant role in the country's economy, accounting for 40% of the annual agricultural output and contributing to 15% of the overall gross domestic product. Given that farming and animal husbandry are vital to the livelihoods of many Ethiopians, livestock holds great socioeconomic importance in the nation (Birhanu, 2014).

According to the Drug Administration and Control Authority of Ethiopia (Ethio, n.d.), cattle diseases were categorized into five groups: non-infectious diseases, infectious diseases, diseases of the respiratory system, diseases of the reproductive system, and ecto-parasites. Infectious diseases are clinically visible illnesses caused by the infection, presence, and spread of harmful biological agents in an individual host organism. Some viruses, bacteria, fungi, protozoa, multicellular parasites, and abnormal proteins are infectious pathogens. On the other hand, noninfectious diseases can’t be passed from one animal to another. Instead, these diseases are caused by environmental, genetic, and lifestyle factors.

However, potential economic benefits were constrained for many reasons. Widespread endemic diseases, including bacterial, viral, and parasite infestation, were the main factors limiting cattle output (Birhanu, 2014). Endemic and transboundary cattle diseases were usually transmittable leading to an outbreak in a short period (Gebre-Amanuel et al., 2018). The outbreak of disease reduced the productivity and quality of animal products such as skin, hides, milk, etc. If it is not controlled, it may cause international trade restrictions on animals and animal products. Therefore, it was essential to know about cattle diseases and solutions to increase productivity.

In addition to this, cattle disease was a significant impediment to domestic animals worldwide, resulting in a high mortality rate (Gebre-Amanuel et al., 2018). To reduce fatality rates, it was established that early disease identification is quite useful. The manual methods that currently used for disease identification was insufficient because naked eyes used for identification.

Occasionally, the disease diagnosis was prone to error, leading to less effective treatments. This resulted in cattle mortality and had a significant influence on economic growth. As a result, more efficient and automated methods for recognizing the disease lesions were required (AlFayez et al., 2020).

The field of veterinary medicine has seen significant advancements in recent years, and the use of deep learning techniques has shown promising results in various applications (Zhang et al., 2019). Deep convolutional neural networks (DCNNs) are a type of deep learning technique that has been widely used for image classification and object detection(Zhou, 2020). In this research, the aim is to explore the potential of DCNNs for the identification of cattle diseases based on their visual symptoms. Cattle disease identification is a crucial aspect of animal health management and early diagnosis can help prevent the spread of the disease, reduce economic losses, and improve overall animal welfare. However, the manual identification of cattle diseases can be time-consuming and often requires specialized knowledge, making it difficult for farmers and veterinarians to accurately diagnose diseases in a timely manner. The use of DCNNs has the potential to address these challenges by providing a fast and reliable method for cattle disease identification (Khan et al., 2020a; Simonyan & Zisserman, 2015).

The use of DCNNs in the field of veterinary medicine is a relatively new area of research (Xu et al., 2020), and there have been few studies on the application of DCNNs for cattle disease identification. However, the use of DCNNs in medical imaging has shown great potential, and the success of these models in other domains suggests that DCNNs could be a promising tool for cattle disease identification. The primary objective of this research is to study the achievability of using DCNNs for the identification of cattle diseases and to evaluate its performance compared to other methods. The study will contribute to the advancement of the field of veterinary medicine and may provide a basis for the development of automated systems for cattle disease identification, which could have significant implications for animal health management and the global economy (Xu et al., 2020).

Deep CNNs have the ability to automatically learn hierarchical representations from raw image data (Lin et al., 2017). Unlike traditional CNNs and machine learning methods, which require manual feature engineering, deep CNNs can automatically extract relevant features at multiple levels of abstraction(Khan et al., 2020a). This enables the model to capture intricate patterns

and discriminative features specific to cattle diseases without relying on handcrafted features (M. Li et al., 2021).

Deep CNNs allow for end-to-end learning, where the entire model is trained jointly with the objective of disease identification (H. Liu et al., 2018). This means that the feature extraction and classification layers are optimized together, resulting in more efficient and accurate predictions. In contrast, traditional CNNs and machine learning methods often involve separate stages for feature extraction and classification, which can introduce suboptimal intermediate representations and reduce overall performance (H. Liu et al., 2018).

Deep CNNs have the advantage of transfer learning, where pre-trained models trained on large-scale image datasets can be used as a starting point for cattle disease identification tasks (Gupta et al., 2022). This allows us to leverage the knowledge and representations learned from a different but related domain, reducing the need for large labeled datasets. Transfer learning can significantly speed up the training process and improve model performance, especially in scenarios where labeled cattle disease images are limited (Kim et al., 2022).

Many studies were conducted on the problems related to recognizing various animal diseases using traditional machine learning with monitoring systems. To the best of our knowledge, there was no previous research reported on five common cattle diseases, and only a few studies were conducted on deep learning to recognize animal diseases, which used manual feature extraction and were not well efficient due to human constraints and limited imagination when designing the feature extractor. Therefore, we proposed to develop deep convolutional neural network (CNN) models for cattle disease identification.

1.2. Motivation

Cattle disease was a big danger to farmers. Some diseases cause the infected cows to become very weak, and lose their ability to reproduce good-quality milk and meat. Sometimes the disease led to the death of the cows. As a result, farmers faced economic losses, which made their lives difficult. If they could detect cattle diseases at an early stage, they could minimize economic losses. It would also help improve their financial situation (Rony et al., 2021). Now is the time of the technological revolution. In today's digital age, agriculture has been connected with the latest technology, and farmers could use the latest technology in their smartphones, which was very important. However, there was still a lack of knowledge to train farmers in

issues like disease detection and classification. Because of this, we wanted to develop a method that would help domain experts in disease identification and farmers who were not familiar with animal diseases. The method that we focused on was studying animal disease identification based on deep learning techniques.

As we mentioned earlier, the research conducted in this area so far has not been sufficient in many ways. The use of deep convolutional neural networks (DCNNs) for cattle disease identification is a relatively new area of research, and there have been few studies on the application of DCNNs for this purpose. Additionally, to the best of our knowledge, few researchers used common deep learning algorithms that focused on common features in cattle images. That was why we were interested in cattle disease research and in using deep learning techniques, which enabled us to capture significant features. Therefore, we proposed to develop deep CNN method from animal disease red–green–blue (RGB) images.

1.3. Statement of Problem

According to (Birhanu, 2014) Livestock in Ethiopia provides drought power, income to farming communities, and means of investment, and is an important source of foreign exchange earnings to the nation (Birhanu, 2014). for example, According to (Gebre-Amanuel et al., 2018) Ethiopia was one of the African nations, with the majority of its citizens living in rural areas and agriculture serving as the foundation of its economy with the highest population of cattle. Despite this fact, the economic contribution made by this enormous amount of cattle was not satisfactory. This leads to lower productivity due to endemic and transboundary cattle diseases. Addressing cattle health problems with a limited number of veterinarians in remote areas was another problem that seriously hinders their potential economic contribution, especially during outbreaks (Gebre-Amanuel et al., 2018).

Various diseases troubled several animals in the cattle business. Similarly, various diseases that affected cattle, such as dermatophytosis, foot-mouth disease (FMD), Lumpy Skin Disease (LSD), dermatophylosis, and Infectious Bovine keratoconjunctivitis (IBK), significantly lowered the quality and output of livestock. Early and accurate identification of such diseases was required for preventing the diseases accordingly before the loss. The disease identification process depended on the knowledge of experts was quite time-consuming, laborious, and subjective. Therefore, advanced solutions were required. The advanced solution was a solution

that was able to automate the disease identification and classification process using different algorithms or approaches.

In previous research, Traditional machine learning methods used for feature extraction were manually engineered to fit the models. This hindered the machine's ability to learn complex patterns and instead forced it to focus on insignificant manual features. Most deep CNN models mainly focus on common spatial and common channel features without assigning special weight to disease-specific details. Previous methods, such as the K-nearest neighbor (KNN) algorithm, achieved good results. However, KNN was not robust enough to capture cattle disease lesion-specific details. The performance of cattle disease recognition in previous works could be further improved by proposing deep CNN method and improving the number of training datasets used in prior research. We proposed to develop a cattle disease identification method that focused on discriminative cattle disease spots using a deep convolutional neural network approach.

1.4. Research Question

The following research questions were answered to the end:

RQ1:- Is the proposed deep convolutional neural network methods could capture disease-specific lesions to leverage better cattle disease recognition performance?

RQ2:- What techniques could improve training dataset for the proposed deep convolutional neural network method?

RQ3:- To what extent did the proposed deep convolutional neural network method improve cattle disease identification compared to other related works?

1.5. Objectives

1.5.1. General Objective

The main objective of this research was to develop a deep convolutional neural network method for cattle disease identification.

1.5.2 Specific Objectives

To achieve the general objective, the following specific objectives are necessary:

- Study deep CNN method that focuses on target cattle disease spots instead of the whole image and image background.

- Study effective mechanisms to improve training dataset for the proposed deep convolutional neural network method.
- Evaluate and compare our proposed method with other related works on cattle disease datasets.

1.6. Scope and Limitation

The focus of this research was aimed to develop modeling that was able to identify cattle disease with better accuracy using deep CNN. However, this thesis work did not include an estimation of the severity of the detected disease or medical treatment recommendations.

Due to dataset problems and the time we had to conduct research, not all cattle diseases were included, but some diseases that occurred in Ethiopia were included. Therefore, the study focused on the five most common cattle diseases, including Lumpy Disease (LD), Infectious Bovine Keratoconjunctivitis (IBK), dermatophytosis, Foot_mouth Disease, and Wart.

1.7. Significance of the Study

This study achieved the ability to enhance the effectiveness of the classification of diseases in the cattle population, which can have positive implications for a society dependent on agriculture and pastoralism. Cattle owners residing in remote areas no longer have to transport sick animals over long distances for disease identification, thanks to this advancement. Additionally, animal health professionals working in rural towns can utilize this methods to provide efficient diagnoses. Moreover, investors engaged in animal farming can employ this method for their benefit. Veterinary and animal science experts can also benefit from this methods, using it as a decision-support tool and for student training. Furthermore, early disease control facilitated an increase in the livestock export market, preventing significant impacts on the cattle population.

1.8. Organization of the Research

This thesis has been structured into seven chapters, as outlined below.

Chapter Two explores the background literature, related works, and effective mechanisms for increasing training dataset. It also provides comprehensive descriptions of cattle diseases and explores the theoretical framework of deep learning, specifically deep convolutional neural networks (CNN).

Chapter Three presents the research methodology employed in developing the solution. It discusses the methods and techniques used, including data collection mechanisms, preprocessing, and augmentation techniques. Furthermore, it covers the development tools utilized and outlines the model and evaluation methods employed.

Chapter Four provides a detailed description of the proposed model. It explains the architecture of the proposed solution, the employed loss functions, the proposed attention mechanism architecture, as well as the utilization of transfer learning for the model. Additionally, it delves into the training process employed.

Chapter Five explains the implementation of the proposed solution, detailing the setup of the working environment. It also showcases the training implementation through the use of sample code.

Chapter Six explains the results obtained from various sources. It includes the results obtained from the baseline model, the proposed model, and related models, while also comparing these results with other existing works.

Chapter Seven serves as a summary and conclusion of the thesis. It encompasses the overall findings, conclusions, and results. Furthermore, it explores any remaining open issues in the research that warrant further investigation.

CHAPTER TWO

LITERATURE REVIEW

2.1. Introduction

This chapter focuses on the description of cattle disease, the data preparation mechanism for improving training data size, building block of CNN architecture, and the studied deep CNN method that focuses on target disease spots, related work. And also transfer learning that used to increases model performance are discussed.

2.2. Cattle Diseases

The cattle have been affected by several diseases (like lumpy, foot-mouth, dermatophytosis or Ringworm, Infectious Bovine Kerato-conjunctivitis, and wart) that highly reduce the quality and quantity of the livestock product. In the early days, the controlling and analysis of cattle diseases were performed manually through visualization. This requires experienced individuals and is laborious, time-consuming, and subjective.

Lumpy disease (LD) is an infectious, eruptive, and sometimes deadly bovine illness caused by a Poxviridae virus (Aiello et al., 2016). Figure 2.1 illustrates the distinctive features of the condition, which include the presence of nodules on the skin and other areas of the body. These nodules are round, slightly raised, firm, well-defined, and often accompanied by pain. The virus responsible for this condition has substantial economic consequences, as it commonly leads to chronic skin damage in animals, consequently reducing the market value of their skin. Additionally, this ailment frequently results in reduced milk production, little growth, infertility, miscarriage, and in severe cases, even death. The skin nodules themselves contain a solid, creamy-gray or yellow tissue mass (Addise Ambilo, 2013).

The initial occurrence of LD in Ethiopia was documented in 1983, specifically in the northwestern region of the country, located southwest of Lake Tana (Addise Ambilo, 2013; Molla et al., 2017). The disease has now expanded its reach to nearly every corner of the country. Considering the widespread occurrence of the disease and the substantial size and structure of the cattle population in Ethiopia, it is highly likely that LD is among the most economically significant livestock diseases in the country (Molla et al., 2017). The control of

LD can be achieved through vaccination, restriction of animal movement, and eradication of infected and exposed animals. However, this requires adequate financial, infrastructural, human resources, and information systems. Since it has not been able to apply all of these techniques in Ethiopia due to the country's current conditions, immunization has been used as the primary practical method of LD control for many years (Molla et al., 2017).

Figure 2. 1:- Lumpy disease

Bovine Papillomatosis, commonly referred to as **warts**, is a condition triggered by a contagious and infectious virus called bovine papillomavirus (BPV). The virus is transmitted when infected cattle come into direct contact with uninfected cattle (Lake, 2013). It is a contagious viral disease in animals characterized by the presence of multiple skin tumors or growths, primarily on the head and neck regions. Although it typically regresses on its own, some cases may persist for an extended period and progress into a malignant form. The disease manifests as either benign nodular lesions, hand-like projections, or small cauliflower-like growths on the skin. These growths originate from the stratified squamous epithelium and can appear as solitary growths or in multiple clusters. Common sites for the development of cutaneous warts include the head, eyelids, ears, neck, dewlap, brisket, shoulders, and legs. Occasionally, the back, para-genital area, and lower abdominal region may also be affected. Warts can occur naturally or be acquired.

They are typically specific to certain species and caused by viruses, with a higher incidence observed in young animals. In winter, young cows commonly develop warts on the skin of their eyelids and along the lower abdomen. However, these growths often disappear on their own when the young animals are exposed to grass in the early spring (Lake, 2013). Figure 2.2 illustrates the symptoms observed in cattle affected by warts. Among the most common tumors in cattle, warts tend to emerge on the head and dewlap when the animals are between 6 and 24

months old. The virus responsible for warts can be transmitted through physical contact or via tools like halters or milking machines. Warts typically exhibit lesions characterized by a flat, wide base and growths resembling cauliflower. In immunosuppressed animals, warts may become larger and more persistent, as is the case with an ongoing bovine viral diarrhea virus infection.

Figure 2. 2:- Bovine Papillomatosis (wart)

Dermatophytosis is known as **Ringworm.** it is a transmissible infectious skin disease caused by Trichophyte verrucous, a spore-forming fungus (Lake, 2013). Ringworm is a skin condition caused by a fungus that is prevalent worldwide. This ailment has significant economic implications for farmers, as it adversely affects growth rates during the active stage of infection and leads to damage to the animals' hides. Ringworm is commonly observed in young animals and can easily be transmitted to humans (a zoonotic disease). Figure 2.3 illustrates the symptoms of ringworm, characterized by slightly raised, well-defined, discolored lesions that are primarily found on the head and neck, although they can also manifest on other parts of the body. In calves, the most frequently affected areas are around the eyes, ears, and back, while adult cattle commonly experience ringworm on the chest and legs (Gebre-Amanuel et al., 2018).

Figure 2. 3:- Ringworm disease

Conjunctivitis, also known as **Infectious Bovine Kerato-conjunctivitis**, is a prevalent and highly transmissible eye ailment that primarily impacts young calves. The main offender behind this condition is Moraxella Bovisa, and in severe cases, it can result in vision loss for the affected animals (Seid, 2019). Typically lasting for a period of 23 days, this condition exhibits intermittent small, slightly translucent areas on the membrane that become noticeable every 2 days. The membrane lesions primarily occur in the central region of the eye (Firdaus et al., 2015).

Figure 2. 4:- Infectious Bovine Kerato-conjunctivitis

Foot_mouth disease (FMD) is a globally prevalent, extremely contagious epidemic that primarily impacts cattle. The primary causative agent of FMD is Aphthovirus (Vyas et al., 2019). The pathogen possesses the ability to travel a distance of 300 kilometers through the air, bridging the gap between infected and unaffected cattle (Vyas et al., 2019). Livestock farming serves as the primary income source for most families within the community, and their livelihoods are put at risk due to the occurrence of cattle deaths. Detecting these deaths at an early stage is crucial for minimizing mortality rates (Fakhrul-Islam et al., 2016).

This external disease specifically affects cattle, adversely affecting livestock production and hindering import and export trade of livestock and animal products. Insufficient knowledge about this condition in cattle makes accurate diagnosis challenging, and failure to diagnose it correctly can lead to significant challenges and complications. Rural communities face the need for efficient and automated methods to detect diseases in cattle. Various external infections can affect domestic animals, including Infectious Bovine Kerato-conjunctivitis (IBK) and noteworthy diseases like Foot_mouth disease (FMD), Bovine Tuberculosis, Anthrax, Lumpy

Disease (LD), and Bovine Viral Diphtheria (BVD). Typical symptoms of FMD include fever, blisters on the lips, mouth, and tongue, as well as pus-filled lesions in the hooves' central area. After recovering from the disease, many affected animals experience weakness and increased vulnerability. While the disease is not highly fatal in adult animals when the mother is affected, it poses a significant risk to very young animals due to their dependency on milk, resulting in a considerably high mortality rate (Fakhrul-Islam et al., 2016).

Figure 2. 5:- Foot and mouth disease

2.3. Data Preparation Mechanism for Cattle Disease Identification

Data preparation is a critical step in the development of deep convolutional neural networks (DCNNs) (Whang & Lee, 2020). The quality and reliability of the input data can significantly impact the performance of a DCNN. To ensure optimal results, it is important to effectively preprocess and prepare the data before training the network. Several techniques and mechanisms can be used for effective data preparation in DCNNs, including data augmentation, Synthetic data generation, Collecting more diverse data, and Careful annotation(Ding et al., 2016). This section will briefly discuss the importance of data preparation in DCNNs and some of the most effective techniques for ensuring high-quality data input. Once the dataset is prepared, it is fed into the DCNN, which consists of several layers of convolution, activation, and pooling operations (Ding et al., 2016). These operations help the DCNN identify and extract important features from the images, such as color, texture, and shape. The DCNN is then trained using an optimization algorithm, such as stochastic gradient descent (SGD), Adam, AdaGrad, and Adadelta to minimize the prediction error between the model's outputs. After training, the model is evaluated on the validation and testing datasets to determine its accuracy in recognizing diseases. Based on the evaluation results, the model can be fine-tuned by adjusting its hyper-parameters or adding more layers to improve its performance. Some effective mechanisms to

improve the size and diversity of training data for deep learning methods are synthetic data generation, collecting more diverse data, and data augmentation.

2.3.1. Synthetic Data Generation

Synthetic data generation is a technique for creating new data samples that are similar to the original training data (Dahmen & Cook, 2019). This can be achieved using generative models such as Generative Adversarial Networks (GANs) or Variation Auto-encoders (VAEs). By generating synthetic data, the size and diversity of the training data can be increased, helping the model to generalize better and overcome the limitations of limited training data (Stasaski et al., 2020). In the context of cattle disease identification, synthetic data can be generated by creating images of cattle with various diseases, allowing the model to be trained on a larger and more diverse set of examples**.**

To train the model to detect patterns and characteristics pertinent to the work at hand, it is necessary to provide synthetic data that is comparable to real data in terms of distribution and variability. Synthetic data generation can be performed using computer graphics, simulation, or other methods (Dahmen & Cook, 2019).

There are several benefits of using synthetic data for training deep learning models (Dahmen & Cook, 2019). One of the main benefits is that synthetic data can be generated in large quantities, allowing the model to learn from a much larger dataset. This can help to overcome the limitations of small training datasets, which can result in overfitting or poor generalization performance. Synthetic data can also be generated with controlled variability, allowing the model to learn from data representing different scenarios or conditions. This can improve the model's robustness and ability to generalize to new data. In addition, synthetic data can be generated with different levels of noise or artifacts, allowing the model to learn to cope with noisy or imperfect data. These benefits make synthetic data generation an attractive technique for improving the performance of deep-learning models for cattle disease recognition.

However, using synthetic data generation in the proposed Deep CNN model for cattle disease identification is that it may not accurately capture the true diversity and variability of real-world data. Synthetic data may lack the refined degrees and complexities that can be found in real data, which can lead to over-generalization and suboptimal performance in real-world scenarios. Additionally, synthetic data generation often requires specialized knowledge and skills to

generate high-quality data, and it may also be computationally expensive. This can make it difficult to scale and distribute the synthetic data generation process for large-scale training datasets. Another disadvantage is that synthetic data may not reflect the real distribution of data, which can lead to bias in the model's predictions and undermine its overall accuracy.

2.3.2. Collecting More Diverse Data

Collecting more diverse data is an important step in improving the performance of a deep learning model (Stasaski et al., 2020). By collecting images of cattle from different sources and environments, and capturing the images under different lighting and weather conditions, the diversity of the training data can be increased, allowing the model to learn a wider range of features and patterns (Whang & Lee, 2020). In the context of cattle disease identification, collecting diverse data can help the model to better handle different types of cattle diseases and improve its accuracy.

However, gathering more diverse data for the proposed Deep CNN model for identifying cattle illness has some drawbacks, including it can take a lot of time and resources, including employees and equipment, to gather vast volumes of different data. Obtaining different data can be difficult, especially if the data is sensitive or specialized to a particular illness or condition. Privacy and security issues might develop during data collection, particularly if the data contains sensitive information or if the persons or organizations involved are worried about how their data will be used. Large-scale, diversified data labeling may be time-consuming and challenging, and it takes a lot of knowledge to ensure that the labels are accurate. Diverse data might be heterogeneous, which could indicate that it has a variety of formats, data kinds, and quality. This could move how well the proposed Deep CNN model performs.

2.3.3. Data Augmentation

Shorten & Khoshgoftaar, Stated that One of the simplest ways to increase the size of the training data is to apply data augmentation techniques such as random rotation, flipping, color change, blur, alter brightness and contrast, and scaling (Shorten & Khoshgoftaar, 2019a). This artificially increases the size of the training data and helps the model generalize better.

One of the main benefits of data augmentation is that it allows for training a deep learning model with limited annotated data (Ding et al., 2016). By artificially expanding the size of the training

set, the model has a better chance of learning the underlying patterns and relationships in the data. This is especially important in domains where annotated data is difficult or expensive to obtain, such as medical imaging or natural language processing. With data augmentation, a deep learning model can still achieve good performance, even with a relatively small annotated dataset (Shorten & Khoshgoftaar, 2019b).

Data augmentation is a technique that is used to increase the size and diversity of the training data for a deep learning model (Ding et al., 2016). There are several advantages to using data augmentation in the proposed Deep CNN methods for cattle disease identification. By augmenting the training data, the model can become more robust to variations in the data, such as changes in lighting or orientation. This helps to make the model more generalizable and less likely to over-fit the training data (Shorten & Khoshgoftaar, 2019b). With larger and more diverse training data, the model can learn a better representation of the data, which can result in improved performance. This is especially important when the original training data is limited in size or diversity. By exposing the model to a wider range of variations in the training data, data augmentation can increase the model's confidence in making predictions. This is because the model has seen a greater variety of data, and is better equipped to handle new, unseen data.

Therefore, data augmentation can be a powerful tool for improving the performance and robustness of a deep learning model, such as the proposed Deep CNN methods for cattle disease identification.

2.4. Deep Learning

Deep learning methods represent a category of machine learning algorithms that demonstrate exceptional proficiency in handling unorganized or unstructured data (Khan et al., 2020b). Deep learning approaches surpass existing machine learning methods by enabling computational models to progressively learn features from data at multiple levels (Moses et al., 2021). With the rise of powerful computing resources, deep learning methods utilizing deep neural networks have gained significant popularity. When confronted with unstructured data, deep learning exhibits enhanced capability and adaptability by efficiently analyzing a vast array of features. The deep learning algorithm traverses multiple layers, each responsible for extracting features in a progressive manner and passing them to subsequent layers. The initial layers focus on

extracting basic features, while subsequent layers combine these features to form a comprehensive representation (Moses et al., 2021).

2.4.1 Conventional Neural Network

Khan et al., stated that CNN is a new system designed for both feature extraction and classification. The convolution and pooling layers of CNN were used for extracting important features of images. In which fully connected layers received the extracted features from all previous neurons as input to map them into the classification layer (Khan et al., 2020b). Deep learning algorithms have been found in high-level feature extractors that learn high-level semantic features using the lower-layer low-level features. Features extracted using the deep CNN have not been invariant to any of the affine transformations. (Ren et al., 2016) stated that the conventional neural network has been recognized as one of the deep neural networks designed for image recognition, detection, and classification (Ren et al., 2016). Therefore, the neuron in a layer was connected to a small region of the layer in the previous layer. Furthermore, CNN has been utilized with element-wise multiplication of matrices followed by submission known as convolution. As (Khan et al., 2020b) clearly stated that CNN contained many components and related elements like convolutional Layers, pooling layers, activation functions, fully connected layers, Los functions, regularizations, and optimizations (Khan et al., 2020b).

2.4.2 Building Block of CNN

A. Convolutional Layer

Many scholars agreed that the convolutional layer has been the basic component of CNN, designed to learn the feature representation of the data by taking row inputs or feature maps to compute the different features in the input with different kernels (filters). The new feature maps that were created after kernel filtering and applying element-wise multiplication and addition were used as the input for the following layer. Because it incorporates filters that change pictures, the convolutional layer's operating principles differ from those of other neural network layers.

B. Stride Number

The stride number indicated the number of pixels skipped during the convolutional filter or kernels slide over the input vector receptive fields. Receptive fields are portions of the input

vector that have the same dimension as the convolutional kernel. The element-wise dot products of the receptive fields and the kernels produced the feature map or convolved features.

C. Padding

On the other hand, a padding operation with zero padding or padding the borders of the original image with zero has been applied to a convolution to keep the original image size. The spatial dimensions of the input volume have been reduced too quickly without zero padding.

D. Pooling Layer

The pooling layer is responsible for reducing the size of the image or feature maps that came from the previous layer. The system has simple operations to compute with common pooling examples of max pooling and average pooling (Khan et al., 2020a)

Max pooling operation was worked by selecting the maximum pixel values of the convolution areas and comparing all the pixels of the convolution areas.

Average pooling has been worked by adding all the pixel values of the convolution area followed by division with the number of pixels in the convolution area.

E. Activation Layer

Furthermore,(Khan et al., 2020a). have developed an activation layer aimed to determine and normalize the output of the network into the same range depending on the activation function (sigmoid, Tanh, and ReLU activation function) (Khan et al., 2020a).

The sigmoid activation function was utilized to normalize the output of the neuron to the range between one and zero. This type of activation function has a vanishing gradient problem for the fact that the derivative function comes to be close to zero as the input value has been very large or very small.

Tanh activation function has been developed with a range between a positive one and a negative one. Even though the function has a vanishing gradient problem, it was better than the sigmoid activation function for the fact that the latter allowed a negative value.

Rectified linear unit (ReLU*)* was developed with a derivative function to overcome the vanishing gradient.

F. Fully Connected Layer

Finally, a fully connected layer (Flatten Layer) used to compute the final output probabilities for each class was developed. It was applied at the end of the CNN model before applying the classifier layer (usually Softmax). The Fully connected Layer was utilized for transforming the feature maps extracted in the final convolution or pooling layer into a one-dimensional (1D) array of numbers.

The activation function used in the last fully connected layer for a multiclass classification problem was a **soft-max function** for normalizing the last fully connected layer output values into target probabilities.

Beyond the activation layers, different **Optimization techniques** were developed by different scholars. Those Optimization algorithms were utilized for decreasing the losses and increasing the accuracy of models. Generally, there have been two optimization algorithms **adaptive optimization algorithms** and **Non- adaptive optimization algorithms**. In the case of non-adaptive optimization algorithms, the learning rate was constant for all features and needed manual selection as indicated in stochastic gradient descent, Momentum, and Nester Momentum algorithms. However, in adaptive optimization algorithms, the learning rate was based on the occurrence of features as clearly stated with AdaGrad, Adadelta, and Adam algorithms.

G. Dropout

Dropout is a fundamental technique employed in neural networks to counteract overfitting. It involves selectively removing both hidden and visible nodes within the network. In our research, we applied dropout specifically to the fully connected layer to mitigate overfitting. This approach effectively regulates the model, reducing overfitting while simultaneously improving the complexity of the convolutional neural network (CNN). To facilitate multi-class classification within CNNs, we utilized the SoftMax activation function in the last fully connected layer. This function offers a comprehensive stochastic activation approach, assigning a probability value ranging from 0 to 1 to each item.

2.4.3. Deep CNN Architecture

The architectures of deep convolutional neural networks (CNNs) have evolved for several years and become a crucial issue more on accuracy and speed. For this reason, different scholars have developed different architectural designs at different times.

A. LeNet-5

One of the most famously known architectures was LeNet-5**.** It was introduced by Yann LeCun and others in the year 1998 to classify handwritten digits(Khan et al., 2020a). LeNet5 has a total of seven layers, with three convolutional layers, two average pooling layers, one fully connected layer, and one output layer. The sigmoid function was used to incorporate nonlinearity before a pooling operation. The output layer was incorporated with RBF to classify 10 digits. **LeNet** was the first architecture of CNN with a reduced number of parameters that automatically learned features from raw pixels

But, The deep CNN LeNet5 was limited to the handwritten digits classification system and it didn't perform well for all image classes(Khan et al., 2020a).

B. AlexNet

The AlexNet CNN was developed and proposed by Alex Krizhevsky in collaboration with Ilya Sutskever and Geoffrey Hinton, who was Krizhevsky's Ph.D. advisor., to enhance the learning capacity of the model by creating a deeper layer (with a total of 11 layers) through the application of different optimization parameters. It consisted of five convolution layers, with three pooling layers and three fully connected layers in between the input layer and the output layer. The input layer was the first layer to define the input dimensions with a 227 x 227 x3 image size and having an output layer as the final layer responsible for classification(Khan et al., 2020b(Khan et al., 2020b)

AlexNet design was employed with ReLU as a non-linear activation function. On the other hand, (Khan et al., 2020b) proposed an effective CNN architecture design called VGGNet for accelerated architectural design. It was the next-best design of the 2014 ILSVRC. The design's main contribution was highlighting the importance of network depth in enhancing CNN's classification and recognition accuracy.

C. VGGNet

The VGGNet architecture has consisted of two convolutional layers; both of them used the ReLU activation function. Besides, the single **max-pooling layer** and the **fully connected layers** were also integrated using a ReLU activation function. The final layer of the model was the **SoftMax** layer utilized for classification. The VGGNet architectural design was deeper than

the AlexNet design. It has exploited a series of one to more convolutional layers with a similar number of filters having a size of 3×3, astride of 1×1, with the same padding. Thus, the output size was the same as the input size for each filter in each convolutional layer. VGGNet design has used a rectified linear activation function followed by a max-pooling layer with a size of 2×2 and astride of the same dimension. The output feature maps of the final pooling layer were typically flattened; transformed into an array of numbers (or vectors), and connected to one or more fully connected layers, also known as dense layers, in that, every input was connected to every output by a learnable weight (Khan et al., 2020b).

However, **VGGNet** was faced with the main drawback; the usage of 138 million trainable parameters, which computationally made it too expensive and difficult to implement on low-resourced systems.

D. Inceptionv3

Inceptionv3 is a convolutional neural network (CNN) developed by Google, consisting of a remarkable depth of 50 layers. It was meticulously constructed and trained, and detailed information about its architecture and training process can be found in the research paper titled "Going Deeper with Convolutions" (Szegedy et al., 2015). The pre-trained version of Inceptionv3 with ImageNet weights can classify a wide range of objects, with the capacity to handle up to 1000 different categories. In contrast to the VGG19 network, Inceptionv3 accepts larger image inputs, specifically 299x299 pixels. In the 2014 ImageNet competition, Inception emerged as the victor, while VGG19 secured the runner-up position (Orhan G, 2020).

E. Deep Residual Network (ResNet)

ResNet (Residual Network) was proposed by 4 scholars from Microsoft Research in 2016 (He et al., 2016a). Reducing network parameters further simplifies the network structure, and connecting the residual element to the output of the element through a short connection effectively solves the gradient disappearance issue brought on by a deep neural network's increased depth. It is a kind of convolutional neural network (CNN) in which the output of the current layer is combined with the input from the previous layer. This skip connection facilitates the network's learning process and improves performance.

The ResNet architecture has shown effective in a variety of applications, including semantic segmentation, object detection, and picture classification. Additionally, ResNets are composed of layers; the depth of these networks may be altered to represent space at any level. The model's success can be attributed to some factors, including the large receptive fields that capture more information about each pixel in an image, the separation of the localization and classification stages, the computational efficiency at higher levels, the effective encoding schemes with simple arithmetic operations, and the increased accuracy as features are extracted further into the network.

Since 1989, significant progress has been made in the architecture of convolutional neural networks (CNNs). These advancements encompass various aspects such as parameter optimization, regularization techniques, structural refinements, and other enhancements. Notably, the improvement in CNN performance can be largely attributed to the reorganization of processing units and the introduction of novel building blocks. The advancements in CNN architecture primarily focus on enhancing depth and spatial utilization. CNN architectures can be classified into seven categories based on the types of architectural modifications employed. These categories include spatial utilization, depth, multi-path, width, feature-map exploitation, channel boosting, and attention-based CNNs. The taxonomy of CNN architectures is presented in Table 2.1 for reference (Khan et al., 2020b).

Table 2. 1: Deep CNN architectures taxonomy

Deep CNN Architecture						
spatial exploitation based on CNN	Depth based CNN	multi-path based	Width based Multi-connection CNN	feature-map exploitation based CNN	channel boosting based on CNN	attention-based CNNs
LeNet	Highway Nets	Highway Nets	WideResNet	Squeeze and Excitation	Channel Boosted CNN using TL	Residual Attention Neural Network
AlexNet	ResNet	ResNet	Pyramidal Net	Competitive Squeeze		Convolutional Block Attention

ZfNet	Inception-V3, V4	DenseNet	Xception	and Excitation		Concurrent Squeeze and Excitation
VGG	Inception-ResNet		Inception Family			
GoogleNet			ResNext			

2.5. Deep CNN Techniques for Cattle Disease Spot Identification

State-of-the-art deep CNN (Convolutional Neural Network) offers several advantages over traditional CNNs and machine learning methods in the context of cattle disease identification (Zhou, 2020). Cattle disease identification requires capturing complex relationships between various disease-specific features and the corresponding diseases. Deep CNNs excel at modeling such complex relationships by leveraging multiple layers of non-linear transformations. The deep architecture enables the network to learn intricate patterns and relationships, resulting in improved disease recognition and classification performance.

Deep CNNs are well-suited for large-scale datasets, which is often the case in cattle disease identification research. The large number of learnable parameters in deep CNNs allows for better utilization of the available data, leading to more accurate models. Deep CNNs have the capacity to learn from a diverse range of samples, making them effective in handling the variability and complexity of cattle disease images. Deep CNNs have the advantage of transfer learning, where pre-trained models trained on large-scale image datasets can be used as a starting point for cattle disease identification tasks (Kim et al., 2022) . This allows us to leverage the knowledge and representations learned from a different but related domain, reducing the need for large labeled datasets. Transfer learning can significantly speed up the training process and improve model performance, especially in scenarios where labeled cattle disease images are limited (Gupta et al., 2022; Kim et al., 2022).

The deep CNNs has good in their ability to automatically learn and extract meaningful features from raw image data, handle complex relationships, adapt to large datasets, and benefit from transfer learning. These capabilities make deep CNNs a powerful approach for cattle disease identification compared to traditional CNNs and machine learning methods

2.5.1. Region-based DCNN

Region-based CNN is a type of DCNN that operates on small regions within an image, rather than the whole image (Chandana et al., 2020). This allows the model to focus on specific regions within an image and reduces the influence of the background. In the context of cattle disease identification, a region-based CNN can be used to identify the target cattle disease spot by operating on small regions of the image that contain the disease spot. Region-based Convolutional Neural Networks (R-CNNs) are a type of deep learning method that focuses on specific regions of interest within an image, rather than processing the entire image (Seetharaman & Mahendran, 2022). R-CNNs can be useful for cattle disease identification, as they allow the model to focus on the target cattle disease spot and ignore the image background (Seetharaman & Mahendran, 2022). This can lead to improved performance, as the model is not distracted by irrelevant information.

There are different variants of R-CNNs, including Fast R-CNN and Faster R-CNN, each with its strengths and weaknesses (Bharati & Pramanik, 2020). Fast R-CNN is a straightforward extension of the Convolutional Neural Network (CNN) architecture, where the entire image is passed through the CNN to generate a feature map, and a region proposal algorithm is used to generate a set of regions of interest. These regions are then fed into a separate fully connected layer for classification. Faster R-CNN, on the other hand, uses a region proposal network (RPN) to generate the regions of interest, allowing the model to learn the regions of interest end-to-end (Y. Liu, 2018).

Using R-CNNs for cattle disease identification involves training the model to identify the regions of interest, which correspond to the target cattle disease spots, and then using these regions to classify the diseases. This can lead to improved performance compared to using traditional CNNs, as the model can focus on the most important parts of the image.

However, the Complexity of Region-based DCNN methods requires additional processing steps to segment and crop the disease spot from the whole image. This increases the complexity of the system and slows down the processing time. Segmentation algorithms used in Region-based DCNN methods may not always provide accurate results, which can negatively impact the recognition accuracy of the model (C.-K. Yang et al., 2022). Annotating the data to extract the regions of interest can be time-consuming and require significant human effort and resources.

Region-based DCNN methods require a large number of annotated images with the disease spot visible, which may not always be readily available for training the model (Bharati & Pramanik, 2020). Region-based DCNN methods may not generalize well to other types of images or datasets, which limits their transferability to different domains.

2.5.2. Segmentation Technique in DCNN

Segmentation is a technique that involves dividing an image into multiple segments or regions (Yogamangalam & Karthikeyan, 2013). In the context of cattle disease identification, image segmentation can be used to separate the target cattle disease spot from the background, allowing the model to focus on the disease spot. Segmentation algorithms such as U-Net, Mask R-CNN, and Deep Lab can be used to implement this approach (Kaur & Kaur, 2014).

Segmentation techniques in deep learning refer to the process of dividing an image into multiple segments or regions of interest, with each segment representing a unique object or region in the image(Khan et al., 2020a). This can be useful for cattle disease identification, as it allows the model to focus on specific regions of the image, such as the cattle, rather than the entire image and background.

Segmentation techniques, although useful. However, it may not always result in accurate segmentation of images, particularly in cases where the boundaries between the affected and healthy tissue are not clearly defined (Kaur & Kaur, 2014). Image segmentation can be a time-consuming process, especially for large datasets, and can slow down the overall disease identification process. Some segmentation techniques may require manual intervention, such as manual correction of incorrect segmentations, which can be time-consuming and prone to human error.

Segmentation techniques may work well for one dataset, but may not perform well on a different dataset, especially if the image characteristics differ greatly. Image segmentation algorithms are often sensitive to image noise, and this can result in incorrect segmentation and, therefore, incorrect disease diagnosis (Yogamangalam & Karthikeyan, 2013). Segmentation techniques can be computationally intensive and require significant computational resources, making it challenging to implement on resource-limited systems. The time required for image segmentation can make it unsuitable for real-time applications where rapid disease diagnosis is crucial.

2.5.3. Attention Mechanism in DCNN

One of the main importance in the field of deep learning nowadays is the attention mechanism (Z. Niu et al., 2021). It takes its cues from human biological systems that, while processing a lot of information, prefer to concentrate on the salient details. The development of deep neural networks has led to the widespread use of the attention mechanism across a range of application fields.

According to how they develop, the attention mechanisms of humans may be split into two kinds (Tsotsos et al., 1995). The first group is saliency-based attention, or bottom-up unconscious attention, which is influenced by outside stimuli. For instance, during a conversation, individuals are more likely to hear loud voices (Hochreiter & Schmidhuber, 1997). The max-pooling and gating process used in deep learning is comparable to this, which passes more appropriate values (i.e., larger values) to the next step. Focused attention, often known as top-down conscious attention, is the second category. Focused attention is defined as attention that has a definite goal and relies on particular tasks. It helps people to choose and actively focus their attention on a certain object. The majority of the attention mechanisms employed in deep learning involve focused attention since they are created with certain goals in mind. Unless there are unique circumstances, the attention mechanism outlined in this study generally relates to focused attention.

As mentioned above, the major method for addressing the issue of information overload is to use the attention mechanism as a resource allocation technique. It can process more significant data with fewer processing resources in the situation of restricted computing power. Consequently, several researchers draw attention to the field of computer vision. (Ramirez-Moreno et al., 2013) A model of visual attention based on saliency that draws out regional low-level visual cues to identify possible salient areas has been developed. In the domain of neural networks, (Mnih et al., 2014) used the recurrent neural network model's attention mechanism to classify photos. (Bahdanau et al., 2014) used the attention mechanism to carry out alignment and translation on machine translation activities at the same time. Subsequently, the Attention mechanism has been used in a variety of activities, including the creation of image captions, text classification, machine translation, action recognition, image-based analysis, speech recognition, recommendation, and graph analysis (Z. Niu et al., 2021).

2.5.3.1 Applications of Attention Mechanism in Deep CNN

The attention mechanism has been widely used in various deep earning tasks such as saliency detection (Du & Li, 2018), crowd counting (X. Yang & Lu, 2021), and facial expression recognition (Fernandez et al., 2019). The operation may determine which features are most relevant for categorization by learning an intermediate attention map, applying the element-wise product to attention maps and source feature maps, and then weighing the relative significance of the various features.

According to authors (J. Li et al., 2020) in their paper entitled "Attention Mechanism-based CNN for Facial Expression Recognition", For the task of facial expression recognition, The essential body components like the eyes, nose, and mouth are where most of the distinguishing traits are located. These important qualities are given more weight by the attention mechanism, which enhances the performance of expression recognition (J. Li et al., 2020).

A. Spatial attention

Spatial allows neural networks to learn the positions that should be focused on, Through this attention mechanism, the spatial information in the original picture is transformed into another space, and the key information is retained (Z. Niu et al., 2021).

Mnih et al., stated that a spatial attention model that uses a single RNN with a peek window as its input and the network's internal state to decide where to focus next and create control signals in a dynamic environment is shown (Mnih et al., 2014). (Souza & Zanchettin, 2019) was introduced a differentiable spatial transformer network (STN) that can find out the areas that need to be paid attention to in the feature map through transformations such as cropping, translation, rotation, scale, and skew. The spatial transformer module, as opposed to pooling layers, is a dynamic device that can dynamically spatially transform a picture (or a feature map) by producing the proper transformation for each input sample (Souza & Zanchettin, 2019).

B. Channel attention

Channel allows neural networks to learn what should be focused on (Z. Niu et al., 2021). (Hu et al., 2020) proposed the squeeze-and-excitation (SE) network that channel-specific feature responses were adaptively re-calibrated using explicit channel interdependency modeling. In that squeeze and excitation module, it used global average-pooled features to compute channel-

wise attention. (H. Yang et al., 2022) proposed the Selective Kernel Network that improved the efficiency and effectiveness of object recognition by adaptive kernel selection in a channel attention manner. Besides, (Stollenga et al., 2014) proposed a channel hard attention mechanism that improved classification performance by allowing the network to iteratively focus on the attention of its filters (Stollenga et al., 2014).

Therefore, the combination of Channel and spatial attention mechanism (CCSAM) was used in this paper to focus on the target area from the cattle disease image. Because Attention mechanism allows the model to focus on the most relevant features and ignore irrelevant information. The attention mechanism allows the model to adapt to different image conditions, such as varying lighting, rotation, and orientation (J. Li et al., 2020). This makes it more robust to different real-world scenarios and less sensitive to input variations (Z. Niu et al., 2021). The attention mechanism provides insight into what the model is focusing on and why. This makes it easier to understand the model's decision-making process and improve it, if necessary. Attention mechanism can be implemented in a computationally efficient manner, as it only requires a small number of additional operations compared to traditional deep learning models(J. Li et al., 2020)(Z. Niu et al., 2021).

Overall, the use of attention mechanisms can greatly enhance the performance and robustness of deep learning models for cattle disease identification, providing a more accurate and interpretable solution.

2.6. Transfer Learning

The transfer is a deep learning strategy that starts with a model that has already been trained to handle a similar but distinct problem(Tan et al., 2018). Instead of building a deep learning model from scratch, transfer learning's basic idea is to use the information gained from one job and apply it to another that is similar (Weiss et al., 2016). As a result, the quantity of data and processing resources needed to train a new model can be greatly reduced. This is because the pre-trained model has already absorbed much of the data's low-level characteristics and patterns (Weiss et al., 2016).

The pre-trained model is often adjusted for the new job in transfer learning by changing the weights of certain of its layers (S. Niu et al., 2020). The objective is to maintain the model's learning from the prior task while adjusting it to the current challenge. Instead of starting from

scratch with a big dataset, this may be accomplished by training the model using a small quantity of annotated data from the new assignment. When training deep learning models from scratch, transfer learning is frequently employed in computer vision, natural language processing, and speech recognition, where huge annotated datasets are frequently needed (Weiss et al., 2016)(Tan et al., 2018). Transfer learning allows researchers and practitioners to make use of already created models and data, cutting down on the time and expense required to create new models and enhancing the effectiveness of deep learning algorithms on brand-new tasks.

The proposed model's implementation of transfer learning has the benefit of greatly decreasing the quantity of training data needed (Tan et al., 2018). This is because the model was trained before on a sizable, varied dataset, and the information learned from that training may be applied to assist the model in more accurately identifying the target disease in the smaller dataset. Transfer learning can also assist in resolving the overfitting problem, in which a model that has been trained on a limited dataset performs well on the training data but badly on unobserved data. The model may be tailored to the specific purpose of identifying cattle diseases by utilizing transfer learning, which improves performance and accuracy.

2.7. Related Works

Shah et al., proposed “animal skin disease detection by using image process operations” (Shah et al., 2019). In this research, their prototypes were performed on six common skin diseases with the help of KNN and SVM algorithms. The proposed system offers a non-invasive and accessible approach to detecting skin diseases. they utilizes image processing techniques such as feature extraction, filtering, image pre-processing, and segmentation to identify the affected area. They used KNN for feature extraction and SVM for the classifier. They employed KNN methods, KNN does not use for diseases with strong correlation and always computes the distance metrics (Deng et al., 2016). In this work, they perform 91 % classification accuracy.

Rai et al., proposed “ deep learning approach to detect lumpy skin disease in cows “(Rai et al., 2021). They used Pre-trained models like VGG-16, VGG-19, and Inception-v3 for feature extraction and then followed by multiple classifiers (k-NN, SVM, and ANN). The proposed architecture leverages machine learning techniques and pre-trained models for efficient feature extraction. Multiple classifiers are utilized for accurate disease detection. but, no standard dataset was mentioned, problem with the dataset is that it can be biased due to lack of proper

dimensions and resolution; but in their dataset, they have chosen only high resolution and proper pixel images and only Lumpy skin diseases are included.

Rony et al., studied “Cattle external disease classification using deep learning techniques” (Rony et al., 2021). They used CNN in both feature extraction and classification. Pooling and traditional CNN layer for feature extraction and fully connected layer for classification. This research is recently studied research on cattle disease classification with the best accuracy by Inception-V3 CNN architecture. The proposed model achieves a high accuracy of 95% and offers a potential solution to detect external epidemic diseases in cattle, aiding farmers and veterinarians in early-stage treatment. However, in feature extraction, CNN focuses only on a common feature without assigning a special weight to disease-specific details. they focus only on three cattle diseases and The paper acknowledges the limitations of limited raw data, leading to decreased accuracy and increased loss during iterations. Inadequate information and imprecise diagnosis may also impact the accuracy of the results. (Khan et al., 2020b).

Vyas et al., suggested “utilizing the Internet of Things to identify FMD and Mastitis illness in cattle (Internet of Things)” (Vyas et al., 2019). To accomplish this task, many types of sensor devices are used to portray various factors in cattle, including temperature, sound, and mobility. For disease detection, machine learning methods (Neural Networks) and a microcontroller are used. This article made no indication of the categorization model's accuracy, and the system is dependent on costly sensor devices that must be integrated into each animal.

Qiao et al., studied “Individual Cattle Identification Using a Deep Learning Based Framework”. The proposed framework is composed of two main parts: the first part is the CNN-based feature extractor that extracts features from the input images, and the second part is the LSTM-based classifier that classifies the extracted features into different categories (Qiao et al., 2019). The proposed framework was tested on a video dataset and achieved an accuracy of 88% for 15 frame video length. However, they only focuses on beef cattle which identify individual cattle body condition, in feature extraction, CNN focuses only on a common feature without assigning a special weight to disease-specific details and collecting real time video dataset for beef cattle is much more difficult.

Table 2. 2: Summary of related works

Authors	**Disease**	**Dataset**	**Method and Algorithm**	**Gap**
Shah et al	Animal Skin Disease, but specific skin disease are not mentioned	Direct captured image data	As a feature extractor, KNN, and a classifier, SVM	They employed KNN methods, KNN does not use for diseases with strong correlation. they utilizes image processing techniques such as feature extraction, filtering, image pre-processing, and segmentation to identify the affected area. However, The paper does not mention a specific dataset and they perform 91 % classification accuracy.
Vyas et al	Foot_mouth disease (FMD) and Mastitis Illness in Cattle	Real time data to find the disease	For disease detection, machine learning methods, & a microcontroller are used.	This article made no indication of the categorization model's accuracy, and the system is dependent on costly sensor devices, Internet of Things (IoT) that must be integrated into each animal
Rai et al	Lumpy Skin Disease	manually collected dataset , but specific details about the dataset are not provided.	For feature extraction VGG-16, VGG-19, and Inception-v3 pre-trained models, followed by several classifiers (k-	They have not mentioned the standard dataset, the problem with the dataset is that may be biased due to lack of proper dimensions and resolution; but in their dataset, They have chosen only high resolution and proper pixel images and also only Lumpy skin diseases are included. The achieved an accuracy of 92.5%.

			NN, SVM, and ANN)	
Rony et al	Lumpy disease(LD), Infectious bovine kerato-conjunctivitis (IBK) and Foot_mouth disease (FMD)	600 raw dataset	CNN in both feature extraction and classification	The training dataset they used are not enough and Only three external cattle diseases are studied. CNN focuses only on a common feature without assigning special weight to disease-specific details. They performs 95% classification accuracy
Qiao et al	Cattle body condition	Video dataset	the first part is the CNN-based feature extractor that extracts features from the input video, and the second part is the LSTM-based classifier that classifies	CNN focuses only on a common feature without assigning a special weight to disease-specific details and Collecting real time video dataset for beef cattle is much more difficult. They only focuses on beef cattle. They identify individual cattle body condition from video dataset. They achieved an accuracy of 88% and 91% for 15 frame video and 20 frame video length, respectively.

To fill the gaps in the above reviewed literature we decided to increase the number of occurred cattle diseases up to five diseases and increase the dataset with the best augmentation techniques and classification that can assign special weight to disease-specific details by using deep convolutional neural network techniques to fulfill essential requirements in classifying cattle disease.

2.8. Summary

We have reviewed different literature on the classification and detection of cattle diseases based on disease type. Theoretical backgrounds in training data processing techniques including data augmentation, Synthetic data generation, Collecting more diverse data, and Careful annotation techniques. Furthermore, we have given a detailed definition and analysis of the building blocks of CNN architectures (convolutional layers, activation layers, pooling layers, and dropout layers). Moreover, the characteristics of the layer and the operations of some layers were explained thoroughly. Finally, we have also reviewed and presented different deep CNN architectures with outstanding results in image recognition as an example. Additionally, we have discussed some information on the attention mechanism in image processing that is used to focus on specific parts of an image for targeting an affected area from the whole image, and Transfer learning is a technique where a pre-trained model on a large dataset is fine-tuned on a smaller dataset are discussed.

In this thesis, deep convolutional neural network with attention mechanism module was used for cattle disease identification. Because, as we have mentioned above using the attention mechanism able the model to focus on the target cattle disease spot instead of the whole image and image background. By selectively focusing on the most important regions of the image, attention mechanisms can prevent overfitting and generalize better to unseen data. Similarly, Attention mechanisms produce a visual representation of what regions of the image the model is focusing on, making it easier to interpret the model's decisions, and attention mechanisms can reduce the computational cost of training and inference, leading to improved performance and faster training times.

CHAPTER THREE

RESEARCH METHODOLOGY

3.1. Introduction

The systematic procedure followed in conducting a given research project is methodology. The success of the study under examination is based on the selection of the proper techniques. It includes variables that can be measured, calculated, and compared. This chapter focuses on the description of approaches and methods of data collection that are used. To accomplish this thesis, including methods to implement the model, data collection, data preparation, what software we use, and the hardware configuration of the system used. Finally, the model evaluation methods. In addition, the materials and methods we have to use for achieving the general and specific objectives of the thesis will be discussed here.

3.2. Overview of Proposed Model Architecture

The proposed cattle disease recognition model development was planned to apply different image processing stages, starting from the cattle disease image acquisition up to the disease type identification (image acquisition, preprocessing, feature extraction, and classification). The overview of the proposed deep CNN model was depicted in Figure 3.1. It had two phases; the first phase was concerned with training parts. The first step in this phase was image data collection, followed by splitting the data into training data and testing data. The collected images were then preprocessed by applying different transformation techniques and data augmentation. The second step was the testing phase, where we had to prepare the data for testing and then test the data using the developed model to assess the accuracy of the model in classifying cattle disease.

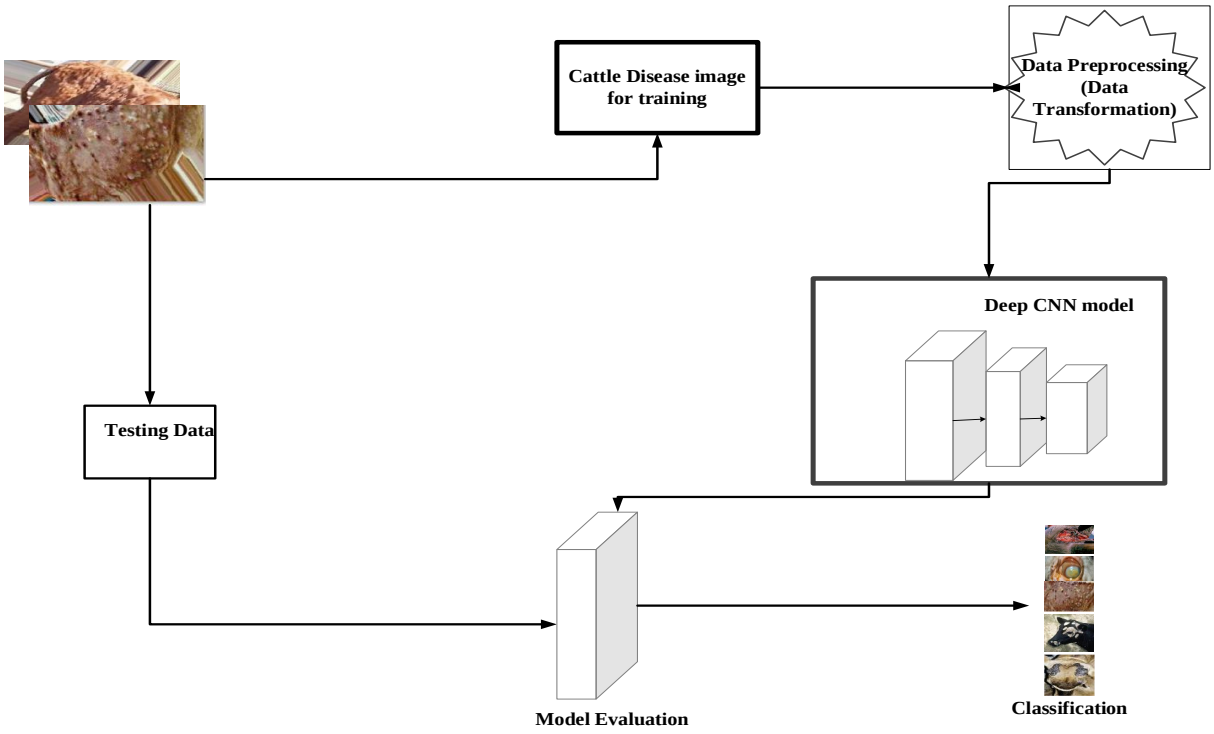


Figure 3. 1:- Overview diagram for proposed DCNN architecture

3.3. Data Collection

The biggest challenge of any research was collecting data. In this thesis, we have used two combined dataset, the first image was taken using a Samsung A32 model smartphone with 128GB memory size and a 480x854 resolution by jpg image format. This first image was collected from Chiro Woreda Veterinary Clinic, the West Hararghe Zone Veterinary Office, and Haramaya University Veterinary Medicine. The organization's professionals are given excellent assistance while gathering images for the purpose of classifying cattle diseases. The second was from the open cattle disease dataset (Three Classes Dataset on External Cattle Disease by Rony et al., 2021).

There are several possible ways to merge or combine the two datasets:

A. Concatenation

One straightforward approach is to concatenate the images and their corresponding labels from both datasets. This involves simply stacking the images and labels from the local dataset and the Kaggle dataset to create a larger combined dataset (Noreen et al., 2020). This can be done using programming libraries such as Numpy or TensorFlow.

B. Data Augmentation

Another approach is to perform data augmentation techniques on both datasets independently before combining them. Data augmentation involves applying various transformations to the images, such as rotations, flips, and zooms, to increase the diversity of the data (Guo & Gould, 2015). After augmenting both datasets separately, we can combine them to create a larger and more diverse dataset.

C. Domain Adaptation

If there are domain differences between the local dataset and the Kaggle dataset, such as differences in lighting conditions or image quality, domain adaptation techniques can be employed (Anwar et al., 2020; Farahani et al., 2021). These techniques aim to align the two datasets by adjusting the features or representations to make them more similar. This can help reduce the domain gap and improve the generalization of the model.

D. Ensemble Methods

Instead of directly combining the datasets, we can also consider using ensemble methods, where we train separate models on each dataset and then combine their predictions during inference (Guo & Gould, 2015). This can be done by averaging the predictions or using more sophisticated ensemble techniques like stacking or boosting. Ensemble methods can leverage the strengths of each dataset and improve the overall performance of the model (Guo & Gould, 2015).

The choice of the combination method depends on the specific characteristics of the datasets and the desired objectives of the research. It is important to carefully consider factors such as class distributions, domain differences, and the need for data diversity when combining the local dataset and the Kaggle dataset to ensure a balanced and representative combined dataset for training the proposed model. Therefore, we have used data augmentation techniques on both datasets independently before combining them.

The number of data collected and taken from secondary sources are shown in table 3.1 below

Table 3. 1: Dataset and Disease numbers for our research

Disease Types	**External Cattle Disease(Rony et al., 2021)**	**Collected Image**	**Increased Raw Image**
Lumpy disease(LD)	200	25	225

Infectious bovine kerato-conjunctivitis (IBK)	150	90	240
Foot_mouth disease (FMD)	250	0	250
Dermatophiytosis (Ringworm)		200	200
Bovine papillomatosis (Wart)		150	150
Total	600	465	1065

3.3.1. Statistical Analysis of Cattle Disease Dataset

We have used the original 1015 images of affected animals from different locations. We used 225 images infected with Lumpy Disease (LD) disease, 240 images infected with Infectious Bovine Kerato-conjunctivitis (IBK) disease, 200 images infected with dermatophytosis (Ringworm) disease, 250 images infected with Foot_mouth disease and 150 images infected with Bovine Papillomatosis (Warts) disease. Descriptive analysis of our datasets is depicted in the table 3.2 below and the picture below we have highlighted the ratio of five diseases through a pie chart in the figure 3.2 below.

Table 3. 2: Descriptive analysis of datasets

Statistical Analysis	
mean	213
standard error	17.8605711
median	225
standard deviations	39.9374511
sample variance	1595
range	100
minimum	150
maximum	250
sums	1065
Count	5
largest(2)	240
smallest(2)	200

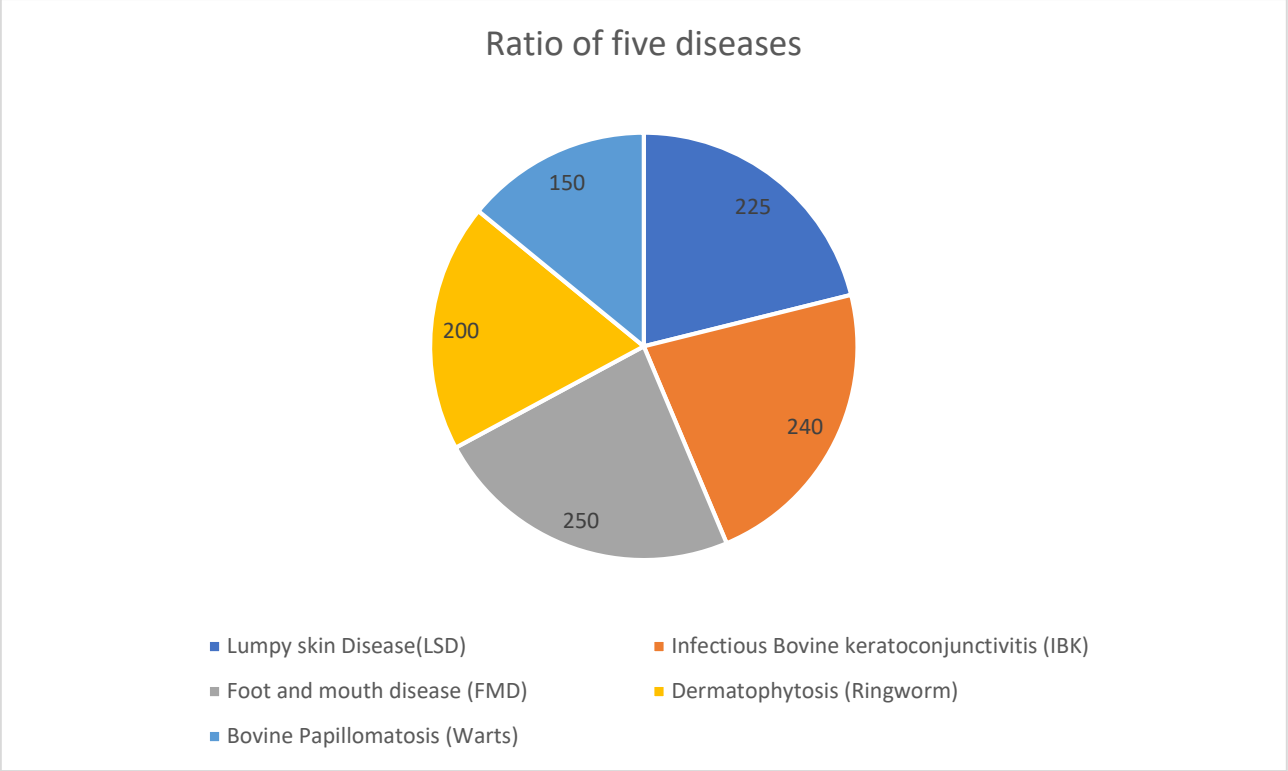


Figure 3. 2:- Highlighted the ratio of five diseases

Additionally, we evaluated the effectiveness of our approach on publicly available skin lesion datasets, namely the International Skin Imaging Collaboration (ISIC) 2019 datasets, The International Symposium on Biomedical Imaging (ISBI) 2016 (Gessert et al., 2020), ISIC 2017 (Codella et al., 2018), ISIC 2018 (Tschandl et al., 2018), and ISIC 2019(Zhao et al., 2021) are four difficult datasets that the International Skin Imaging Collaboration (ISIC) released. ISBI 2016, which had two classes and an image size of 1200, was the first challenge. In the second challenge, ISIC 2017, the number of classes and photos grew to three and 2000 respectively. Seven classifications of skin lesions make up ISIC 2018, which has 10000 photos. Nevus (NV), Basal Cell Carcinoma (BCC), Actinic Keratosis Intra Epithelial Carcinoma (AKIEC), Benign Keratosis (BKL), Dermatofibroma (DF), and Vascular lesion (VASC) are among these classifications. ISIC 2019 is the most recent demanding dataset, and it contains 25,331 photos grouped into eight classes (the seven classes that were specified in ISIC 2018 with an additional new class called Squamous Cell Carcinoma (SCC))

This dataset was facilitate image classification for 8 skin diseases size 25,356 images belonging to 8 classes. Images are classified into labeled folders. Those images were originally part of the International Skin Imaging Collaboration (ISIC)[1] skin disease dataset.

Table 3. 3: ISIC skin disease dataset.

Skin disease name	**Size of images**
Actinic keratosis	867
Basal cell carcinoma	3323
Benign keratosis	2624
Dermatofibroma	239
Melanocytic nevus	12900
Melanoma	4522
Squamous cell carcinoma	628
Vascular lesions	253

3.4. Preprocessing

Image preprocessing was an important step in the proposed model to improve the performance of the Deep CNN with the CCSAM attention module for cattle disease identification. This step helps to enhance the quality of the input images, making them suitable for training the model. It involves several operations like resizing, normalization, and data augmentation.

Resizing the input images to a standard size was important to ensure that the model receives inputs of a consistent size (Fadnavis, 2014). This facilitates and improves the training process. Normalization helps to standardize the pixel values and makes the training data more consistent. To prepare the images for training and testing in the proposed deep CNN model, it is important to perform certain preprocessing steps. The first step in this process is to resize the images to a uniform size (224 x 224) so that all the images have the same height and width. This is important because the model is designed to take inputs of a specific size.

Another important preprocessing step is to normalize the pixel values of the images. This means converting the pixel values to the range [0, 255]. It helps to standardize the input data and can

[1] ISIC skin disease image dataset labelled. These images were originally part of the ISIC skin disease dataset. https://www.kaggle.com/datasets/riyaelizashaju/isic-skin-disease-image-dataset-labelled

improve the performance of the model. Also Gaussian noise removal is applied using spatial filtering techniques that can remove Gaussian noise. Additionally, these preprocessing steps are crucial for improving the accuracy and performance of the proposed model for cattle disease identification.

3.5. Data Augmentation

Data augmentation is a technique that generates additional training samples by randomly transforming the original images (Shorten & Khoshgoftaar, 2019a). It helps to increase the size and diversity of the training data, making the model more robust and capable of recognizing cattle diseases under different conditions we need a large data collection for deep learning. As a result, we employ data augmentation. Data augmentation is the practice of creating new data from an existing training sample in order to increase the number of training data points in a dataset. It's critical to expand the amount of data points. It's also used to extract complicated features from data while avoiding the problem of overfitting. The solution to the problem of limited data is data augmentation, which is a data-space solution. The phrase "data augmentation" refers to a variety of techniques for enhancing the amount and quality of training datasets so that more powerful deep learning models may be created from them. Various data augmentation techniques were used on the original photos in this thesis to obtain additional images for our data set. There are a lot of augmentation techniques like zooming, cropping, flipping, rotation, color change, noise injection, saturation, altering brightness and contrast, shifting, and others.

By creating additional and diverse instances to train datasets, data augmentation is beneficial to enhance the performance and results of deep learning models. Data augmentation can be done both before and after the data is fed into our model (Image-Data Generator augmentation) and during the training process. Using Keras libraries, data augmentation is conducted during network training in this thesis. The original image generates each image that was delivered into the network during the training. This can be accomplished in Keras by defining a set of random transformations to be applied to the images read by the Image-Data-Generator instance. Additionally, we have used an augmenter from imgaug library.

In this thesis, augmentation techniques were divided into two groups depending on their transformation forming. Therefore, we have geometrical augmentation and photometric

augment techniques for evaluating the effectiveness of augmentation techniques which of these was more effective for training Deep CNN models.

Geometrical augmentation: - Geometrical augmentation techniques are contains rotation, shearing, Zooming, flipping, and shifting. These techniques are known as geometry/position augmentation (Shorten & Khoshgoftaar, 2019a; Wang & Perez, 2017). Because they augment images by changing the positions of the original images.

Photometric augmentations: - Photometric augmentation techniques are contains image blurring, noise injection, color change, altering brightness, contrast, and also saturation. These techniques of augmentation are known as photometric/color augmentation (Shorten & Khoshgoftaar, 2019a; Wang & Perez, 2017). Because they augment images without changing the positions of the original images they apply only illumination and color to the original to generate new images.

3.6. Development Tools

3.6.1. Hardware Tools

Some hardware tools are listed below for different functions in the research processes.

Table 3. 4: Hardware tools

Device Name	Purpose
RAM	To store large dataset
Flash Disk	To transfer data
Hard Disk	To increase the computation and to fasten the training
GPU	To accelerate the computation and training process

3.6.2. Software Tools

The study process for Coding, labeling, displaying outcomes, and recording reporting was due by using software tools, programming tools, and libraries we wanted to use listed below as follows:

Anaconda is a Python programming language distribution for scientific computing (data science and machine learning applications). In order to simplify packaging and maintenance, It

offers a navigator program to display many settings, including Jupiter Notebook, Spider, VS Code, and environment-installed modules.

Jupyter Notebook*:* is an open-source web application that helps to load, configure Python API, and write Python code.

TensorFlow*:* TensorFlow is an open-source platform for machine learning and will be used to build a model and deploy it in client environments like other real-world applications.

Keras*:* Keras is one of the deep learning frameworks that run on top of TensorFlow that is powerful and easy to use. It is preferable due to its user-friendly, modular, and extensible work on CPU and GPU though it depends on the task to execute.

OpenCV*:* **OpenCV** (*Open-Source Computer Vision Library*) is a library mainly aimed to provide shared infrastructure for image processing and computer vision applications.

3.7. Evaluation Metrics

Every project has to evaluate machine learning models or algorithms. To test a model, there are different evaluation measures are available. These consist of confusion matrix, logarithmic loss, classification accuracy, and other metrics. Classification accuracy is the ratio of the number of accurate predictions to the total number of input samples. The method used by logarithmic loss, often known as log loss, is to penalize erroneous classifications. The full performance of the model is described by a confusion matrix, which provides us with a matrix as output. There are additional evaluation measures available that are not mentioned here. It is possible to test a model or algorithm using a combination of these different assessment metrics.

Based on the counts of test records that the classification model correctly and incorrectly predicted, the performance of the model is assessed. In addition to showing how well a predictive model performs, the confusion matrix also shows which classes are correctly and mistakenly predicted as well as the kind of errors that are being made. Measurements of Recall (also known as Sensitivity), Precision, Specificity, Accuracy, and, most importantly, the AUC-ROC Curve may all be made using the confusion matrix.

In this thesis, we have used the mean aggregate value of the accuracy Acc, precision Pr, recall Rec, and F1 score to calculate the classification performance.

$$Pr_j = \frac{C_{ii}}{\sum_i C_{ji}} \ , Rec_j = \frac{C_{jj}}{\sum_i C_{ji}} \text{ and } F1_j = \frac{2\ x\ Pr_j XRec_j}{Pr_j + Rec_j}$$

Where C is class, i and j denoted the index of each class value, Cij denoted the number of i_{th} class but predicted in j_{th} class and vice versa, C_{ii} and C_{jj} denote correctly predicted as i_{th} and j_{th} class respectively, the mean values of F1, Rec, and Pr calculated to obtain the aggregate mean values

$mean_{Pr} = \frac{1}{n}\sum_{n=1}^{n} Pr$,$mean_{Rec} = \frac{1}{n}\sum_{n=1}^{n} Rec$ and $mean_{F1} = \frac{1}{n}\sum_{n=1}^{n} F1$. Where n denoted the numbers of categories.

The precision metric measures the proportion of accurate positive predictions to all positive predictions produced. It was used to measure how many correct positive predictions are made. (Saito & Rehmsmeier, 2015).

A recall is a metric that measures the proportion of accurate positive predictions among all possible positive predictions. Recall shows the missed positive predictions, as opposed to precision, which only discusses the right positive predictions among all positive predictions. In this way, recall gives an idea of the coverage of the positive class.(Saito & Rehmsmeier, 2015).

The model's performance was a summed up using the F-Score, which combines the results of precision and recall into a single score that accounts for both features. (Saito & Rehmsmeier, 2015).

Accuracy shows the classification problem's correct prediction value and calculates as the total numbers of the model's correct prediction divided by all number of data instances used for the model (Saito & Rehmsmeier, 2015).

CHAPTER FOUR

PROPOSED SOLUTION

4.1. Chapter Overview

In this chapter, presents the proposed solution architecture with mathematical expressions and its necessary diagrammatic representation. Introduced the proposed model architecture, Data augmentation for the proposed model, Attention mechanism for the proposed model, Model selection, and model developing, explains the model learning function, and also the process training model will be discussed.

4.2. Data Augmentation for Proposed Model

For cattle disease identification, data augmentation was used to generate a variety of different images of cattle with the different degrees of transformation to increase the diversity of the training data. We have divided augmentation techniques into two group's geometric and photometric groups. Two groups of augmentation techniques were used to evaluate the effectiveness of augmentation techniques, and to determine which of these groups were more effective for training Deep CNN models. These augmented images were then used to train the ResNet18 with the CCSAM attention module, leading to improved performance on the test data. It was important to apply data augmentation carefully, as applying too many transformations can lead to overfitting.

Table 4. 1: Geometrical Transformation method

Classes of Disease	**Raw Data**	**Geometrical augmentation**
Lumpy Disease(LD)	225	width_shift, height_shift, rotation, shear, zoom, horizontal_flip,
Infectious Bovine kerato-conjunctivitis (IBK)	240	
Foot_mouth disease (FMD)	250	
Dermatophytosis (Ringworm)	200	
Bovine Papillomatosis (Warts)	150	
Total	1065	

Table 4. 2: Photometric Transformation method

Classes of Disease	Raw Data	Photometric augmentation
Lumpy Disease(LD)	225	Blur, contrast, color change, noise injection, and brightness.
Infectious Bovine kerato-conjunctivitis (IBK)	240	
Foot_mouth disease (FMD)	250	
Dermatophytosis (Ringworm)	200	
Bovine Papillomatosis (Warts)	150	
Total	1065	

4.3. Attention Mechanism for Proposed Model

A. Channel attention module

Channel attention in our proposed models are operated as follows:

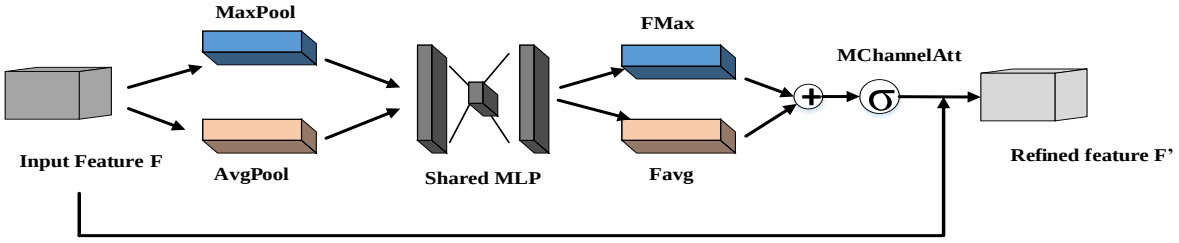


Figure 4. 1:- Channel attention diagram

Mathematical description of our channel attention modules that we have used in this thesis is defined as the following equations:

$$M_{ChannelAtt(F)} = \sigma\left[MLP\left(AvgPool(F)\right) + MLP\left(MaxPool(F)\right)\right]$$

$$= \sigma\left[\left(W_1\left(W_0\left(F_{AvgPool}^{ChannelAtt}\right)\right)\right) + \left(W_1\left(W_0\left(F_{MaxPool}^{ChannelAtt}\right)\right)\right)\right] \qquad (1)$$

$$W_0 \in R^{\frac{C}{r}*C}, and\ W_1 \in R^{C*\frac{C}{r}}$$

Where ∈ denoted element-wise multiplication, σ denoted sigmoid operation, r represented the reduction rate, M_C ($M_{channelAtt}$ (F)) denoted weight coefficient of channel attention, Multi-Layer

Perceptron (MLP) is represents shared two-layer neural network, C/ r represented number of neurons in the first layers of MLP, C was represented the number of neurons in the second layer of the MLP and where W0 needs to be activated by ReLu. W_0 was the first layer of the Multi-Layer Perceptron (MLP), and W_1 was the second layer of the MLP. F and F' on the diagram were denoted the original feature map and the new feature map respectively.

As shown in Figure 4.1, Channel attention diagram. The first step is to input the feature graph F of the height (H) x weight (W) x channel, after which the global maximum pooling and global average pooling are carried out, respectively. The results of the calculations are then processed through the shared two-layer neural network MLP (which has dimensions of 1 x 1 x N (number of channels) and has C/r neurons in the first layer with ReLu as its activation function and C neurons in the second layer with F_{Max} and F_{Avg} as its maximum and average pooling characteristics). In the following step, the two features are combined, and the weight coefficient Mc is then calculated using the sigmoid activation function. The new feature F' is then entered into the subsequent spatial attention module by multiplying the weight coefficient by the original feature F and applying various weights to the feature pictures of each channel.

B. Spatial attention module

Spatial attention in our proposed models is operated as follows:

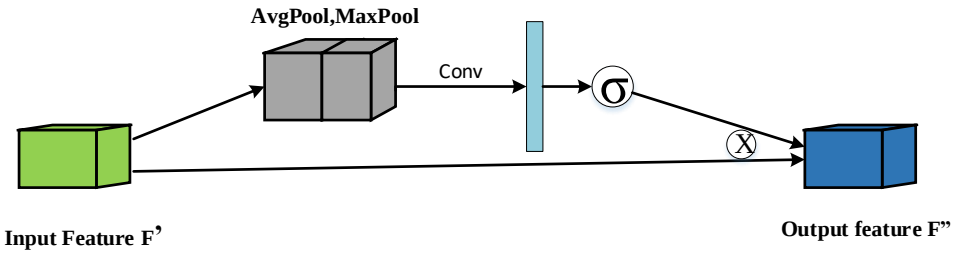


Figure 4. 2 :- Diagram of spatial attention operation process.

The mathematical description of our spatial attention modules that we have used in this thesis is defined as the following equations:

$$M_{spatial}(F') = \sigma[f^{7x7}([AvgPool(F'); MaxPool(F')])]$$

$$= \sigma\left(f^{7x7}\left(\left[F^S_{Avg}; F^S_{Max}\right]\right)\right) \quad (2)$$

Where σ is a sigmoid operation, **F'** represented the feature map refined by the channel attention module, and the convolution kernel of 7 × 7 sizes is selected for the spatial attention feature map to further encode the locations that need to be activated or suppressed in the spatial dimension.

As shown in Figure 4.2: representation diagram of spatial attention operation process. The feature map **F**' with different channel weights output by the channel attention module is taken as the input feature graph of this module. Firstly, the channel dimension is based on global maximum pooling and global average pooling, and then the two results are combined array operation. After the convolution procedure, the number of dimensions is decreased to one channel, and sigmoid-generated spatial attention characteristics are produced. Lastly, the newly produced features are multiplied by the input features of the module, and the feature graph **F**" with different spatial weights is the output.

C. Combination of Channel and Spatial Attention Mechanism (CCSAM)

The combination of channel and spatial attention mechanism (CCSAM) is an Attention mechanism Module that combines the channel Attention Module and spatial Attention Module. The calculation efficiency is improved while reducing the number of parameters, which is simple and effective. It is included into the current network together with the residual convolution unit to successfully change the feature graph's weight and produce more useful detail information. Figure 4.3, shows the structural representation diagram of CCSAM.

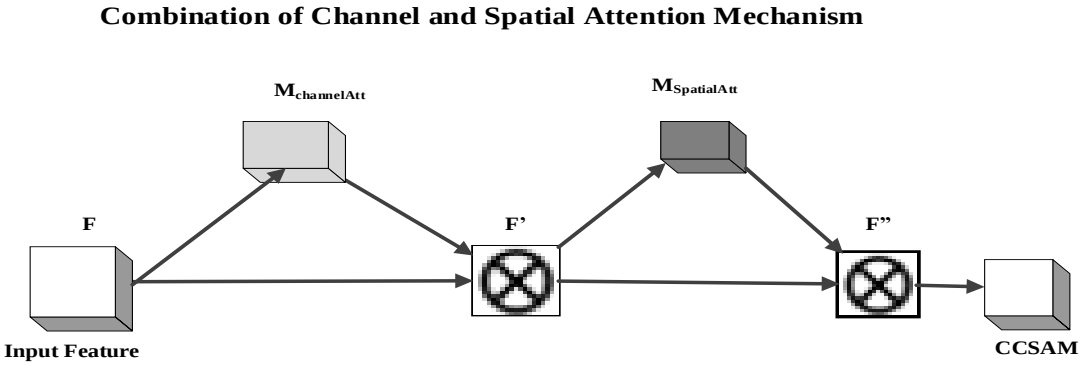


Figure 4. 3:- Proposed Combination of Channel and Spatial Attention Mechanism

The mathematical description of Figure 4.3, is the multiplications of equations (1) and (2) within the input channel defined in the following equations:

$$CCSAM(x) = F' * \text{F"} \qquad (3)$$

$$Where, M_{ChannelAtt}(F) * F = F' \; and \; M_{spatial}(F') * F' = F''$$

Where the input feature map is F, F' represents the multiplication of an input feature map with the weight coefficient of channel attention, and also F" represented the multiplication of channel attention module refined feature F' with the weight coefficient of spatial attention.

The combination of channel and spatial attention mechanisms (CCSAM) was used in this paper to focus on the target area from the cattle disease image. Because the Attention mechanism allows the model to focus on the most relevant features and ignore irrelevant information. The attention mechanism allows the model to adapt to different image conditions, such as varying lighting, rotation, and orientation (J. Li et al., 2020). This makes it more robust to different real-world scenarios and less sensitive to input variations(Z. Niu et al., 2021). The attention mechanism provides insight into what the model is focusing on and why. This makes it easier to understand the model's decision-making process and improve it, if necessary. Attention mechanism can be implemented in a computationally efficient manner, as it only requires a small number of additional operations compared to traditional deep learning models(J. Li et al., 2020)(Z. Niu et al., 2021).

4.4. Model Selection

In this thesis, the most suitable deep learning model architecture is selected based on the problem requirements and the nature of the data. In order to prevent overfitting, we choose the network structure of Resent 18 as the baseline model. Because Resent 18 has the small number of parameters and fast calculation(Mahmud et al., 2021). ResNet18 is a convolutional neural network architecture designed for image classification tasks. It was introduced by Microsoft Research in 2015 and has 18 layers, including convolutional layers, batch normalization layers, and fully connected layers(He et al., 2016b).

ResNet18, a variant of the Residual Network architecture, offers several advantages over other models like VGG, AlexNet, and LeNet for cattle disease identification. ResNet18 has a deep architecture with shortcut connections that mitigate the vanishing gradient problem and enable effective training of deep networks (Mahmud et al., 2021; Subramanian et al., 2022). The skip connections allow the network to learn residual mappings, making it easier to optimize and improve the flow of gradients during training. This enables ResNet18 to effectively capture

intricate patterns and hierarchical representations, enhancing its performance in complex tasks like cattle disease identification. ResNet18 achieves good performance with relatively fewer parameters compared to other deeper architectures like VGG (Aremu & Nandakumar, 2023). The skip connections allow for the reuse of features from previous layers, reducing the redundancy of parameters and improving parameter efficiency. This makes ResNet18 more computationally efficient and easier to train, especially when working with limited computational resources or smaller datasets.

ResNet18 has been pre-trained on large-scale datasets like ImageNet, making it an excellent candidate for transfer learning. The pre-trained ResNet18 model can be used as a starting point for cattle disease identification tasks, allowing researchers to benefit from the learned features and representations from a diverse range of images. This significantly speeds up the training process and improves model performance, particularly when limited labeled cattle disease images are available. ResNet18 has proven to be highly effective at feature extraction, capturing both low-level and high-level features important for image recognition tasks. The residual blocks in ResNet18 enable the model to learn and extract intricate patterns and disease-specific features from cattle disease images. This leads to better discrimination and recognition of different cattle diseases, enhancing the overall performance of the model.

ResNet18 exhibits robustness to variations in input images, such as changes in scale, rotation, and viewpoint. The deep and hierarchical nature of the architecture allows ResNet18 to learn invariant representations that are resilient to such variations. This robustness is particularly important in cattle disease identification, where images may exhibit variations in terms of perspective, lighting conditions, and image quality.

Generally, ResNet18's advantages include its deep architecture with shortcut connections, parameter efficiency, transfer learning capabilities, strong feature extraction abilities, and robustness to variations. These factors contribute to its effectiveness in cattle disease identification tasks, making it a preferred choice over other models like VGG, AlexNet, LeNet, and others.

The detailed architecture of ResNet18 is as follows:

Input: The input image is of size 224x224x3 (RGB image).

Convolutional layer: The first layer is a convolutional layer with 64 filters of size 7x7 with a stride of 2 and padding of 3, but other convolutional layers filtered with kernel size 3x3 with a stride 1 and padding 1 and down sampling convolutional layers filtered with kernel size 1x1 with a stride 2 and padding 1.

Batch normalization: The output of the convolutional layer is passed through a batch normalization layer.

ReLU activation: The output of the batch normalization layer is passed through a ReLU activation function.

Residual blocks: ResNet18 has four layers, each consisting of two residual blocks which contain two convolutional layers and one shortcut connection. The first layer has 64 filters, and the rest have 128, 256, and 512 filters, respectively.

Average pooling: After the residual blocks, the output is passed through a global average pooling layer, which averages the values in each feature map.

Fully connected layer: The output of the global average pooling layer is flattened and passed through a fully connected layer with 1000 neurons, corresponding to the number of classes in the ImageNet dataset.

Softmax activation: The output of the fully connected layer is passed through a softmax activation function, which converts the output into a probability distribution over the classes.

Overall, ResNet18 uses residual connections to enable the training of much deeper neural networks while maintaining good accuracy. The shortcut connections in the residual blocks allow the gradients to flow directly from the output to the input, which helps in training deep neural networks without the vanishing gradient problem.

Detailed ResNet18 parameters are depicted in table below:

Table 4. 3: Resnet18 structural details

Layers			Input	Output	Kernel size		Stride	Padding	Parameters
Conv1			3	64	7	7	2	3	9472
Maxpool			64	64	3	3	2	0.5	0
Layer1	Basic	Conv2-1	64	64	3	3	1	1	36928
	Block 1	Conv2-2	64	64	3	3	1	1	36928

	Basic	Conv2-3	64	64	3	3	1	1	36928
	Block 2	Conv2-4	64	64	3	3	1	1	36928
Layer2	Basic	Conv2-1	64	128	3	3	2	1	73856
	Block 1	Conv2-2	128	128	3	3	1	1	147584
	Basic	Conv2-3	128	128	3	3	1	1	147584
	Block 2	Conv2-4	128	128	3	3	1	1	147584
Layer3	Basic	Conv2-1	128	256	3	3	2	1	295168
	Block 1	Conv2-2	256	256	3	3	1	1	590080
	Basic	Conv2-3	256	256	3	3	1	1	590080
	Block 2	Conv2-4	256	256	3	3	1	1	590080
Layer4	Basic	Conv2-1	256	512	3	3	2	1	1180160
	Block 1	Conv2-2	512	512	3	3	1	1	2359808
	Basic	Conv2-3	512	512	3	3	1	1	2359808
	Block 2	Conv2-4	512	512	3	3	1	1	2359808
FC			512	1000					513000
Total parameter size									11511784

Once the model architecture was selected, the next step was to fine-tune the pre-trained model for the cattle disease recognition task. This was done by updating the model's parameters with the training data to learn the specific patterns associated with the target diseases. The model's performance was evaluated during the training process using a validation set, and the parameters were adjusted accordingly to optimize the model's performance. The training process continued until the model converged, which was typically determined by observing the improvement in performance on the validation set.

4.5. Proposed model Architecture

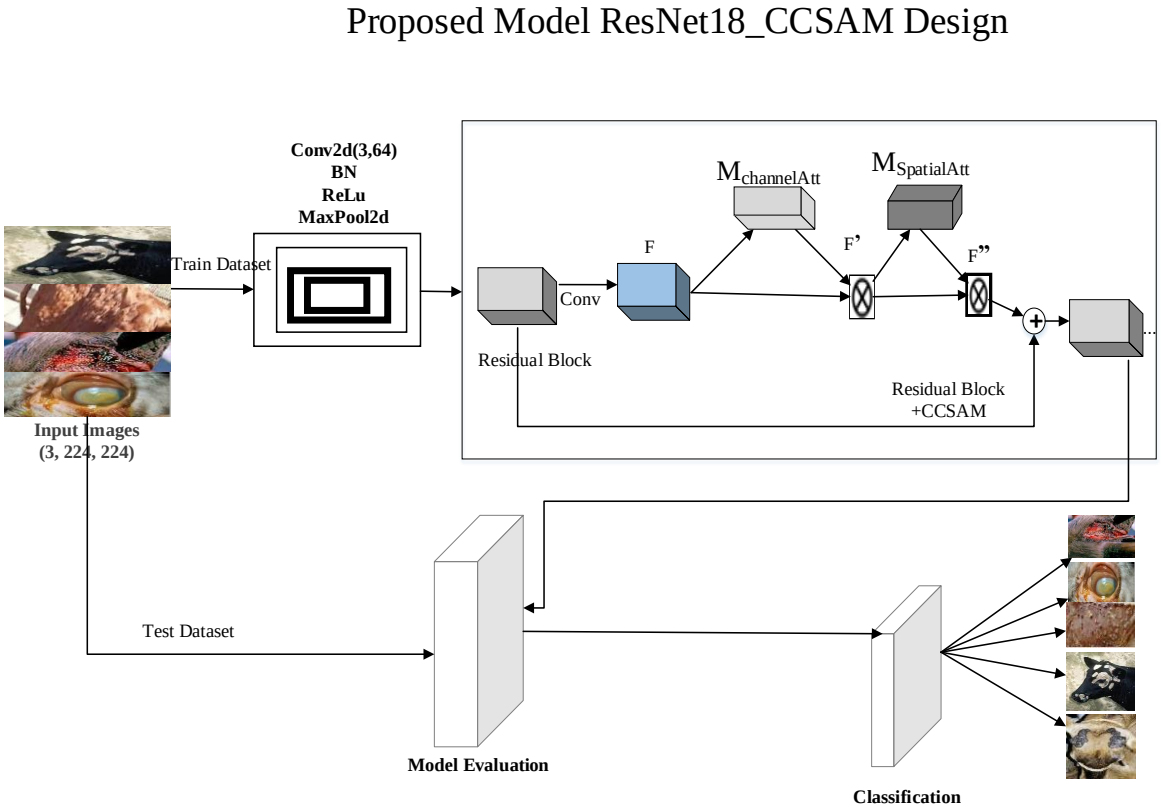


Figure 4. 4:- Proposed model Resnet18_CCSAM architecture

In the above Figure 4.4, Conv2d represented convolutional layers, BN represents Batch Normalization, Relu represented rectified unit activation layers, MaxPool represented max pooling layers, CCSAM represented combination of channel attention and spatial attention mechanism, F represents input feature map, F' represents multiplication of input feature map with weight coefficient of channel attention and also F" represented represents multiplication of channel attention module refined feature F' with weight coefficient of spatial attention.

4.5.1. Detail Description of Proposed Model

In this paper, we have used ResNet18 with a Combination of Channel and Spatial Attention Mechanisms (CCSAM). We select ResNet18's network structure as the framework to avoid overfitting. Due to Resnet18's few parameters and quick computation. The proposed CCSAM, was added to the Basic Block of the Residual network, more comprehensive picture information is learnt and image recognition mistakes are decreased by improving the representation of the feature map from the spatial dimension, and channel dimension of the feature map.

ResNet-18 consists of eight residual building components and a convolutional layer. ResNet-18 is made up of 17 convolutional layers, a fully connected layer, and a max-pooling layer with a different filter size. Traditional ResNet-18 models include 11.6 million parameters and utilize batch normalization (BN) and ReLU activation functions to completely cover the back of the convolutional layers in the "basic block." (Sarwinda et al., 2021). The ResNet-18 network's basic architecture can be thought of as a residual building component. The structure of the residual building block (He et al., 2016a) is shown in Figure 4.5. The convolutional layer is skipped using a sort of shortcut. Direct addition of the input vector and the vector produced by the convolutional layer can produce the output vector, which is then produced by the rectified linear unit (ReLU) activation function.

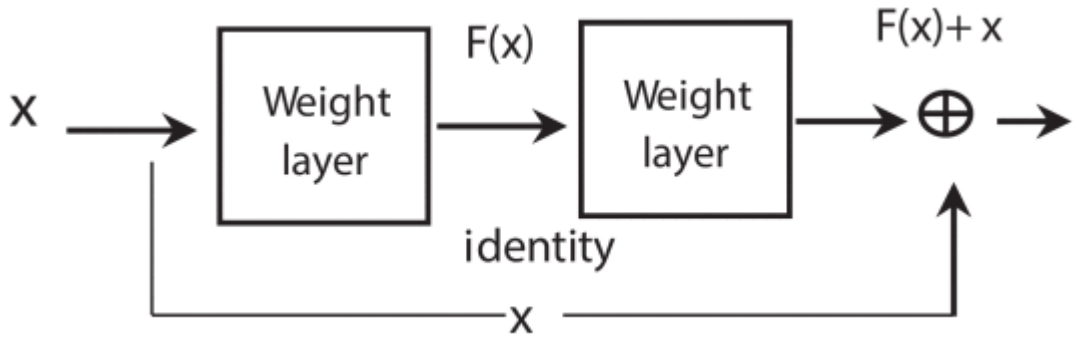


Figure 4. 5 :- Basic block of Residual network

For Example, the learning idea of the residual unit is F(x) = H(x) - x, so that the original learning feature may be derived as F(x) + x. The feature learned from the residual unit with a short connection is denoted as H(x) when the input is x. Only the identity mapping is created to preserve stable network performance when the residual is 0, as residual learning is simpler than direct learning of the original features. Since the actual residual will not be zero, H(x) can be viewed as the sum of the identity mapping and residual mapping F(x) and thus allows the residual unit to learn new features based on the input features for improved performance.

How does CCSAM work in the proposed model?

The CCSAM module has two main components a channel attention mechanism and a spatial attention mechanism. The channel attention mechanism assigns a weight to each channel of the feature map based on the importance of each channel, while the spatial attention mechanism assigns a weight to each spatial location of the feature map based on its importance.

Both Channel Attention and Spatial Attention are components of the CCSAM module. Attention was distributed among channels using the Channel Attention class. As arguments, it accepts two numbers of input channels and the reduction factor. The generation of channel-wise feature representations was accomplished via adaptive average pooling and adaptive max pooling. The dimensionality of the feature representations was decreased by applying two convolutional layers. A sigmoid activation function and element-wise summing are both applied to the output of the two convolutions. The input feature map was element-wise multiplied with the output channel attention map to produce the final product.

Attention was distributed across spatial dimensions using the Spatial Attention class. Arguments are not required. It figures out the mean and maximum activations throughout the input feature map's channels. Along the channel dimension, the average and maximum activations are added together. A convolutional layer and sigmoid activation function are applied to the concatenated tensor. The input feature map is element-wise multiplied by the produced spatial attention map. The Channel Attention and Spatial Attention processes are combined in the CCSAM module. The Basic Block module, a fundamental piece of the ResNet architecture, is used in the ResNet-18 model with CCSAM. Two convolutional layers with batch normalization and ReLU activation, followed by a CCSAM module, make up each Basic Block. The output of the CCSAM module is combined with the input feature map.

The ResNet-18 model is built in four phases, with a specific amount of Basic Blocks in each stage. The final step involves using an adaptive average pooling layer to shrink the spatial dimensions to 1x1. A new completely connected layer with 512 output features is used to replace the existing fully connected layer. To further process the features, a dense layer, a dropout layer, and a third fully linked layer are added. Finally, the predicted class probabilities are obtained by using a softmax activation. For a specific classification task with 5 classes.

Generally, the output of the CCSAM module was obtained by multiplying the feature map by the channel attention, and spatial attention weights could be inserted into the ResNet18 model after every residual block, which typically included two or more convolutional layers. By adding the CCSAM module to the ResNet18 model, we could improve its performance by selectively attending to important parts of the feature map.

4.6. Transfer Learning for Proposed Model

Transfer learning is a technique that involves using a pre-trained deep learning model as the starting point for a new task. In the context of cattle disease identification, transfer learning can be used to leverage the knowledge learned from pre-training the model on a large and diverse dataset, such as ImageNet, to improve the performance of the model on the cattle disease identification task.

The ResNet18 architecture, which was proposed to be used in this study, had been pre-trained on ImageNet, a large dataset of natural images. The pre-trained model could be fine-tuned for cattle disease identification by updating the weights of the model with the new task-specific data. The benefit of transfer learning was that the model had already learned a good representation of the image data from the pre-training phase, allowing the model to converge faster on the cattle disease identification task. Additionally, transfer learning could help to overcome the issue of insufficient training data, as the model had already learned from a large and diverse dataset. The fine-tuned ResNet18 with the CCSAM attention module was then used for cattle disease identification, leading to improved performance compared to training the model by freezing layers.

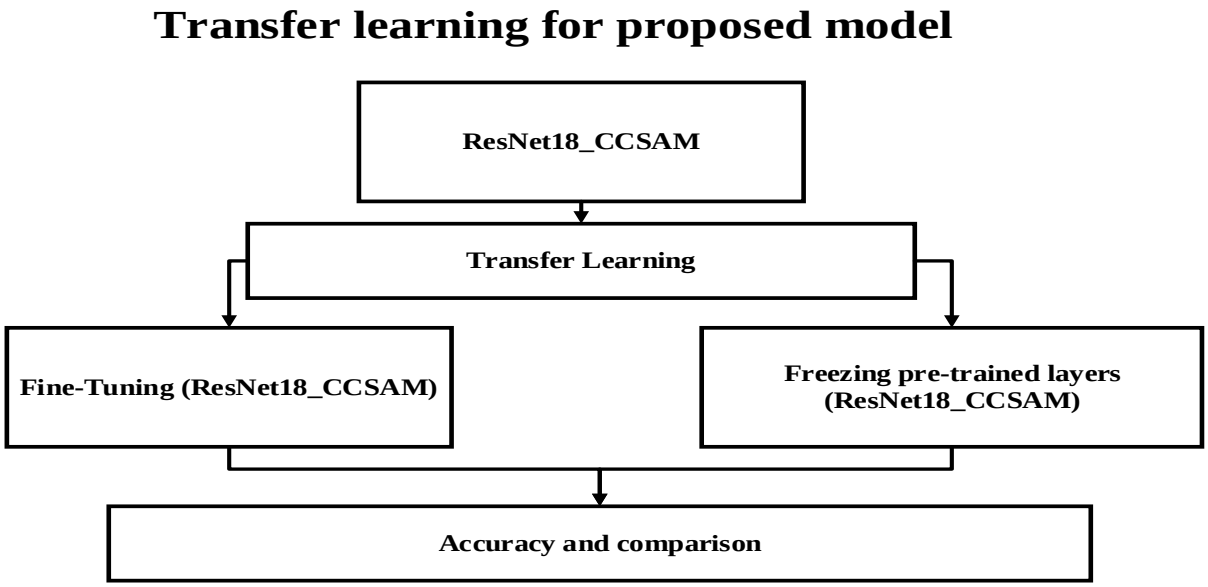


Figure 4. 6 :- Transfer learning for proposed model

4.6.1 Fine-Tuning Transfer learning for proposed model

The Fine Tuning technique was flexible, as it involved using a model with the same design as a pre-trained model that had demonstrated outstanding results in performing specific tasks similar to the one we were attempting to achieve and learn from scratch. We deleted the weights of every layer from the pre-trained model and retrained the entire model based on our own results. In this process, the technique had to be customized according to the specific requirements of our model.

Detail parameters fine-tuned transfer models of proposed models were here:

Table 4. 4: Parameters fine-tuned transfer models of proposed models

Layers			Input	Output	Kernel size		Stride	Padding	Parameters
Conv1			3	64	7	7	2	3	9472
Maxpool			64	64	3	3	2	0.5	0
Layer1	**Basic**	Conv2-1	64	64	3	3	1	1	36928
	Block 1	Conv2-2	64	64	3	3	1	1	36928
CCSAM (64)									679
	Basic	Conv2-3	64	64	3	3	1	1	36928
	Block 2	Conv2-4	64	64	3	3	1	1	36928
CCSAM (64)									679
Layer2	**Basic**	Conv2-1	64	128	3	3	2	1	73856
	Block 1	Conv2-2	128	128	3	3	1	1	147584
CCSAM (128)									2283
	Basic	Conv2-3	128	128	3	3	1	1	147584
	Block 2	Conv2-4	128	128	3	3	1	1	147584
CCSAM (128)									2283
Layer3	**Basic**	Conv2-1	128	256	3	3	2	1	295168
	Block 1	Conv2-2	256	256	3	3	1	1	590080
CCSAM (256)									8563
	Basic	Conv2-3	256	256	3	3	1	1	590080
	Block 2	Conv2-4	256	256	3	3	1	1	590080

CCSAM (256)									33411
Layer4	**Basic**	Conv2-1	256	512	3	3	2	1	1180160
	Block 1	Conv2-2	512	512	3	3	1	1	2359808
CCSAM (512)									33411
	Basic	Conv2-3	512	512	3	3	1	1	2359808
	Block 2	Conv2-4	512	512	3	3	1	1	2359808
CCSAM (512)									33411
Average pooling			512	512	7	7	7	0	0
FC			512	512					262656
Dense			512	256					131328
FC2			256	5					1285
Number of trainable parameters:									11483925
Number of non-trainable parameters:									0

In above table 4.4, CCSAM attention modules were added after all basic blocks of Resnet18 to proposed model ResNet18_CCSAM. Resnet18 is pre-trained but every layer's weights from the pre-trained model are retrained(model learn from scratch) and other fully-connected layers, dense layers, and dropout layers are added to classify five cattle diseases.in the above fine-tuned transfer models of proposed models, all parameters 11483925 are trained

4.6.2. Freezing Layers Transfer Learning for Proposed Model

A layer can be frozen to prevent weight changes. When the underlying model (trained on any other dataset) is made frozen and its layers cannot be modified any further, this strategy is occasionally utilized in transfer learning. In the case of images, some are run through the layers without updating weights.

In freezing pre-trained layers of Transfer learning for the proposed model, adding CCSAM attention modules the same as fine-tuned CCSAM attention modules were added after all basic blocks of Resnet18 to the proposed model ResNet18_CCSAM. But only added CCSAM layers and modified fully connected layers are trained. Other layers are frozen because Resnet18 is pre-trained every layer's weights from the pre-trained model are not retrained means the weights

of pre-trained layers cannot be updated. The number of trainable parameters is 483093 and the number of non-trainable parameters is 11000832

4.7. Model Learning Functions

Categorical Cross-entropy loss: Loss/cost functions are used to optimize the model during training when working on a Machine Learning or Deep Learning problem. Almost always, the goal is to minimize the loss function. The better the model, the lower the loss. Categorical cross-entropy is a widely accepted loss function used in classification tasks. It is used in this research for computing the loss of multi-classification actions.

The following sum is computed by the categorical cross-entropy loss function to determine the loss of an example:

$$\boldsymbol{CCe} = -\frac{\boldsymbol{1}}{\boldsymbol{N}} * \sum_{\boldsymbol{i=1}}^{\boldsymbol{N}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{C}} \boldsymbol{Y_{ij}} * \boldsymbol{log}(\boldsymbol{p_{ij}}) \qquad (\mathbf{4})$$

Where CCe represented the categorical cross-entropy loss, N denoted the total number of samples or instances in the dataset, C denoted the number of classes or categories, Y_{ij} was denoted the ground truth label for sample i and class j. It was equal to 1 if the sample belongs to class j, and 0 otherwise, p_{ij} was denoted the predicted probability of sample i belonging to class j. This probability was typically obtained from a softmax activation function applied to the output of the model for each class. The categorical cross-entropy loss function calculates the average cross-entropy loss over all the samples and classes. It penalizes the model more heavily when the predicted probability diverges from the ground truth label.

In the thesis paper, we have used stochastic gradient descent (SGD) optimizer functions. The torch. Optim. SGD function is an optimizer that implements the SGD algorithm, which is a popular stochastic gradient descent optimization algorithm. It takes in the model parameters and hyper-parameters such as learning rate and momentum, and updates the model parameters based on the gradients of the loss function with respect to the parameters. The SGD optimizer allows for efficient training on large-scale datasets. In cattle disease identification, where datasets can contain a vast number of samples, SGD's stochastic nature becomes beneficial. By randomly selecting a subset of samples, or mini-batches, at each iteration, SGD enables efficient exploration of the dataset, making it easier to generalize patterns and identify disease-related

features. This enabled the ResNet18 model with CCSAM to effectively learn and discriminate between different cattle diseases, leading to improved identification accuracy.

4.8. Hyper-parameters

The hyper-parameters function as handles that can be adjusted during model training which would lead to improved accuracy and reduce the risk of over-fitting. There are two kinds of hyper-parameters. The first defines the network structure, including the number of filters, kernel size, and the hidden layer between input and output. The other indicates network training parameters such as the number of epochs, batch size, and learning rate (Lateef & Abbas, 2022). Selecting the best collection of hyper-parameters for a learning algorithm is known as hyper-parameter tuning. A model input called a hyper-parameter has its value predetermined before the learning process even starts. Hyper-parameter tuning is essential for machine learning/ deep earning algorithms to work. And for the essential parameters for the model performance such as epoch, batch size, number of neurons, and batch size has been tuned for the ResNet18-CCSAM model.

4.9. Training Process of Proposed Model

To train a ResNet18 model with a CCSAM attention mechanism, we had to modify the standard ResNet18 architecture to include the CCSAM attention module. The CCSAM module took in the feature maps produced by the ResNet18 model and performed spatial and channel-wise attention on the feature maps to produce the final attention-enhanced feature maps. These attention-enhanced feature maps were then passed through the final dense layer to produce the final predictions.

Here are the general steps to train a ResNet18 with the CCSAM model:

- First, load the dataset that we would use for training the model. This dataset should have contained images of cattle and the corresponding labels indicating which disease the cattle were suffering from.
- Next, we preprocessed the data by performing tasks such as normalizing the pixel values and resizing the images.
- Then, we modified the standard ResNet18 architecture to include the CCSAM attention module. This was done by adding the CCSAM module after each residual block in the ResNet18 architecture.

- After modifying the architecture, we compiled the model by specifying the optimizer, loss function, and metrics that we would be using for training the model.
- Finally, we trained the model by passing the preprocessed data through the modified ResNet18 with CCSAM architecture.

The model was typically trained for a fixed number of epochs, and the training process was monitored by observing the training and validation accuracy. After training the model, we evaluated its performance on test set to obtain an estimate of the model's generalization ability. Once the model had been trained, it could be used to make predictions on new images of cattle and classify them based on the presence of a disease.

CHAPTER FIVE

IMPLEMENTATION OF THE PROPOSED SOLUTION

5.1. Chapter Overview

In this chapter, the implementation of the proposed model architecture was discussed. The working environment, Environment setup, Dataset Preparation Implementation, Proposed Model Implementation, Learning Hyper-parameters, and experiments undertaken would have discussed here.

5.2. Working Environments

This part clarifies the hardware tools that have been used to execute model experiments just as depicting the hardware stack.

❖ Laptop: The Laptop is utilized for fostering a model developing.

- Operating system: Windows 10
- Processor: Intel ® Core™ i5-7200U CPU @ 2.50GHz 2.70GHz
- Graphics: Intel ® Graphics 3000
- Primary Memory (RAM): 8.00 GB
- System Type: x64-based processor, 64-bit operating system

❖ Desktop: The desktop computer is used for training model on the given dataset.

- Operating system: Windows 10
- Processor: Intel ® Core™ i5-4580 CPU @ 3.29GHz x 4
- Graphics: Intel ® HD Graphics 4600
- GPU: GeForce RTX 2070 Super 6 GB RAM
- Primary Memory (RAM): 8.00 GB
- System Type: x64-based processor, 64-bit operating system

5.3. Environment Setup

Anaconda is a software distribution platform that provides an environment for deep learning research processing. It is used by millions of students, developers, and researchers to build deep learning environments in Python[2].

Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations, and narrative text an open-source, tool that integrates software code, eases the level of development and modularity, and computational output[3].

Google Colab is a cloud-based platform that allows you to write and run Python code in your browser. It is a free service that provides access to a virtual machine with pre-installed deep learning frameworks such as TensorFlow and PyTorch[4].

5.4. Dataset Preparation Implementation

To ensure that the model receives inputs of a consistent size, it is important to resize the input images to a standard size (Fadnavis, 2014). Performing certain preprocessing steps is important to prepare the images for training/testing with the proposed deep CNN model. This includes normalization which helps to standardize the pixel values and makes the training data more consistent. This facilitates and improves the training process. In order to ensure that all the images have the same height and width, it is important to resize the images to a uniform size (224 x 224) as the model is developed to take inputs of a specific size. This is the first step in this process.

Converting the pixel values of the images to the range [0, 255] is an important preprocessing step as it helps to standardize the input data and can improve the performance of the model. This process is known as normalizing the pixel values of the images. Additionally, the accuracy and performance of the proposed model for cattle disease identification can be improved by taking certain preprocessing steps.

Steps or Algorithm of data loading for preprocessing data

Algorithm 1:- Steps or Algorithm of data loading for preprocessing data

[2] website link: https://www.anaconda.com/
[3] Website link: https://www.techrepublic.com/article/google-colab-vs-jupyter-notebook/
[4] Website link : https://neptune.ai/blog/how-to-use-google-colab-for-deep-learning-complete-tutorial

Step 1: Import required libraries

Step 2: Define the transforms for training and validation data

Step 3: Define the dataset for training data, validation, and testing using `datasets.ImageFolder` with the root directory

Step 4: Create a data loader for training data, validation, and testing dataset using `DataLoader` with a batch size of 8.

Step 5: Define the class labels as a list of strings

Applying to preprocess on loaded data

Algorithm 2:- Applying preprocess on loaded data

Step 1:Import required libraries

Step 2: Define the `transform_train` object using `transforms. Compose`

Step 3: Add the `transforms. Resize` method to resize the images to 224x224 pixels

Step 4: Add the `transforms.ToTensor` method to convert the image to a tensor

Step 5: Add the `transforms. Normalize` method to normalize the pixel values with a mean and standard deviation of (0.5, 0.5, 0.5)

5.5. Data Augmentation Implementation

Data augmentation is a method that can create more training samples by randomly transforming the original images (Shorten & Khoshgoftaar, 2019a). This technique can help to increase the size and diversity of the training data, which can make the model more robust and able to identify cattle diseases under different conditions. Since the original datasets are too small, data augmentation is used to make the dataset larger. Image augmentation techniques can produce different variations of the images.

Algorithm of geometrical augmentations implementation

Algorithm 3:- Geometrical augmentations implementation

Start:

Import the required modules: os, ImageDataGenerator, load_img, img_to_array from tensorflow.keras.preprocessing.image.

Step 1: Create an instance of the ImageDataGenerator class and assign it to the variable datagen.

Step 2: Set the desired augmentation parameters for datagen:

rotation_range=90/180

shear_range=0.6

zoom_range=0.5

width_shift_range=0.3

height_shift_range=0.2

horizontal_flip=true

fill_mode=nearest.

End.

Some examples of geometric augmentation techniques

Figure 5. 1:- Examples of geometric augmentation techniques

Algorithm for photometric augmentations implementation

Algorithm 4:- Photometric augmentations implementation

Start: Import the required modules: cv2, os, and numpy as np.

Step 1: Define the function add noise that takes an image as input and adds random noise to it using cv2.randn and cv2.add.

Return the resulting image.

Step 2: Define the function alter contrast that takes an image as input and randomly alters the contrast using cv2.convertScaleAbs with random alpha and beta values.

Return the resulting image.

Step 3: Define the function alter brightness that takes an image as input and randomly alters the brightness using cv2.convertScaleAbs with random alpha and beta values.

Return the resulting image.

Step 4: Define the function alter saturation that takes an image as input, converts it to HSV color space using cv2.cvtColor, and randomly alters the saturation of the image by scaling the saturation channel.

Return the resulting image.

Step 5: Define the function blur that takes an image as input and applies Gaussian blur to it using cv2.Gaussian Blur with a randomly chosen kernel size.

Return the resulting image

END

Some examples of photometric augmentation techniques

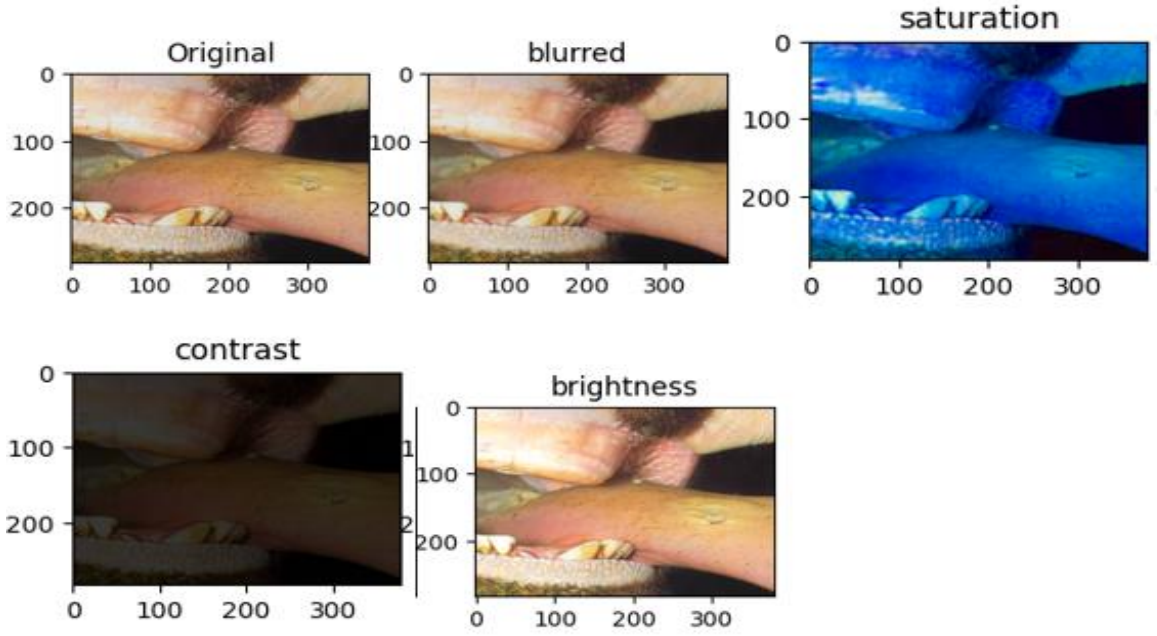


Figure 5. 2:- Examples of photometric augmentation techniques

Generally, the original image used in this thesis is 1065 by augmenting the original image to 10650. Transformations of these augmentations are performed carefully because over-augmenting may lead the model to overfitting and under fitting because of this we have apply each augmentation separately and carefully.

Table 5. 1: Geometric and photometric dataset size

Classes of Disease	Raw Data	Geometric Transformation	Photometric Transformation
Lumpy Skin Disease(LSD)	225	2250	2250
Infectious Bovine kerato-conjunctivitis (IBK)	240	2400	2400
Foot_mouth disease (FMD)	250	2500	2500
Dermatophytosis (Ringworm)	200	2000	2000
Bovine Papillomatosis (Warts)	150	1500	1500
Total	1065	10650	10650

From the total of datasets 70% used for training purpose, 20% for testing purpose and 10% used for validation from training data.

```
!pip install split-folders
import splitfolders
input_folder="/content/drive/MyDrive/DataSet_CD"
output="/content/drive/MyDrive/DataSet_CD/splitted_dataset"
splitfolders.ratio(input_folder, output,seed=42, ratio=(.7, .1, .2))
```

Sample code 1: Sample code for splitting dataset

5.6. Proposed Model Implementation

Our proposed model was the integration of Resnet18 with CCSAM attention modules. The mathematical improvement are discussed in the previous section chapter 4, in this chapter the implementation of channel attention and spatial attention and then the combination of both of them are discussed. Then the implementation of Fine-tuned Resnet18 with CCSAM and pre-trained ResNet18 with CCSAM were discussed.

A. Channel Attention implementation

```
class ChannelAttention(nn.Module):
    def __init__(self, channels, reduction=16):
        super(ChannelAttention, self).__init__()
        self.avg_pool = nn.AdaptiveAvgPool2d(1)
        self.max_pool = nn.AdaptiveMaxPool2d(1)
        self.fc1 = nn.Conv2d(channels, channels // reduction, kernel_size=1, bias=False)
        self.relu = nn.ReLU(inplace=True)
        self.fc2 = nn.Conv2d(channels // reduction, channels, kernel_size=1, bias=False)
        self.sigmoid = nn.Sigmoid()

    def forward(self, x):
        avg_out = self.fc2(self.relu(self.fc1(self.avg_pool(x))))
        max_out = self.fc2(self.relu(self.fc1(self.max_pool(x))))
        channel_out = self.sigmoid(avg_out + max_out)
        return channel_out
```

Sample code 2: Sample code of Channel Attention implementation

B. Spatial attention implementation

```
class SpatialAttention(nn.Module):
    def __init__(self):
        super(SpatialAttention, self).__init__()
        self.conv = nn.Conv2d(2, 1, kernel_size=7, stride=1, padding=3)
        self.sigmoid = nn.Sigmoid()
    def forward(self, x):
        avg_weighted = torch.mean(x, dim=1, keepdim=True)
        max_weighted, _ = torch.max(x, dim=1, keepdim=True)
        spatial_descriptor = torch.cat([avg_weighted, max_weighted], dim=1)
        spatial_out = self.conv(spatial_descriptor)
        spatial_out = self.sigmoid(spatial_out)
        return spatial_out
```

Sample code 3: Sample code of spatial attention

C. Combination of channel attention and spatial attention implementation

```
class CCSAM(nn.Module):
    def __init__(self, in_planes, ratio=16):
        super(CCSAM, self).__init__()
        self.channel_attention = ChannelAttention(in_planes, ratio)
        self.spatial_attention = SpatialAttention()

    def forward(self, x):
        x = self.channel_attention(x) * x
        x = self.spatial_attention(x) * x
        return x
```

Sample code 4: Sample code of CCSAM

In order to integrate the above CCSAM attention module with ResNet18. We have classified Resnet18 pre-trained model into Fine-tuning and Frozen layers. Because we have to compare their accuracy of them for selecting a better model for cattle disease identification.

5.6.1. Fine-tuned Proposed Model Implementation

In this fine-tuning implementation all layers are re-defined for training the model from scratch

```
def conv3x3(in_planes, out_planes, stride=1):
    return nn.Conv2d(in_planes, out_planes, kernel_size=3, stride=stride, padding=1, bias=False)
class BasicBlock(nn.Module):
    expansion = 1
    def __init__(self, inplanes, planes, stride=1, downsample=None, groups=1,
                 base_width=64, dilation=1, norm_layer=None,
                 *, reduction=16):
        super(BasicBlock, self).__init__()
        self.conv1 = conv3x3(inplanes, planes, stride)
        self.bn1 = nn.BatchNorm2d(planes)
        self.relu = nn.ReLU(inplace=True)
        self.conv2 = conv3x3(planes, planes, 1)
        self.bn2 = nn.BatchNorm2d(planes)
        self.ccsam = CCSAM(planes, reduction)
        self.downsample = downsample
        self.stride = stride
    def forward(self, x):
        residual = x
        out = self.conv1(x)
        out = self.bn1(out)
        out = self.relu(out)
        out = self.conv2(out)
        out = self.bn2(out)
        out = self.ccsam(out)
        if self.downsample is not None:
            residual = self.downsample(x)
        out += residual
        out = self.relu(out)
        return out
```

After the Basic Block of ResNet was defined we have created another def class which was used to modify the final fully connected layers of Resnet and used to display the Basic Block class within number classes of our proposed classes(num_classes=5)

```
def resnet18_ccsam(num_classes=5):
    model = ResNet(BasicBlock, [2, 2, 2, 2], num_classes=num_classes)
    model.avgpool = nn.AdaptiveAvgPool2d(1)
    model.fc = nn.Linear(model.fc.in_features, 512)
    # Add a dropout layer with probability 0.5
    model.dropout = nn.Dropout(p=0.5)
    # Add a new fully connected layer with 256 output features
    model.dense = nn.Linear(512, 256)
    # Add a new fully connected layer with 5 output classes and softmax activation
    model.fc2 = nn.Linear(256, 5)
    model.softmax = nn.Softmax(dim=1)

    return model
model=resnet18_ccsam(num_classes=5)
print(model)
```

Sample code 5: Sample code of fine-tuned proposed Resnet_CCSAM model

5.6.2. Freezing Layers Proposed Model Implementation

In this freezing layers pre-trained layers Implementation, first loading pre-trained model, then freeze all pre-trained layers, and also add CCSAM on all Basic Block of Resnet18 we can add directly without defining new class:

```
# Load the pre-trained ResNet18 model
model = models.resnet18(pretrained=True)

# Freeze the pre-trained layers
for name, param in model.named_parameters():
    if 'layer1' in name or 'layer2' in name or 'layer3' in name or 'layer4' in name:
                param.requires_grad = False
#add ccsam on basicblock 0 of layer1
model.layer1[0].add_module("CCSAM", CCSAM(64))
#add ccsam on basicblock 1 of layer1
model.layer1[1].add_module("CCSAM", CCSAM(64))
#add ccsam on basicblock 0 of layer2
model.layer2[0].add_module("CCSAM", CCSAM(128))
#add ccsam on basicblock 1 of layer2
model.layer2[1].add_module("CCSAM", CCSAM(128))
#add ccsam on basicblock 0 of layer3
model.layer3[0].add_module("CCSAM", CCSAM(256))
#add ccsam on basicblock 1 of layer3
model.layer3[1].add_module("CCSAM", CCSAM(256))
#add ccsam on basicblock 4 of layer4
model.layer4[0].add_module("CCSAM", CCSAM(512))
#add ccsam on basicblock 1 of layer4
model.layer4[1].add_module("CCSAM", CCSAM(512))
```

Then, modifying the last fully connected layers because pre-trained layers are trained on 1000 class but in our thesis we would change to 5 classes

```
# Replace the last fully connected layer with a new one that has 512 output features
num_ftrs = model.fc.in_features
model.fc = nn.Linear(num_ftrs, 512)

# Add a dropout layer with probability 0.5
model.dropout = nn.Dropout(p=0.5)

# Add a new fully connected layer with 256 output features
model.dense = nn.Linear(512, 256)

# Add a new fully connected layer with 5 output classes and softmax activation
model.fc2 = nn.Linear(256, 5)
model.softmax = nn.Softmax(dim=1)

# Print the model architecture
print(model)
```

Sample code 6: Freezing layers pre-trained layers of Resnet18_CCSAM Implementation

5.7. Learning Hyper-parameters

A hyper-parameter refers to a predetermined parameter that is set prior to the start of the learning procedure, encompassing elements like batch size, epoch count, optimization method, learning rate, and loss weight. These variables can be modified and significantly influence the effectiveness of model training. Optimizers, on the other hand, are methods or strategies employed to fine-tune specific attributes of the neural network, such as weights and learning rate, with the goal of minimizing losses. Adam, stochastic gradient descent (SGD), RMSProp, and AdaGrad etc. many optimizations are there (Kingma, D. and Ba, 2015).

In this thesis, a Stochastic gradient descent optimizer with learning-rate=0.0001 was utilized to update the network weight (Kingma, D. and Ba, 2015). The step size, commonly referred to as the "learning rate," determines the extent of weight adjustments during training. The training process involves using a batch size of one over a span of 50 epochs. Feature down-sampling is performed using a stride of 2, while a kernel size of 1 is utilized.

SGD provided control over the learning rate and momentum, allowing us to fine-tune these hyper-parameters based on the specific requirements of the cattle disease identification task. The learning rate determined the step size in the optimization process, while momentum helped accelerate convergence and escaped local minima. By carefully adjusting these parameters, we

could optimize the ResNet18 model with CCSAM to achieve better performance. Additionally, SGD's simplicity and ease of implementation made it easier to interpret and reproduce results, which was important in the field of cattle disease identification research.

In this thesis, the advantages of SGD in cattle disease identification research when using the ResNet18 model with CCSAM in the Py-Torch library lay in its efficient handling of large-scale datasets and the control it provided over the learning rate and momentum. By randomly sampling mini-batches and fine-tuning hyper-parameters, SGD enabled effective exploration of the dataset and improved performance in identifying cattle diseases, making it a suitable optimization algorithm for this research task.

Table 5. 2: Hyper-parameters

Hyperparameter	Value
batch size	8
learning rate (lr)	0.0001
model optimizer	SGD
epochs	50

5.8. Experimental Classes

Lists of experimental classes

Table 5. 3: Lists of experimental classes

Experiment	**Datasets**	**model**
Pre-trained Resnet18	Geometric and photometric Dataset	Resnet18
Fine-tuning pre-trained Proposed model	Geometric and photometric Dataset	ResNet18_CCSAM
Freezing layers pre-trained on proposed model	Geometric and photometric Dataset	ResNet18_CCSAM
Fine-tuning Proposed model	Cattle External Disease Dataset	ResNet18_CCSAM
Fine-tuning Proposed model	ISIC skin disease dataset	ResNet18_CCSAM

Related work	Cattle External Disease Dataset	Other Model
Related work	ISIC skin disease dataset	Other Model

CHAPTER SIX

RESULTS AND DISCUSSION

6.1. Chapter Overview

This chapter, first, describes the experiments carried out to evaluate cattle disease training image transformation techniques. Second, cattle disease classification methods using pre-trained models weights, proposed models with a combination of channel attention and spatial attention module and without attention modules were discussed. Thirdly, cattle disease classification model performance and its evaluation results on different training sample transformation results by the proposed model and by baseline models were discussed in detail.

6.2. Results and Discussions

The baseline model used in this thesis was Resnet18, which is a pre-trained model that has been trained on a large dataset of 1000 classes called ImageNet. ResNet-18 is composed of 18 layers and is commonly used for various computer vision tasks such as image classification and feature extraction. The proposed model, ResNet18_CCSAM, has a combination of channel attention and spatial attention modules. The proposed model was classified into freezing pre-trained layers and fine-tuned proposed model for evaluating cattle disease identification. In the experiment, the images were resized to the same size of 224 x 244 and were used on both geometric and photometric transformation techniques.

As presented in Table 6.1, the original image used in this thesis was augmented to 10650. Therefore, the total size of the dataset used in this thesis is 10650 for both geometric and photometric augmentation techniques.

Table 6. 1: Geometric and photometric dataset size

Classes of Disease	Raw Data	Geometric Transformation	Photometric Transformation
Lumpy Skin Disease(LSD)	225	2250	2250
Infectious Bovine kerato-conjunctivitis (IBK)	240	2400	2400
Foot_mouth disease (FMD)	250	2500	2500

Dermatophytosis (Ringworm)	200	2000	2000
Bovine Papillomatosis (Warts)	150	1500	1500
Total	1065	10650	10650

6.2.1. Classification Performance for the Proposed and Baseline Method

As shown in Table 6.2, the Resnet18_CCSAM model and baseline model on both geometric and photometric augmentations techniques classification report for the mean aggregate Accuracy, Precision, Recall, and F1-Score values are presented. As experimental result in Table 6.2, demonstrated the fine-tuned proposed model (ResNet18_CCSAM) on photometric augmentation techniques (a), outperformed the baseline model by +5% aggregate accuracy, the freezing layers Resnet18_CCSAM by +3.5% aggregate accuracy. On geometric augmentation techniques (b), outperformed the baseline model by +5.27% aggregate accuracy and freezing layers Resnet18_CCSAM by +4.07% accuracy. The fine-tuned Resnet18_CCSAM with geometric augmentation techniques has scored high performance than photometric augmentation techniques. The proposed model with a combination of channel and spatial attention mechanism training with all augmentation techniques achieved sufficiently high results for all metrics. This mainly shows that the CCSAM attention module added on deep CNN method can focus on target cattle disease spots instead of the whole image and image background.

Table 6. 2: Classification performance for the proposed and baseline models

Transformation	**Metrics**	**ResNet18**	**ResNet18_CCSAM**	
		Baseline	Freezing Layers	Fine-Tuned(Ours)
(a) Photometric	Precision	0.93	0.944	0.98
	Recall	0.93	0.944	0.98
	F1 Score	0.93	0.942	0.98
	Accuracy	0.93	0.945	**0.98**
(b) Geometric	Precision	0.938	0.944	0.99
	Recall	0.938	0.944	0.99
	F1 Score	0.936	0.944	0.99
	Accuracy	0.937	0.949	**0.9897**

Table 6.2 shows that the proposed fine-tuned Resnet18_CCSAM methods have produced competitive results with both geometric (b) and photometric augmentation (a). However, geometric augmentation in (b) has a better-aggregated accuracy of 98.97% for the proposed Resnet18_CCSAM. The photometric augmentation shown in Table 6.2 (a) has shown a 0.97% performance reduction to geometric augmentation (b) for the proposed Resnet18_CCSAM method. This is because simple photometric data augmentation techniques duplicate existing training samples. While freezing layer Resnet18_CCSAM with both augmentation (a) and (b), it attends spatial information without considering the global disease context. The proposed fine-tuned Resnet18_CCSAM captures global disease characteristics from the cattle disease images. From the experiment result, the nature of augmentation and the structure of developed models have impacted disease recognition performance.

NB: - the classification report values of Precision, Recall, F1-Score and Accuracy for each class by baseline and proposed model on both geometric and photometric augmentation were listed in the Appendix part.

6.2.2. Proposed Model and Baseline Model Confusion Matrix

Figure 6.1, (a) to (f) shows the confusion matrix for the proposed Resnet18_CCSAM model and baseline model on both geometric and photometric augmentations. As can be seen from Figure 6.1(a), the performance of the fine-tuned proposed model on geometric augmentation using confusion matrix, the target class are seen as five cattle disease namely FMD (foot and mouth disease), KCD(Infectious bovine kerato-conjunctivitis disease), LD(lumpy disease), RWD(Ringworm disease), and WD (bovine papillomatosis or warts disease) respectively.

As such 99.60%, 100%, 99.78%, 97.25%, and 97.35% of testing dataset instances are correctly classified as FMD (foot and mouth disease), KCD (Infectious bovine kerato-conjunctivitis disease), LD (lumpy disease), RWD (Ringworm disease), and WD (bovine papillomatosis or warts disease) respectively. Where the model misclassifies only 0.2% instances of FMD (foot and mouth disease) as RWD (Ringworm disease) and 0.2% instances as WD (bovine papillomatosis or warts disease), 0.22% instances of LD (lumpy disease) into KCD (Infectious bovine kerato-conjunctivitis disease), 0.33% and 2.32% instances of WD into LD and RWD respectively. And 1.5%, 0.50%, and 0.75% instances of RWD as FMD, LD, and WD respectively. But the baseline model and freezing layers Resnet18_CCSAM model have more

confusion on both augmentation techniques than the proposed fine-tuned Resnet18_CCSAM model. For example, the baseline model in Figure 6.1(e) on geometric augmentation misclassifies 0.81%, 0.2%, 3.63%, and 1.81% instances of FMD classified into KCD, LD, RWD, and WD respectively. 0.83%, 0.42%, 0.42%, and 0.83% instances of KCD as FMD, LD, RWD, and WD respectively. 1.55%, 0.88%, 3.54%, and 6.42% instances of LD classified as FMD, KCD, RWD, and WD respectively. 0.5% of instances of RWD classify as WD. And 0.33% and 3.97% instances of WD classifies as LD, RWD respectively. This shows misclassification or confusion in the baseline model was more when compared to the fine-tuned proposed Resnet18_CCSAM model.

In general, as illustrated in Figure 6.1(a) and (c) the fine-tuned Resnet18_CCSAM demonstrated high-performance classification on both geometric and photometric augmentation. The confusion matrix for freezing layers Resnet18_CCSAM (b) and (d), and the confusion matrix for the baseline model (e) and (f) showed good performance but there were more confusions than the fine-tuned ResNet18_CCSAM in Figure 6.1 (a) and (c). Because the behaviors of some diseases are somewhat closest and also the result demonstrates that the training model with channel and spatial attention mechanism has a more powerful discriminative learning ability on targeted cattle disease spots area than the pre-trained model which not trained with an attention mechanism.

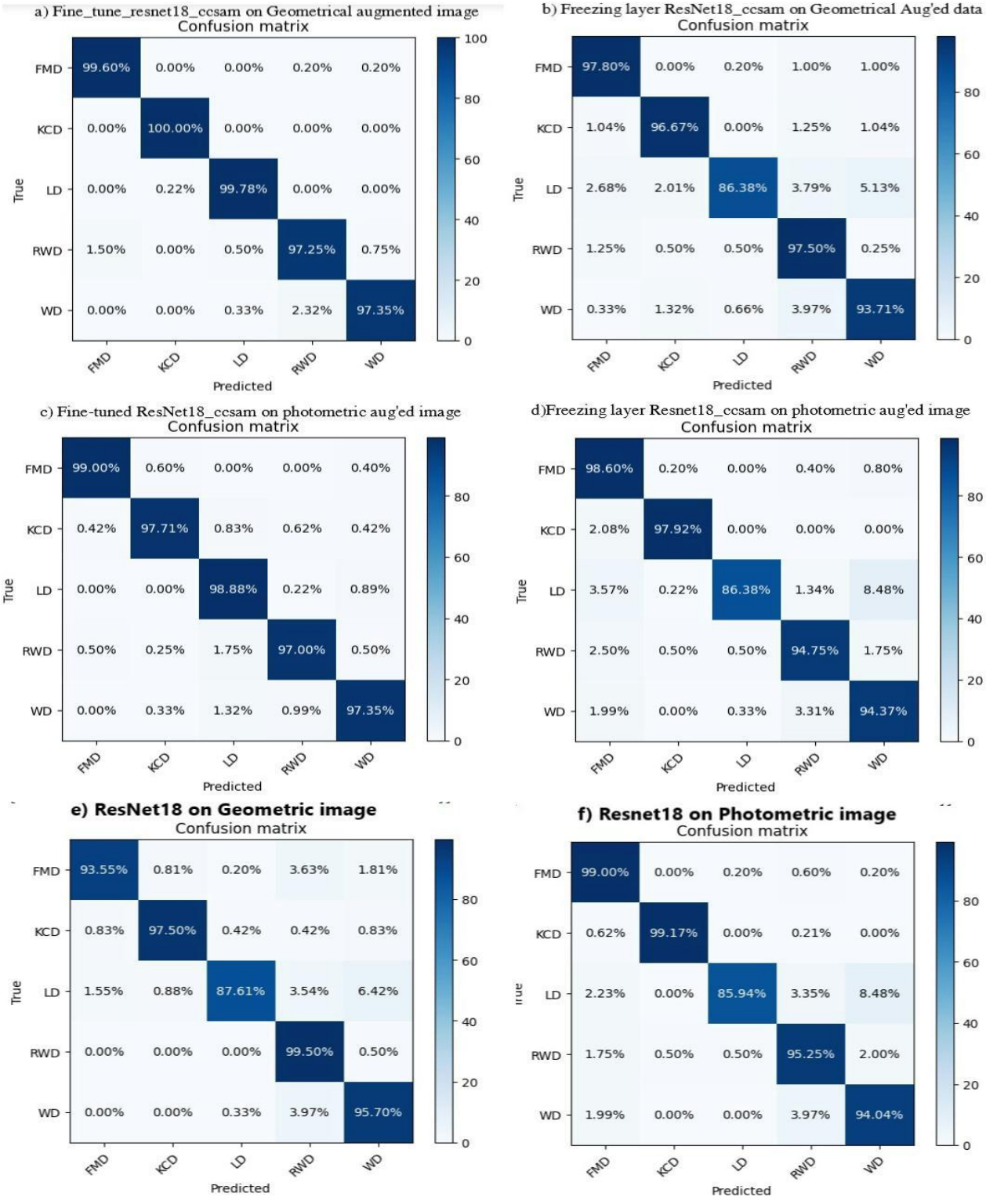


Figure 6. 1:- Confusion matrix for the proposed model and baseline model

6.2.3. Heat Map Visualization for the Proposed Model and Baseline Model

Figure 6.2 displays a sample of the original image, as well as the heat map generated on the original image using Resnet18, freezing layer Resnet18 with CCSAM, and the fine-tuned Resnet18 with CCSAM.

The Resnet18 model's attention heat map visualization failed to identify the disease spot, which demonstrates poor heat map visualization. Figure 6.2 shows that the Freezing Layer Resnet18 with a combination channel and spatial attention mechanism model heat map visualization has fewer abilities on some disease images than Resnet18. Both Resnet18 and Freezing Layer Resnet18_CCSAM focus on the whole image rather than the spot area from cattle disease images. However, the fine-tuned Resnet18_CCSAM model has enabled clear disease spot visualization, which confirms the model's ability to capture global and local features. The fine-tuned Resnet18_CCSAM model's attention heat maps are clearer and bigger compared to the baseline model, as shown in Figure 6.2. The combination of channel and spatial attention mechanisms with the deep convolutional neural network has a strong capability to model local and global features jointly, which is robust enough to target cattle disease spots instead of the whole image and image background.

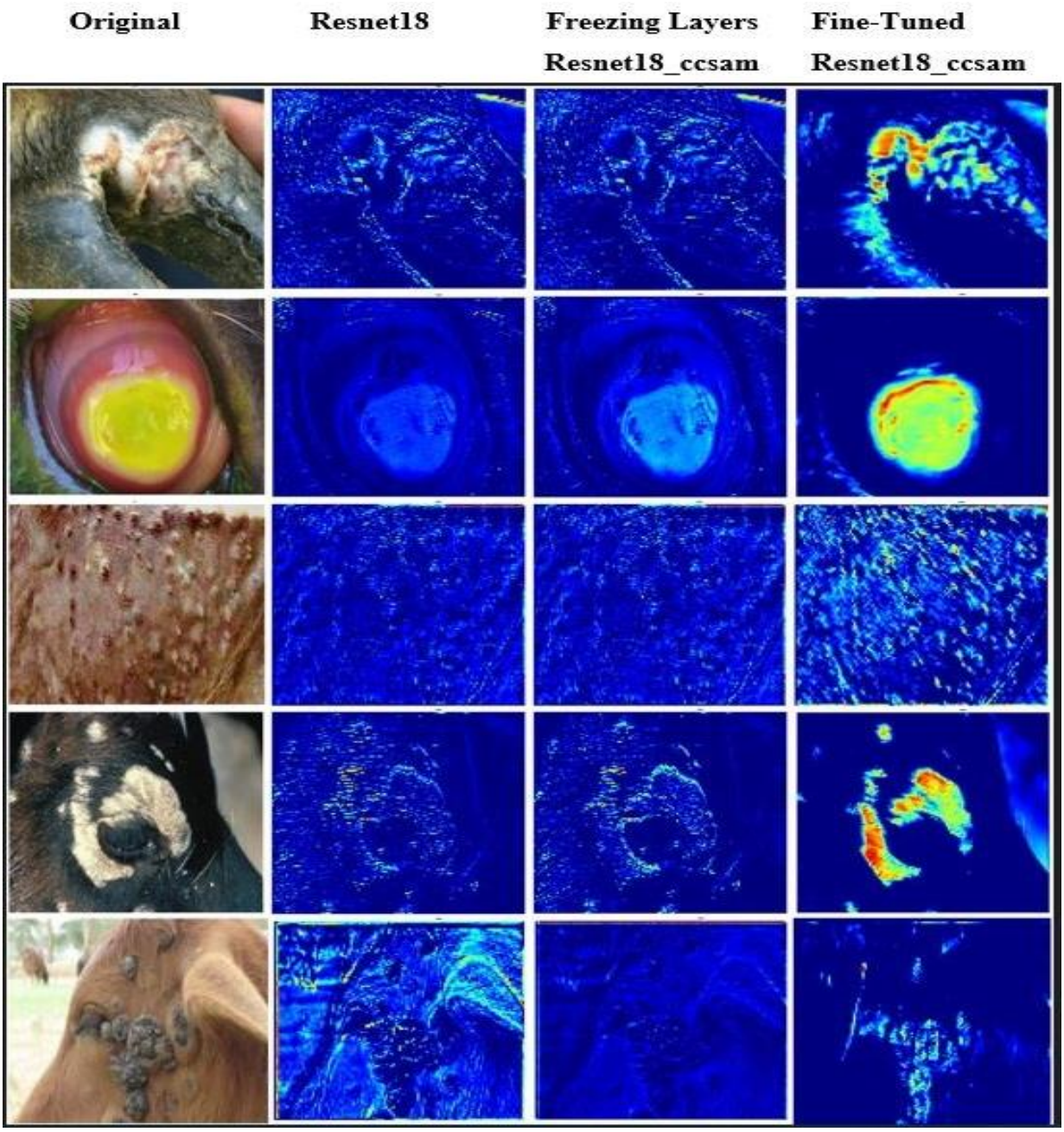


Figure 6. 2:- Attention Heat Map Visualization for Proposed Model and Baseline Model

6.2.4. Accuracy and Loss Plot for the Model Training and Validation

Figure 6.3, Figure 6.4, and Figure 6.5 presented the training and validation accuracy/loss curve and their trend across epochs. The training accuracy curve is consistent with that of validation as shown in Figure 6.4 and Figure 6.3. This indicates that the model does not have an overfitting problem with the parameters chosen during the training process.

Figure 6.3: illustrates training, validation accuracy, and loss observed on geometric and photometric augmentations techniques by using a baseline model without an attention mechanism.

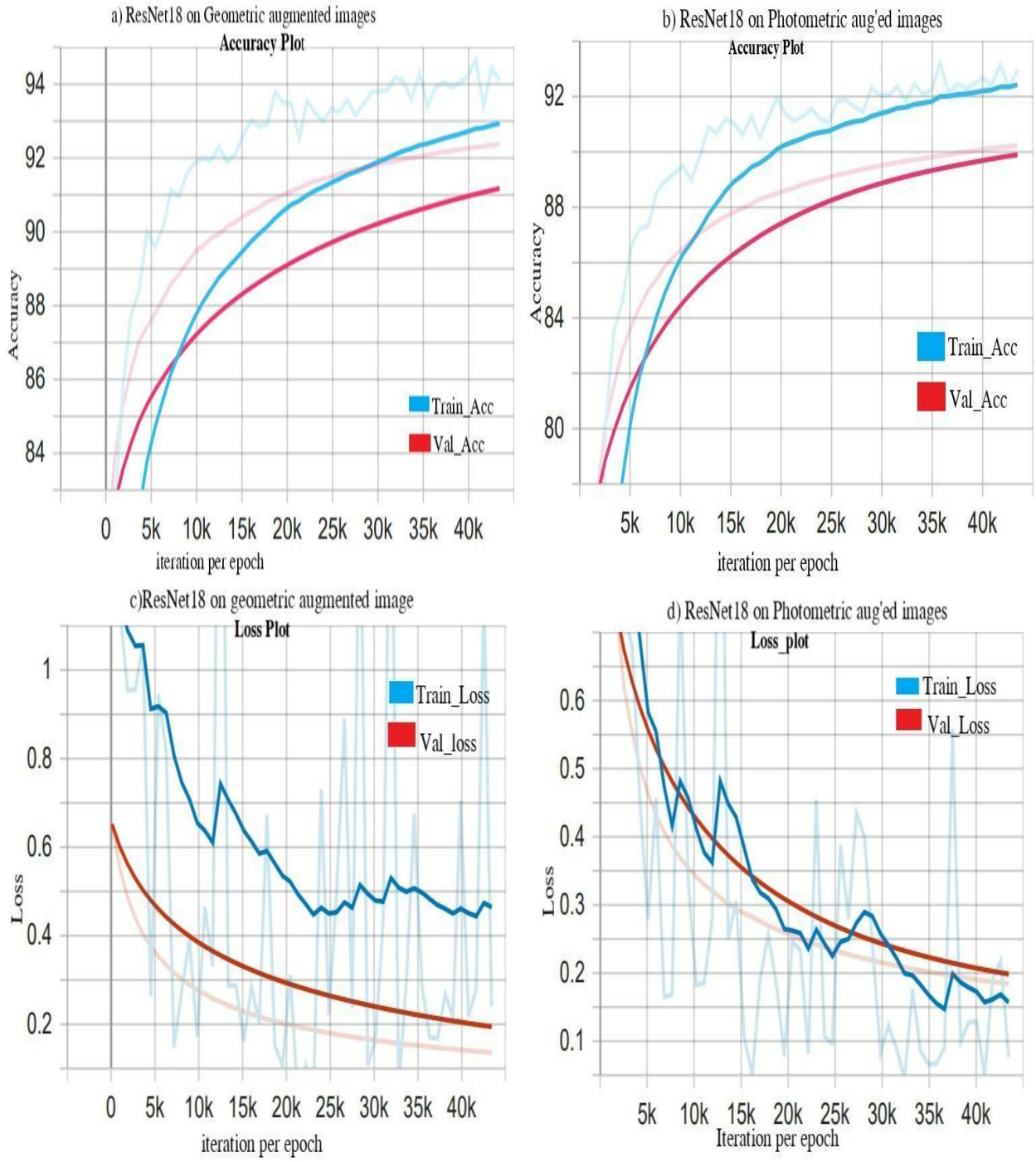


Figure 6. 3:- Training, validation accuracy and loss on geometric and photometric Aug data by baseline model

Figure 6.4: illustrates training and validation accuracy observed on geometric and photometric augmentations techniques by using fine-tuned proposed model and freezing layers proposed model with the combination of channel attention and spatial attention mechanisms (ResNet18_CCSAM).

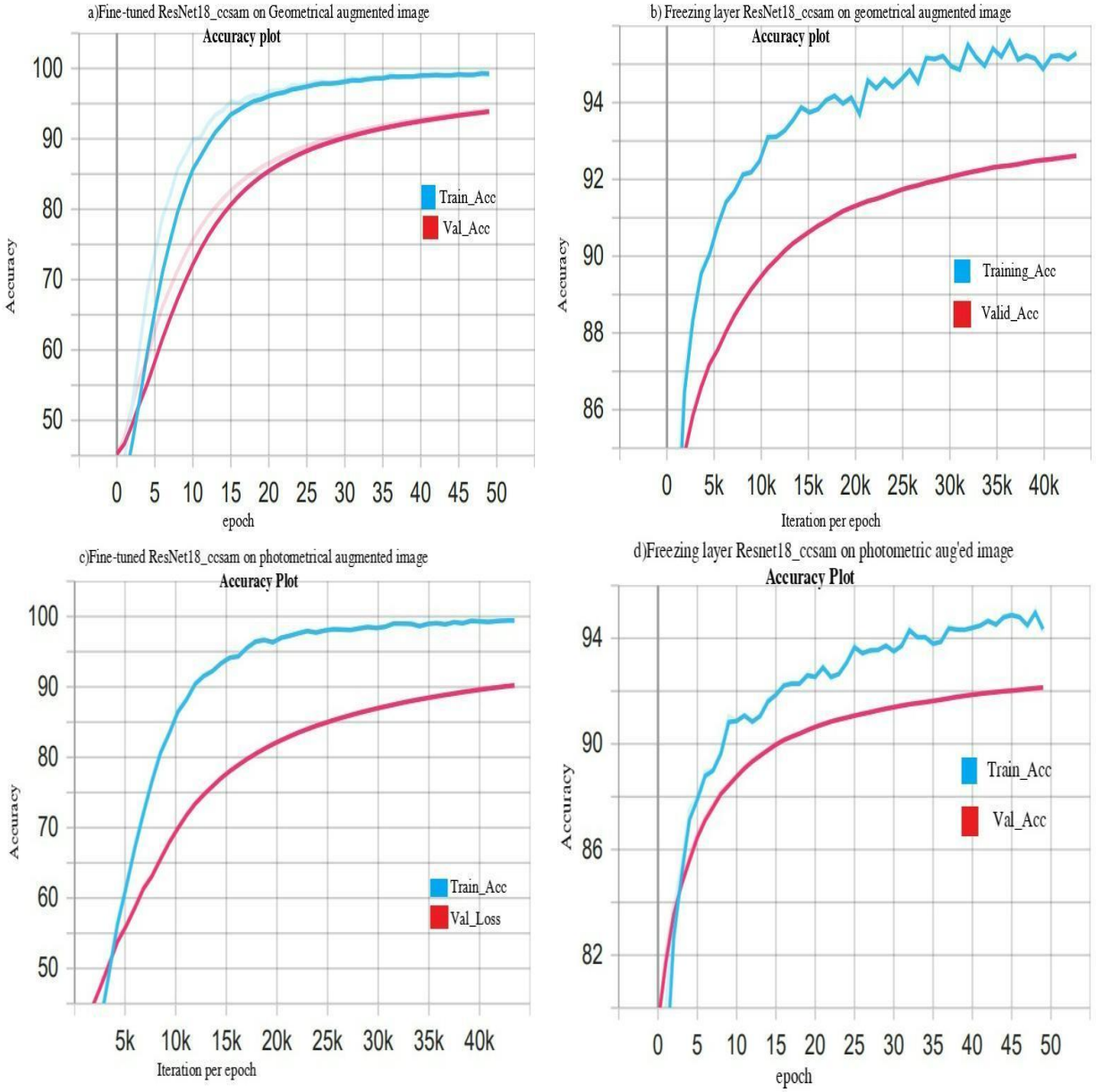


Figure 6. 4:- Training and validation accuracy of proposed model fine-tuned and freezing layer model on geometrical and photometric data

Figure 6.5: illustrates training and validation loss observed on geometric and photometric augmentations techniques by using fine-tuned proposed model and freezing layers model with the combination of channel attention and spatial attention mechanism (ResNet18_CCSAM).

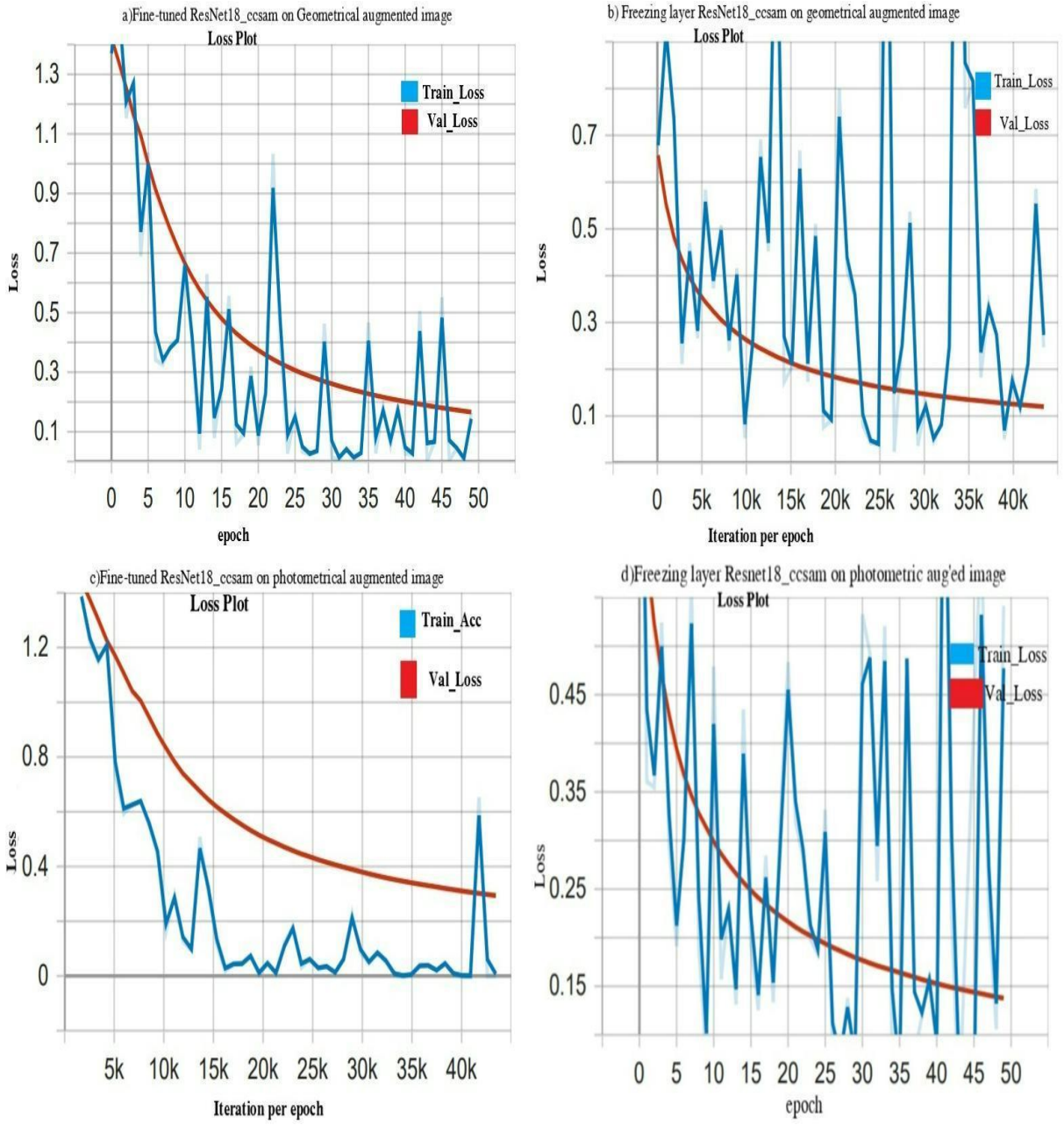
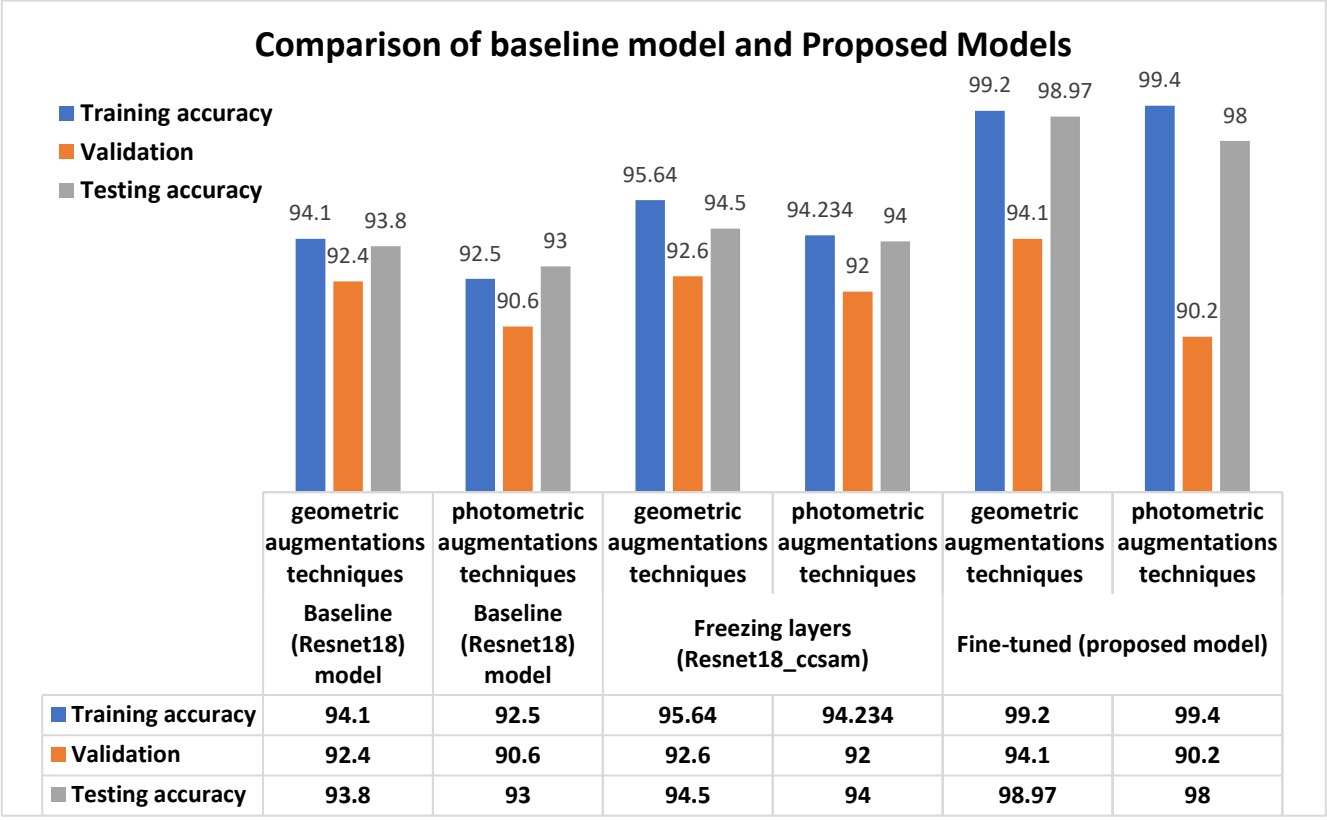


Figure 6. 5:- Training and validation accuracy loss of proposed model fine-tuned and freezing layer on geometrical and photometric data

6.2.5. Comparison of the Proposed Model with Baseline Model

The results of training, validation, and testing accuracy for the proposed fine-tuned model, freezing layers model, and baseline model were presented in Figure 6.5.

As depicted in Figure 6.6 below, the fine-tuned proposed model with geometric augmentations techniques has obtained +5.17% testing accuracy over baseline testing accuracy and also obtained +4.47% testing accuracy over the freezing layer Resnet18_CCSAM model. The fine-tuned proposed model with photometric augmentations techniques has obtained +5% testing accuracy over baseline model testing accuracy and also obtained +4% testing accuracy over the freezing layer Resnet18_CCSAM model. The experiment result shows the proposed model Resnet18_CCSAM with geometric augmentation has good performance when compared with the baseline model and freezing layers model. From the result, it is clear that the training model with a combination of channel and spatial attention mechanism (CCSAM) has a great advantage in classifying cattle disease and the result shows that the augmentation techniques we use during the training model have an impact on classification cattle disease.



	geometric augmentations techniques	photometric augmentations techniques	geometric augmentations techniques	photometric augmentations techniques	geometric augmentations techniques	photometric augmentations techniques
	Baseline (Resnet18) model	Baseline (Resnet18) model	Freezing layers (Resnet18_ccsam)		Fine-tuned (proposed model)	
Training accuracy	94.1	92.5	95.64	94.234	99.2	99.4
Validation	92.4	90.6	92.6	92	94.1	90.2
Testing accuracy	93.8	93	94.5	94	98.97	98

Figure 6. 6:- Comparison of the proposed model with baseline Model

6.2.6. Related Work Comparison

In this experiment, the classification accuracy of the proposed model on a public dataset was presented and compared with the accuracy of related work models. Firstly, from related work, we took the paperwork by an author (Rony et al., 2021) using an inception-v3 model on three classes of external cattle diseases. Secondly, the public dataset we took was the International Skin Imaging Collaboration (ISIC) skin disease dataset[6]. In this comparison experiment, related work model accuracy and the public dataset we used for comparison were not the same with our model and with our dataset. But we want to show existing related work classification accuracy with our model classification accuracy and evaluate the performance of our proposed model on other public datasets.

Table 6. 3: Related Work Comparison

Model	**Dataset**	**Authors**	**Classes**	**Accuracy**
Inception V3	Cattle External Disease [5]	(Rony et al., 2021)	3	95
Google Net	ISIC Skin disease[6]	(Kassem et al., 2020)	8	94
Proposed model (our)	Cattle External Disease [5]	our	3	98.5
Proposed model (our)	ISIC Skin disease[6]	our	8	98
Proposed model (our)	Our Dataset	our	5	98.97

As clearly shown in Table 6.3, the proposed model on the cattle external disease[5] dataset outperformed +3.5% accuracy over related work (Rony et al., 2021). When comparing the classification performance of our proposed model on public dataset[6] with related work in (Kassem et al., 2020) that was conducted on a similar dataset, the proposed model achieved +4% greater accuracy over related work in (Kassem et al., 2020). The proposed method has robust classification ability due to its channel and spatial attention mechanism discriminative learning nature despite the high accuracy in other related works. Therefore, the combination of

[5] Three classes, Dataset on Cattle External Disease by Rony et al., 2021

[6] ISIC skin disease image datasets labelled. These images were originally part of the ISIC skin disease dataset. https://www.kaggle.com/datasets/riyaelizashaju/isic-skin-disease-image-dataset-labelled

channel and spatial attention mechanism with deep convolutional neural network method demonstrated robust performance on several evaluation metrics.

6.3. Summary

The objective of this thesis is to develop a deep convolutional neural network model for cattle disease identification. As stated in the problem statement, the study focuses on modeling the best-performed methods that can focus on target cattle disease spots instead of the whole image and image background. The selected method to achieve this goal is the Deep CNN method. As mentioned in the introduction to this research, cattle disease classification or identification is a major problem in Ethiopian societies. Cattle are harmed by various diseases such as lumpy disease (LD), bovine papillomatosis (warts) disease, Foot_mouth disease (FMD) disease, infectious bovine keratoconjunctivitis (IBK), and dermatophytosis (ringworm), resulting in a wide range of productivity losses.

Disease detection and classification can be done with the naked eye, but this method is ineffective when it comes to huge cattle populations. To answer this challenge, numerous researchers used various approaches and algorithms to conduct the study. K-Nearest Neighbor, Inception-v3, supportive vector machine, VGG, and previous research has utilized other methods to handle this problem. However, few researchers used common deep learning algorithms that focused on common features in cattle disease images. The research conducted in this area was not sufficient. The identification and classification of cattle diseases were the roots of the problems that received attention in the current economic development sector. As a response to this difficulty, this work developed and demonstrated a model for classifying cattle disease images using deep convolutional neural network methods and attention mechanisms to focus on spot areas in cattle disease images. Deep convolutional neural networks and attention mechanisms are the best methods of deep learning as discussed in the literature part of this study.

The model created as a result of this study is Resnet18 with the combination of channel attention and spatial attention mechanisms, or Resnet18_CCSAM. Therefore, it is selected to conduct this research experiment. Different tools and libraries are used during implementation as discussed in Chapter 5. The results of the experiments are mentioned in the result portion. Different data augmentation techniques, different pre-trained models, and different datasets were used to explore the experiments. All these experiments were used to select the best-performing model,

which is the solution to the stated problem. The experiment results were evaluated using different evaluation metrics as proposed. Additionally, the confusion matrix was another method used to evaluate the classification performance of the model. According to the model evaluations, the best-fit model found has 99.4 training accuracy and 94.1 validation accuracy with geometric data augmentation techniques.

As presented in Table 6.3, disease classification research was conducted previously. However, the method they used, the data they used, and the model performance they found are different from this study. For example, the paper conducted by (Rony et al., 2021) used the external cattle disease dataset, and the performance achieved was 95%. The paper conducted by (Kassem et al., 2020) used the International Skin Imaging Collaboration (ISIC) skin disease dataset, and the performance achieved was 94%. However, the proposed model of this thesis achieved 98.5 and 98% performance on the external cattle disease dataset and the International Skin Imaging Collaboration (ISIC) skin disease dataset, respectively. This means the proposed method has robust classification ability due to its channel and spatial attention mechanisms and discriminative learning nature. Therefore, the combination of channel and spatial attention mechanisms with the deep convolutional neural network method has the great advantage of focusing on the spot area of the image instead of the whole image.

CHAPTER SEVEN

CONCLUSION AND FUTURE WORKS

7.1. Conclusions

Cattle disease is a major problem in societies that live on farms around the world. Various diseases such as lumpy disease (LD), bovine papillomatosis (warts) disease, dermatophytosis (ringworm), infectious bovine kerato-conjunctivitis (IBK), and Foot_mouth disease (FMD) harm cattle and result in a wide range of productivity losses. Disease detection and classification can be done with the naked eye, but when it comes to huge cattle populations, this method is ineffective. To answer this challenge, numerous researchers used various approaches and algorithms to conduct the study. K-Nearest Neighbor, Inception-v3, supportive vector machine, VGG, and previous research has utilized other methods to handle this problem. However, few researchers used common deep learning algorithms that focused on common features in cattle disease images.

The research conducted in this area was not sufficient. Therefore, this research is conducted using deep convolutional neural network for cattle disease identification and attention mechanisms to focus on spot areas in cattle disease images. Particularly, the deep residual CNN model ResNet18 with channel and spatial attention mechanisms (CCSAM) added at each residual network basic block was proposed. The experiments were studied using different data augmentation techniques, different pre-trained models, and different datasets. In this thesis, datasets are collected from Haramaya University Veterinary Medicine, Chiro Woreda Veterinary Clinic, West Hararghe Zone Veterinary Office, and also from the Kaggle open cattle disease dataset by (Rony et al., 2021). The original image used in this thesis is 1065 by augmenting the original image to 10650 for both geometric and photometric augmentation techniques separately under two groups. The targeted classes in this thesis are the five most common cattle diseases, including lumpy disease (LD), bovine papillomatosis (warts) disease, dermatophytosis (ringworm), infectious bovine keratoconjunctivitis (IBK), and Foot_mouth disease (FMD).

To measure the overall performance of the proposed method, the mean aggregate values of precision, recall, F1 score, and accuracy were calculated. To further strengthen our evaluation,

the confusion matrix and classification report metrics were used. In general, the proposed ResNet18-CCSAM method, fine-tuned from a pre-trained ResNet18 model on a geometrically augmented dataset, has obtained a precision of 99%, a recall of 98%, an F1 score of 98%, and 98.97% accuracy. Compared to the baseline model and other related research works, the proposed method has robust classification ability due to its channel and spatial attention mechanisms and discriminative learning nature. Therefore, the combination of channel and spatial attention mechanisms with the deep convolutional neural network method has the great advantage of focusing on the spot area of the image instead of the whole image.

7.2. Contributions

This research work has the following contributions:

- The research identify Deep CNN methods which contains small numbers of parameters have more advantages over large parameters Deep CNN in cattle disease identification when the number of datasets is not too large.
- The research contributes to the development of a deep convolutional neural network model, specifically ResNet18 with the combination of channel and spatial attention mechanisms (CCSAM), for cattle disease identification. This model leverages the power of deep learning and attention mechanisms to accurately classify cattle diseases based on image data. The model's architecture and implementation provide a valuable contribution to the field of livestock disease identification.
- The research increased number of disease to five cattle diseases Additionally, data augmentation techniques are employed to increase the diversity and size of the training dataset, leading to enhanced model generalization and robustness.

7.3. Future Works

There are many research areas on different animals to protect them from the loss of their productivity. In this study, we have used the five cattle diseases only used in lumpy disease (LD)**,** bovine papillomatosis **(**warts) disease, dermatophytosis (Ringworm), Infectious bovine kerato-conjunctivitis (IBK), and Foot_mouth disease (FMD). Therefore, future work will be focused on all animal diseases by improving the model in the best way. Furthermore, we will find appropriate training data generation techniques and collect more original datasets rather than augmenting more times for the training model.

This research is sponsored by **Adama Science and Technology University** by the following Grant Number:

ASTU/SM-R/826/23

REFERENCES

Addise Ambilo, A. M. (2013). *Major Skin Diseases of Cattle : Prevalence and Risk Factors in and around*. *3*, 147–153.

Aiello, S. E., Moses, M. A., & Allen, D. G. (2016). *The Merck veterinary manual*. Merck & Company, Incorporated White Station, NJ, USA.

AlFayez, F., El-Soud, M. W. A., & Gaber, T. (2020). Thermogram Breast Cancer Detection: a comparative study of two machine learning techniques. *Applied Sciences*, *10*(2), 551.

Anwar, S., Tahir, M., Li, C., Mian, A., Khan, F. S., & Muzaffar, A. W. (2020). Image colorization: A survey and dataset. *ArXiv Preprint ArXiv:2008.10774*.

Aremu, T., & Nandakumar, K. (2023). PolyKervNets: Activation-free Neural Networks For Efficient Private Inference. *2023 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, 593–604. https://doi.org/10.1109/SaTML54575.2023.00045

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural Machine Translation by Jointly Learning to Align and Translate. *ArXiv*, *1409*.

Bharati, P., & Pramanik, A. (2020). Deep learning techniques—R-CNN to mask R-CNN: a survey. *Computational Intelligence in Pattern Recognition: Proceedings of CIPR 2019*, 657–668.

Birhanu, T. (2014). Prevalence of the major infectious animal diseases affecting livestock trade industry in Ethiopia. *Journal of Biology, Agriculture and Healthcare*, *4*(17), 76–82. http://www.iiste.org/Journals/index.php/JBAH/article/view/15171

Chandana, P., Ghantasala, G. P., Jeny, J. R. V., Sekaran, K., Deepika, N., Nam, Y., & Kadry, S. (2020). An effective identification of crop diseases using faster region based convolutional neural network and expert systems. *International Journal of Electrical and Computer Engineering (IJECE)*, *10*(6), 6531–6540.

Codella, N. C. F., Gutman, D., Celebi, M. E., Helba, B., Marchetti, M. A., Dusza, S. W., Kalloo, A., Liopyris, K., Mishra, N., Kittler, H., & Halpern, A. (2018). Skin lesion analysis toward melanoma detection: A challenge at the 2017 International symposium on biomedical

imaging (ISBI), hosted by the international skin imaging collaboration (ISIC). *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, 168–172. https://doi.org/10.1109/ISBI.2018.8363547

Dahmen, J., & Cook, D. (2019). SynSys: A synthetic data generation system for healthcare applications. *Sensors*, *19*(5), 1181.

Deng, Z., Zhu, X., Cheng, D., Zong, M., & Zhang, S. (2016). Efficient kNN classification algorithm for big data. *Neurocomputing*, *195*, 143–148.

Ding, J., Chen, B., Liu, H., & Huang, M. (2016). Convolutional neural network with data augmentation for SAR target recognition. *IEEE Geoscience and Remote Sensing Letters*, *13*(3), 364–368.

Du, H., & Li, Y. (2018). *Feature Extraction and Image Recognition with Convolutional Neural Networks Feature Extraction and Image Convolutional Neural Networks Recognition with*.

Ethio, D. A. and C. A. (n.d.). *STANDARD VETERINARY TREATMENT GUIDELINES (Drug Administration and Control Authority of Ethiopia) | PDF | Veterinary Medicine | Infection*. 2022-12-07. Retrieved December 7, 2022, from https://www.scribd.com/doc/48645004/STANDARD-VETERINARY-TREATMENT-GUIDELINES-Drug-Administration-and-Control-Authority-of-Ethiopia

Fadnavis, S. (2014). Image Interpolation Techniques in Digital Image Processing: An Overview. *International Journal Of Engineering Research and Application*, *4*, 2248–962270.

Fakhrul-Islam, K., Jalal, M. S., Poddr, S., Quader, M. N., Sahidur-Rahman, M., Dutta, A., & Mazumder, S. (2016). Clinical Investigation of Foot and Mouth Disease of Cattle in Batiaghata Upazilla Veterinary Hospital, Bangladesh. *Veterinary Sciences: Research and Reviews*, *2*(3), 76–81. https://doi.org/10.17582/journal.vsrr/2016.2.3.76.81

Farahani, A., Voghoei, S., Rasheed, K., & Arabnia, H. R. (2021). A brief review of domain adaptation. *Advances in Data Science and Information Engineering: Proceedings from ICDATA 2020 and IKE 2020*, 877–894.

Fernandez, P. D. M., Pena, F. A. G., Ren, T. I., & Cunha, A. (2019). FERAtt: Facial expression recognition with attention net. *IEEE Computer Society Conference on Computer Vision*

and Pattern Recognition Workshops, *2019-June*, 837–846. https://doi.org/10.1109/CVPRW.2019.00112

Firdaus, F., Abdullah, J., Sadiq, M. A., Darshini, D., Tijjani, A., Abba, Y., Mohammed, K., Lim, E., Chung, T., Jefri, M., Norsidinand, M., & Rahman, A. (2015). *Prevalence Study of Infectious Bovine Keratoconjunctivitisin Dairy cattle under the Ladang Angkat Programme of University Veterinary Hospital of the Universiti Putra Malaysia* . *8*(11), 95–98. https://doi.org/10.9790/2380-081119598

Gebre-Amanuel, E. K., Taddesse, F. G., & Assalif, A. T. (2018). Web Based Expert System for Diagnosis of Cattle Disease. *Proceedings of the 10th International Conference on Management of Digital EcoSystems*, 66–73. https://doi.org/10.1145/3281375.3281400

Gessert, N., Nielsen, M., Shaikh, M., Werner, R., & Schlaefer, A. (2020). Skin Lesion Classification Using Ensembles of Multi-Resolution EfficientNets with Meta Data. *MethodsX*, *7*, 100864. https://doi.org/10.1016/j.mex.2020.100864

Guo, J., & Gould, S. (2015). Deep CNN ensemble with data augmentation for object detection. *ArXiv Preprint ArXiv:1506.07224*.

Gupta, J., Pathak, S., & Kumar, G. (2022). Deep learning (CNN) and transfer learning: a review. *Journal of Physics: Conference Series*, *2273*(1), 12029.

He, K., Zhang, X., Ren, S., & Sun, J. (2016a). Deep residual learning for image recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2016-Decem*, 770–778. https://doi.org/10.1109/CVPR.2016.90

He, K., Zhang, X., Ren, S., & Sun, J. (2016b). Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Hu, J., Shen, L., Albanie, S., Sun, G., & Wu, E. (2020). Squeeze-and-Excitation Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *42*(8), 2011–2023. https://doi.org/10.1109/TPAMI.2019.2913372

Kassem, M. A., Hosny, K. M., & Fouad, M. M. (2020). Skin Lesions Classification into Eight Classes for ISIC 2019 Using Deep Convolutional Neural Network and Transfer Learning. *IEEE Access*, *8*, 114822–114832. https://doi.org/10.1109/ACCESS.2020.3003890

Kaur, D., & Kaur, Y. (2014). International Journal of Computer Science and Mobile Computing Various Image Segmentation Techniques: A Review. *International Journal of Computer Science and Mobile Computing*, *3*(5), 809–814. www.ijcsmc.com

Khan, A., Sohail, A., Zahoora, U., & Qureshi, A. S. (2020a). A survey of the recent architectures of deep convolutional neural networks. *Artificial Intelligence Review*, *53*(8), 5455–5516. https://doi.org/10.1007/s10462-020-09825-6

Khan, A., Sohail, A., Zahoora, U., & Qureshi, A. S. (2020b). A survey of the recent architectures of deep convolutional neural networks. *Artificial Intelligence Review*, *53*(8), 5455–5516. https://doi.org/10.1007/s10462-020-09825-6

Kim, H. E., Cosa-Linan, A., Santhanam, N., Jannesari, M., Maros, M. E., & Ganslandt, T. (2022). Transfer learning for medical image classification: a literature review. *BMC Medical Imaging*, *22*(1), 69.

Kingma, D. and Ba, J. (2015). *Kingma, D. and Ba, J. (2015) Adam A Method for Stochastic Optimization. Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015). - References - Scientific Research Publishing*. https://www.scirp.org/(S(351jmbntvnsjt1aadkozje))/reference/ReferencesPapers.aspx?ReferenceID=1611214

Lake, D. (2013). Web-based Expert System for Cattle Diseases Diagnose. *Unpublished Masters Thesis,Addis Abeba University*, *November*.

Lateef, R., & Abbas, A. (2022). Tuning the Hyperparameters of the 1D CNN Model to Improve the Performance of Human Activity Recognition. *Engineering and Technology Journal*, *40*(4), 547–554. https://doi.org/10.30684/etj.v40i4.2054

Li, J., Jin, K., Zhou, D., Kubota, N., & Ju, Z. (2020). Attention mechanism-based CNN for facial expression recognition. *Neurocomputing*, *411*, 340–350. https://doi.org/10.1016/j.neucom.2020.06.014

Li, M., Han, J., & Yang, J. (2021). Automatic feature extraction and fusion recognition of motor imagery EEG using multilevel multiscale CNN. *Medical & Biological Engineering & Computing*, *59*(10), 2037–2050.

Lin, Y., Nie, Z., & Ma, H. (2017). Structural damage detection with automatic feature-extraction through deep learning. *Computer-Aided Civil and Infrastructure Engineering*, *32*(12), 1025–1046.

Liu, H., Chen, T., Shen, Q., Yue, T., & Ma, Z. (2018). Deep image compression via end-to-end learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2575–2578.

Liu, Y. (2018). An improved faster R-CNN for object detection. *2018 11th International Symposium on Computational Intelligence and Design (ISCID)*, *2*, 119–123.

Mahmud, M. S., Zahid, A., Das, A. K., Muzammil, M., & Khan, M. U. (2021). A systematic literature review on deep learning applications for precision cattle farming. *Computers and Electronics in Agriculture*, *187*, 106313.

Mnih, V., Heess, N., Graves, A., & Kavukcuoglu, K. (2014). Recurrent models of visual attention. *Advances in Neural Information Processing Systems*, *3*(January), 2204–2212.

Molla, W., de Jong, M. C. M., & Frankena, K. (2017). Temporal and spatial distribution of lumpy skin disease outbreaks in Ethiopia in the period 2000 to 2015. *BMC Veterinary Research*, *13*(1), 1–9.

Niu, S., Liu, Y., Wang, J., & Song, H. (2020). A decade survey of transfer learning (2010–2020). *IEEE Transactions on Artificial Intelligence*, *1*(2), 151–166.

Niu, Z., Zhong, G., & Yu, H. (2021). A review on the attention mechanism of deep learning. *Neurocomputing*, *452*, 48–62. https://doi.org/10.1016/j.neucom.2021.03.091

Noreen, N., Palaniappan, S., Qayyum, A., Ahmad, I., Imran, M., & Shoaib, M. (2020). A deep learning model based on concatenation approach for the diagnosis of brain tumor. *IEEE Access*, *8*, 55135–55144.

Orhan G, Y. (2020). *4 Pre-Trained CNN Models to Use for Computer Vision with Transfer*

Learning | by Orhan Gazi Yalçın | Towards Data Science. Sep 23, 2020. https://towardsdatascience.com/4-pre-trained-cnn-models-to-use-for-computer-vision-with-transfer-learning-885cb1b2dfc

Qiao, Y., Su, D., Kong, H., Sukkarieh, S., Lomax, S., & Clark, C. (2019). Individual Cattle Identification Using a Deep Learning Based Framework. *IFAC-PapersOnLine*, *52*(30), 318–323. https://doi.org/10.1016/j.ifacol.2019.12.558

Rai, G., Naveen, Hussain, A., Kumar, A., Ansari, A., & Khanduja, N. (2021). *A Deep Learning Approach to Detect Lumpy Skin Disease in Cows BT - Computer Networks, Big Data and IoT* (A. P. Pandian, X. Fernando, & S. M. S. Islam (eds.); pp. 369–377). Springer Singapore.

Ramirez-Moreno, D., Schwartz, O., & Ramirez-Villegas, J. (2013). A saliency-based bottom-up visual attention model for dynamic scenes analysis. *Biological Cybernetics*, *107*. https://doi.org/10.1007/s00422-012-0542-2

Rony, M., Barai, D., Riad, & Hasan, M. Z. (2021). Cattle External Disease Classification Using Deep Learning Techniques. *2021 12th International Conference on Computing Communication and Networking Technologies, ICCCNT 2021*, 2–8. https://doi.org/10.1109/ICCCNT51525.2021.9579662

Saito, T., & Rehmsmeier, M. (2015). The Precision-Recall Plot Is More Informative than the ROC Plot When Evalua. In *PLoS ONE* (Vol. 10, Issue 3, pp. 1–21).

Sarwinda, D., Paradisa, R. H., Bustamam, A., & Anggia, P. (2021). Deep Learning in Image Classification using Residual Network (ResNet) Variants for Detection of Colorectal Cancer. *Procedia Computer Science*, *179*, 423–431. https://doi.org/https://doi.org/10.1016/j.procs.2021.01.025

Seetharaman, K., & Mahendran, T. (2022). Leaf Disease Detection in Banana Plant using Gabor Extraction and Region-Based Convolution Neural Network (RCNN). *Journal of The Institution of Engineers (India): Series A*, *103*(2), 501–507.

Seid, A. (2019). *Review on Infectious Bovine Keratoconjunctivitis and its Economic Impacts in Cattle*. *9*(5). https://doi.org/10.19080/JDVS.2019.09.555774.

Shah, M., Patel, T., Joshi, M., & Parmar, N. (2019). *To Identify Animal Skin Disease Model Using Image Processing*.

Shorten, C., & Khoshgoftaar, T. M. (2019a). A survey on image data augmentation for deep learning. *Journal of Big Data*, *6*(1), 1–48.

Shorten, C., & Khoshgoftaar, T. M. (2019b). A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*. https://doi.org/10.1186/s40537-019-0197-0

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 1–14.

Souza, T. V. M., & Zanchettin, C. (2019). Improving Deep Image Clustering with Spatial Transformer Layers. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 11730 LNCS*, 641–654. https://doi.org/10.1007/978-3-030-30490-4_51

Stasaski, K., Yang, G. H., & Hearst, M. A. (2020). More diverse dialogue datasets via diversity-informed data collection. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 4958–4968.

Stollenga, M. F., Masci, J., Gomez, F., & Schmidhuber, J. (2014). Deep networks with internal selective attention through feedback connections. *Advances in Neural Information Processing Systems*, *4*(January), 3545–3553.

Subramanian, M., Shanmugavadivel, K., & Nandhini, P. S. (2022). On fine-tuning deep learning models using transfer learning and hyper-parameters optimization for disease identification in maize leaves. *Neural Computing and Applications*, *34*(16), 13951–13968.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *07-12-June*, 1–9. https://doi.org/10.1109/CVPR.2015.7298594

Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., & Liu, C. (2018). A survey on deep transfer learning. *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th*

International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part III 27, 270–279.

Tschandl, P., Rosendahl, C., & Kittler, H. (2018). The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Scientific Data*, *5*(1), 180161. https://doi.org/10.1038/sdata.2018.161

Tsotsos, J. K., Culhane, S. M., Kei Wai, W. Y., Lai, Y., Davis, N., & Nuflo, F. (1995). Modeling visual attention via selective tuning. *Artificial Intelligence*, *78*(1–2), 507–545. https://doi.org/10.1016/0004-3702(95)00025-9

Vyas, S., Shukla, V., & Doshi, N. (2019). FMD and mastitis disease detection in cows using internet of thingh(iot). *Procedia Computer Science*, *160*(2018), 728–733. https://doi.org/10.1016/j.procs.2019.11.019

Wang, J., & Perez, L. (2017). The Effectiveness of Data Augmentation in Image Classification using Deep Learning. *ArXiv:1712.04621v1*.

Weiss, K., Khoshgoftaar, T. M., & Wang, D. (2016). A survey of transfer learning. *Journal of Big Data*, *3*(1), 1–40.

Whang, S. E., & Lee, J.-G. (2020). Data collection and quality challenges for deep learning. *Proceedings of the VLDB Endowment*, *13*(12), 3429–3432.

Xu, Z., Guo, X., Zhu, A., He, X., Zhao, X., Han, Y., & Subedi, R. (2020). Using deep convolutional neural networks for image-based diagnosis of nutrient deficiencies in rice. *Computational Intelligence and Neuroscience*, *2020*.

Yang, C.-K., Lee, C.-Y., Wang, H.-S., Huang, S.-C., Liang, P.-I., Chen, J.-S., Kuo, C.-F., Tu, K.-H., Yeh, C.-Y., & Chen, T.-D. (2022). Glomerular disease classification and lesion identification by machine learning. *Biomedical Journal*, *45*(4), 675–685.

Yang, H., Li, C., Liang, Y., Liu, W., & Meng, F. (2022). *applied sciences*. 1–11.

Yang, X., & Lu, X. (2021). Multi Scale Attention Network for Crowd Counting. *ACM International Conference Proceeding Series*. https://doi.org/10.1145/3487075.3487097

Yogamangalam, R., & Karthikeyan, B. (2013). Segmentation techniques comparison in image

processing. *International Journal of Engineering and Technology*, *5*(1), 307–313.

Zhang, S., Zhang, Q., & Li, P. (2019). Apple disease identification based on improved deep convolutional neural network. *Journal of Forestry Engineering*, *4*(4), 107–112.

Zhao, C., Shuai, R., Ma, L., Liu, W., Hu, D., & Wu, M. (2021). Dermoscopy Image Classification Based on StyleGAN and DenseNet201. *IEEE Access*, *9*, 8659–8679.

Zhou, D.-X. (2020). Theory of deep convolutional neural networks: Downsampling. *Neural Networks*, *124*, 319–327. https://doi.org/https://doi.org/10.1016/j.neunet.2020.01.018

APPENDIX

A. Important Python Libraries used

```
import argparse

import csv

import os

import tqdm

import pdb

import shutil

import time

import torch

import numpy as np

import torchvision

import torch.nn as nn

import os.path as osp

import torch.optim as optim

from torchvision import datasets, models

from functools import partial

import torchvision.models as models

import torch.optim as optim

from torch.autograd import Variable

from torch.backends import cudnn

from torch.optim.lr_scheduler import ReduceLROnPlateau
```

```
from torch.utils.tensorboard import SummaryWriter

from datetime import datetime

import matplotlib.pyplot as plt

import numpy as np

from sklearn.metrics import confusion_matrix

from sklearn.metrics import confusion_matrix, classification_report,
accuracy_score,precision_recall_fscore_support
```

B. Python code for CCSAM implementation

```
import torch

import torch.nn as nn

import torch

import torchvision.models as models

import torch.nn as nn

import numpy as np

class ChannelAttention(nn.Module):

    def __init__(self, channels, reduction=16):

        super(ChannelAttention, self).__init__()

        self.avg_pool = nn.AdaptiveAvgPool2d(1)

        self.max_pool = nn.AdaptiveMaxPool2d(1)

        self.fc1 = nn.Conv2d(channels, channels // reduction, kernel_size=1, bias=False)

        self.relu = nn.ReLU(inplace=True)

        self.fc2 = nn.Conv2d(channels // reduction, channels, kernel_size=1, bias=False)

        self.sigmoid = nn.Sigmoid()
```

```
    def forward(self, x):
        avg_out = self.fc2(self.relu(self.fc1(self.avg_pool(x))))
        max_out = self.fc2(self.relu(self.fc1(self.max_pool(x))))
        channel_out = self.sigmoid(avg_out + max_out)
        return channel_out
class SpatialAttention(nn.Module):
    def __init__(self):
        super(SpatialAttention, self).__init__()
        self.conv = nn.Conv2d(2, 1, kernel_size=7, stride=1, padding=3)
        self.sigmoid = nn.Sigmoid()
    def forward(self, x):
        avg_weighted = torch.mean(x, dim=1, keepdim=True)
        max_weighted, _ = torch.max(x, dim=1, keepdim=True)
        spatial_descriptor = torch.cat([avg_weighted, max_weighted], dim=1)
        spatial_out = self.conv(spatial_descriptor)
        spatial_out = self.sigmoid(spatial_out)
        return spatial_out
class CCSAM(nn.Module):
    def __init__(self, in_planes, ratio=16):
        super(CCSAM, self).__init__()
        self.channel_attention = ChannelAttention(in_planes, ratio)
        self.spatial_attention = SpatialAttention()
```

```
    def forward(self, x):
        x = self.channel_attention(x) * x
        x = self.spatial_attention(x) * x
        return x
```

C. Proposed Model Summary

```
ResNet(
  (conv1): Conv2d(3, 64, kernel_size=(7, 7), stride=(2, 2), padding=(3, 3), bias=False)
  (bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
  (relu): ReLU(inplace=True)
  (maxpool): MaxPool2d(kernel_size=3, stride=2, padding=1, dilation=1, ceil_mode=False)
  (layer1): Sequential(
    (0): BasicBlock(
      (conv1): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
      (conv2): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (ccsam): CCSAM(
        (channel_attention): ChannelAttention(
          (avg_pool): AdaptiveAvgPool2d(output_size=1)
```

(max_pool): AdaptiveMaxPool2d(output_size=1)

(fc1): Conv2d(64, 4, kernel_size=(1, 1), stride=(1, 1), bias=False)

(relu): ReLU(inplace=True)

(fc2): Conv2d(4, 64, kernel_size=(1, 1), stride=(1, 1), bias=False)

(sigmoid): Sigmoid()

)

(spatial_attention): SpatialAttention(

(conv): Conv2d(2, 1, kernel_size=(7, 7), stride=(1, 1), padding=(3, 3))

(sigmoid): Sigmoid()

)))

……

(layer4): Sequential(

(0): BasicBlock(

(conv1): Conv2d(256, 512, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False)

(bn1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)

(relu): ReLU(inplace=True)

(conv2): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)

(ccsam): CCSAM(

(channel_attention): ChannelAttention(

(avg_pool): AdaptiveAvgPool2d(output_size=1)

```
        (max_pool): AdaptiveMaxPool2d(output_size=1)
        (fc1): Conv2d(512, 32, kernel_size=(1, 1), stride=(1, 1), bias=False)
        (relu): ReLU(inplace=True)
        (fc2): Conv2d(32, 512, kernel_size=(1, 1), stride=(1, 1), bias=False)
        (sigmoid): Sigmoid()
      )
      (spatial_attention): SpatialAttention(
        (conv): Conv2d(2, 1, kernel_size=(7, 7), stride=(1, 1), padding=(3, 3))
        (sigmoid): Sigmoid()
      ) )
    (downsample): Sequential(
      (0): Conv2d(256, 512, kernel_size=(1, 1), stride=(2, 2), bias=False)
      (1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    ) )
  (1): BasicBlock(
    (conv1): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
    (bn1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (relu): ReLU(inplace=True)
    (conv2): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
    (bn2): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (ccsam): CCSAM(
```

```
      (channel_attention): ChannelAttention(
        (avg_pool): AdaptiveAvgPool2d(output_size=1)
        (max_pool): AdaptiveMaxPool2d(output_size=1)
        (fc1): Conv2d(512, 32, kernel_size=(1, 1), stride=(1, 1), bias=False)
        (relu): ReLU(inplace=True)
        (fc2): Conv2d(32, 512, kernel_size=(1, 1), stride=(1, 1), bias=False)
        (sigmoid): Sigmoid()
      )
      (spatial_attention): SpatialAttention(
        (conv): Conv2d(2, 1, kernel_size=(7, 7), stride=(1, 1), padding=(3, 3))
        (sigmoid): Sigmoid()
      ) ) ) )
  (avgpool): AdaptiveAvgPool2d(output_size=1)
  (fc): Linear(in_features=512, out_features=256, bias=True)
  (dropout): Dropout(p=0.5, inplace=False)
  (fc2): Linear(in_features=256, out_features=5, bias=True)
  (softmax): Softmax(dim=1)
)
```

D. Classification report

The classification report values of Accuracy, Precision, Recall, and F1-Score for each individual classes by baseline and proposed model on both geometric and photometric augmentation.

	precision	recall	f1-score	support
0.0	0.99	1.00	0.99	500
1.0	1.00	1.00	1.00	480
2.0	0.99	1.00	1.00	448
3.0	0.98	0.97	0.98	400
4.0	0.99	0.97	0.98	302
accuracy			0.99	2130
macro avg	0.99	0.99	0.99	2130
weighted avg	0.99	0.99	0.99	2130

a) Fine-tuned ResNet18_ccsam on geometric aug'ed image

	precision	recall	f1-score	support
0.0	0.96	0.98	0.97	500
1.0	0.97	0.97	0.97	480
2.0	0.99	0.86	0.92	448
3.0	0.91	0.97	0.94	400
4.0	0.89	0.94	0.91	302
accuracy			0.95	2130
macro avg	0.94	0.94	0.94	2130
weighted avg	0.95	0.95	0.94	2130

b) Freezing layer Resnet18_ccsam on geometric

	precision	recall	f1-score	support
0.0	0.99	0.99	0.99	500
1.0	0.99	0.98	0.98	480
2.0	0.97	0.99	0.98	448
3.0	0.98	0.97	0.98	400
4.0	0.97	0.97	0.97	302
accuracy			0.98	2130
macro avg	0.98	0.98	0.98	2130
weighted avg	0.98	0.98	0.98	2130

c) Fine-tuned ResNet18_ccsam on photometric aug'ed image

	precision	recall	f1-score	support
0.0	0.92	0.99	0.95	500
1.0	0.99	0.98	0.99	480
2.0	0.99	0.86	0.92	448
3.0	0.95	0.95	0.95	400
4.0	0.85	0.94	0.90	302
accuracy			0.95	2130
macro avg	0.94	0.94	0.94	2130
weighted avg	0.95	0.95	0.95	2130

d)Freezing layer Resnet18_ccsam on photometric aug'ed image

E. Dataset Collection Approval papers