الجمهورية الجزائرية الديمقراطية الشعبية

MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

YAHIA FARES UNIVERSITY OF MEDEA

FACULTY OF TECHNOLOGY

Department of Electrical Engineering

End of Study Project

Sector: Telecommunication

Specialty: Telecommunications System

Theme

Design of an intelligent poultry disease detection system based on deep learning and IoT

Presented by :

CHOUAOU Marwa

HAMMANI Yasmine

Directed by:

Dr . BENYEZZA Hamza

Co-directed by :

Mr. ANIBA Yahia

Class 2024

THANKS

First and foremost, we extend our deepest gratitude to ALLAH, the Almighty, for giving us faith and enabling us to reach this milestone.

We are especially grateful to our supervisor, Mr. BENYEZZA Hamza, and our co-supervisor, Mr. ANIBA Yahya, for their guidance and the keen interest they have shown in our work.

We sincerely thank the members of the jury for kindly agreeing to be part of the examination committee.

Our heartfelt thanks go to all our teachers who have played a significant role in our education throughout our years of study.

we also thank our families for our encouragement and support

We also wish to express our appreciation to all our fellow students, with whom we had the pleasure of sharing this educational journey.

Lastly, we thank everyone who, in any way, contributed to the completion of this work.

DEDICATION

With deep gratitude, I dedicate this humble work to those to whom, regardless of the words used, I will never be able to fully express my sincere love.

ALLAH, the Almighty

For granting me the strength, knowledge, and perseverance to complete this work.

To My Father's Pure Spirit:

You are the guiding light that continues to illuminate my path, even in your absence. Your wisdom, love, and sacrifices have shaped me into the person I am today, and your memory will forever be cherished in my heart.

To the woman who endured without letting me feel the pain, who never denied my requests, and who spared no effort to bring happiness into my life**: My adorable mother**.

To my dear brothers Nabil and Walid, may Allah protect them and grant them happiness and success.

To all my teachers: I take this opportunity to express my respect and consideration.

To my partner Yassmine, with whom I shared joys and challenges during this project, may Allah fulfill your dreams and protect you.

I also dedicate this work to my precious friends: Bouchra, Fatima, Feriel, Radhia, Chorouk, Lina, Sanaa, Samia, Dalel, and Soumia, and to everyone who knows me; to my dear colleagues in the Systems telecommunication class, with whom I shared wonderful moments and beautiful memories.

To all those I love and those who love me Marwa

و آخر دعواهم ان الحمدلله رب العالمي

الى من بلغ الرسالة و أدى الامانة الى نبي الرحمة و نور العالمين سيدنا محمد صلى الله عليه و سلم

الى المعلم الاول الذي انار بضياء هداه بصائرنا و اخرجنا من ظلمات الجهل الى نور العلم

من قال انا لها "نالها " و انا لها و ان ابت رغما عنها اتيت بها

لم تكن الرحلة قصيرة ولا الطريق محفوفة بالتسهيلات و لكنني فعلتها فالحمدلله الذي يسر لنا البدايات و بلغنا النهايات

اهدي هذا النجاح لنفسي الطموحة اولا و الى كل من سعى معي لإتمام مسيرتي الدراسية

الى من احمل اسمه بكل عز و إفتخار ، لمن مهد لنا الطريق بجهده و تعبه الى تاج رأسي "أبي " حفظك الله و اطال في عمرك

الى العظيمة التي صنعت مني فتاة طموحة و غرست فيّ القيم و المبادئ ، الى من كان حلمها أن تعتلي منصة التخرج لكن الظروف منعتها اهديك تخرجي و نجاحي فلولاك لما كان ، الى ملكة قلبي "أمي " اطال الله في عمرك و أدامك شمعة لدربي

" الى الكتف الذي لا يميل و الضلع الثابت إخوتي "عبدالمالك " و " عبدالله " و الى قطعة قلبي اختي مؤنستي "ايناس

الى الرجل الذي اخرج اجمل مافي داخلي و شجعني للوصول "زوجي " ادامك الله سندا لا يميل ولا يمل

الى الايادي الطاهرة التي ساندتني في ضعفي أحبتي " منال" و "دلال" و الى القلب الذي غمرني ب دعاءه عزيزتي جدتي حفظك الله و كل اقربائي الذين وسعتهم ذاكرتي و لم تسعهم مذكرتي

"الى الارواح الغالية على قلبي صديقاتي " فريال " و "مروة" و " بشرى " و " فاطمة الزهراء

و الى اغلى و اعز شريكة نجاح صديقتي و اختي "مروة " ادام الله مابيننا في الخير

الى استانتي المخلصين الذين مروا علي لكن اثرهم لم يمر

الى كل من احبني و دعا لي بالسداد، الى زميلاتي و زملائي تخصص اتصالات سلكية و اللاسلكية انظمة اتصالات دفعة 2024

اهدي هذا العمل

ياسمين

ملخص

يقدم هذا المشروع تطوير نظام ذكي للكشف عن أمراض الدواجن، يجمع بين إنترنت الأشياء والتعلم العميق. يستخدم النظام أجهزة استشعار لجمع البيانات البيئية (درجة الحرارة والرطوبة) وكاميرا لمراقبة الدواجن في الوقت الفعلي، واكتشاف العلامات المبكرة للأمراض من خلال تحليل الصور الملتقطة باستخدام خوارزميات الشبكة العصبية الالتفافية

بالإضافة إلى ذلك، تم استخدام قرادیو و نودراد لإنشاء واجهة تفاعل بين الإنسان والآلة تفاعلية. يوفر قرادیو واجهة ويب سهلة الاستخدام لعرض نتائج تحليل ، بينما يتم استخدام نودراد لعرض البيانات البيئية والصور/الفيديوهات في الوقت الفعلي. يجمع هذا المزيج من التقنيات بين حل شامل وفعال لمراقبة وإدارة صحة الدواجن في الوقت الفعلي

الكلمات المفتاحية إنترنت الأشياء , التعلم العميق, الشبكة العصبية الالتفافية , واجهة التفاعل بين الإنسان والآلة , قرادیو ,نودراد

Abstract

This end-of-study project presents the development of an intelligent system for poultry disease detection, combining the Internet of Things (IoT) and Deep Learning. The system uses sensors to collect environmental data (temperature and humidity) and a camera to monitor the poultry in real time, detecting early signs of diseases by analyzing the captured images through Convolutional Neural Network (CNN) algorithm. Additionally, Gradio and Node-RED were utilized to create an interactive Human-Machine Interface (HMI). Gradio provides a user-friendly web interface for visualizing the results from the CNN analysis, while Node-RED is used to display the environmental data and real-time images/videos. This combination of technologies results in a comprehensive and efficient solution for real-time monitoring and management of poultry health.

Keywords: IoT, CNN, Deep learning, Node-RED, Gradio, HMI

Résumé

Ce projet de fin d'études présente le développement d'un système intelligent pour la détection des maladies chez la volaille, combinant l'Internet des Objets (IoT) et l'Apprentissage Profond. Le système utilise des capteurs pour collecter les données environnementales (température et humidité) et une caméra pour surveiller la volaille en temps réel, détectant les premiers signes de maladies en analysant les images capturées à l'aide d'algorithmes de Réseau Neuronal Convolutionnel (CNN). De plus, Gradio et Node-RED ont été utilisés pour créer une Interface Homme-Machine (IHM) interactive. Gradio fournit une interface web conviviale pour visualiser les résultats de l'analyse CNN, tandis que Node-RED est utilisé pour afficher les

données environnementales et les images/vidéos en temps réel. Cette combinaison de technologies aboutit à une solution complète et efficace pour la surveillance et la gestion en temps réel de la santé des volailles

Mot clé : internet des objets , apprentissage profond , réseau neuronal convolutif,IHM ,

Node-RED, Gradio

Summary

List of Figures
List of table

List of Abbreviations

General Introduction

I. Chapter I:General information on poultry diseases, Deep learning, CNN, and IO . 4

I.1 Introduction 4
I.2 Poultry Disease 4
I.2.1 Harms of Poultry Disease 11
I.2.2 Symptoms of Poultry Disease 13
I.3 Deep learning 14
I.3.1 Definition 14
I.3.2 Advantages of Deep Learning 15
I.3.3 Applications of deep learning 15
I.3.4 Types of deep learning 17
I.4 CNN 17
I.4.1 Introduction 17
I.4.2 Definition 18
I.4.3 Function of CNN 18
I.4.4 Application of CNN 20
I.4.5 CNN Architecture 21
I.5 Internet of Things 22
I.5.1 Definition 22
I.5.2 Characteristics of IoT 22
I.5.3 Architecture of IoT 23
I.5.4 Applications of IoT 25
I.5.5 Benefits of IoT 29
I.6 Conclusion 30

II. Chapter II: Hardware and Software Description 31

II.1 Introduction 31
II.2 Hardware 31
II.2.1 Cameras 31
II.2.2 Sensors 34

II.2.3 Raspberry 36
II.3 Software 45
II.3.1 Python 45
II.3.2 Gradio 46
II.3.3 Node-RED 48
II.3.4 Usefulness of Anaconda and Google Collab for Implementation 50
II.3.5 Predefined Model 52
II.3.6 Performances Criteria 57
II.4 Conclusion. 62
III. Chapter III: Implementation and Results 65
III.1 Introduction 65
III.2 Proposed Diagram for our system Poultry Diseases Detection 65
III.3 Dataset preparation 67
III.4 Model Selection and Implementation 68
III.4.1 VGG16 model 68
III.4.2 MobileNet 73
III.4.3 Comparison Results: VGG16 vs MobileNet 78
III.5 Human-Machine Interface (HMI) 79
III.6 Interactive HMI and Gradio Implementation for Poultry Disease Classification .. 79
III.6.1 Gradio 79
III.6.2 Node-RED 85
III.7 Conclusion 87
General conclusion

List of figure

Figure I.1: Respiratory diseases [4] **4**
Figure I.2: Gastrointestinal [4] **5**
Figure I.3: Reproductive diseases [5] **5**
Figure I.4: Neurological diseases [6] **6**
Figure I.5: External parasites [7] **6**
Figure I.6: Internal parasite [7] **7**
Figure I.7: New castle chicken diseases **7**
Figure I.8: Salmonella poultry diseases **9**
Figure I.9: Coccidiosis poultry diseases **10**
Figure I.10: Image recognition of chicken disease [10] **15**
Figure I.11: Natural language processing [11] **16**
Figure I.12: Graphical representation of finance.[12] **16**
Figure I.13: General architecture of CNN **21**
Figure I.14: Architecture of IoT [21] **24**
Figure I.15: Smart home [22] **25**
Figure I.16: Industrial IoT [22] **26**
Figure I.17: Smart Cities [22] **27**
Figure I.18: Smart healthcare [22] **28**
Figure I.19: Smart agriculture [23] **29**
Figure II.1: A thermal imaging camera is used to detect hot spots in commercial poultry houses [25] **32**
Figure *II.2*: Standard Vision Camera [26] **33**
Figure II.3: humidity and temperature sensor (DHT11) **34**
Figure II.4: Comparison between Raspberry Pi models in a table **37**
Figure II.5: Standard components of a Raspberry Pi **39**
Figure II.6: Raspberry Pi 4B Gpios **41**
Figure II.7: Connecting Peripherals **43**
Figure II.8: Raspberry Pi **44**
Figure II.9: Google Collab logo **50**
Figure II.10: anaconda Navigator **51**

Figure II.11: Basic VGG16 architecture [37] **54**
Figure II.12: Mobilenet architecture [39] **56**
Figure II.13:Roc Curve [40] **59**
Figure II.14:Model accuracy and loss[41] **60**
Figure II.15: General confusion matrix **61**
Figure III.1: Proposed system general architecture **66**
Figure III.2: Salmonella ..... **67** **Figure III.3: Coccidiosis**
Figure III.4: New Castle Disease **Figure III.5: healthy** **68**
Figure III.6: Accuracy and loss curves **69**
Figure III.7: Confusion matrix (VGG16) **70**
Figure III.8: ROC curve (VGG16) **72**
Figure III.9: Accuracy and loss curves (MobileNet) **73**
Figure III.10: Confusion matrix (MobileNet) **75**
Figure III.11: ROC curve (MobileNet) **76**
Figure III.12: Gradio results: second test (Coccidiosis) **83**
Figure III.13: Gradio results: fifth test (new castle disease) **84**
Figure III.14:example node-red IHM for temperature and humidity **85**

List of Tables

Table III-1: comparaison results(VGG16 and MobileNet).....78

List of Abbreviations

AI : artificial intelligent

API: application programming interface

AUC: Area under the curve

CNN: convolutional neural networks

DL: Deep learning

GPIO: General purpose input/output

GPU: a graphics processing unit

HDMI: high-definition multimedia interface

HTML: hypertext markup language

IoT: internet of things

IBD: infectious brutal disease

IBV: infectious bronchitis virus

IDE: integrated development environment

I2C:inter-integrad circuit

NLTK: natural language toolkit

NDV: New castle disease virus

NLP: Natural language processing

OvA: one-vs-all

OvO: one-vs-one

PWM: pulse width modulation

RNN: Recurrent neural network

ROC: Receiver operating characteristic

SPI: serial peripheral interface

UART: Universal asynchronous receiver/transmitter

UI: user interface

XML: extensible markup language

General Introduction

The significance of animal agriculture cannot be overlooked, especially in light of the expanding global population. The consumption of animal products plays a vital role in providing nutritionally rich meals and supporting the well-being of individuals of all ages in various communities worldwide. To meet the growing demand for animal proteins, the agricultural industry must prioritize enhancing efficiency and ramping up production levels[1].

Poultry is renowned for its abundant protein content and will likely see a surge in production as governments prioritize its output. This may increase densely populated poultry farms. In addition to its nutritional benefits, rearing poultry plays a crucial role in driving the socioeconomic progress of developing nations by providing a reliable supply of eggs and meat. This not only helps to secure household, regional, and national food and nutrition needs but also generates vital income for local communities. Poultry production contributes significantly to a country's gross domestic product, with an estimated contribution of over a hundred million dollars[2] [3].

As the poultry farming industry grows, it also brings the risk of spreading infectious diseases among birds. This can lead to devastating consequences, including high mortality rates and significant financial losses. The main contributing factors are inadequate biosecurity protocols, limited vaccination programs, and poor poultry management techniques. Additionally, the extensive production sector lacks essential veterinary interventions. Some of the most prevalent poultry diseases are fowl cholera, helminth infestations, salmonella infections, avian coccidiosis, and Newcastle disease. It is crucial to prioritize disease prevention and management strategies in poultry farming to protect the overall health of the birds.

Therefore, in this study, we propose an innovative solution based on DL and (IoT) technology to enhance disease detection in poultry farms. This proposal is a response to the challenges faced by the poultry farming industry, aiming to improve the efficiency of disease detection processes and mitigate the economic losses incurred by farms. The objective of this project is to develop an intelligent system capable of early identification of poultry health status and receiving data and signals from them over the internet, enabling farmers to take immediate preventive measures.

- **Problem statement**

poultry industry is a vital component of the global food supply chain, providing a significant source of protein. However, the industry faces significant challenges in disease detection, as

traditional methods have proven to be inefficient. This inefficiency leads to the spread of diseases, economic losses, and risks to food safety.

Among these methods, we find:

Visual Inspection: This type of examination relies on observing and noting poultry for any signs of illness with the naked eye. This includes looking for changes in the overall appearance of the poultry and their behavior, such as body shape, skin color, feather condition, eye movement, and sensitive areas. This type of inspection heavily depends on the expertise of the breeders and their careful observation.

Physiological Examination: This type of examination involves measuring the pulse rate, respiration rate, and body temperature of poultry to determine their health status and detect any signs of illness. This type of examination can provide important indicators of the overall health status of the flock.

Laboratory Tests: Laboratory tests involve taking samples of blood, feces, or tissues from poultry and analyzing them in the laboratory to accurately diagnose diseases. A variety of laboratory tests such as viral, bacterial, and parasitic tests are used to determine the causative agents of diseases.

These methods are characterized by ease, low cost, and widespread availability. However, they may be ineffective in accurately diagnosing some diseases and may require a long time to obtain results.

Furthermore, the limited real-time monitoring capabilities play a crucial role as they are one of the key challenges facing the poultry farming industry. This means that effective and immediate monitoring of poultry health and disease detection is difficult to achieve. This leads to delayed disease detection and increased spread, resulting in deteriorating poultry health and increased economic losses for farms.

These limited capabilities may be due to the following factors:

Lack of Technologies: There may be a lack of advanced monitoring technologies such as the (IoT) and artificial intelligence (AI).

Equipment Costs: The costs of the necessary hardware and equipment for real-time monitoring may be high and not affordable.

Training and Qualification: There may be a lack of training and qualification for workers in real-time poultry monitoring.

Weak Infrastructure: Some farms may face difficulties in establishing the necessary infrastructure for real-time monitoring, such as communication networks and cloud monitoring.

To improve real-time monitoring capabilities, innovative and effective technological solutions must be developed and implemented, such as smart monitoring systems based on IoT and AI technologies. These solutions can help improve early disease detection and reduce its spread, thereby enhancing poultry health and achieving sustainability in the poultry farming industry

Therefore, there is an urgent need for an AI system that can address these challenges and enhance disease detection capabilities in poultry farms effectively. By utilizing real-time monitoring technologies and data analysis, this system can contribute to preventing the spread of diseases and reducing the negative impacts on poultry health, human health, and the agricultural economy.

Objectives of the Project

This research aims to design an intelligent system for detecting diseases in poultry using deep learning techniques and the (IoT). This system monitors the health of poultry in real time and detects early signs of diseases. By leveraging smart cameras and IoT sensors, data on the behavior and appearance of poultry are collected. Subsequently, this data is analyzed using deep learning algorithms to detect any abnormal changes. The project aims to improve poultry health, reduce economic losses for breeders, and enhance the management of avian diseases.

Organization of the thesis

This research consists of four chapters organized as follows:

- In the first chapter, we present some fundamental concepts related to the IOT, DL, CNN, and chicken diseases, and the relationship between them.
- In the second chapter, we detail the hardware and software tools we used.
- In the third chapter, we detail the implementation and results of our project on detecting chicken diseases using IoT and deep learning. We outline the integration of various hardware components, including IoT sensors for real-time data collection and high-performance GPUs for efficient model training

And finally, we will conclude this thesis with a general conclusion and future perspectives

I. Chapter I: General information on poultry diseases, Deep learning, CNN, and IoT

I.1 Introduction

Poultry farming is vital for global food production, but diseases like avian influenza and Newcastle disease can cause significant economic losses. The integration of deep learning and the Internet of Things (IoT) offers innovative solutions to this problem. IoT devices, such as sensors and cameras, collect real-time data on environmental conditions and chicken behavior. Convolutional Neural Networks (CNNs), a type of deep learning algorithm, analyze this data to detect early signs of disease. This technology allows farmers to take swift action, improving the health and productivity of poultry farms.

I.2 Poultry Disease

poultry diseases can have significant impacts on poultry farming, leading to economic losses and affecting food production. Here's some general information about poultry diseases:

- **Respiratory Diseases:** Respiratory infections are common in poultry and can be caused by various pathogens, including bacteria, viruses, and fungi. Examples of respiratory diseases include infectious bronchitis, Newcastle disease, avian influenza, and infectious laryngotracheitis. These diseases can cause respiratory distress, coughing, sneezing, nasal discharge, and decreased egg production. Figure I-1 represents respiratory chicken disease.



Figure I.1: Respiratory diseases [4]

- ✓ **Gastrointestinal Diseases:** Gastrointestinal disorders are also prevalent in poultry and can result from bacterial, viral, or parasitic infections. Diseases such as coccidiosis, infectious bursal disease (Gumboro disease), necrotic enteritis, and infectious coryza can affect the digestive system, leading to diarrhea, dehydration, weight loss, and decreased feed intake. Here is Figure I-2 which represents gastrointestinal poultry disease



Figure I.2: **Gastrointestinal** [4]

- ✓ **Reproductive Disorders:** Disorders affecting the reproductive system can lead to decreased fertility and hatchability rates, as well as poor egg quality. Common reproductive diseases in poultry include infectious bronchitis virus (IBV), infectious bursal disease (IBD), egg drop syndrome, and avian influenza. These diseases can cause egg abnormalities, such as soft-shelled eggs, shell-less eggs, or misshapen eggs. Here is the figure I-3 that represents reproductive chicken disease.



Figure I.3: Reproductive diseases [5]

- ✓ **Neurological Diseases:** Neurological disorders can affect the central nervous system of poultry, leading to symptoms such as paralysis, tremors, ataxia, and convulsions. Examples of neurological diseases include Marek's disease, avian encephalomyelitis, botulism, and avian leukosis. These diseases can have significant welfare implications and may require euthanasia of affected birds. Figure I-4 represents neurological chicken disease.

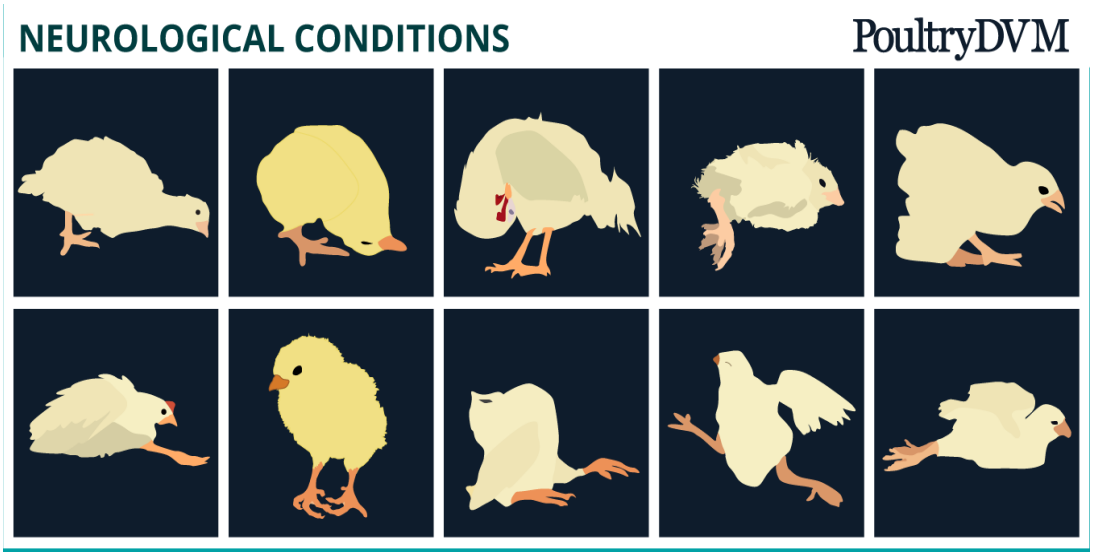


Figure I.4: Neurological diseases [6]

- ✓ **External Parasites:** Poultry are susceptible to infestations by external parasites such as lice, mites, fleas, and ticks. These parasites can cause irritation, feather loss, skin lesions, anemia, and decreased productivity. Regular monitoring and appropriate treatment are essential for controlling parasite infestations in poultry flocks. Here is the figure I-5 that represents external parasites in chicken disease.



Figure I.5: External parasites [7]

- ✓ **Internal Parasites:** Internal parasites, including worms (such as roundworms, tapeworms, and cecal worms) and protozoa (such as coccidia), can affect the gastrointestinal tract of poultry. Parasitic infections can lead to poor growth, weight loss, decreased egg production, and increased susceptibility to other diseases. Proper management practices, including deworming and sanitation, are crucial for controlling internal parasite infections in poultry. Here is figure I-6 represents internal parasites in chicken disease.



Figure I.6: Internal parasite [7]

- **New castle chicken diseases:** Newcastle disease is a highly contagious viral disease that affects birds, including chickens. It is caused by the Newcastle disease virus (NDV), which is a paramyxovirus. The disease can vary in severity from mild to highly lethal, depending on the strain of the virus and the susceptibility of the bird species. Here's an overview of Newcastle disease, its impact, and management strategies. As shown in the figure I-7



Figure I.7: New castle chicken diseases

- **Symptoms**

Newcastle disease can manifest in various forms, depending on the strain of the virus and the age and species of the bird. The symptoms can be classified into three main forms:

- ✓ **Velogenic (highly virulent):** Respiratory symptoms: Gasping, coughing, and sneezing.
- ✓ **Nervous symptoms:** Tremors, paralysis of wings and legs, twisting of the head and neck, and complete paralysis.
- ✓ **Digestive symptoms:** Greenish, watery diarrhea.
- ✓ **General symptoms:** Severe drop in egg production, swelling of tissues around the eyes and neck, sudden death.
- ✓ **Mesogenic (moderately virulent):** Mild respiratory and nervous symptoms.

Reduced egg production.

- ✓ **Lentogenic (low virulence):** Mild or inapparent respiratory symptoms.

Often used as live vaccines due to their mild nature.

- **Transmission**

- ✓ **Direct contact:** Infected birds spread the virus through secretions from the respiratory tract, feces, and contaminated feed, water, equipment, and clothing.
- ✓ **Airborne spread:** The virus can be transmitted through the air, especially in densely populated bird areas.
- ✓ **Wild birds:** Wild birds can carry and spread the virus without showing symptoms.

- **Salmonella chicken diseases:** Salmonella infections in poultry are caused by various serotypes of the bacterium Salmonella, which can lead to illness in both poultry and humans. Poultry can be asymptomatic carriers of Salmonella, posing a risk to food safety through contaminated eggs and meat. Here's a detailed overview of Salmonella infections in chickens As shown in Figure I-8



Figure I.8: Salmonella poultry diseases

- **Symptoms**

Poultry infected with Salmonella can exhibit a range of symptoms, but they can also be asymptomatic carriers. When symptoms do occur, they may include:

General Symptoms

- ✓ Weakness and lethargy
- ✓ Decreased feed intake
- ✓ Poor growth or weight loss
- ✓ Ruffled feathers

Gastrointestinal Symptoms

- ✓ Diarrhea, which may be greenish in color
- ✓ Increased thirst

Reproductive Symptoms

- ✓ Drop in egg production
- ✓ Production of misshapen or discolored eggs

- **Transmission**

- ✓ **Horizontal Transmission:** The most common way Salmonella spreads in poultry is through fecal-oral contamination. Birds can ingest the bacteria through contaminated feed, water, or the environment.

- ✓ **Vertical Transmission:** Infected hens can transmit Salmonella to their eggs, leading to infected chicks.
- ✓ **Human Interaction:** Humans can spread Salmonella to birds through contaminated equipment, clothing, or hands.

- **Coccidiosis poultry diseases:** Coccidiosis is a common and significant parasitic disease in poultry caused by protozoan parasites of the genus Eimeria. These parasites invade the intestinal lining of poultry, leading to various health problems. Coccidiosis is highly contagious and can cause substantial economic losses in poultry production due to decreased growth rates, poor feed conversion, and increased mortality. As shown in figure I-9



Figure I.9: Coccidiosis poultry diseases

➢ **Symptoms**

The symptoms of coccidiosis in poultry can vary depending on the Eimeria species involved and the severity of the infection. Common symptoms include:

✓ **Intestinal Signs**

- Diarrhea, which may contain blood or mucus.
- Droppings that are loose or watery.
- Presence of undigested feed in the droppings.

✓ **General Signs**

- Weight loss or poor weight gain.
- Decreased feed and water intake.
- Lethargy and depression.
- Ruffled feathers.
- Pale comb and wattles due to anemia.

✓ **Severe Cases**

- Sudden death, particularly in young chicks.
- Severe dehydration and emaciation.

➢ **Transmission**

✓ **Fecal-Oral Route:** Coccidiosis spreads through the ingestion of sporulated oocysts (the infective stage of Eimeria) present in contaminated feed, water, litter, or the environment.

✓ **Contaminated Environment:** High stocking densities, poor sanitation, and wet litter conditions favor the survival and spread of oocysts.

Poultry farmers need to work closely with veterinarians and extension specialists to develop comprehensive disease prevention and control strategies tailored to their specific production systems and environmental conditions.

I.2.1 Harms of Poultry Disease

Poultry diseases can cause various harms, impacting both the poultry industry and consumers. Here are some of the main harms associated with poultry diseases:

✓ **Economic Losses:** Poultry diseases can lead to significant economic losses for poultry farmers due to reduced productivity and increased mortality rates. Decreased egg production, poor growth rates, and losses in the market value of affected birds can result in financial hardships for poultry producers.

✓ **Reduced Food Supply:** Disease outbreaks in poultry farms can reduce the availability of poultry products such as eggs and meat in the market. This can lead to shortages of poultry products, which may result in higher prices for consumers and food insecurity for communities that rely on chicken as a primary protein source.

- ✓ **Animal Suffering:** Poultry diseases can cause suffering and distress to affected birds. Respiratory infections, gastrointestinal disorders, and other diseases can lead to pain, discomfort, and impaired welfare for poultry, compromising their quality of life.

- ✓ **Zoonotic Transmission:** Some poultry diseases have the potential to spread to humans, posing risks to public health. Zoonotic pathogens transmitted from poultry to humans can cause illnesses such as salmonellosis, campylobacteriosis, and avian influenza, leading to human health problems and medical expenses.

- ✓ **Trade Disruptions:** Disease outbreaks in poultry-producing regions can result in trade disruptions and restrictions imposed by importing countries to prevent the spread of pathogens. These trade barriers can affect international poultry trade, leading to economic losses for exporting countries and impacting global food supply chains.

- ✓ **Biosecurity Risks:** Chicken diseases can pose biosecurity risks by increasing the likelihood of pathogen transmission within and between poultry farms. Poor biosecurity practices can contribute to the spread of diseases among poultry populations, necessitating costly control measures and interventions.

- ✓ **Environmental Impact**: Disease management practices in poultry farming, such as the use of antimicrobials and disinfectants, can have environmental consequences. Improper disposal of poultry waste and runoff from farms can contribute to water and soil pollution, affecting ecosystems and human health.

- ✓ **Social and Community Impact:** Poultry diseases can have social and community impacts, particularly in rural areas where poultry farming is a significant source of livelihood. Disease outbreaks can disrupt local economies, strain healthcare systems, and erode community cohesion and well-being.

I.2.2 Symptoms of Poultry Disease

The symptoms of poultry diseases can vary depending on the specific pathogen involved, the affected organ systems, and the severity of the illness. However, some common symptoms of poultry diseases include:

I.2.2.1 Respiratory Symptoms

- Coughing
- Sneezing
- Nasal discharge
- Labored breathing or gasping
- Wheezing or rattling sounds

I.2.2.2 Gastrointestinal Symptoms

- Diarrhea
- Watery or bloody droppings
- Vomiting
- Decreased appetite or anorexia
- Weight loss or poor growth

I.2.2.3 Neurological Symptoms

- Lack of coordination (ataxia)
- Seizures or convulsions
- Head tilting or twisting

I.2.2.4 Reproductive Symptoms

- Abnormalities in egg quality (soft shells, shell-less eggs)
- Reproductive tract infections
- Egg peritonitis (accumulation of fluid in the abdomen)

I.2.2.5 Behavioral Changes

- Lethargy or depression

- Huddling or puffing up feathers
- Isolation or avoidance of other birds
- Aggression or unusual pecking behavior
- Excessive vocalization or distress calls

I.2.2.6 External Signs

- Swelling or inflammation of the eyes, comb, or wattles
- Lesions or sores on the skin or mucous membranes
- Feather loss or ruffled feathers
- Abnormalities in the appearance or color of feathers
- Lumps or abscesses under the skin

I.2.2.7 General Signs of Illness

- Reduced activity or lethargy
- Weight loss or emaciation
- Pale comb and wattles (indicating anemia)
- Dehydration (sunken eyes, dry and tacky mouth)
- Poor grooming or unkempt appearance

I.3 Deep learning

I.3.1 Definition

DL is a form of machine learning and (AI) that mimics the way humans acquire certain types of knowledge. DL models can be trained to perform classification tasks and identify patterns in photos, text, audio, and other miscellaneous data. It is also used to automate tasks that typically require human intelligence, such as describing images or transcribing audio files.

DL is an important element of data science, including statistics and predictive modeling. This is extremely beneficial for data scientists who need to collect, analyze, and interpret large amounts of data. DL makes this process faster and easier.

The human brain has millions of interconnected neurons that work together to learn information, while deep learning has neural networks built from multiple layers of collaborating software nodes. Use large amounts of labeled data and neural network architectures to train DL models[8].

I.3.2 Advantages of Deep Learning

High accuracy: Deep Learning algorithms can achieve state-of-the-art performance in various tasks, such as image recognition and natural language processing.

Automated feature engineering: Deep Learning algorithms can automatically discover and learn relevant features from data without the need for manual feature engineering.

Scalability: Deep Learning models can scale to handle large and complex datasets, and can learn from massive amounts of data.

Flexibility: Deep Learning models can be applied to a wide range of tasks and can handle various types of data, such as images, text, and speech.

Continual improvement: Deep Learning models can continually improve their performance as more data becomes available[9].

I.3.3 Applications of deep learning

DL can be used in a wide variety of applications, including:

- Image recognition: To identify objects and features in images, such as people, animals, places, etc. as Figure II-10 is represented below image recognition on chicken disease.

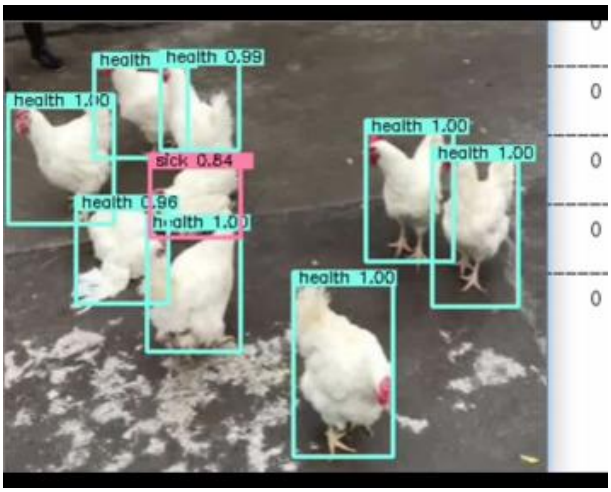


Figure I.10: Image recognition of chicken disease [10]

- **Natural language processing:** To help understand the meaning of text, such as in customer service chatbots and spam filters. as figure I-11 is represented below nlp.

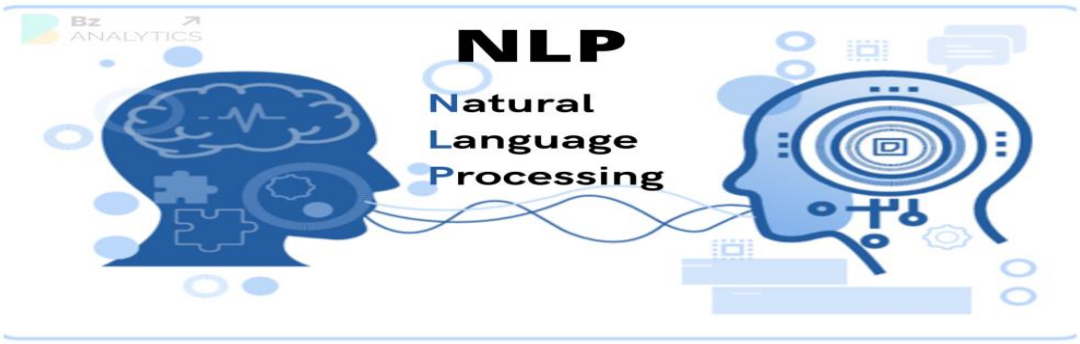


Figure I.11: Natural language processing [11]

- **Finance:** To help analyze financial data and make predictions about market trends. figure I-12 is represented below a graphical representation of finance.

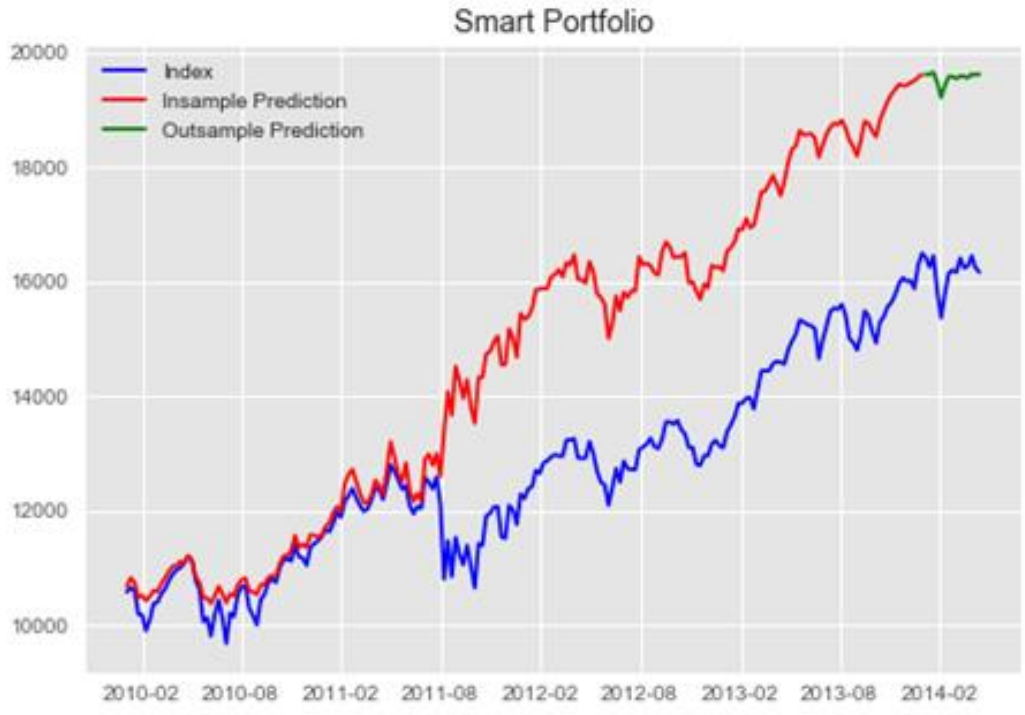


Figure I.12: Graphical representation of finance.[12]

- **Text to image:** Convert text into images, such as in the Google Translate app[13].

I.3.4 Types of deep learning

There are many different types of DL models. Some of the most common types include:

- ✓ **Convolutional neural networks (CNNs)**

CNNs are used for image recognition and processing. They are particularly good at identifying objects in images, even when those objects are partially obscured or distorted.

- ✓ **Deep reinforcement learning**

Deep reinforcement learning is used for robotics and game playing. It is a type of machine learning that allows an agent to learn how to behave in an environment by interacting with it and receiving rewards or punishments.

- ✓ **Recurrent neural networks (RNNs)**

RNNs are used for natural language processing and speech recognition. They are particularly good at understanding the context of a sentence or phrase, and they can be used to generate text or translate languages[8].

I.4 CNN

I.4.1 Introduction

For many years in the society, farmers have relied on experts to diagnose and detect chicken diseases. As a result, farmers lose many domesticated birds due to late diagnoses or a lack of reliable experts. With the available tools from AI and machine learning based on computer vision and image analysis, the most common diseases affecting chickens can be identified easily from the images of chicken droppings. In this study, we propose a DL solution based on (CNN) to predict whether the feces of chicken belong to either of the three classes. We also leverage the use of pre-trained models and develop a solution for the same problem. Based on the comparison, we show that the model developed from the XceptionNet outperforms other models for all metrics used. The experimental results show the apparent gain of transfer learning (validation accuracy of 94% using pretraining over its contender 93.67% developed CNN from full training on the same dataset). In general, the developed fully trained CNN comes second

when compared with the other model. The results show that the pre-trained XceptionNet method has overall performance and highest prediction accuracy, and can be suitable for chicken disease detection applications [14].

I.4.2 Definition

A (CNN) is a category of machine learning model, namely a type of DL algorithm well suited to analyzing visual data. CNNs -- sometimes referred to as convents -- use principles from linear algebra, particularly convolution operations, to extract features and identify patterns within images. Although CNNs are predominantly used to process images, they can also be adapted to work with audio and other signal data.

CNN architecture is inspired by the connectivity patterns of the human brain -- in particular, the visual cortex, which plays an essential role in perceiving and processing visual stimuli. The artificial neurons in a CNN are arranged to efficiently interpret visual information, enabling these models to process entire images. Because CNNs are so effective at identifying objects, they are frequently used for computer vision tasks such as image recognition and object detection, with common use cases including self-driving cars, facial recognition, and medical image analysis.

Unlike CNNs, older forms of neural networks often needed to process visual data in a piecemeal manner, using segmented or lower-resolution input images. A CNN's comprehensive approach to image recognition lets it outperform a traditional neural network on a range of image-related tasks and, to a lesser extent, speech and audio processing[15].

I.4.3 Function of CNN

Networks (CNNs) serve several functions in the context of machine learning and computer vision tasks. Here are some of the key functions of CNNs:

- **Feature Extraction:** CNNs excel at automatically learning and extracting hierarchical representations of features from input data, particularly in the case of images. Through a series of convolutional and pooling layers, CNNs detect and capture various low-level and high-level features such as edges, textures, shapes, patterns, and objects present in the input images.

- **Pattern Recognition:** CNNs are capable of recognizing complex patterns and structures within visual data. By learning from large datasets, CNNs can identify specific objects, scenes, or concepts present in images, making them highly effective for tasks like image classification, object detection, and image segmentation.

- **Localization:** CNNs can accurately localize objects within images by predicting bounding boxes or segmentation masks around objects of interest. This capability is essential for tasks such as object detection, where the network must not only classify objects but also precisely locate their positions within the image.

- **Classification:** One of the primary functions of CNNs is image classification, where the network assigns a label or category to an input image based on its contents. CNNs learn to associate specific features extracted from the input image with predefined classes or categories, allowing them to classify images into different classes with high accuracy.

- **Detection and Recognition:** CNNs can detect and recognize various objects, faces, text, or other visual elements within images. This function is critical for applications such as facial recognition, scene understanding, optical character recognition (OCR), and visual search, where the network needs to identify specific entities or attributes within the input data.

- **Segmentation:** CNNs can perform semantic segmentation, which involves partitioning an image into meaningful segments or regions and assigning each pixel a class label. This function enables tasks such as image segmentation, where the network delineates different objects or regions within an image, providing detailed pixel-level information about the scene.

- **Feature Visualization and Interpretation:** CNNs can help visualize and interpret the learned features and representations within images, shedding light on how the network perceives and processes visual information. Techniques such as feature visualization, activation maximization, and gradient-based attribution methods can provide insights into the inner workings of CNNs and improve model understanding.

Overall, CNNs play a crucial role in various computer vision tasks, offering powerful capabilities for feature extraction, pattern recognition, object detection, and image understanding, among others. Their versatility and effectiveness have made them indispensable tools in fields such as image analysis, medical imaging, autonomous vehicles, robotics, and many others.

I.4.4 Application of CNN

Because processing and interpreting visual data is such a common task, CNNs have a wide range of real-world applications, from healthcare and automotive to social media and retail.

Some of the most common fields in which CNNs are used include the following:

• Healthcare. In the healthcare sector, CNNs are used to assist in medical diagnostics and imaging. For example, a CNN could analyze medical images such as X-rays or pathology slides to detect anomalies indicative of disease, thereby aiding in diagnosis and treatment planning.

• Automotive. The automotive industry uses CNNs in self-driving cars that navigate their environments by interpreting camera and sensor data. CNNs are also useful in AI-powered features of nonautonomous vehicles, such as automated cruise control and parking assistance.

• Social media. On social media platforms, CNNs are employed in a range of image analysis tasks. For example, a social media company might use CNN to suggest people tag in photographs or to flag potentially offensive images for moderation.

• Retail. E-commerce retailers use CNNs in visual search systems that let users search for products using images rather than text. Online retailers can also use CNNs to improve their recommender systems by identifying products that visually resemble those a shopper has shown interest in.

• Virtual assistants. Although CNNs are most often used to work with image data, virtual assistants are a good example of applying CNNs to audio processing problems. CNNs can

recognize spoken keywords and help interpret users' commands, enhancing a virtual assistant's ability to understand and respond to its user[15].

I.4.5 CNN architecture

CNN consists of multiple layers like the input layer, Convolutional layer, Pooling layer, and fully connected layers.

Simple CNN architecture in Figure I-13

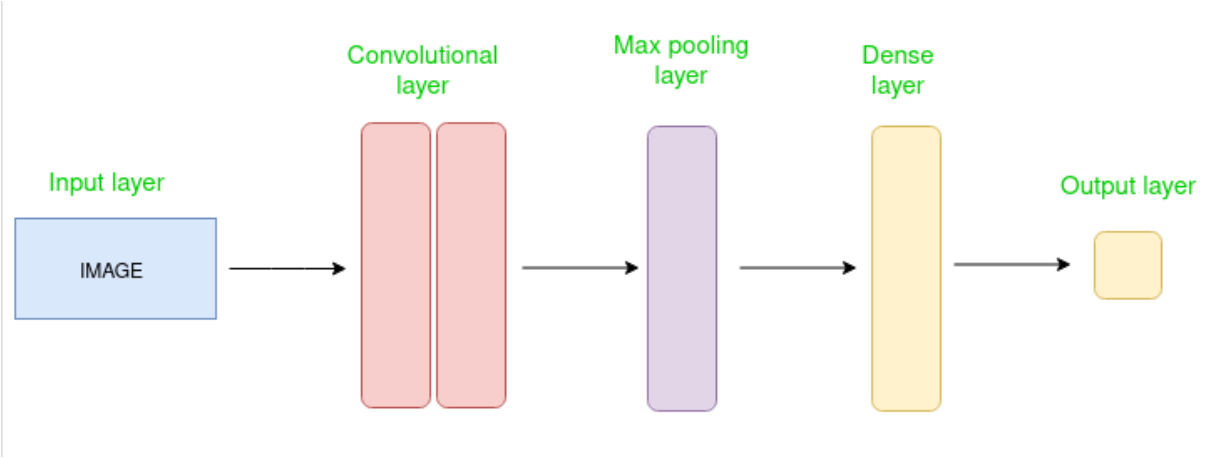


Figure I.13: General architecture of CNN

The Convolutional layer applies filters to the input image to extract features, the Pooling layer downsamples the image to reduce computation, and the fully connected layer makes the final prediction. The network learns the optimal filters through backpropagation and gradient descent [16].

- **Input Layers:** It's the layer in which we give input to our model. In CNN, Generally, the input will be an image or a sequence of images. This layer holds the raw input of the image with width 32, height 32, and depth 3.

- **Convolutional Layers:** This is the layer, which is used to extract the feature from the input dataset. It applies a set of learnable filters known as the kernels to the input images. The filters/kernels are smaller matrices usually 2×2, 3×3, or 5×5 shape. it slides over the input image data and computes the dot product between kernel weight and the corresponding input image

patch. The output of this layer is referred ad feature maps. Suppose we use a total of 12 filters for this layer we'll get an output volume of dimension 32 x 32 x 12.

- **Pooling layer:** This layer is periodically inserted in the covnets and its main function is to reduce the size of volume which makes the computation fast reduces memory and also prevents overfitting. Two common types of pooling layers are max pooling and average pooling. If we use a max pool with 2 x 2 filters and stride 2, the resultant volume will be of dimension 16x16x12.

- **Output Layer:** The output from the fully connected layers is then fed into a logistic function for classification tasks like sigmoid or softmax which converts the output of each class into the probability score of each class.

I.5 Internet of Things

I.5.1 Definition

The (IoT) is a modern phenomenon that encompasses the interconnection of everyday objects, or "things," through the use of sensors, software, and other innovative technologies. This interconnected network allows for the exchange of data between devices and systems via the Internet. These devices can range from common household items to complex industrial tools. With a current 7 billion connected IoT devices and projections of 10 billion by 2020 and 22 billion by 2025, the growth and impact of IoT is undeniable.

IoT has swiftly emerged as a crucial technology in the 21st century. With the ability to link ordinary items such as kitchen gadgets, vehicles, thermostats, and even baby monitors to the internet through embedded devices, a seamless exchange of information is now within reach, connecting people, procedures, and objects [17].

I.5.2 Characteristics of IoT

I.5.2.1 Interconnectivity

Everything can be interconnected with the global infrastructure of information and communication.

I.5.2.2 Object-related services

IoT can offer services related to objects within the limitations of these objects, including privacy protection and semantic consistency between physical objects and their associated virtual counterparts [18].

I.5.2.3 Heterogeneity

IoT devices are heterogeneous because they are based on different hardware platforms and networks. They can interact with other devices or service platforms via different networks [19].

I.5.2.4 Dynamic changes

Dynamic changes refer to the fluid alterations in the state of devices, such as transitioning between sleep and wake modes, connectivity status (connected or disconnected), and contextual variables like location and speed. Moreover, the quantity of devices in operation can vary dynamically[20].

I.5.2.5 Large-scale

The number of devices that need to be managed and communicate with each other will be at least an order of magnitude larger than those connected to the current internet. Managing the generated data and interpreting it for application purposes will be even more critical. This involves data semantics as well as efficient data processing

I.5.2.6 Connectivity

IoT devices are interconnected through various communication protocols such as Wi-Fi, Bluetooth, Zigbee, and cellular networks, enabling seamless data exchange.

I.5.2.7 Security and Privacy

IoT devices and networks implement robust security measures to protect data privacy and prevent unauthorized access, ensuring the integrity and confidentiality of information.

I.5.3 Architecture of IoT

The architecture of the (IoT) typically consists of several layers, each serving specific functions to enable communication, data processing, and interaction between connected devices. While there can be variations in IoT architectures based on specific applications and requirements, a typical architecture includes the following layers:

Perception Layer: This layer comprises sensors, actuators, and other physical devices that collect data from the environment or interact with it. Sensors detect changes in the physical world, such as temperature, humidity, or motion, while actuators enable devices to perform actions based on received instructions.

Network Layer: The network layer facilitates communication between IoT devices and enables data transmission over various communication protocols such as Wi-Fi, Bluetooth, Zigbee, or

cellular networks. This layer ensures reliable and secure connectivity between devices and the IoT infrastructure.

Middleware Layer: The middleware layer acts as an intermediary between the perception layer and the application layer, providing services such as data filtering, aggregation, protocol translation, and device management. It helps in managing the complexity of IoT deployments and ensures interoperability between different devices and protocols.

Application Layer: The application layer is where data collected from IoT devices is processed, analyzed, and utilized to derive actionable insights or perform specific tasks. It includes applications, services, and analytics platforms that enable monitoring, control, automation, and decision-making based on IoT data. As shown in figure I-14

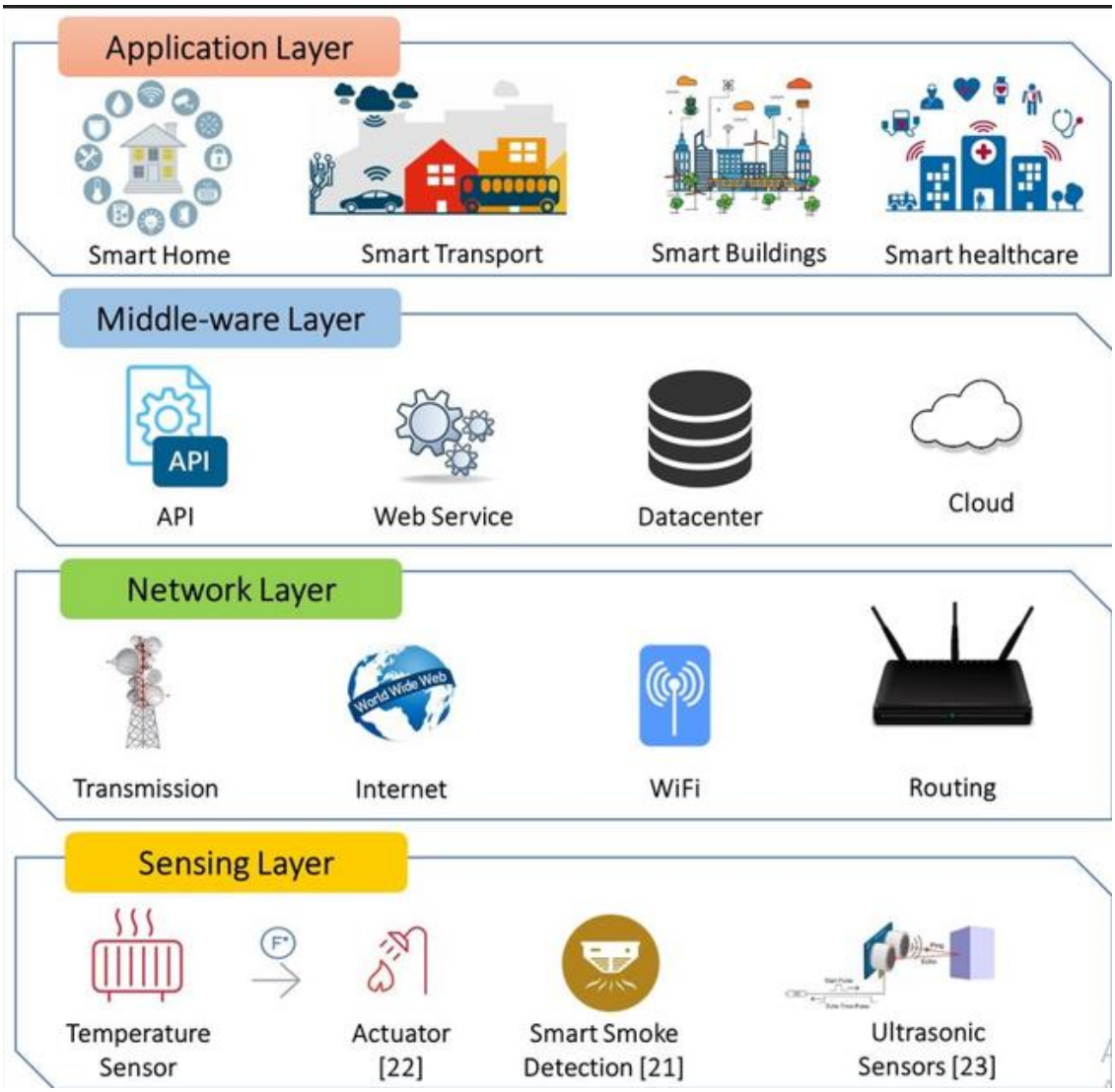


Figure I.14: Architecture of IoT [21]

I.5.4 Applications of IoT

Smart Home Automation: IoT enables the automation and remote control of various home devices such as thermostats, lights, locks, and security cameras. Figure I-15 illustrates that



Figure I.15: Smart home [22]

Industrial IoT (IIoT): In industries, IoT facilitates predictive maintenance, asset tracking, inventory management, and process optimization, leading to improved efficiency and cost savings. As shown in figure I-16



Figure I.16: Industrial IoT [22]

Smart Cities: IoT technologies are used to manage urban infrastructure, including traffic lights, waste management systems, public transportation, and utilities, to create sustainable and efficient urban environments. As shown in figure I-17

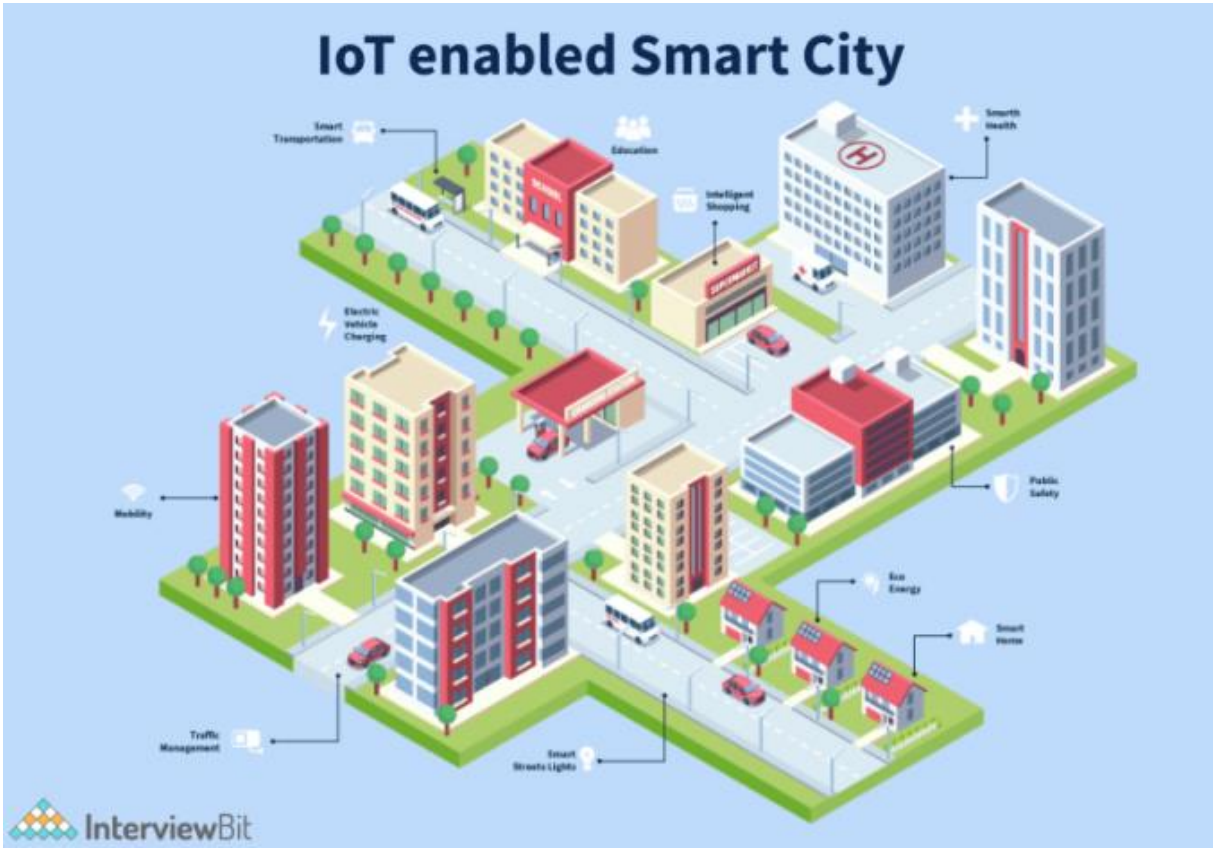


Figure I.17: Smart Cities [22]

Healthcare: IoT devices monitor patients' health remotely, track medical assets, manage inventories, and enhance the efficiency of healthcare delivery systems. As shown in the figure I-18

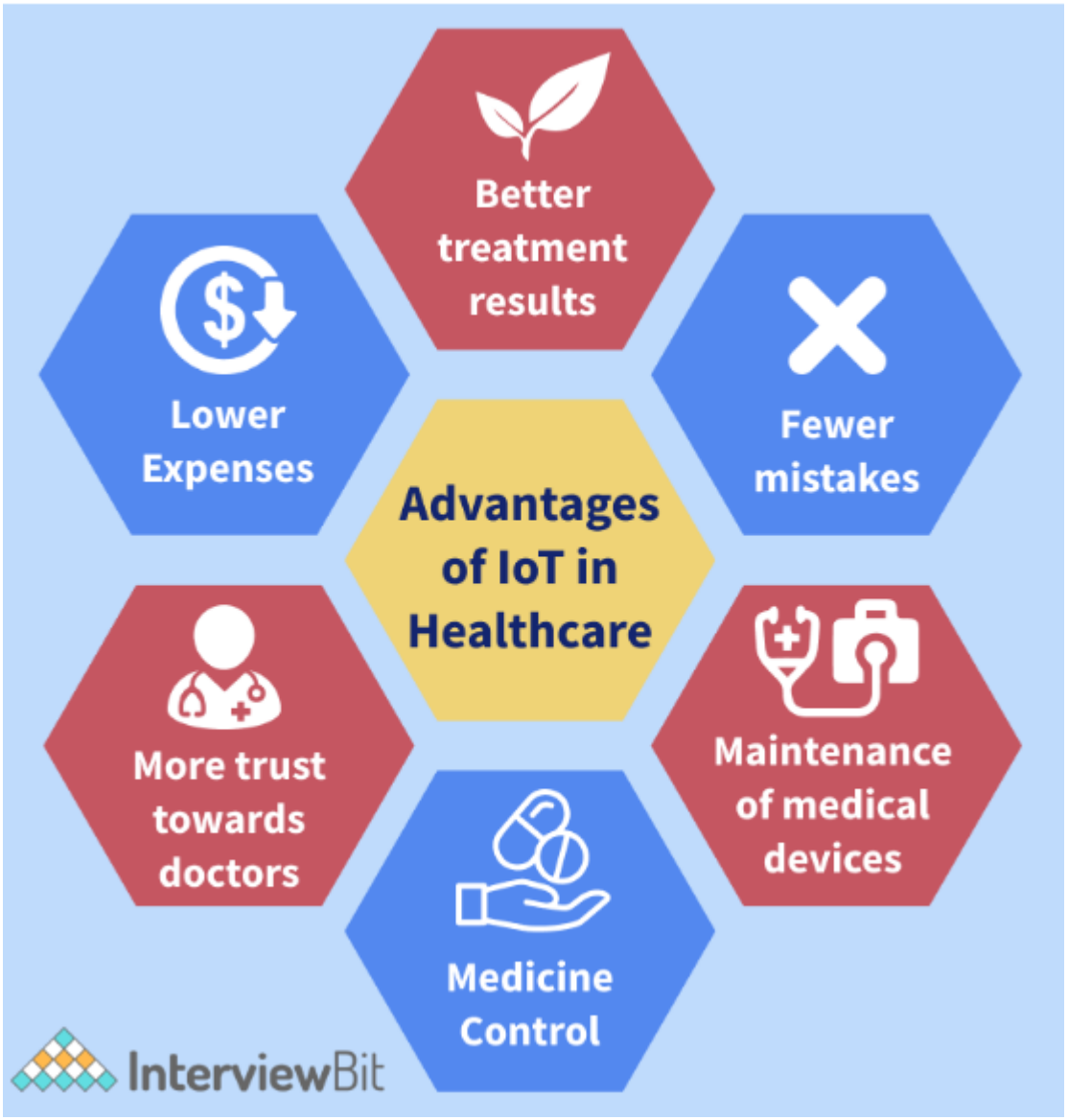


Figure I.18: Smart healthcare [22]

Agriculture: IoT solutions in agriculture include precision farming techniques, crop monitoring, livestock tracking, and environmental monitoring to optimize resource utilization and improve yields. As shown in figure I-19

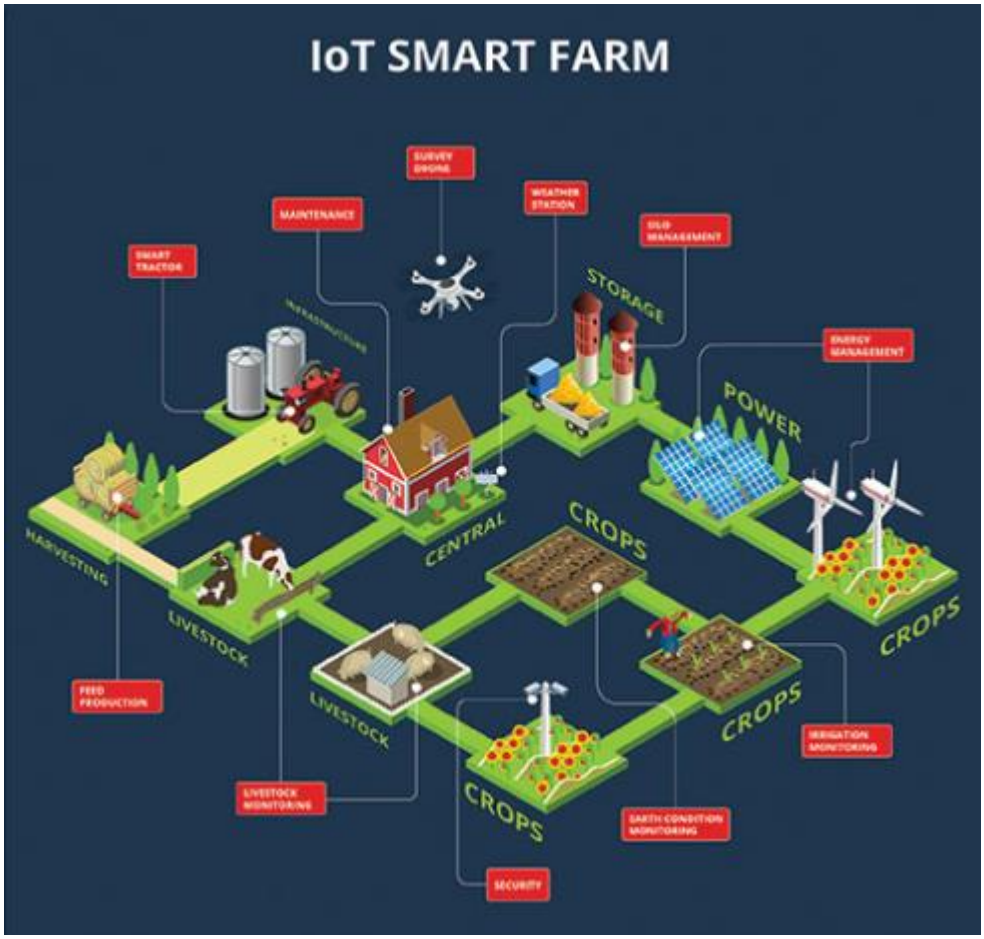


Figure I.19: Smart agriculture [23]

I.5.5 Benefits of IoT

IoT has numerous benefits and perks that can make the lives of businesses and consumers easier. Voicing a few of them:

- Having access to the IOT allows people to work smarter, live smarter, and better manage their lives.
- Aside from automating the home, IoT is critical to business. Using IoT data analytics, companies can gain deep insights into everything from the performance of machines to supply chain and logistics operations in real-time.
- By automating tasks, businesses can provide better services to their clients while lessening the need for human involvement.
- It also improves service delivery, which makes manufacturing and delivering goods less expensive.

- As a result of IoT, companies are forced to rethink how they run their business and to better formulate their strategies.
- Streamlines communication between electronic devices.
- A traditional data collection system has its limitations and is designed for passive usage, whereas an IoT system enables immediate action on data.
- With IoT, you can detect problems and improve processes to enhance customer experience.
- Through IoT, the access control system is capable of providing additional security to an organization and the general public. Incorporating IoT into surveillance can help to increase security in an organization and find any suspicious activity.

I.6 Conclusion

This research aims to develop a smart system that relies on (AI) and (IoT) to achieve early detection of diseases in poultry farms. The system involves using smart sensors and web cameras connected to an IoT network to monitor poultry health in real-time. Data collected from these devices is analyzed using DL techniques to identify any abnormal changes in the behavior or appearance of the poultry that may indicate the presence of diseases. Through this smart system, poultry can be accurately and rapidly monitored, enabling the detection of diseases at early stages and the implementation of necessary measures to prevent their spread, safeguard poultry health, and mitigate economic losses for farms

II. Chapter II: Hardware and Software Description

II.1 Introduction

In today's digital era, the integration of advanced software and hardware plays a crucial role in developing innovative solutions across various industries. Agriculture, particularly poultry farming, significantly benefits from technological advancements. In this chapter, we explore how we can harness modern software and hardware to enhance farming operations, improving both efficiency and effectiveness in farm management.

We aim to present a detailed overview of the tools and technologies we utilize to create an integrated farm management system. By leveraging cutting-edge software and robust hardware, we strive to transform traditional farming practices into more data-driven and automated processes. This integration not only helps in better disease detection and monitoring but also in optimizing overall farm performance. Through this exploration, we demonstrate the potential of modern technology in revolutionizing agricultural practices, making them more sustainable and productive.

II.2 Hardware

Poultry farming is a cornerstone of the global food industry, supplying essential protein to populations worldwide. However, the presence of diseases within poultry flocks presents formidable obstacles for farmers, resulting in economic losses and endangering food security. To combat this issue, the incorporation of cutting-edge technologies, such as cameras and sensors, into poultry farming practices has emerged as a promising solution. This chapter delves into the various applications, advantages, and challenges associated with the integration of cameras and sensors for the detection of poultry diseases.

II.2.1 Cameras

The camera is an essential hardware element in the intelligent poultry disease detection system based on DL and IoT. It plays a vital role in capturing high-quality images of poultry within the farm environment. These images serve as critical input data for the DL model, enabling it to analyze and identify signs of diseases or abnormalities in the poultry

Key features of the camera used in this system include:

High Resolution: The camera should have high resolution to capture clear and detailed images of the poultry, allowing for accurate analysis by the DL model.

Wide Field of View: A wide field of view enables the camera to cover a larger area within the poultry farm, ensuring comprehensive monitoring of the flock.

Low-Light Performance: Since poultry farms may have varying lighting conditions, including low-light environments, the camera should have good low-light performance to capture clear images in all conditions.

Weatherproofing: Outdoor poultry farms are exposed to various weather conditions, so the camera should be weatherproof to withstand elements like rain, dust, and extreme temperatures.

Compatibility with IoT Systems: The camera should be compatible with IoT systems and protocols for seamless integration into the overall intelligent disease detection system.

By incorporating cameras with these features, the poultry disease detection system can effectively monitor the health and well-being of the poultry flock, enabling early detection of diseases and timely intervention.

II.2.1.1types of cameras used in this field and their roles

- **Thermal Imaging Cameras:** Thermal imaging cameras are used to measure the body temperature of poultry, and they are useful in detecting infections and diseases such as bronchitis and arthritis. These cameras rely on thermal radiation to generate images that illustrate temperature differences, allowing for the detection of changes in the body temperature of poultry. figure II-1 represents the thermal imaging cameras [24].



Figure *II.1*: A thermal imaging camera is used to detect hot spots in commercial poultry houses [25]

- **Standard Vision Cameras:** Standard vision cameras are used to analyze the appearance of poultry, including color, shape, and movement. These cameras can be used to monitor any visible signs of diseases such as hair loss or changes in feather color, which may indicate health issues. figure II-2 illustrates the standard vision cameras [24].



Figure *II.2*: Standard Vision Camera [26]

- **Smart Vision Cameras:** These cameras are equipped with AI technologies and rely on DL algorithms to analyze poultry images advancedly. These cameras can detect subtle changes in behavior and appearance that may indicate health problems, allowing for accurate monitoring of bird health.

- **Continuous Video Cameras:** These cameras rely on continuous video recording of poultry activity on the farm. Video can be used to track movement and behavior around the clock, providing detailed data on bird health and facilitating the detection of any abnormal changes.

- **Ultraviolet Imaging Cameras:** These cameras are used to detect hidden skin marks and other health problems that may not be visible to the naked eye. This technology provides slow and accurate illumination that allows for seeing changes at the skin level [27].

II.2.2 Sensors

In poultry farming, various types of sensors are used to monitor the health and well-being of chickens and detect any signs of diseases.. the DHT11 temperature and humidity sensor was chosen for the system

II.2.2.1Dht11 sensor

The DHT11 is a cost-effective digital sensor for detecting temperature and humidity. The sensor can be easily connected to any microcontroller such as Arduino, Raspberry Pi, etc. to measure humidity and temperature instantly.

The humidity and temperature sensor DHT11 is available as a sensor and module. The sensor differs from the module in the pull-up resistors and operating LED. DHT11 is a relative humidity sensor. To measure the ambient air, the sensor uses a thermistor and a capacitive humidity sensor. Figure II-3 represent DHT11 sensor

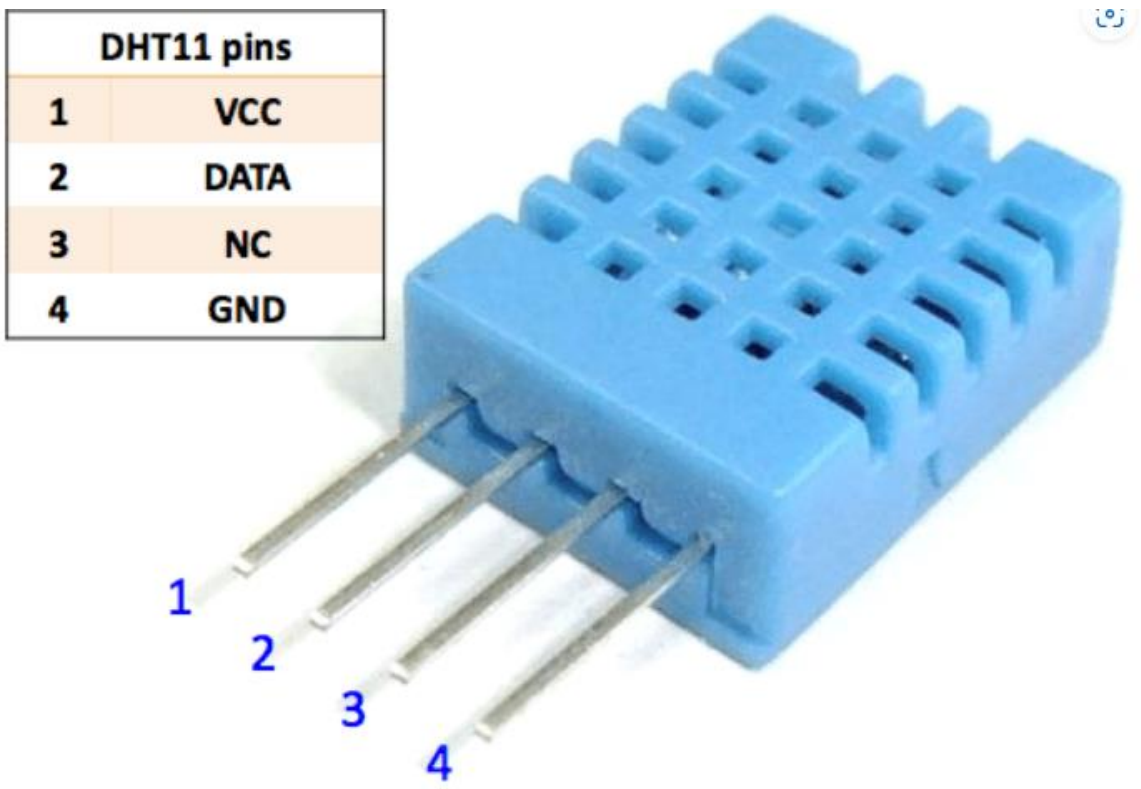


Figure II.3: humidity and temperature sensor (DHT11)

II.2.2.2Characteristics of DHT11

The DHT11 sensor is known for its specific characteristics, which make it suitable for various applications, including use in poultry farms. Here are the key characteristics of the DHT11 sensor:

Temperature Range: The DHT11 can measure temperatures ranging from 0 to 50 degrees Celsius with an accuracy of ±2 degrees Celsius.

Humidity Range: It measures humidity levels from 20% to 90% with an accuracy of ±5%.

Digital Output: The sensor provides a digital output, making it easy to interface with microcontrollers like Arduino and Raspberry Pi.

Low Cost: It is an economical option for sensing temperature and humidity.

Size: The DHT11 is compact, making it suitable for various applications where space is limited.

Operating Voltage: It operates at a voltage range of 3.3V to 5.5V.

Sampling Rate: The sensor has a sampling rate of 1 Hz, meaning it can provide one reading per second.

Long-Term Stability: The DHT11 is designed for long-term stability and reliability in monitoring environmental conditions.

Simple Interface: It uses a single-wire interface for communication, simplifying connections and wiring.

Power Consumption: The sensor has low power consumption, making it ideal for battery-powered devices.

These characteristics make the DHT11 a versatile and practical sensor for monitoring environmental conditions in various applications, including agricultural and industrial settings.

II.2.2.3Benefits of Cameras and Sensors

Early Disease Detection: Continuous monitoring enabled by cameras and sensors allows for early detection of diseases, enabling prompt intervention and treatment, thus reducing the spread of infection and minimizing economic losses.

Data-Driven Decision Making: The data collected by cameras and sensors provide valuable insights into poultry health and environmental conditions. Farmers can make data-driven decisions regarding disease management, health preservation measures, and resource allocation based on this information.

Improving Efficiency: Automated monitoring tasks reduce the need for manual surveillance, saving time and labor costs for farmers. It also facilitates proactive management practices, leading to overall efficiency improvement in poultry production.

II.2.3 Raspberry

In the following, we will present the Raspberry Pi that we used in our project. This device is considered the brain of our system because it acts as a web server displaying in real-time several values stored in a database.

II.2.3.1Presentation of Raspberry

The Raspberry Pi is indeed a mini-computer; it is a small single-board computer capable of running operating systems like Linux and even a pared-down version of Windows 10. This board is manufactured to assist in computer studies and to serve as a means of learning computer programming in various languages (such as Python, Scratch, etc.). It features USB ports, an Ethernet connector, an HDMI port, and audio output, much like a desktop computer. The Raspberry Pi is intended for embedded computing applications. The core of the Raspberry Pi is an FPGA (Broadcom 2835) integrating an ARM11 processor clocked at 700MHz and numerous peripherals [28].

II.2.3.2Hardware Specifications and Architectures

There have been several different models of the Raspberry Pi since its inception:

Model A: Use this model for a low-cost project that requires a full computer without networking capabilities and decent I/O support.

Model B: This model can be used for a project where price is not an issue and where the most powerful Pi is needed. This model also contains easy-to-use I/O, making it suitable for early Pi projects.

Model Zero: This model is ideal for a very low-cost project, in a confined space, that requires a fully functional computer and would benefit from wireless connectivity.

Figure II-4 shows some differences between certain models of Raspberry Pi

	Générations Zéro	Génération 1		Génération 2	Génération 3	Génération 4
Modéle	zéro	A	B	2 B	3 A+	4 B
RAM	512 Mo	256 Mo	512 Mo	1 Go	512 Mo	1Go / 2Go / 4Go / 8Go
Processor	1GHz ARM11	700 MHz ARM11	700 MHz ARM11	900 quadricœur ARM Cortex-A7	1,4 quadricœur ARM Cortex-A53	1,5 quadricœur ARM Cortex-A72
Sorties vidéo	Mini HDMI et Composite via soudures	HDMI et Composite via une Price RCA		HDMI et Composite (via un connecteur Jack		2 Micro HDMI et Composite via un connecteur Jack
Sorties audio	Mini-HDMI	stéréo Jack 3,5 sortie son 5.1 sur la prise HDMI				
USB	1 Micro-USB 2.0	1 (2.0)	2 (2.0)	4 (2.0)	1 (2.0)	2 (2.0) 2 (3.0)
poids	9 g	45 g	45 g	45 g	29 g	46 g
Reseau	non		10/100 Mbits/s Ethernet		Wifi 802.11 ac Bluetooth 4.2	Ethernet Wifi Bluetooth

Figure II.4: Comparison between Raspberry Pi models in a table

When looking at the different Raspberry Pi models, you'll notice that some have a "+" symbol in their name, indicating design improvements. For example, the Raspberry Pi Model A offers 256 MB of RAM, while the Model A+ has 512 MB of RAM, a microSD slot, and additional GPIO pins. We continue with the Raspberry Pi 3 Model B, which heralds the beginning of the third generation and replaces its predecessor, the Raspberry Pi 2 Model B. This model features a Broadcom BCM2837 64-bit quad-core processor clocked at 1.2GHz and 1GB of RAM, with built-in Wi-Fi and Bluetooth Low Energy (BLE). In comparison, the Raspberry Pi 3 Model B+ features a Broadcom BCM2837B0 Cortex-A53 (ARMv8) 64-bit 1.4GHz processor with 1GB RAM and dual-band 2.4GHz and 5GHz IEEE 802.11.b/g/n/. AC WiFi, Bluetooth 4.2, and BLE.

II.2.3.3Standard Components of Raspberry Pi

To get a good understanding of what can be added to the bare Raspberry Pi board, **Figure 19** represents both its general organization and the various connectors available on a standard Raspberry Pi:

The center of the board is occupied by the processor, or more precisely, by a stack of the processor and its memory, which are mounted on top of each other.

The other major integrated circuit visible on this side of the printed circuit board is the network controller, obviously absent from Model A.

The majority of the card's surface is occupied by the connectors rather than active components, and some rather curious choices have been made in terms of connectors, some of which are quite archaic for such a recent product.

The network socket accepts a standardized RJ45 plug and any Ethernet network at 10 Mbits or 100 Mbits per second.

The adjacent corner of these connectors features five LED status indicators for the Raspberry Pi:

1. A green LED indicating processor activity.
2. A red LED indicating power-on status.
3. A green LED indicating full duplex mode for the network connection.
4. Another green LED indicates network activity.
5. A yellow LED indicating that the network is operating at 100 megabits per second.

The board features:

1. **An ARM processor:** ARM architectures are low-power processor architectures introduced in 1983 by "Acorn Computers" and developed in 1990 by "ARM Ltd."
2. **Random Access Memory (RAM):** This is the memory where the Raspberry Pi places data during processing.
3. **A variety of connectors:**

USB port: Used to connect the Raspberry Pi to other devices or to connect a mouse and keyboard.

SD card slot: The Raspberry Pi requires additional external memory to function. This slot allows for the connection of external memory where the operating system and files are stored.

Ethernet port: A port corresponding to the international Ethernet protocol for local packet communication

Audio socket or jack: It is a 3.5mm diameter female stereo connector designed for connecting a headset, a pair of amplified speakers like those used with PCs, or even an external amplifier.

HDMI port: "High Definition Multimedia Interface" used to connect the Raspberry Pi to a compatible display capable of providing full HD 1080p video and audio.

RCA port: "Radio Corporation of America," used to transmit video signals generated by the Raspberry Pi in PAL or NTSC composite video format, allowing for display on old cathode ray tube monitors in low resolution.

Micro USB power connector: Used to connect to a power source. Always do this last, after connecting all other components.

GPIO ports: "General Purpose Input/Output" ports that allow us to connect electronic components such as LEDs and buttons to the Raspberry Pi. figure II-5 shows the components of Raspberry Pi

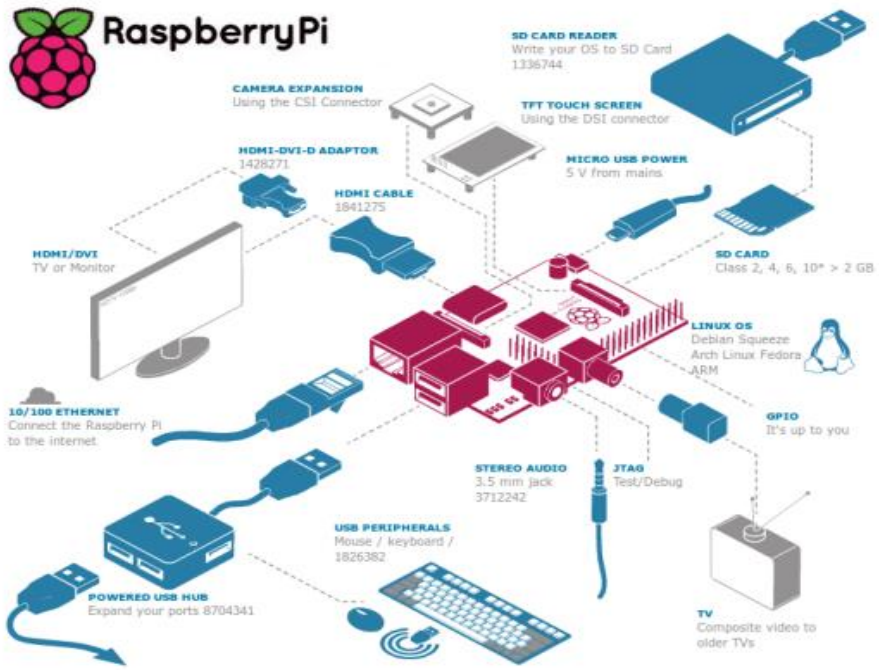


Figure II.5: Standard components of a Raspberry Pi

II.2.3.3.1 GPIOs

GPIOs (General Purpose Input/Output) are versatile configurable digital outputs found on embedded controllers and System-on-chip (SoC) processors like those used in Raspberry Pi boards. These outputs can be configured to act as either inputs or outputs, allowing communication with external components such as sensors, actuators, LEDs, motors, and more. GPIOs provide great flexibility in electronic project design, enabling users to control and monitor various electronic devices according to their needs.

Here are common types of GPIO

Digital GPIO: Digital GPIOs are used for simple digital inputs/outputs, such as reading the state of an ON/OFF switch, or a push button, or controlling the on/off of a light or motor. They operate by transmitting binary digital signals, typically 0 or 1.

PWM (Pulse Width Modulation): PWM GPIOs generate output signals with variable pulse width, allowing control of the average power supplied to an external device. They are commonly used to control motor speed, LED brightness, and other applications where power variation is needed.

UART (Universal Asynchronous Receiver/Transmitter): UART GPIOs facilitate the exchange of asynchronous serial data between the Raspberry Pi and other external devices. They are used to establish bidirectional serial connections and are commonly used to communicate with devices such as GPS modules and microcontrollers.

I2C (Inter-Integrated Circuit): I2C GPIOs enable synchronous serial communication between multiple devices connected to the same bus. They are often used to connect sensors and other low-speed data transmission devices.

SPI (Serial Peripheral Interface): SPI GPIOs facilitate high-speed synchronous serial communication between a Raspberry Pi and external devices. They are used to transfer data at high speeds and are commonly used with sensors, displays, and other devices requiring fast data transmission.

Analog GPIOs: Some GPIOs can measure analog signals, such as variable voltages or currents. They are used to capture data from analog sensors and can also be used to generate analog output signals.

These different types of GPIOs offer considerable flexibility for interacting with various electronic components and meeting the specific requirements of applications. As shown in figure II-6.

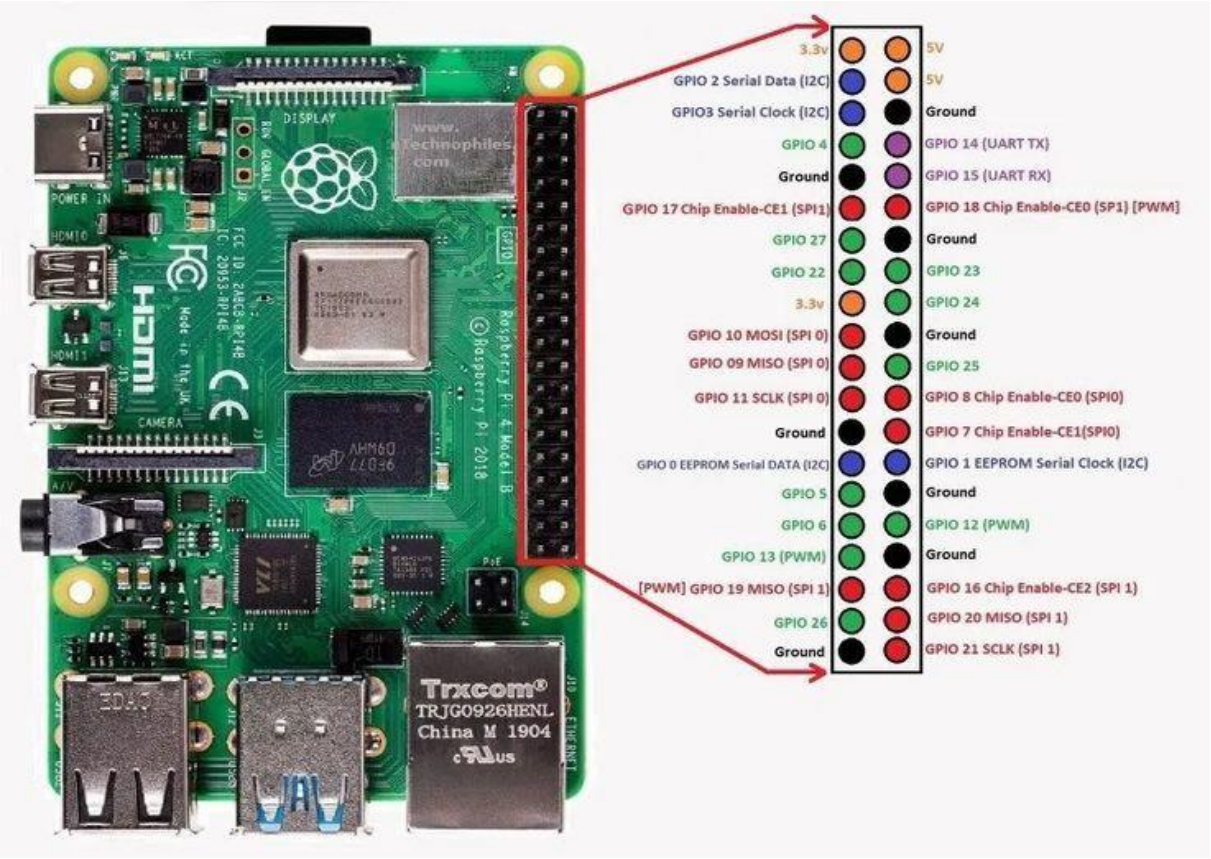


Figure II.6: Raspberry Pi 4B Gpios

II.2.3.4 Starting a Raspberry Pi

The Raspberry Pi can be directly connected to a traditional user interface (UI) setup, including a mouse, keyboard, HDMI, or composite video screen. However, like any Linux computer, the Raspberry Pi can integrate its development tools and a user interface based on SSH, controllable from another computer via Ethernet or Wi-Fi.

II.2.3.4.1 First Boot of Raspberry Pi

We'll need several things. Firstly, a Raspberry Pi and its power supply. Secondly, a MicroSD card. Additionally, you'll also need a USB keyboard and mouse for your Raspberry Pi, and finally a screen.

II.2.3.4.2 Preparing the SD Card

Just as the PC's hard drive must be equipped with the required operating system, the memory card used with the Raspberry Pi must be initialized with the chosen operating system. Without this, it won't start. The foundation recommends using a card with at least 4 GB and at least Class 4.

The first step is to download the Raspberry Pi operating system from the Raspberry Pi Foundation. We use "Raspberry Pi Imager" on Ubuntu. Just download the ZIP file and extract it to a new folder. You will then have an IMG image file containing the operating system.

Next, we need a program that can write the operating system image file to the microSD card. For this, we use the balenaEtcher software. So download and install it.

II.2.3.4.3 Connecting the Raspberry Pi

To set up and power on the Raspberry Pi, start by inserting your microSD card into the designated slot on the Raspberry Pi.

Connect your USB mouse and keyboard to the USB ports on the Raspberry Pi; any USB keyboard and mouse will work.

Connect the display cable to the HDMI port on the Raspberry Pi, using an adapter if necessary, or use the RCA output for lower-quality composite video.

If you want to connect to the internet via cable, use an Ethernet cable to connect the Pi’s port to the Ethernet socket, although this is not necessary if you don't need internet access.

All the software for the Raspberry Pi can be downloaded or updated online, making internet access essential. Fortunately, connecting to a wired network is straightforward with any standard RJ45 cable, as the Raspberry Pi's network interface accepts both straight and crossover cables.

The Raspberry Pi does not include a WiFi module, so a USB-WiFi dongle is needed for WiFi. If your monitor has built-in speakers, the Raspberry Pi can play sound through them, or you can use the audio port for headphones or speakers. Once all cables are connected and the microSD card is inserted, plug in the micro USB power supply to power up and start your Raspberry Pi.

You should see a red light on the Raspberry Pi and a Raspberry logo on the screen, followed by the graphical interface and desktop. On the first boot, log in with the default credentials: "pi" for the username and "raspberry" for the password. As shown in Figure II-7

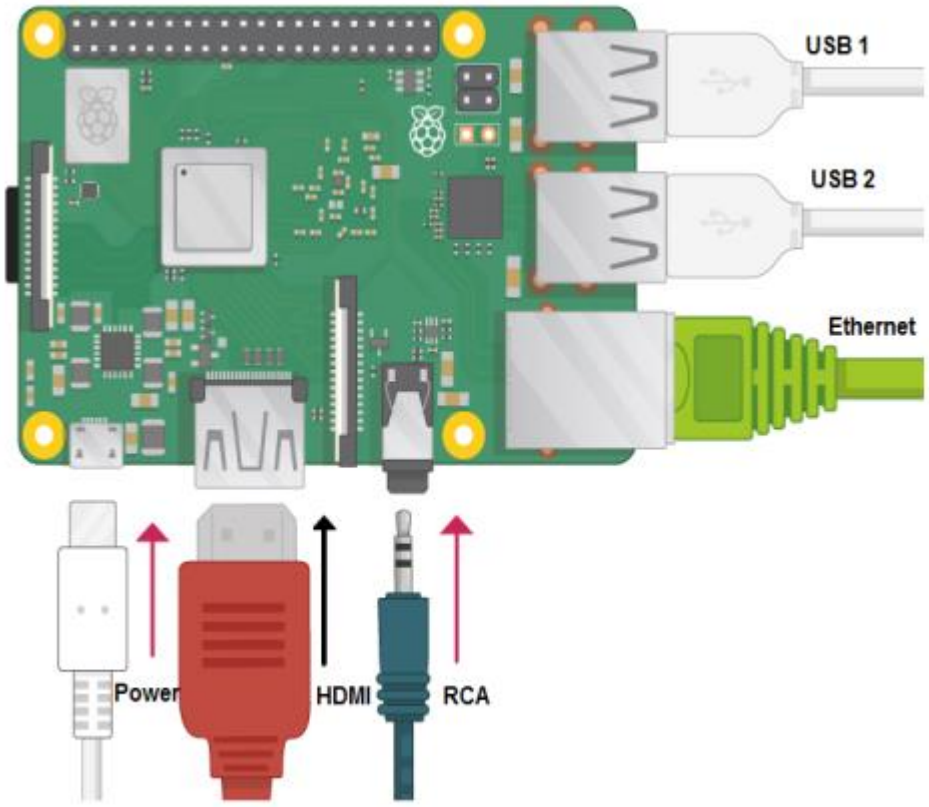


Figure II.7: Connecting Peripherals

II.2.3.5Raspberry pi in poultry disease detection

Using a Raspberry Pi in poultry disease detection is a promising application of embedded computing. Here are some ways a Raspberry Pi can be used in chicken monitoring and disease detection:

- **Video surveillance system:** A Raspberry Pi can be configured with a camera to continuously monitor the chicken environment. Captured images can be analyzed to detect abnormal behaviors or signs of illness.
- **Environmental data collection:** By using sensors connected to a Raspberry Pi, it's possible to monitor environmental parameters such as temperature, humidity, and air quality in the chicken coop. Variations in these parameters can indicate the presence of diseases or uncomfortable living conditions for the chickens.
- **Real-time data analysis:** The Raspberry Pi can be programmed to analyze data collected by sensors in real time. Machine learning algorithms can be used to detect patterns or anomalies that could indicate the presence of diseases.
- **Alerts and notifications:** In case of anomaly detection, the Raspberry Pi can send alerts to farm owners or managers via smartphone notifications or email. This allows for quick intervention to take preventive or corrective measures.
- **User-friendly interface:** The Raspberry Pi can be equipped with a user-friendly interface, accessible via a web browser or mobile app. This allows users to easily monitor the health status of chickens and make informed decisions in real time. figure II-8 represents the Rasberry Pi [29].



Figure II.8: Raspberry Pi

By combining the video surveillance, data collection, and real-time analysis capabilities of the Raspberry Pi, it's possible to set up an intelligent and efficient avian surveillance system for detecting and preventing diseases in chickens.

II.3 **Software**

In the realm of poultry farming, the management of poultry health is paramount for ensuring optimal production, welfare, and profitability. However, the complex interplay of various factors such as environmental conditions, infectious agents, and management practices poses significant challenges in maintaining flock health. To address these challenges, the integration of software solutions has emerged as a valuable tool in the arsenal of poultry health management.

Software applications tailored to poultry diseases offer a multifaceted approach to disease detection, diagnosis, prevention, and management. These tools harness the power of data analytics, machine learning, and decision support systems to provide actionable insights for poultry farmers, veterinarians, and researchers. By leveraging software solutions, stakeholders in the poultry industry can enhance their ability to monitor flock health, implement timely interventions, and mitigate the impact of diseases on production.

II.3.1 **Python**

II.3.1.1Definition

Python is a general-purpose, high-level programming language. With a strong emphasis on indentation, its design philosophy prioritizes code readability.

Python uses garbage collection and dynamic typing. It is compatible with various programming paradigms, such as object-oriented, functional, and structured (especially procedural). Because of its extensive standard library, it is frequently referred to as a "batteries included" language [30].

II.3.1.2Advantages of Python

- Presence of third-party modules: Python has a rich ecosystem of third-party modules and libraries that extend its functionality for various tasks.
- Extensive support libraries: Python boasts extensive support libraries like NumPy for numerical calculations and Pandas for data analytics, making it suitable for scientific and data-related applications.

- Open source and large active community base: Python is open source, and it has a large and active community that contributes to its development and provides support.
- Versatile, easy to read, learn, and write: Python is known for its simplicity and readability, making it an excellent choice for both beginners and experienced programmers.
- User-friendly data structures: Python offers intuitive and easy-to-use data structures, simplifying data manipulation and management.

- High-level language: Python is a high-level language that abstracts low-level details, making it more user-friendly.
- Dynamically typed language: Python is dynamically typed, meaning you don't need to declare data types explicitly, making it flexible but still reliable.
- Object-oriented and Procedural programming language: Python supports both object-oriented and procedural programming, providing versatility in coding styles.
- Portable and interactive: Python is portable across operating systems and interactive, allowing real-time code execution and testing.
- Ideal for prototypes: Python's concise syntax allows developers to prototype applications quickly with less code.
- Highly efficient: Python's clean design provides enhanced process control, and it has excellent text processing capabilities, making it efficient for various applications
- (IoT) opportunities: Python is used in IoT applications due to its simplicity and versatility.
- Interpreted language: Python is interpreted, which allows for easier debugging and code development[31].

II.3.2 Gradio

Gradio is an open-source Python library that allows you to quickly create UIs (user interfaces) for your machine-learning models. With Gradio, you can build interactive interfaces for your models with just a few lines of code, enabling users to easily interact with and explore the functionality of your models through web browsers[32].

II.3.2.1Features of Gradio

- **Simple Interface Creation:** Gradio allows you to create user interfaces for your machine-learning models with just a few lines of code. It abstracts away the complexities of web development, making it easy for users to interact with your models.

- **Support for Various Input Types:** Gradio supports a wide range of input types, including text, images, audio, and video. This flexibility enables you to build interfaces for a variety of machine-learning tasks, from image classification to natural language processing.
- **Interactive Visualization**: Gradio provides interactive visualization options for displaying the results of your models. You can choose from various output types, such as text, images, plots, and more, to present the model's predictions or outputs intuitively and understandably.
- **Real-time Updates:** Gradio allows for real-time updates of model outputs as users interact with the interface. This feature enables users to see how changes in input data affect the model's predictions or outputs instantly.
- **Customization Options:** Gradio offers a range of customization options, allowing you to tailor the interface to your specific needs and preferences. You can customize the layout, styling, and functionality of the interface to create a seamless user experience.

- **Built-in Deployment:** Gradio provides built-in deployment options, making it easy to share your models with others. You can deploy your interfaces locally or host them on the web, enabling users to access them from anywhere with an internet connection.
- **Integration with Existing Models:** Gradio seamlessly integrates with popular machine learning libraries like TensorFlow, PyTorch, and sci-kit-learn, allowing you to use your existing models with minimal modifications.
- **Community Support**: Gradio has a growing community of users and contributors who actively contribute to its development and provide support through forums, documentation, and tutorials.

Overall, Gradio is a powerful tool for democratizing machine learning by making it easy to create, share, and interact with machine learning models through intuitive user interfaces.

II.3.3 Node-RED

II.3.3.1Definition

Node-RED is a flow-based, low-code development tool for visual programming developed originally by IBM for wiring together hardware devices, APIs, and online services as part of the Internet of Things.

Node-RED provides a web browser-based flow editor, which can be used to create JavaScript functions. Elements of applications can be saved or shared for reuse. The runtime is built on Node.js. The flows created in Node-RED are stored using JSON [33].

II.3.3.2Node-RED function

1. Flow-Based Programming

Visual Workflow Design: Allows users to create applications by connecting predefined nodes representing different functionalities, such as data input/output, processing, and control flow.

Drag-and-Drop Interface: Simplifies the process of building and modifying workflows without needing extensive coding knowledge.

2. Integration and Connectivity

Device and Service Integration: Easily integrates with a wide range of devices, online services, and APIs, facilitating communication between different systems.

Protocol Support: Supports various protocols such as HTTP, MQTT, WebSockets, TCP/UDP, and more, enabling seamless data exchange in IoT and automation scenarios.

3. Event-Driven Processing

Real-Time Data Handling: Designed to handle asynchronous events, making it suitable for real-time data processing tasks.

Trigger-Based Execution: Nodes can be configured to execute actions based on specific events, such as receiving a message or reaching a scheduled time.

4. Data Transformation and Processing

Data Flow Management: Manages the flow of data between nodes, allowing for efficient data transformation and processing.

Function Nodes: Provides JavaScript function nodes where users can write custom code to perform more complex data manipulations.

5. Automation and Orchestration

Task Automation: Automates repetitive tasks by creating workflows that can handle various automation scenarios, from simple to complex.

Scheduling: Includes nodes for scheduling tasks at specific times or intervals, which is essential for automating time-based activities.

6. User Interface and Dashboard

UI Dashboard: Allows users to create custom dashboards to visualize data, control devices, and interact with their workflows.

Real-Time Monitoring: Dashboards can be used for real-time monitoring and control of IoT systems and automation processes.

7. Logging and Debugging

Debug Nodes: Provides tools for logging data and debugging workflows, helping users identify and resolve issues efficiently.

Error Handling: Includes nodes and mechanisms for handling errors, ensuring that workflows can manage unexpected situations gracefully.

8. Modularity and Reusability

Reusable Subflows: Allows the creation of sub-flows, which are reusable components that can be incorporated into multiple workflows, enhancing modularity and maintainability.

Node Reusability: Users can create custom nodes and share them with the community, promoting reusability and collaboration.

9. Extensibility and Customization

Custom Nodes and Plugins: Users can develop custom nodes to extend Node-RED's functionality, tailored to specific requirements.

Open-Source Flexibility: As an open-source tool, Node-RED can be customized and extended to meet unique needs and integrated into various environments.

II.3.4 Usefulness of Anaconda and Google Collab for Implementation

- **Google Collab:** Google Colab, short for Google Colaboratory, is a platform offered for free by Google that allows you to write and execute Python code in your browser. It enables you to run Jupyter notebooks without needing to worry about your hardware or the software installed on your computer. Google Colab also facilitates access to computational resources and common machine-learning libraries. Figure II-9 represents the Google Colab logo [34].



Figure II.9: Google Collab logo

- **Anaconda:** Anaconda is a free and open-source distribution tool designed for programming in Python and R environments. It is widely used in data science, artificial intelligence, and machine learning. This scientific distribution of Python includes numerous packages necessary for data analysis. Anaconda also serves as an open-source environment manager[35].

  Anaconda includes a wide range of essential software and libraries for data science, artificial intelligence, and machine learning. Here are some of the main software and libraries included:

  Jupyter Notebook: An interactive environment for writing and executing code in Python.

  Spyder: An integrated development environment (IDE) for Python, specifically designed for data analysis.

  NumPy: A library for numerical computations with support for multidimensional arrays and matrices.

Pandas: A library for data manipulation and analysis, offering flexible data structures.

SciPy: A library used for scientific and technical computing.

Matplotlib: A library for creating static, animated, and interactive visualizations in Python.

Seaborn: A data visualization library based on Matplotlib, with a high-level interface for drawing statistical graphics.

Scikit-learn: A library for machine learning and data mining.

TensorFlow: An end-to-end open-source platform for machine learning.

Keras: A high-level neural networks API running on top of TensorFlow.

PyTorch: A machine learning library based on Torch.

NLTK (Natural Language Toolkit): A suite of libraries and programs for natural language processing.

Statsmodels: A library for estimating and testing statistical models.

Beautiful Soup: A library for parsing HTML and XML documents.

Dask: A library for parallel and distributed computing.

H2O.ai: An open-source platform for machine learning and predictive analytics.

This software and libraries included in Anaconda provide users with a comprehensive set of tools for development and analysis in the field of data science.

As shown in Figure II-10

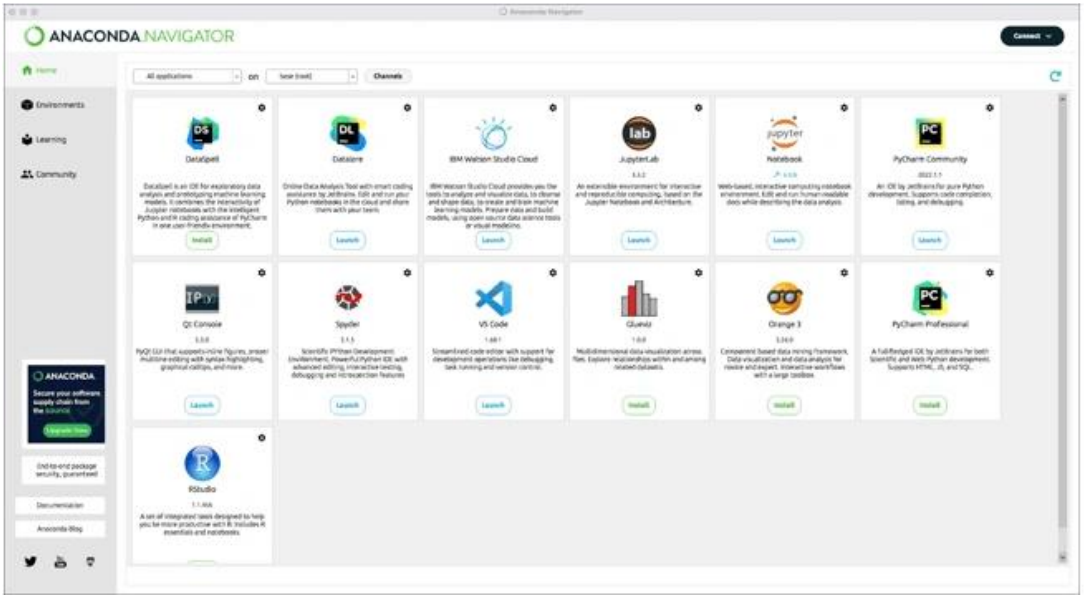


Figure II.10: anaconda Navigator

We used Google Colab in our research project on poultry disease detection due to its powerful features and easy accessibility. Google Colab was particularly helpful because it provides free access to high-performance graphics processing units (GPUs), greatly speeding up the training of our machine-learning models. This capability is crucial when dealing with large datasets and complex algorithms, significantly reducing processing time compared to running them on local machines.

While Anaconda is a powerful tool for package management and creating isolated environments for various project aspects, the cloud-based platform of Google Colab allowed us to execute code easily without worrying about limitations related to local hardware. We could easily share our work and collaborate with team members through integration with Google Drive, increasing the efficiency of our project workflow.

Overall, the high-performance graphics processing power available in Google Colab made it a more practical choice for our needs, enabling more efficient training and testing of models. By leveraging Google Colab, we were able to achieve faster results and focus more on improving our models to increase the accuracy of poultry disease detection. Additionally, Anaconda takes a long time to produce results, often exceeding 24 hours (compared to Google Colab, which takes only a few minutes).

II.3.5 Predefined Model

This study uses architectures designed for image classification including VGG16 and MobileNet

II.3.5.1Vgg16 model (Visual Geometry Group 16)

II.3.5.1.1 Definition

VGG16 is a deep convolutional neural network model for image classification tasks. The network consists of 16 layers of artificial neurons, each dedicated to incrementally processing image information and improving the accuracy of predictions.

Instead of using a lot of hyperparameters, VGG16 uses convolutional layers with 3x3 filters and step 1 (in the same padding), and maxpool layers with 2x2 filters and step 2. This

arrangement of convolutional and maxpool layers is followed throughout the architecture. At the end, it has two fully connected layers followed by a softmax for the output.[36]

In VGG16, "VGG" refers to the Visual Geometry Group at the University of Oxford, and "16" refers to the 16 layers of the network with weights. This network is a fairly large one, with about 138 million parameters

II.3.5.1.2 Architecture of Vgg16

The VGG16 architecture consists of 16 weighted layers (13 convolutional layers, 2 fully connected layers, and 1 softmax layer). Below is a detailed explanation of each layer type in VGG16

1. **Input Layer**

Shape: Typically accepts input images of size 224x224x3 (width, height, and RGB channels).

2. **Convolutional Layers**

Purpose: These layers apply convolution operations on the input and extract features from the image.

Filter Size: All convolutional layers use a filter size of 3x3.

Stride: The stride is set to 1.

Padding: “Equal” padding is used, which means that the spatial dimensions of the input are preserved.

Activation Function: ReLU (Rectified Linear Unit) is applied after each convolution to introduce non-linearity.

3. **Max Pooling Layers**

Purpose: These layers reduce the spatial dimensions (width and height) of the feature map, thereby reducing the number of parameters and computations in the network.

Filter Size: 2x2

Stride: 2

4. **Fully Connected (Dense) Layers**

Purpose: These layers are used to combine the features extracted by the convolutional layers and make predictions.

Number of Neurons: The first two fully connected layers each have 4096 neurons.

Purpose: The final layer of the network that provides the probabilities for each class.

Number of Outputs: Depends on the number of classes (e.g., 1000 for ImageNet).

- **Summary of VGG16 Architecture**

The VGG16 model is structured to handle the input image through a series of convolutional layers with small receptive fields (3x3) followed by max-pooling layers to reduce the spatial dimensions.

This process is repeated five times, with the number of filters increasing at each block to capture more complex features.

After the convolutional layers, the output is flattened and passed through two fully connected layers, each containing 4096 neurons.

Finally, a softmax layer is used to produce the classification probabilities for each class.

The simplicity and consistency of the VGG16 architecture make it an effective and widely used model for image classification tasks.

Figure II-11 shows the basic architecture of the VGG16 model

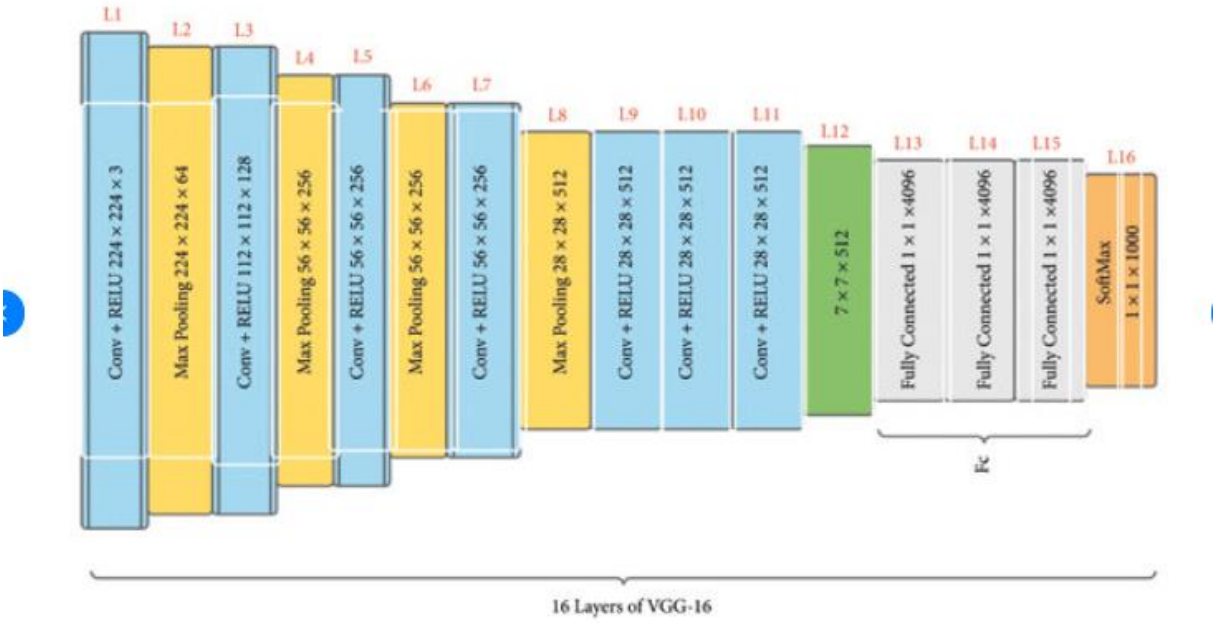


Figure II.11: Basic VGG16 architecture [37]

II.3.5.2MobileNet model

II.3.5.2.1 Definition

MobileNet is a convolutional neural network designed for mobile and embedded vision applications. It is based on an optimized architecture that uses depthwise separable convolutions to create a lightweight deep neural network that can provide low latency for mobile and embedded devices[38].

II.3.5.2.2 Key Features of MobileNet

1) **Depthwise Separable Convolutions:** Instead of using standard convolutions, MobileNet employs depthwise separable convolutions, which break the convolution process into two steps: depthwise convolution and pointwise convolution. This separation significantly reduces the computational cost and the number of parameters.
2) **Width Multiplier:** A hyperparameter α (alpha), known as the width multiplier, is used to thin the network. It uniformly scales the number of filters in each layer, reducing both the computational cost and the number of parameters.
3) **Resolution Multiplier:** A hyperparameter ρ (rho), known as the resolution multiplier, is used to reduce the input image resolution. This decreases the computational cost by reducing the size of the input images.
4) **Efficient Design:** MobileNet is designed to be lightweight and efficient, making it suitable for mobile and embedded devices with limited computational resources.

II.3.5.2.3 Architecture

The MobileNet architecture is composed of a series of layers, including:

1) **Standard Convolution:** The first layer is a standard convolution layer that processes the input image.
2) **Depthwise Separable Convolutions:** The core of the architecture consists of multiple depthwise separable convolution layers, each followed by a batch normalization layer and a ReLU activation function.
3) **Fully Connected Layers:** The final layers are fully connected layers that combine the features extracted by the convolutional layers and output the class probabilities using a softmax activation function. The basic building block of MobileNet is the depthwise separable convolution, which is divided into:

4) **Depthwise Convolution:** Applies a single filter to each input channel.
5) **Pointwise Convolution:** Uses a 1x1 convolution to combine the outputs of the depthwise convolution.

This separation drastically reduces the number of computations and parameters compared to traditional convolutional layers.

As shown in figure II-12

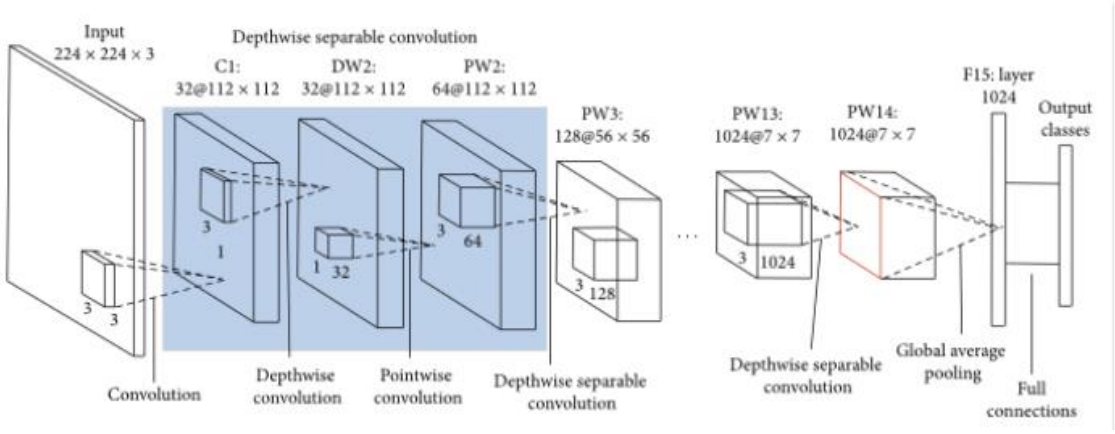


Figure II.12: Mobilenet architecture [39]

II.3.5.2.4 Performance Metrics

MobileNet achieves a balance between high accuracy and low computational requirements. Here are some key metrics:

1) **Accuracy:** While slightly lower than some of the more computationally intensive models, MobileNet still achieves high accuracy on standard benchmarks.
2) **Speed:** MobileNet is significantly faster than traditional convolutional networks, making it ideal for real-time applications.
3) **Model Size:** The reduced number of parameters results in a smaller model size, which is crucial for deployment on mobile devices.

II.3.5.2.5 Applications

MobileNet is widely used in various applications due to its efficiency and effectiveness:

1) **Image Classification:** MobileNet can classify images into thousands of categories, making it suitable for a wide range of applications.
2) **Object Detection:** Combined with frameworks like SSD (Single Shot MultiBox Detector), MobileNet can be used for real-time object detection on mobile devices.
3) **Face Recognition:** Its lightweight nature makes MobileNet suitable for face recognition applications on mobile devices.
4) **Embedded Systems:** MobileNet's low computational requirements make it ideal for deployment in embedded systems with limited resources

II.3.6 Performances criteria

- **ROC Curve**

The "ROC curve" stands for the Receiver Operating Characteristic curve. It's a graphical representation used in binary classification tasks to evaluate the performance of a model. The ROC curve plots the true positive rate (sensitivity) against the false positive rate (1 - specificity) at various threshold settings.

In simpler terms, the ROC curve helps to understand how well a model can distinguish between two classes (e.g., positive and negative) by varying the classification threshold. A perfect classifier would have a curve that reaches the top-left corner of the plot (100% true positive rate and 0% false positive rate), while a random classifier would produce a diagonal line from the bottom-left to the top-right of the plot.

The area under the ROC curve (AUC) is often used as a summary measure of the model's performance, with a higher AUC indicating better discrimination ability. It's a widely used metric in fields like machine learning, medicine (e.g., for diagnostic tests), and many others where binary classification is relevant.[40]

- **Translation of the curve**

Translating the AUC (Area Under the Curve) involves understanding its significance as a metric for evaluating the performance of a classification model. Here's how to interpret the AUC:

- **Understanding the Range**

AUC values range from 0 to 1, where 0.5 indicates a model that performs no better than random chance (i.e., equivalent to flipping a coin), and 1 represents a perfect model that perfectly separates the classes.

- **Interpreting AUC Values**

AUC values closer to 1 indicate better discrimination ability of the model, meaning the model is better at distinguishing between positive and negative instances.

AUC values closer to 0.5 suggest that the model's performance is no better than random chance.

AUC values below 0.5 indicate that the model is performing worse than random guessing, implying that the model's predictions are inversely related to the true class labels.

- Comparison with Random Classifier

AUC serves as a comparison to a random classifier. A random classifier would have an AUC of 0.5, as it has no discriminatory power.

Therefore, a model with an AUC greater than 0.5 is performing better than random guessing, while a model with an AUC less than 0.5 is performing worse than random guessing.

- **Practical Interpretation**

In practical terms, an AUC value of:

0.7 to 0.8 is considered good.

0.8 to 0.9 is considered very good.

Above 0.9 is considered excellent.

Below 0.7 may still have some utility but might require further improvement.

- **Assessment of Discrimination Ability**

AUC measures the model's ability to discriminate between positive and negative instances across all possible classification thresholds.

It indicates the probability that the model will rank a randomly chosen positive instance higher than a randomly chosen negative instance.

- **Real-world Application**

In real-world applications, a higher AUC generally corresponds to a model that is more reliable and accurate in its predictions.

For tasks like disease diagnosis or anomaly detection, a higher AUC implies better performance in correctly identifying diseased individuals or detecting anomalies while minimizing false positives. figure II-13 represents the general Roc curve

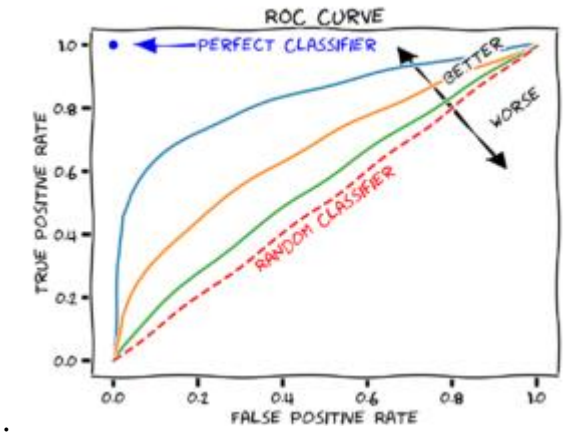


Figure II.13:Roc Curve [40]

➢ Accuracy and Loss

• Accuracy

Accuracy is a method for measuring a classification model's performance. It is typically expressed as a percentage. Accuracy is the count of predictions where the predicted value is equal to the true value. It is binary (true/false) for a particular sample. Accuracy is often graphed and monitored during the training phase though the value is often associated with the overall or final model accuracy. Accuracy is easier to interpret than loss[41].

• Loss

A loss function, also known as a cost function, takes into account the probabilities or uncertainty of a prediction based on how much the prediction varies from the true value. This gives us a more nuanced view of how well the model is performing.

Unlike accuracy, loss is not a percentage — it is a summation of the errors made for each sample in training or validation sets. Loss is often used in the training process to find the "best" parameter values for the model (e.g. weights in a neural network). During the training process, the goal is to minimize this value.

The most common loss functions are log loss and cross-entropy loss (which yield the same result when calculating error rates between 0 and 1), as well as mean squared error, and likelihood loss.

Unlike accuracy, loss may be used in both classification and regression problems. figure II-14 represents accuracy and loss models[41].

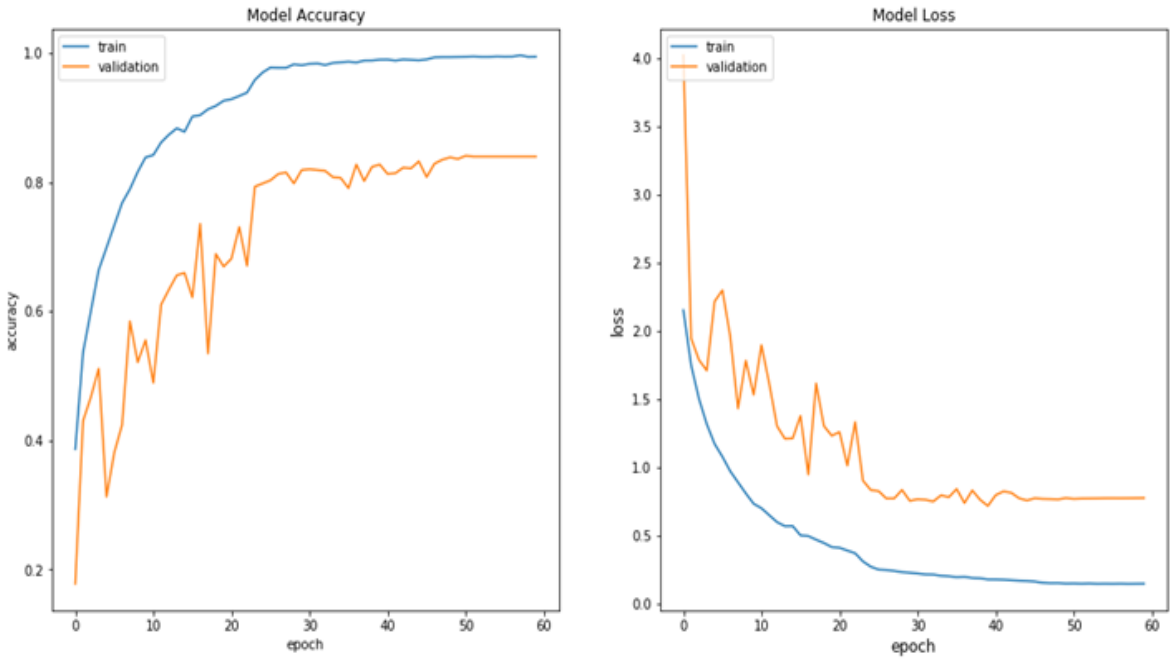


Figure II.14:Model accuracy and loss[41]

➢ Confusion matrix

A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known. It allows visualization of the performance of an algorithm.

The confusion matrix itself is relatively simple to understand, but the related terminology can be confusing:

True Positives (TP): These are cases in which the model correctly predicted the positive class.

True Negatives (TN): These are cases in which the model correctly predicted the negative class.

False Positives (FP): These are cases in which the model incorrectly predicted the positive class (i.e., the model predicted a positive result when it should have been negative).

False Negatives (FN): These are cases in which the model incorrectly predicted the negative class (i.e., the model predicted a negative result when it should have been positive).

figure II-15 illustrates the confusion matrix

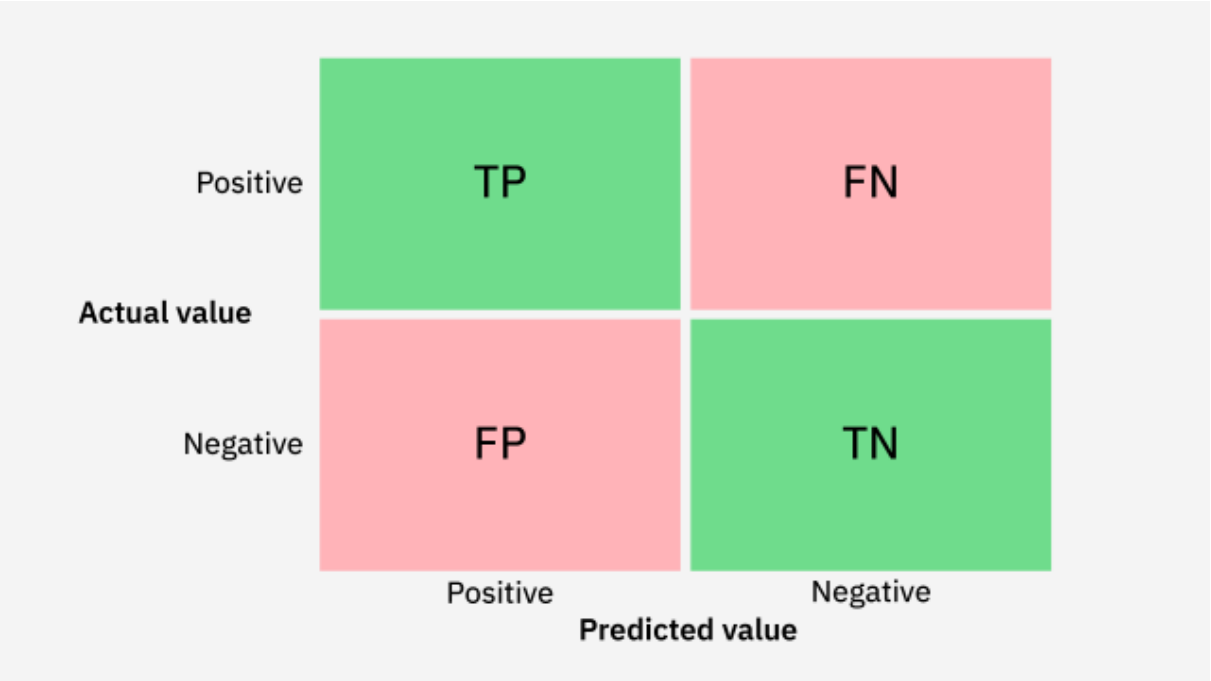


Figure II.15: General confusion matrix

From the confusion matrix, various performance metrics can be calculated:

Accuracy: The proportion of correctly classified instances (TP + TN) to the total number of **instances.Precision:** The proportion of true positive predictions out of all positive predictions (TP / (TP + FP)).

Recall (or Sensitivity): The proportion of true positive predictions out of all actual positives (TP / (TP + FN)).

F1 Score: The harmonic mean of precision and recall, which provides a balance between precision and recall.

Confusion matrices are especially useful in binary classification problems, but they can also be extended to multiclass classification problems by considering each class separately or by using other techniques like one-vs-all (OvA) or one-vs-one (OvO) strategies.

Each cell in the confusion matrix represents the count of instances belonging to a particular combination of true class and predicted class. This matrix helps in evaluating the performance of the classification model across all classes. You can compute various evaluation metrics like accuracy, precision, recall, and F1-score based on this confusion matrix.

II.4 Conclusion.

In conclusion, we have demonstrated in this chapter how the integration of advanced software and hardware can significantly enhance the efficiency and effectiveness of poultry farming. By leveraging these technologies, we have developed a comprehensive farm management system that allows for precise monitoring and rapid detection of poultry diseases. Our approach highlights the transformative potential of modern technology in agriculture, enabling more sustainable and productive farming practices. Through this exploration, we emphasize our commitment to advancing agricultural innovation and improving farm management outcomes

III. Chapter III: Implementation and Results

III.1 Introduction

This chapter details the implementation and results of our project on detecting chicken diseases using IoT and deep learning. The motivation behind this project stems from the significant impact that timely and accurate detection of poultry diseases can have on the agricultural industry. By leveraging advanced computational methods, we aim to provide an innovative solution that enhances health monitoring processes within poultry farms.

To achieve our objectives, we utilized Google Collab as our primary platform for programming and implementing machine learning models. Google Collab offers a cloud-based environment that is both powerful and user-friendly, allowing us to harness substantial computational resources without the constraints of local hardware limitations. This capability was crucial for processing our extensive dataset and training complex neural network models efficiently.

III.2 Proposed Diagram for our system Poultry Diseases Detection

Figure III. 1 illustrates the general architecture of the proposed system. The system integrates multiple sensors, a Raspberry Pi / a PC, and cloud services to monitor and analyze poultry health in real time. Here's a detailed breakdown of each component and their interactions:

- ✓ **Poultry Farm**

The diagram starts with a poultry farm where the chickens are housed. This is the environment that needs to be monitored.

- ✓ **Sensors**

Temperature and Humidity Sensor: Measures the temperature and humidity levels in the poultry environment.

Camera: Captures images or videos of the poultry farm to monitor the chickens' activity and health.

- **Information flow from sensors and the camera**

The sensors and the camera collect data from the poultry farm. This data includes environmental parameters (humidity and temperature) and visual information (images or videos).

- ✓ **Raspberry Pi / PC**

The collected data are sent to a Raspberry Pi or a PC. This device is responsible for processing the data (images) based on the developed CNN algorithm.

- **IoT Capabilities**: The device connects to the internet and other devices, enabling remote monitoring and control.

✓ **Router**

The router facilitates the connection between the local network (where the Raspberry Pi or PC is located) and the internet, enabling data to be sent to the cloud and accessed remotely.

✓ **Cloud**

The cloud stores the data collected from the poultry farm. It can provide additional processing power for data analysis, long-term data storage, and data accessibility from anywhere in the world.

✓ **Local HMI (Human-Machine Interface)**

The local HMI allows farm workers or managers to interact with the system directly on-site. It provides real-time data visualization and control over the farm's environment.

✓ **Remote Access**

Through the internet and the cloud, authorized users can access the data and monitor the poultry farm remotely. This could include farm managers or veterinarians who need to check on the farm's conditions from a different location.

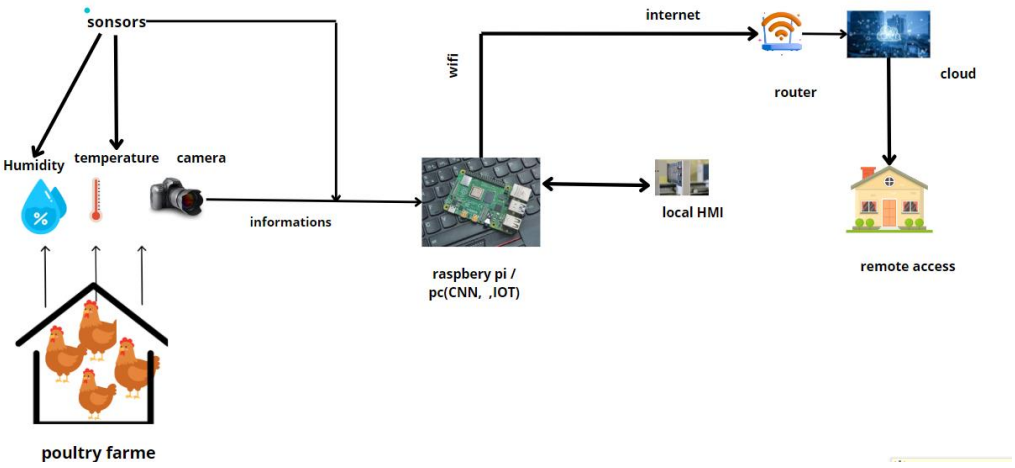


Figure III.1*: Proposed system general architecture*

III.3 Dataset preparation

We began by downloading the chicken disease dataset from our local PC to Google Colab. Our dataset is meticulously organized into three primary subsets: training, validation, and testing. Each subset is further divided into four distinct categories: New Castle Disease, Healthy, Coccidiosis, and Salmonella, ensuring a comprehensive representation of each health condition across all phases of model development. Specifically, we compiled a dataset that includes 6,508 images for the training subset, 778 images for the validation subset, and 781 images for the test subset, all categorized into the same four classes. This organization enhances the accuracy and effectiveness of our diagnostic efforts by ensuring that each class is adequately represented, allowing our models to handle real-world data effectively. The images in the dataset have dimensions of 224x224 pixels and three color channels (RGB). We sourced this dataset from the following references: [42], [43] and [44]

Proper organization was crucial for effective training and evaluation.

The training set (70% of the data): was used to train the models, enabling them to learn a variety of patterns associated with chicken diseases.

The validation set (10%): was used to fine-tune the models and prevent overfitting, providing immediate feedback on the models' performance during training.

The test set (20%): was reserved for evaluating the models' performance on unseen data, offering an unbiased assessment of their accuracy.

The figures III-2 ,III-3, III-4 ,III-5 below are taken from our dataset:

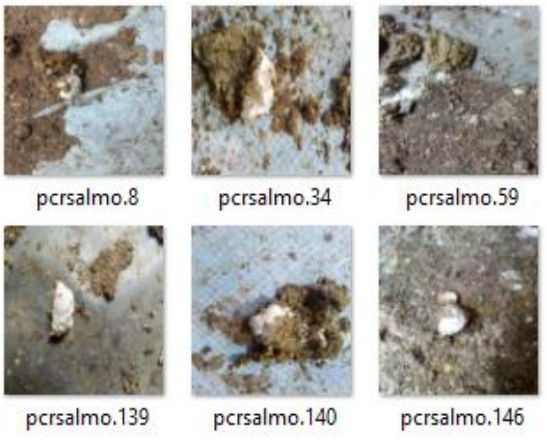


Figure III.2: Salmonella

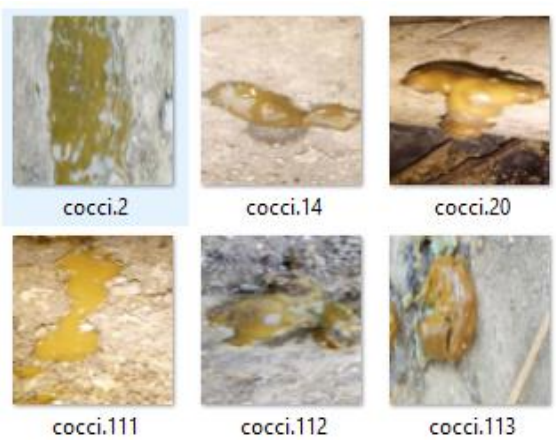


Figure III.3: Coccidiosis

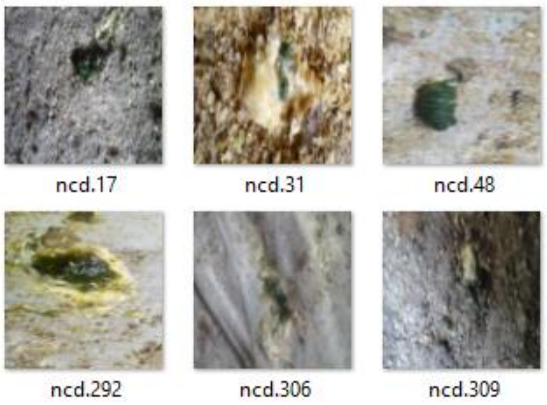


Figure III.4: New Castle Disease



Figure III.5: healthy

III.4 Model Selection and Implementation

For this project, we chose two well-regarded convolutional neural network architectures: VGG16 and MobileNet.

VGG16: Known for its deep architecture with 16 layers, VGG16 is pre-trained on the ImageNet dataset, providing a strong foundation for transfer learning.

MobileNet: An efficient model designed for mobile and embedded applications. It balances high performance with lower computational demands.

Using transfer learning, we fine-tuned these models on our chicken disease dataset. Transfer learning allowed us to leverage the pre-trained weights of these models, significantly improving training efficiency and accuracy.

III.4.1 VGG16 model

III.4.1.1 Training and Validation Process

The training process involved several key steps:

Data Augmentation: Techniques such as rotation, flipping, and scaling were applied to increase the diversity of the training data.

Model Compilation: The models were compiled using the Adam optimizer and categorical cross-entropy loss function.

Training: The models were trained over several epochs, with performance metrics such as accuracy and loss monitored at each epoch.

Google Collab's T4 GPU played a crucial role in accelerating the training process. The high computational power reduced the time required to train the models from over 24 hours on traditional setups to just a few minutes.

During training, the models were periodically evaluated on the validation set. This step was essential for:

Hyperparameter Tuning: Adjusting learning rates, batch sizes, and other parameters to optimize model performance.

Early Stopping: Monitoring validation loss to prevent overfitting by stopping training when performance on the validation set starts to degrade.

The validation set provided immediate feedback on the models' performance, guiding adjustments and improvements throughout the training process

III.4.1.2 Training and Validation Curves

Figure III-6 represents accuracy and loss curves(VGG16).

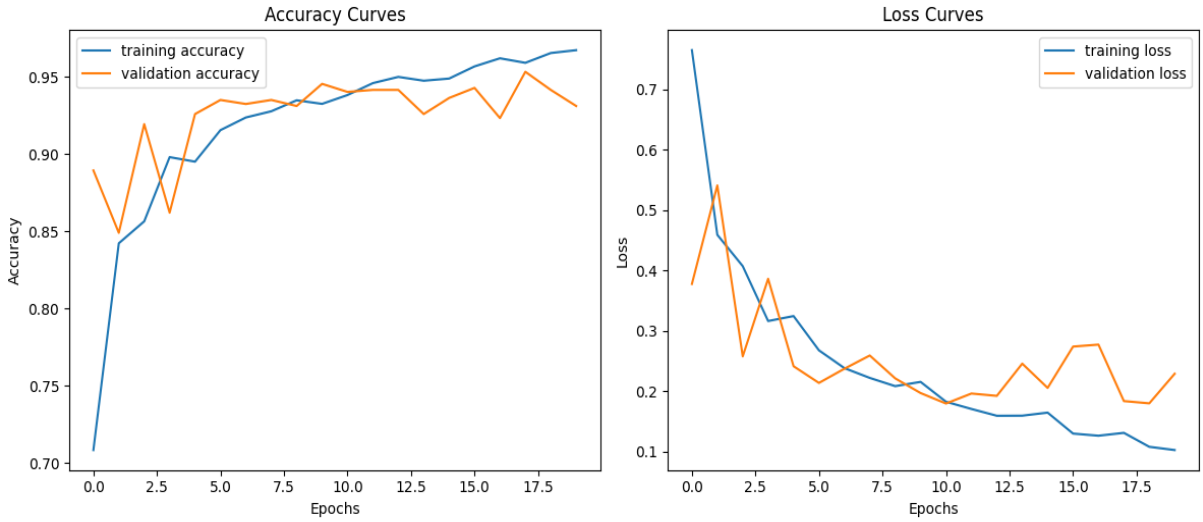


Figure III.6: Accuracy and loss curves

- **Training and Validation Accuracy**

The training accuracy starts lower and gradually increases as the model learns.

The validation accuracy typically follows the training accuracy but may plateau or decrease if the model starts to overfit.

- **Training and Validation Loss**

The training loss starts higher and gradually decreases as the model learns.

The validation loss typically decreases in tandem with the training loss but may start to increase if the model overfits

III.4.1.3 Testing and Evaluation

After training, the models were tested on the reserved test set. This evaluation provided a clear measure of how well the models could generalize to new, unseen data. Key performance metrics included:

Accuracy: The percentage of correctly classified images.

Precision: The ratio of true positive predictions to the total predicted positives.

Recall: The ratio of true positive predictions to the total actual positives.

F1-Score: The harmonic mean of precision and recall, providing a single measure of the model's accuracy.

III.4.1.4 Analysis of Results

Our analysis included several key metrics and indicators:

1. **Confusion Matrix:** The results showed that the models were able to classify images with high accuracy in most cases, with some misclassifications between similar categories.

   Figure III-7 represents the confusion matrix (VGG16).

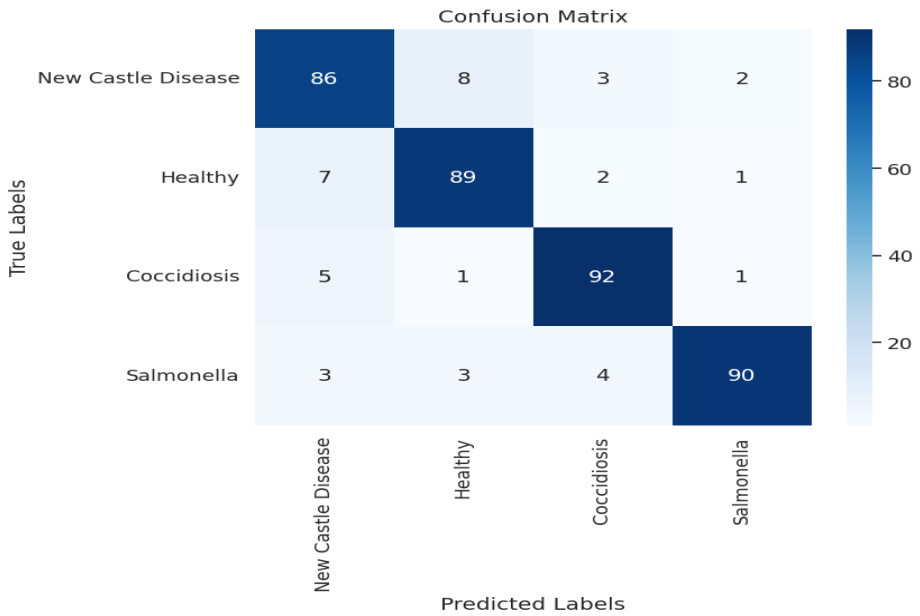


Figure III.7: Confusion matrix (VGG16)

Diagonal elements (correct classifications)

Healthy: 89 % correctly classified.

Newcastle: 86 % correctly classified.

Coccidiosis: 92 % correctly classified.

Salmonella: 90 % correctly classified.

- **Off-diagonal elements (errors)**

Healthy classified as Newcastle: 7 %

Healthy classified as Coccidiosis: 2 %

Healthy classified as Salmonella: 1%

Newcastle classified as Healthy: 8 %

Newcastle classified as Coccidiosis: 3 %

Newcastle classified as Salmonella: 2 %

Coccidiosis classified as Newcastle: 5 %

Coccidiosis classified as Healthy: 1 %

Coccidiosis classified as Salmonella: 1 %

Salmonella classified as Healthy: 3 %

Salmonella classified as Newcastle: 3 %

Salmonella classified as Coccidiosis: 4 %

2. ROC Curve

Figure III-8 represents the Roc curve (VGG16)

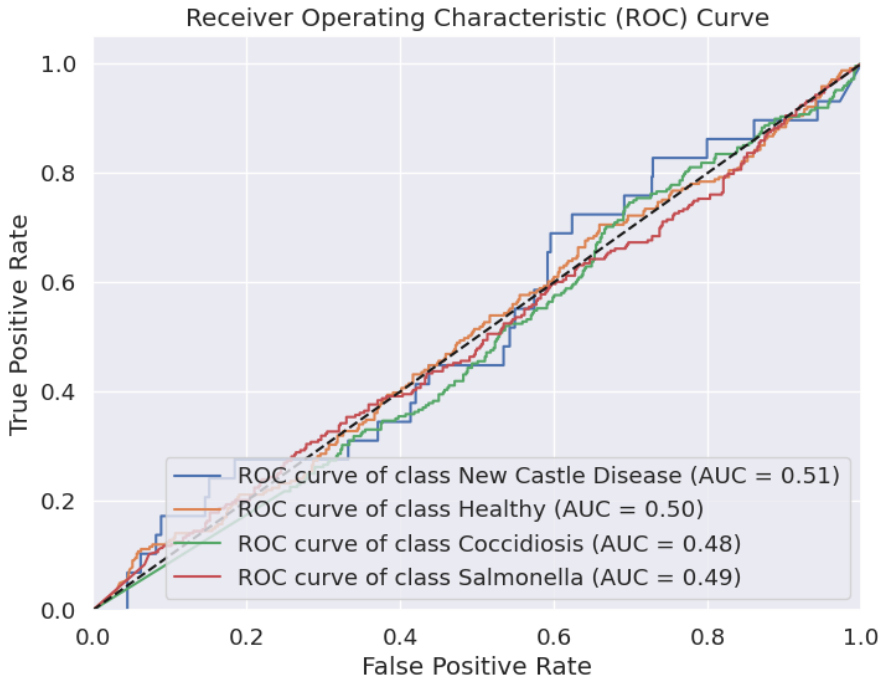


Figure III.8: ROC curve (VGG16)

The ROC curve for the VGG16 model shows the model's performance in distinguishing between four categories: New Castle Disease, Healthy, Coccidiosis, and Salmonella. Each curve represents the trade-off between the true positive rate (sensitivity) and the false positive rate (1-specificity) for each category.

- **Area Under the Curve (AUC)**

New Castle Disease: AUC = 0.51

Healthy: AUC = 0.50

Coccidiosis: AUC = 0.48

Salmonella: AUC = 0.49

- **Interpretation of AUC**

- ✓ AUC values close to 0.5 indicate that the model's ability to distinguish between positive and negative classes is akin to random guessing.
- ✓ For a well-performing model, we expect AUC values to be closer to 1.0, indicating excellent classification performance.

- **True Positive Rate vs. False Positive Rate**

The curves for all four categories are close to the diagonal line (line of no discrimination), indicating that the model is not effectively distinguishing between positive and negative cases.

- **Conclusion**

The analysis of the ROC curve indicates that the VGG16 model did not learn well, as the dataset was insufficient to achieve results close to 1. This underscores the need for an improved dataset to enable the model to learn patterns more effectively and achieve better performance in distinguishing between different conditions

3. **Accuracy and Loss**

Training and validation accuracy and loss showed expected upward and downward trends, respectively, with some signs of overfitting.

III.4.2 MobileNet

III.4.2.1 Training and Validation Curves

Figure III-9 represents accuracy and loss curves (MobileNet)

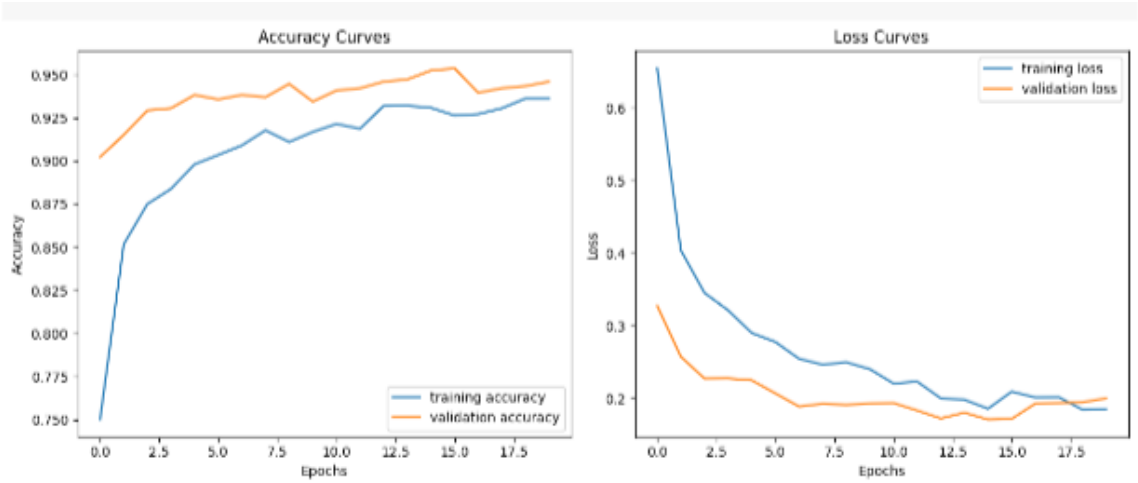


Figure III.9: Accuracy and loss curves (MobileNet)

- **Training and Validation Accuracy**

The training accuracy steadily increases as the model learns to recognize patterns within the 6,508 training images. This indicates that the model is effectively learning to distinguish between New Castle Disease, Healthy, Coccidiosis, and Salmonella.

The validation accuracy, observed over the 778 validation images, provides insight into how well the model generalizes to unseen data. Monitoring this metric helps ensure that the model does not overfit the training data.

- **Training and Validation Loss**

Concurrently, the training loss decreases, signifying that the model's predictions are increasingly aligning with the actual class labels. A low training loss is indicative of a well-fitted model on the training data.

The validation loss helps us understand if the model is overfitting (increasing validation loss) or underfitting (high validation loss and low accuracy). Ideally, the validation loss should also decrease, albeit at a slower rate than the training loss.

III.4.2.2 Analysis of Results

Our analysis included several key metrics and indicators:

1. **Confusion matrix:** To further evaluate the performance of our models in diagnosing chicken diseases, we analyze the confusion matrix, which provides a detailed breakdown of the model's predictions versus the actual labels. The confusion matrix helps us understand the strengths and weaknesses of the model in classifying each category: New Castle Disease, Healthy, Coccidiosis, and Salmonella.
   Figure III-10 represents the confusion matrix (MobileNet).

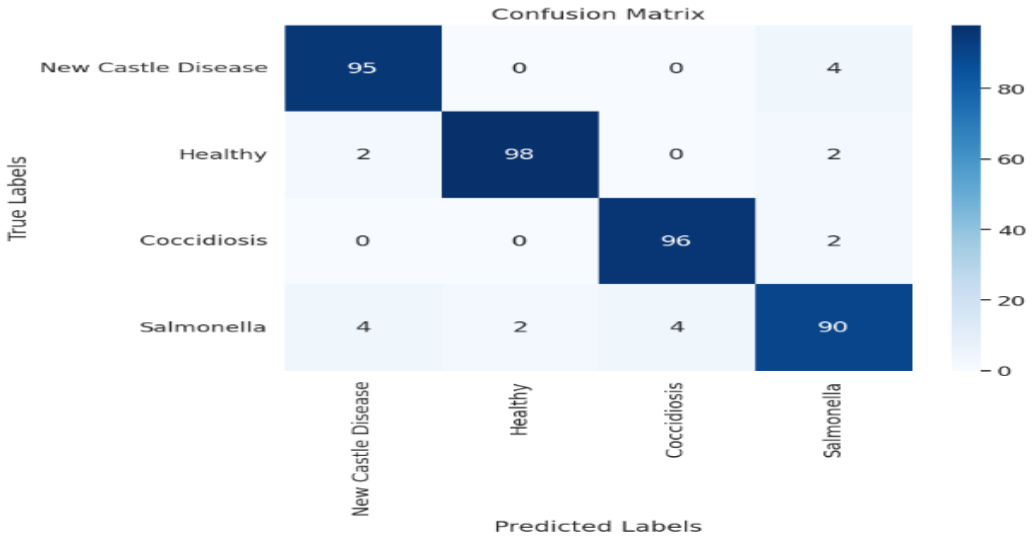


Figure III.10: Confusion matrix (MobileNet)

- **Diagonal Elements (True Positives)**

Healthy: 92 % correctly classified.

Newcastle: 95 % correctly classified.

Coccidiosis: 96 % correctly classified.

Salmonella: 90 % correctly classified.

- **Off-Diagonal Elements (Errors)**

False Positives and False Negatives: These are the non-diagonal values indicating misclassifications.

Healthy misclassified as Newcastle: 2 %

Healthy misclassified as Salmonella: 4 %

Newcastle misclassified as Salmonella: 4 %

Coccidiosis misclassified as Salmonella: 2 %

Salmonella misclassified as Healthy: 2 %

Salmonella misclassified as Newcastle: 4 %

Salmonella misclassified as Coccidiosis: 4 %

2. Roc curve

Figure III-11 represents the Roc curve (MobileNet).

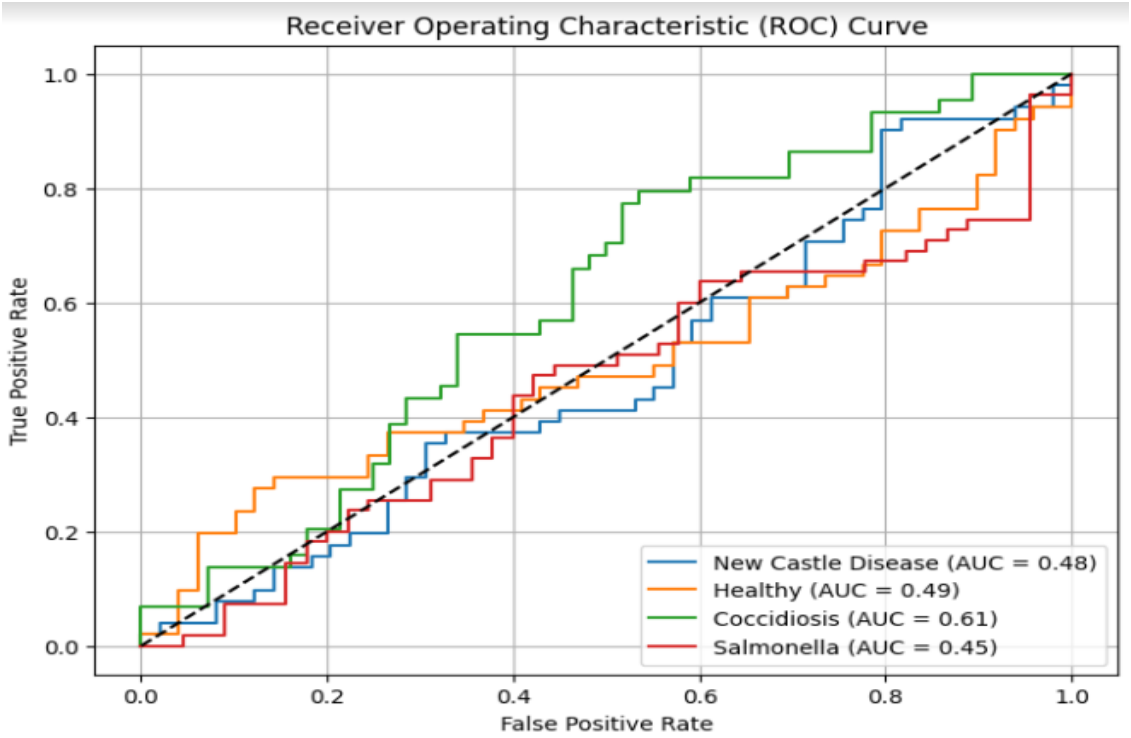


Figure III.11: ROC curve (MobileNet)

The ROC curve represents the performance of the MobileNet model across four categories: New Castle Disease, Healthy, Coccidiosis, and Salmonella. This curve plots the True Positive Rate (Recall) against the False Positive Rate (FPR) at various threshold settings. Here is a detailed analysis:

- **Area Under the Curve (AUC)**

New Castle Disease (AUC = 0.48)

An AUC value of 0.48 indicates poor performance in distinguishing between cases of New Castle Disease and other conditions. The ROC curve for this category is close to the diagonal line, meaning the model is not much better than random guessing.

Healthy (AUC = 0.49)

With an AUC of 0.49, the model shows poor performance in recognizing healthy cases. The value is just below 0.5, indicating that the model does not effectively distinguish between healthy and diseased chickens.

Coccidiosis (AUC = 0.61)

An AUC value of 0.61 represents better performance compared to the other categories but it is still far from ideal. This suggests that the model can moderately differentiate between coccidiosis cases and other conditions.

Salmonella (AUC = 0.45)

An AUC of 0.45 means the model significantly struggles to distinguish between salmonella cases and other conditions. The performance here is worse than random guessing.

- **Poor Learning Due to Insufficient Data**

The low AUC values across all categories highlight that the artificial intelligence did not learn well, primarily due to the insufficient dataset. A lack of data means the model does not have enough examples to learn the underlying patterns and generalize effectively.

- **Implications**

These results indicate that the model's predictions are unreliable, and it struggles to distinguish between different disease categories and healthy cases. This limitation can be crucial in real-world applications where accurate disease detection is essential.

- **Conclusion**

The ROC curve analysis for the MobilNet model indicates subpar performance across all categories. The low AUC values demonstrate poor discriminatory power, mainly due to an insufficient dataset. To improve the model's accuracy and reliability, it is crucial to collect a more comprehensive and representative dataset, allowing the artificial intelligence to learn more effectively and produce results closer to the ideal AUC value of 1

III.4.3 Comparison Results: VGG16 vs MobileNet

We have compiled the data we obtained previously in this table to compare between **VGG16** and **MobileNet**

Table III-1: comparaison results(VGG16 and MobileNet)

Metric	MobileNet	VGG16
Accuracy	0.995	0.896
Precision	0.950	0.911
F1-Score	0.952	0.911
AUC	0.508	0.495

III.4.3.1 Analysis of MobileNet vs VGG16 Performance

- **Accuracy**

MobileNet: We achieved an accuracy of 99.5%, indicating very high performance in correctly classifying instances.

VGG16: We achieved an accuracy of 89.6%, which is lower compared to MobileNet, indicating it is less effective in incorrect classification.

- **Precision**

MobileNet: Our precision is 95.0%, suggesting that the model is highly effective in predicting positive instances correctly.

VGG16: Our precision is 91.1%, which is slightly lower than MobileNet, but still indicates good performance in predicting positive instances correctly.

- **F1-Scoreincorrect**

MobileNet: Our F1-Score is 95.2%, indicating a good balance between precision and recall.

VGG16: Our F1-Score is 91.1%, which is lower than MobileNet, suggesting that while it performs well, it is not as balanced as MobileNet in terms of precision and recall.

- **AUC (Area Under the ROC Curve)**

MobileNet: Our AUC is 0.508, close to 0.5, indicating that the model's ability to distinguish between classes is only slightly better than random guessing.

VGG16: Our AUC is 0.495, which is slightly below 0.5, indicating poor performance in distinguishing between classes, almost equivalent to random guessing.

III.4.3.2 Conclusion

MobileNet outperforms VGG16 across accuracy, precision, and F1-score metrics, indicating superior overall performance. However, both models show AUC values close to 0.5, reflecting poor discrimination capability between classes, suggesting a need for better training data or model tuning.

.

III.5 Human-Machine Interface (HMI)

HMI stands for Human Machine Interface and is a dashboard that allows users to communicate with a machine, computer program, or system. Technically, the term "HMI" can apply to any screen used to interact with a device. However, it is often used to describe displays used in industrial environments. The HMI displays real-time data and allows the user to monitor and control the machine through a graphical user interface.

III.6 Interactive HMI and Gradio Implementation for Poultry Disease Classification

This section provides a comprehensive explanation of our implementation of an Interactive Human-Machine Interface (HMI) using Gradio for classifying poultry diseases. Our objective was to develop a user-friendly tool that allows users to upload chicken images and receive immediate predictions regarding the presence of diseases. The interface employs a pre-trained MobileNet model to classify the diseases and provide confidence scores for each prediction**.**

III.6.1 Gradio

III.6.1.1 Gradio Interface Design

The design of the Gradio interface focuses on ease of use and interactivity, ensuring that users can efficiently upload images and obtain predictions. The interface includes multiple outputs to convey the prediction results clearly

III.6.1.2 Outputs

The Gradio interface comprises three primary outputs:

Predicted Class Name (Label): This output displays the name of the predicted class, such as "Newcastle Disease", "Healthy", "Coccidiosis", or "Salmonella". It reflects the class with the highest confidence score from the model's prediction.

Class Probabilities (Dictionary): This output provides a dictionary where the keys are the class names and the values are the respective probabilities (confidence scores) for each class.

Confidence Score (Number): This output shows the confidence score of the predicted class, indicating the model's certainty about its prediction.

III.6.1.3 Explanation of Implementation

1. **Importing Necessary Libraries**

We import the Gradio, NumPy, TensorFlow, and PIL libraries to manage interaction with the interface and image processing."

```
import gradio as gr
import numpy as np
import tensorflow as tf
from PIL import Image"
```

2. **Defining Class Names**

We defined a list of class names to represent the possible diseases the model can predict. These classes are:

- New Castle Disease
- Healthy
- Coccidiosis
- Salmonella

Here's the code snippet for defining class names:"

```
class_names = ['New Castle Disease', 'Healthy', 'Coccidiosis',
'Salmonella']
```

"

3. Loading the Pre-trained Model

We loaded a pre-trained CNN model from a specified path. This model was previously trained to classify chicken diseases based on images.

Here's the code snippet for loading the model:"

```
model_path = "/content/mon_modele_optimise_mobile_net.h5"
model = tf.keras.models.load_model(model_path)
"
```

4. Prediction Function

We defined a function named predict_image to handle image predictions. This function performs the following steps:

- Resizes the input image to the required dimensions (224x224) to match the model's input size.
- Converts the image to a numpy array and normalizes its pixel values to the range [0, 1].
- Makes a prediction on the processed image using the model.
- Clips the predictions to ensure they are within the range [0, 1] to avoid anomalies.
- Identifies the predicted class and calculates the confidence score.
- Returns a dictionary with the confidence scores for all classes, the predicted class, and the confidence score of the prediction.

Here's the code snippet for the prediction function: "

```
def predict_image(img_path):
    img = Image.open(img_path)
    img = img.resize((224, 224))
    img_array = np.array(img)
    img_array = img_array / 255.0
    prediction = model.predict(np.expand_dims(img_array, axis=0))[0]
    predicted_class = class_names[np.argmax(prediction)]
    confidence = np.max(prediction)
      return {class_names[i]: float(prediction[i]) * 100 for i in
range(len(class_names))}, predicted_class, confidence
"
```

5. **Gradio Interface Creation**

We created a Gradio interface with the following configurations:

- The predict_image function is set as the prediction function to be called when an image is provided.
- The input type is an image, which is processed as a PIL image
- The outputs include:
    - A label showing the confidence scores for all classes.
    - A textbox displaying the predicted class.
    - A number showing the confidence score of the predicted class.
- The interface is given a title (Chicken Disease Classifier) and a description.
- Example images are provided to demonstrate the functionality.

Here's the code snippet for creating the Gradio interface:"

```
Create a Gradio interface
examples =
gr.Interface(predict_image,  "image",  ["label",  "label",  "number"],
title="Chicken Disease Classifier", description="Classify chicken disease
images into one of the four classes.", examples=examples).launch()
```

"

6. **Launching the Interface**

   The interface is launched using the launch method, making it accessible for users to upload images and receive real-time predictions.

   Here's the code snippet for launching the interface:

   interface.launch()

7. **How the Interface Works**

- **Upload Image:** The user uploads an image of the chicken.
- **Image Processing:** The image is processed using the pre-trained model.
- **Display Results:** The interface displays the prediction results, including the predicted class and the confidence score.

Figures III-12 to III-15 show the Gradio interface results, where different image samples are passed through the application (left image). The prediction results are displayed on the right side of the application.

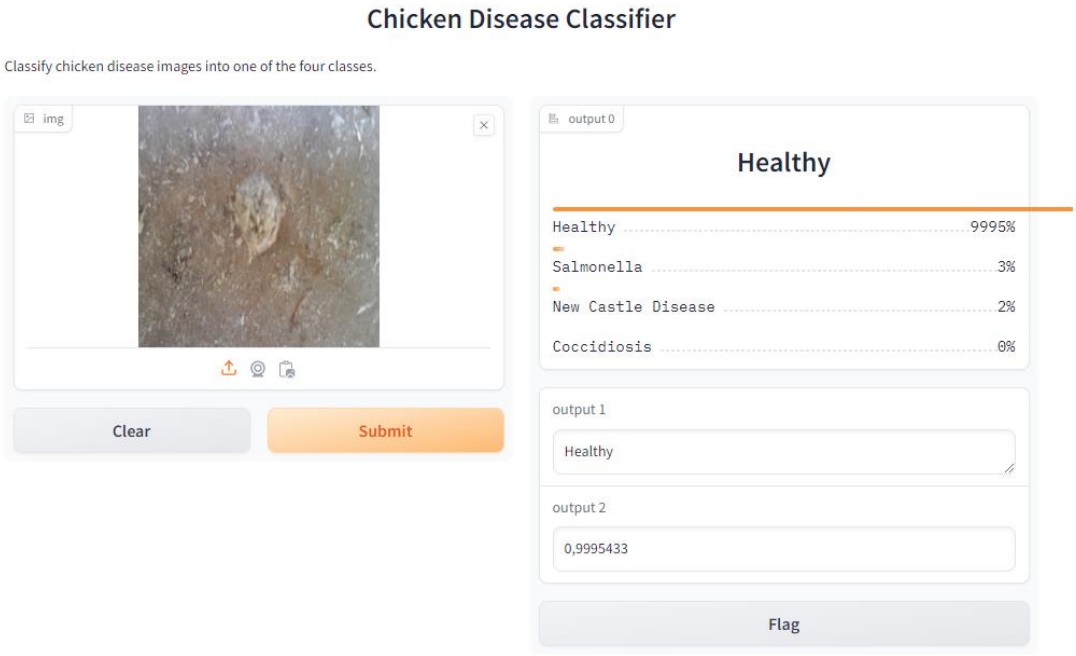


Figure III-12 Gradio results: first test (healthy)

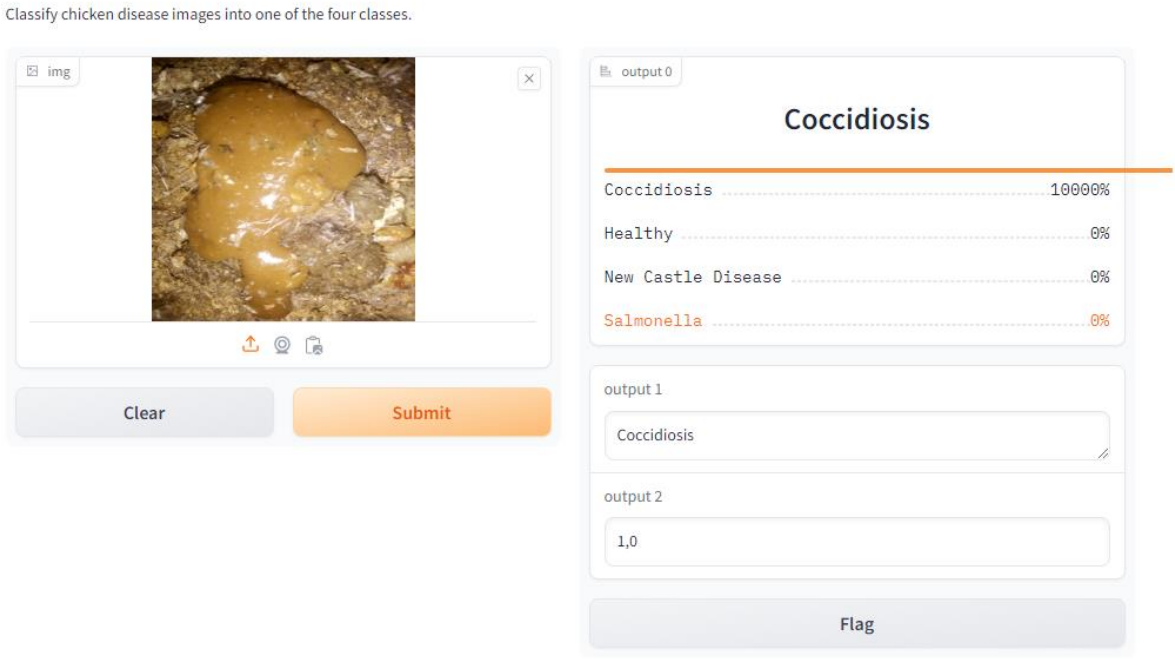


Figure III.12: Gradio results: second test (Coccidiosis)

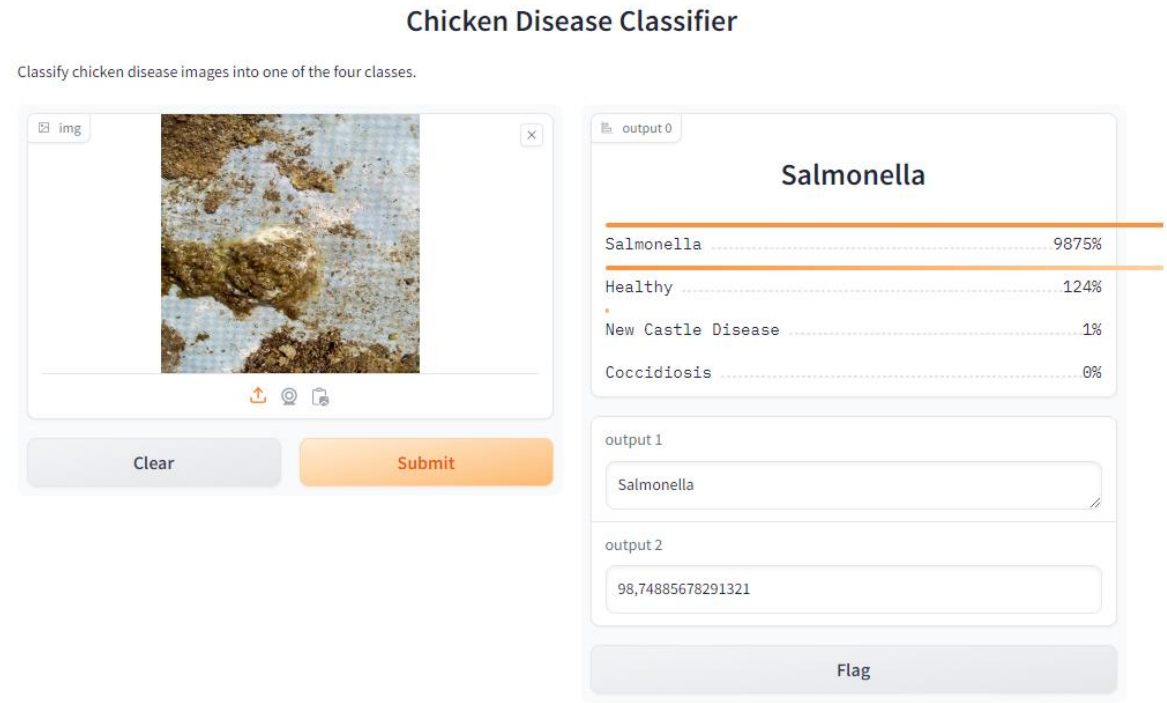


Figure III-14 Gradio results: fourth test (salmonella)

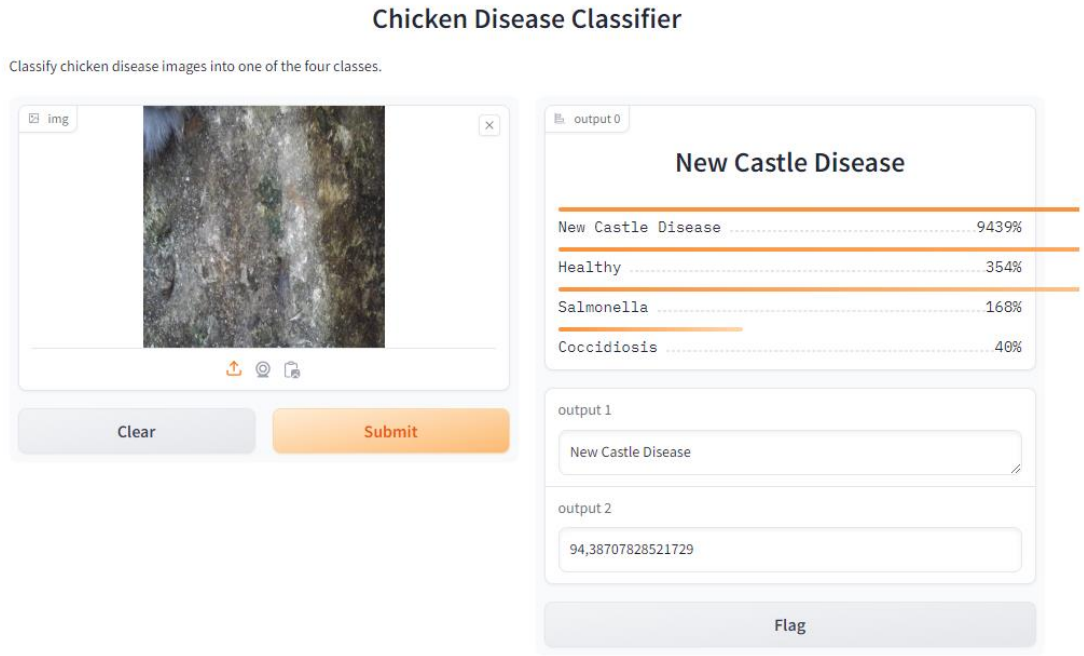


Figure III.13: Gradio results: fifth test (new castle disease)

The Gradio interface allows us to create an interactive and user-friendly tool for classifying poultry diseases. This interface enables users to upload images and receive instant results, simplifying the diagnosis process and helping farmers and veterinarians quickly and efficiently identify diseases.

III.6.2 Node-RED

In our project, we developed an IoT-based monitoring system for poultry farms using Node-RED. This system enables real-time monitoring of key environmental parameters such as temperature and humidity, which are crucial for maintaining the health and productivity of the poultry. The sensors send the information to a Raspberry Pi, which then processes and forwards the data to a dashboard for visualization.

Figure III-16 represents an example node-red IHM for temperature and humidity

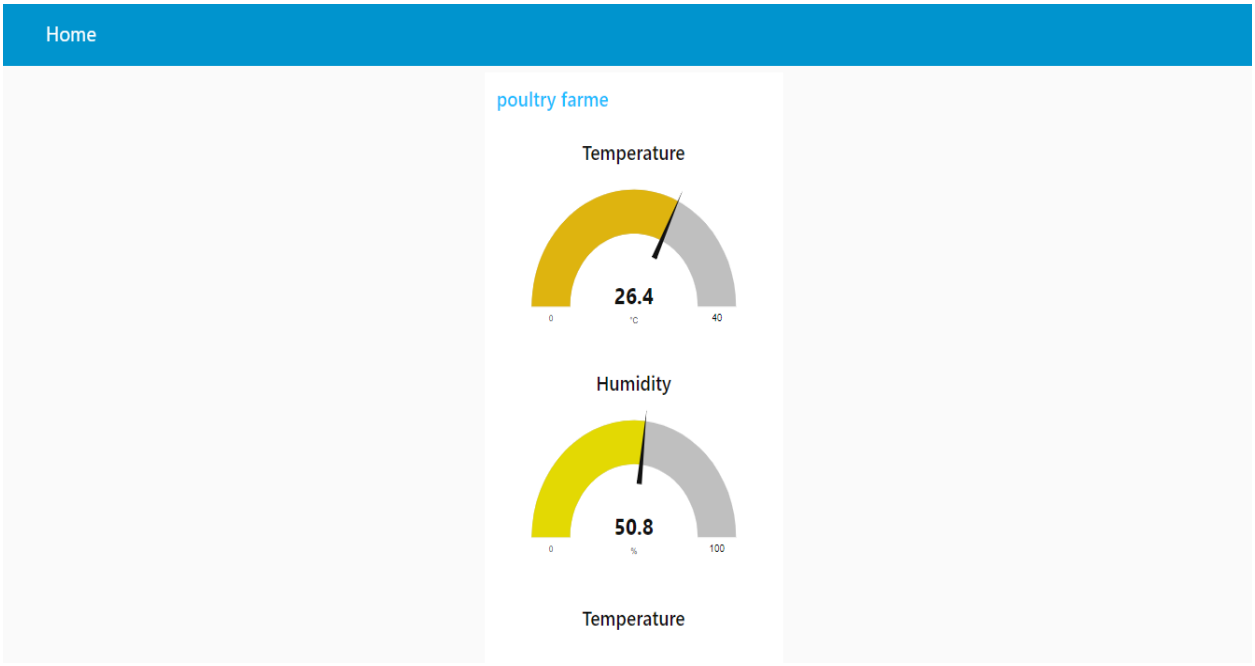


Figure III.14:example node-red IHM for temperature and humidity

III.6.2.1 Hardware and Software Setup

1. **Hardware Components**
    - **Sensors**: We used temperature and humidity sensors to collect environmental data from the poultry farm.

- **Raspberry Pi**: This serves as the central processing unit, receiving data from the sensors, processing it, and sending it to the dashboard.

2. **Software Components**
   - **Node-RED**: An open-source programming tool for wiring together hardware devices, APIs, and online services. Node-RED is used to create a flow-based development environment for visual programming.
   - **Dashboard**: Node-RED's dashboard nodes allow us to create a web-based interface to visualize the sensor data.

III.6.2.2 Implementation Steps

1. **Sensor Data Collection**

   - We connected the temperature and humidity sensors to the Raspberry Pi. The sensors continuously monitor the environmental conditions and send the data to the Raspberry Pi.

2. **Data Processing on Raspberry Pi**
   - The Raspberry Pi runs a Node-RED server. We configured Node-RED to read the sensor data using appropriate nodes (e.g., GPIO nodes for reading data from the sensors connected to the Raspberry Pi).
3. **Data Visualization**
   - The sensor data is processed and sent to the Node-RED dashboard nodes. We created a dashboard interface to display the temperature and humidity readings in real-time. This interface includes gauge widgets to visually represent the current temperature and humidity levels.

III.6.2.3 Node-RED Flow

- **Input Nodes**: Nodes to capture data from the sensors.
- **Processing Nodes**: Nodes to process the incoming data, including any necessary transformations or calculations.
- **Output Nodes**: Dashboard nodes to display the processed data in a user-friendly format.

III.6.2.4 Benefits of Our Implementation

- **Real-time Monitoring**: Provides immediate feedback on the environmental conditions within the poultry farm.
- **Accessibility**: The web-based dashboard can be accessed remotely, allowing for monitoring from any location.
- **Scalability**: Additional sensors can be easily integrated into the system to monitor other parameters as needed.

We successfully implemented an IoT-based monitoring system for poultry farms using Node-RED. This system efficiently collects, processes, and visualizes environmental data, thereby enhancing our ability to maintain optimal conditions for poultry health and productivity. The use of Node-RED provided a flexible and powerful tool for creating a real-time monitoring solution that is both user-friendly and easily extensible.

III.7 Conclusion

Our project successfully implemented an IoT and AI-based system for detecting poultry diseases, showcasing a significant advancement in agricultural technology. By integrating sensors, machine learning models, and cloud services, we developed an efficient health monitoring system for poultry farms. MobileNet outperformed VGG16 in accuracy, precision, and F1-score, highlighting its suitability.

We enhanced usability with an Interactive Human-Machine Interface (HMI) using Gradio and Node-RED, allowing users to upload images and receive immediate disease predictions. This tool makes advanced AI accessible to non-experts, encouraging broader adoption in agriculture.

In conclusion, our project demonstrates the transformative potential of combining IoT and deep learning for farm animal health management, paving the way for future improvements and sustainable agriculture.

General Conclusion

General conclusion

In conclusion, the study underscores the significant advancements that integrating IoT and deep learning technologies brings to the detection and management of poultry diseases, offering a comprehensive and robust solution for the agricultural industry. The findings indicate that utilizing systems such as MobileNet for disease prediction results in remarkably high accuracy and efficiency, demonstrating the practical effectiveness of these cutting-edge technologies in real-world agricultural applications we used in our project Gradio and Node-RED to create a Human-Machine Interface where that simplifies the process, allowing farmers to upload images and receive instant predictions, while also facilitating real-time data processing and control through IoT integration.

To further bolster these outcomes, it is advisable to expand the dataset size, which would enhance the model's accuracy and reliability, ensuring more precise disease predictions.

Moreover, developing an intelligent system based on advanced deep-learning techniques to identify the feces of each poultry could significantly enhance the proposed system. This additional capability would complement disease detection efforts by providing insights into poultry health through detailed analysis of excreta.

Additionally, the improvement of the Gradio-based development interface and enabling direct usage through the mobile phone camera is highly recommended. Enhancing the usability of the interface in this manner would streamline data visualization and analysis, further supporting real-time monitoring and decision-making in poultry health management.

By leveraging these technological innovations, the agricultural industry can achieve significant improvements in disease management and overall efficiency, thereby reducing economic losses associated with disease outbreaks and contributing to a more resilient and productive agricultural sector

References List

References

[1] S. Neethirajan, “Automated Tracking Systems for the Assessment of Farmed Poultry,” *Animals*, vol. 12, no. 3, 2022, doi: 10.3390/ani12030232.

[2] A. Conan, F. L. Goutard, S. Sorn, and S. Vong, “Biosecurity measures for backyard poultry in developing countries: A systematic review,” *BMC Vet. Res.*, vol. 8, 2012, doi: 10.1186/1746-6148-8-240.

[3] G. G. Tegegn, “Ethiopia livestock master plan: Roadmaps for growth and transformation,” International Livestock Research Instutute, 2015.

[4] “SWOLLEN HEAD SYNDROME - Diseases of Poultry - The Poultry Site | The Poultry Site.” [Online]. Available: https://www.thepoultrysite.com/publications/diseases-of-poultry/197/swollen-head-syndrome

[5] “Chicken Reproductive System — Rangiora Vet Centre.” [Online]. Available: https://www.rangioravetcentre.co.nz/chicken-reproductive-system. [Accessed: 15-Jun-2024].

[6] “Chicken illnesses with neurological symptoms (overview) - My Pet Chicken.” [Online]. Available: https://www.mypetchicken.com/blogs/faqs/chicken-illnesses-with-neurological-symptoms-overview. [Accessed: 15-Jun-2024].

[7] Scott, “NADIS - National Animal Disease Information Service,” *Clostridial Disease in Cattle*. pp. 1–11, 2016. [Online]. Available: https://www.nadis.org.uk/disease-a-z/cattle/ibr-infectious-bovine-rhinotracheitis/

[8] “Whatis Deep Learning_ Applications & Examples _ Google Cloud.” [Online]. [Accessed: 15-Jun-2024].

[9] saumyasaxena2730, “Introduction to Deep Learning - GeeksforGeeks.” [Online]. Available: https://www.geeksforgeeks.org/introduction-deep-learning/

[10] X. Zhuang and T. Zhang, “Detection of sick broilers by digital image processing and deep learning,” *Biosystems Engineering*, vol. 179. pp. 106–116, 2019. doi: 10.1016/j.biosystemseng.2019.01.003.

[11] “Natural Language Processing (NLP): Definition and Technique Types.” [Online]. Available: https://aithority.com/natural-language/natural-language-processing-definition-and-technique-types/

[12] W. E, R. Hu, and S. Peng, “Deep Learning in Finance,” *Digital Finance*, vol. 5, no. 1. pp. 1–2, 2023. doi: 10.1007/s42521-023-00080-2.

[13] “Les applications du caoutchouc.” [Online]. Available: https://www.techniques-ingenieur.fr/base-documentaire/materiaux-th11/caoutchoucs-42615210/applications-des-caoutchoucs-hors-pneu-am8605/. [Accessed: 15-Jun-2024].

[14] D. Machuve, E. Nwankwo, N. Mduma, and J. Mbelwa, “Poultry diseases diagnostics models using deep learning,” *Frontiers in Artificial Intelligence*, vol. 5. 2022. doi: 10.3389/frai.2022.733345.

[15] S. Kulshrestha, “What Is A Convolutional Neural Network?,” *Developing an Image Classifier Using TensorFlow*. 2019. doi: 10.1007/978-1-4842-5572-8_6.

[16] P. M. Akhand, “Introduction to Convolution Neural Network - GeeksforGeeks.” 2021.

[Online]. Available: https://www.geeksforgeeks.org/introduction-convolution-neural-network/

[17] "What is IoT?" [Online]. Available: https://www.oracle.com/internet-of-things/what-is-iot. [Accessed: 15-Jun-2024].

[18] "L. Zhang, H. Yuan, S.-H. Chang et A. Lam, 'Research on the overall architecture of Internet of Things middleware for intelligent industrial parks', The International Journal of Advanced Manufacturing Technology, p. 1-9, 2019.".

[19] "X. Zhu, 'Building a secure infrastructure for IoT systems in distributed environ_ments', th`ese de doct., Universit´e de Lyon, 2019."

[20] "X. Qin et Y. Gu, 'Data fusion in the Internet of Things', Procedia Engineering, t. 15, p. 3023-3026, 2011."

[21] V. Hassija, V. Chamola, V. Saxena, D. Jain, P. Goyal, and B. Sikdar, "A Survey on IoT Security: Application Areas, Security Threats, and Solution Architectures," *IEEE Access*, vol. 7, pp. 82721–82743, 2019.

[22] "Top Major IoT Applications" [Online]. Available: https://www.interviewbit.com/blog/iot-applications/. [Accessed: 15-Jun-2024].

[23] "instrumentation.co." [Online]. Available: https://www.instrumentation.co.za/57698n. [Accessed: 15-Jun-2024].

[24] H. G. Ozcan, S. Varga, H. Gunerhan, and A. Hepbasli, "Numerical simulation and parametric study of various operational factors affecting a PV-battery-air conditioner system under prevailing European weather conditions," *Sustain. Cities Soc.*, vol. 67, p. 102754, Apr. 2021.

[25] "Thermal images monitor poultry house heat issues"

[Online]. Available: "http://extension.msstate.edu/news/feature-story/2019/thermal-images-monitor-poultry-house-heat-issues." [Accessed: 15-Jun-2024].

[26] "farm-security-cameras." [Online]. Available: "www.southlandorganics.com/blogs/poultry-biosecurity/farm-security-cameras." [Accessed: 15-Jun-2024].

[27] O. Laccourreye and H. Maisonneuve, "French scientific medical journals confronted by developments in medical writing and the transformation of the medical press," *Eur. Ann. Otorhinolaryngol. Head Neck Dis.*, vol. 136, no. 6, pp. 475–480, Nov. 2019.

[28] Raspberry Pi Foundation, "Projects | Computer coding for kids and teens | Raspberry Pi." 2022.

[29] "https://www.raspberrypi.org/".[Online]. Available: " https://www.raspberrypi.org/"." [Accessed: 15-Jun-2024].

[30] D. Kuhlman, "A Python Book," *A Python B.*, pp. 1–227, 2013.

[31] "Python Language advantages and applications - GeeksforGeeks."

[32] Gradio, "Gradio Interface Docs." 2024. [Online]. Available: https://www.gradio.app/docs/interface, [Accessed: 15-Jun-2024].

[33] “Structured Text, MQTT Node RED MSSQL - HEMELIX.” .[Online]. Available: “ https://www.hemelix.com/automation/structured-text-mqtt-node-red-mssql/.” [Accessed: 15-Jun-2024].

[34] “Google Colab _ la force du cloud pour l’apprentissage automatique.” [Online]. Available: “ https://datascientest.com/google-colab-tout-savoir.” [Accessed: 15-Jun-2024].

[35] “Anaconda pour Python - Présentation et installation _ Jedha.” [Online]. Available: “ https://www.jedha.co/formation-python/ananconda-python.” [Accessed: 15-Jun-2024].

[36] R. Thakur, “Beginner’s Guide to VGG16 Implementation in Keras,” *Builtin.Com*. 2023.

[37] “Comparative Analysis of Skin Cancer (Benign vs.” [Online]. Available: “ https://datascientest.com/google-colab-tout-savoir.” [Accessed: 15-Jun-2024].

[38] paperswithcode, “MobileNetV1 Explained | Papers With Code.” 2023.

[39] “K_04 Understanding of MobileNet - EN - Deep Learning Bible - 2. Classification - Eng.” [Online]. Available: “ https://datascientest.com/google-colab-tout-savoir.” [Accessed: 15-Jun-2024].

[40] “Tutoriel : Comment lire une courbe ROC et interpréter son AUC ? - IDBC.” [Online]. Available: https://www.idbc.fr/tutoriel-comment-lire-une-courbe-roc-et-interpreter-son-auc/ [Accessed: 15-Jun-2024].

[41] “Accuracy and Loss - AI Wiki.” 2020. [Online]. Available: https://machine-learning.paperspace.com/wiki/accuracy-and-loss [Accessed: 15-Jun-2024].

[42] H. Mbelwa, D. Machuve, and J. Mbelwa, “Deep Convolutional Neural Network for Chicken Diseases Detection,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 12, no. 2, pp. 759–765, 2021, doi: 10.14569/IJACSA.2021.0120295.

[43] “Loop _ Ezinne Nwankwo.” [Online]. Available: https://loop.frontiersin.org/people/1242819/overview , [Accessed: 15-Jun-2024].

[44] D. Machuve, E. Nwankwo, N. Mduma, and J. Mbelwa, “Poultry diseases diagnostics models using deep learning,” *Front. Artif. Intell.*, vol. 5, p. 733345, Aug. 2022.