الجمـــهورية الجزائرية الديمـــقراطية الشعـــبية
Democratic And Popular Republic Of Algeria
وزارة التعـلـيـم الـعـالـــي والـبـحـث الـعـلـمي
Ministry Of Higher Education And Scientific Research
جـامـعـة عمّـــار ثـلـيـــجـــي بالاغـــــــــواط
University Of Amar Telidji Laghouat
كـلـيـــــة التكنولوجـــــــيـــا
Faculty Of Technology
قســـم الالكترونيك
Electronics Department

MASTER DEGREE THESIS

Option: Networks and telecommunications

Presented by:
Atallah Abdelali

THEME

Face sheep recognition using artificial intelligence

Publicly defended before a jury composed of :

Surname and First Name	Grade	Quality
Nedjma Abdelhafidhi	MCB	President
Abdelkader Birane	MCB	Examiner
Kamal Sehairi	MCA	Supervisor
Mohammed Redha Bouzidi	MCB	Co-Supervisor

Academic Year 2024/2025
28/06/2025

بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ

الإهداء

الحمد لله رب العالمين،
حمدًا طيبًا مباركًا فيه، ملء السماوات والأرض، عدد ما كان وعدد ما سيكون،
لك الحمد يا رب على ما أنعمتَ به من توفيق وتيسير، وعلى ما رزقتني من الصبر والعزم لإتمام هذا العمل.

ثم...

إلى والدي العزيز:
مصدر قوتي وقدوتي في الكفاح والعمل،
إليكَ يا من غرست فيَّ القيم والمبادئ، وكنتَ لي سندًا لا يزول،
أهدي هذا العمل المتواضع تقديرًا لعطائك وتضحياتك.

وإلى والدتي الحبيبة:
نبع الحنان وروح الدعاء،
يا من كنتِ السند في كل مراحل الحياة،
أهديكِ هذا الإنجاز عرفانًا لحنانكِ ودعمكِ اللامحدود.

والى اخي واخواتي الاعزاء:
أنتم الرفاق في رحلة العمر،
لكم مني هذا الإهداء تقديرًا لكل لحظة قرب ومحبة وتشجيع.

وإلى أعمامي (احمد و عبدالمالك) وعمّاتي:
أهديكم هذا العمل احترامًا لمكانتكم، ووفاءً لكل ما قدمتموه من دعم ومساندة.

وإلى أخوالي (عبدالكريم و الطاهر) وخالاتي الكرام:
لكم جزيل الشكر والامتنان، وهذا الإهداء رمز محبة وتقدير.

وإلى جدتي الغالية:
أهديكِ هذا العمل وفاءً لحنانكِ ودعواتكِ.

وإلى خالتي الكبرى:
أنتِ التي كنتِ دومًا أمًّا ثانية وملجأ حنان،
لكِ مني هذا الإهداء عرفانًا لطيبة قلبكِ ومواقفك النبيلة.

وإلى أبناء أعمامي الذين كنتم دومًا شركاء اللحظات التي لا تُنسى،
أهديكم هذا الجهد تعبيرًا عن اعتزازي بصلتنا، ووفائي لأخوتنا.

إلى أصدقائي الأعزاء:
أنتم الذين كنتم لي عونًا في طريق العلم والحياة،
بصدق مشاعركم ونُبل صحبتكم.

وإلى أصدقاء والدي الأعزاء:
من كان لهم في حياتنا الأثر الطيب، والدعم الصادق،
أهديكم هذا العمل تقديرًا لمواقفكم النبيلة، وامتنانًا لصلتكم الوفية التي كانت دومًا مصدر احترام واعتزاز.

وإلى كل من عرفني من قريب أو بعيد،
أهديكم هذا العمل المتواضع شكرًا لكل كلمة طيبة، وكل دعم صادق، وكل دعاء خفي.

شكر وتقدير

بحمد الله وتوفيقه، وبتيسيرٍ منه، أنهينا هذا العمل بعد رحلة مليئة بالتحديات والعزم. فله الحمد أولاً وآخراً، أن منحنا القوة والصبر للوصول إلى هذه المرحلة

نتوجه بخالص الشكر والتقدير إلى **الدكتور كمال سحيري**، الذي كان خير موجّه ومرشد، فبفضل توجيهاته السديدة، ومتابعته الدقيقة، وتشجيعه المتواصل، استطاع هذا العمل أن يرى النور بالشكل الذي نأمل. لقد كان لدعمه العلمي والإنساني بالغ الأثر في هذا العمل.

كما نُعبّر عن شكرنا لـ **الدكتور محمد بوزيدي**،على توجيهاته القيمة والذي لم يبخل علينا بعلمه ، ومساهمته في إطار هذا العمل.

ولا يفوتنا أن نتوجه بالشكر لجميع أساتذتنا الكرام الذين ساهموا في تكويننا، ولكل أعضاء لجنة المناقشة الذين تكرموا بقبول تقييم هذه الأطروحة وإثرائها بملاحظاتهم القيمة.

كما نخص بالشكر عائلاتنا وأصدقاءنا الأوفياء، الذين دعمونا معنويًا طوال فترة الدراسة، وشاركونا لحظات التعب والنجاح.

ولكل من ساهم بكلمة أو بدعاء أو بموقف، نُهدي له بالغ امتناننا، ونعترف له بالفضل.

Abstract

This thesis focuses on designing and developing an intelligent system for sheep identification using artificial intelligence and computer vision techniques.

The system follows multiple stages, starting with data collection, face detection from images, feature extraction for each sheep, and finally classification using machine learning algorithms.

The main goal of this work is to help farmers by providing an automatic and accurate tool for sheep identification without the need for manual checking each time.

The system was applied and tested on real sheep images collected in a local environment. The results showed promising performance despite some challenges.

Key words: Computer Vision ,Face Recognition ,Few-Shot Learning ,Sheep Identification ,Feature Extraction ,Support Vector Machine (SVM) ,FaceNet ,SSD MobileNet

ملخص

تتناول هذه المذكرة تصميم وتطوير نظام ذكي للتعرف على الأغنام باستخدام تقنيات الذكاء الاصطناعي والرؤية الحاسوبية.

يعتمد النظام على مراحل متعددة تبدأ بجمع البيانات، ثم معالجة الصور لاكتشاف الوجوه، واستخراج الخصائص المميزة ، ثم تصنيفها باستخدام خوارزميات تعلم الآلة.

الهدف من هذا العمل هو تحسين طرق إدارة وتربية الأغنام من خلال تقديم أداة ذكية تساعد المربين على التعرف على الاغنام بطريقة أوتوماتيكية ودقيقة، دون الحاجة إلى تدخل بشري مباشر في كل مرة.

تم تطبيق واختبار النظام على مجموعة من الصور الحقيقية للأغنام في بيئة محلية، وقد أظهرت النتائج أداءً واعدًا رغم بعض التحديات.

Résumé

Cette thèse porte sur la conception et le développement d'un système intelligent d'identification des moutons à l'aide de techniques d'intelligence artificielle et de vision par ordinateur.

Le système suit plusieurs étapes, en commençant par la collecte des données, la détection des visages à partir des images, l'extraction des caractéristiques de chaque mouton, et enfin la classification à l'aide d'algorithmes d'apprentissage automatique.

L'objectif principal de ce travail est d'aider les agriculteurs en leur fournissant un outil automatique et précis pour l'identification des moutons, sans qu'il soit nécessaire de procéder à une vérification manuelle à chaque fois.

Le système a été appliqué et testé sur des images réelles de moutons collectées dans un environnement local. Les résultats ont montré des performances prometteuses malgré quelques difficultés.

CONTENTS

CHAPTER 3

LIST OF FIGURES

CHAPTER I

CHAPTER II

LIST OF FIGURES

CHAPTRE III

LISTE OF TABLES

ABBREVIATIONS AND ACRONYMS

Abbreviation		Full Term
AI	:	Artificial Intelligence
ML	:	Machine Learning
DL	:	Deep Learning
NLP	:	Natural Language Processing
SVM	:	Support Vector Machine
k-NN	:	k-Nearest Neighbors
CNN	:	Convolutional Neural Network
SVR	:	Support Vector Regression
PCA	:	Principal Component Analysis
DBSCAN	:	Density-Based Spatial Clustering of Applications with Noise
RFID	:	Radio-Frequency Identification
GLCM	:	Gray Level Co-occurrence Matrix
LBP	:	Local Binary Pattern
SVD	:	Singular Value Decomposition
SIFT	:	Scale-Invariant Feature Transform
DoG	:	Difference of Gaussian
HOG	:	Histogram of Oriented Gradients
ANNs	:	Artificial Neural Networks
DNNs	:	Deep Neural Networks
RNNs	:	Recurrent Neural Networks
GANs	:	Generative Adversarial Networks
GPUs	:	Graphics Processing Units
ReLU	:	Rectified Linear Unit
GloVe	:	Global Vectors for Word Representation

ABBREVIATIONS AND ACRONYMS

GNNs	:	Graph Neural Networks
R-CNN	:	Regions with Convolutional Neural Network Features
YOLO	:	You Only Look Once
RPN	:	Region Proposal Network
SSD	:	Single Shot MultiBox Detector
COCO	:	Common Objects in Context (Dataset)
mAP	:	Mean Average Precision
IoT	:	Internet of Things
DSS	:	Decision Support System
OpenCV	:	Open Source Computer Vision Library
ONNX	:	Open Neural Network Exchange
NumPy	:	Numerical Python
FSL	:	Few-Shot Learning
MAML	:	Model-Agnostic Meta-Learning
VAEs	:	Variational Autoencoders
PASCALVOC	:	PatternAnalysis, Statistical Modelling, and Computational Learn ing– Visual Object Classes.

INTRODUCTION

GENERAL INTRODUCTION

In recent years, Artificial Intelligence (AI) has become very important in many fields like healthcare, transportation, industry, and agriculture. AI allows machines to learn from data and make decisions without human help. One important part of AI is Deep Learning, which is used for tasks like image classification, object detection, and face recognition [1].

Face recognition is already used in many areas such as security systems, mobile phones, and smart devices . However, most face recognition systems focus on humans. Animal face recognition, especially for livestock like sheep, is still a new research field with many challenges . Identifying individual sheep can help farmers with better flock management, disease monitoring, and breeding control .

In this thesis, we present a system for automatic sheep face detection and recognition using AI techniques. The system works in real-time with a webcam. First, it detects sheep faces using the SSD MobileNet model in ONNX format for fast and efficient detection [2]. Then, it extracts face features (embeddings) using the FaceNet model . Finally, a Support Vector Machine (SVM) classifier recognizes the identity of each sheep by comparing new face features with a trained model .

This project shows how AI can be applied in agriculture to solve real problems. It offers a non-invasive and accurate solution for animal identification, helping farmers save time and reduce errors compared to traditional methods like ear tags or RFID chips .

This thesis is organized into three main chapters:

- **Chapter 1:** Presents the theoretical background of AI, machine learning, deep learning, and object detection methods.
- **Chapter 2:** Discusses the use of AI in agriculture, especially in livestock management.
- **Chapter 3:** Explains the system implementation, describes the source code, shows experimental results, and discusses limitations and future improvements.

Through this research, we hope to contribute to the development of smart farming solutions that support farmers and improve productivity.

Chapter I

OBJECT DETECTION

1 Introduction

Livestock farming is one of the most important sectors of agriculture. Sheep farming, in particular, provides meat, milk, and wool, which are essential for the economy in many countries. Traditionally, farmers identify sheep using physical tags or marks. However, these methods are not always reliable because tags can fall off, and marks can fade.

Today, artificial intelligence (AI) offers new ways to solve this problem. AI can help recognize individual sheep using images or videos. This is faster and more accurate than traditional methods. [3]

Problem Statement

Manual identification of sheep is time-consuming and can lead to mistakes. Farmers face difficulties when managing large numbers of animals. Losing or misidentifying sheep can cause economic loss and affect the health of the flock. [3]

The problem is: how can we create a system that identifies sheep automatically and accurately using artificial intelligence?

2 Artificial Intelligence

Artificial intelligence means machines that can do tasks that normally need human intelligence. These tasks include understanding language, recognizing images, playing games, or driving cars. The goal is to make machines that can think and act intelligently. [4]

2.1 Types of Artificial Intelligence

There are three main types of AI:

Narrow AI: Also called Weak AI, it is designed to do one task. For example, a face recognition system only identifies faces. [4]

General AI: This AI can do any task a human can do. It is still being researched and is not yet real. [4]

Super AI: More intelligent than humans. It is a future idea and does not exist yet. [5]

2.2 Applications of Artificial Intelligence

AI is used in many real-world systems:

In Healthcare: AI helps doctors find diseases in medical images . [6]
In Farming: AI helps count animals, check health, and predict food needs . [1]
In Transport: Self-driving cars use AI to detect roads, signs, and people . [7]
In Smart assistants: Devices like Siri or Alexa understand voice and answer questions.

2.3 AI Technologies

AI uses different methods to work. The main ones are:

Machine Learning (ML): A technique where machines learn from data and get better with time. [4]
Deep Learning (DL): A type of ML that uses neural networks with many layers to understand data. [4]
Natural Language Processing (NLP): Helps machines understand and respond to human language. [8]

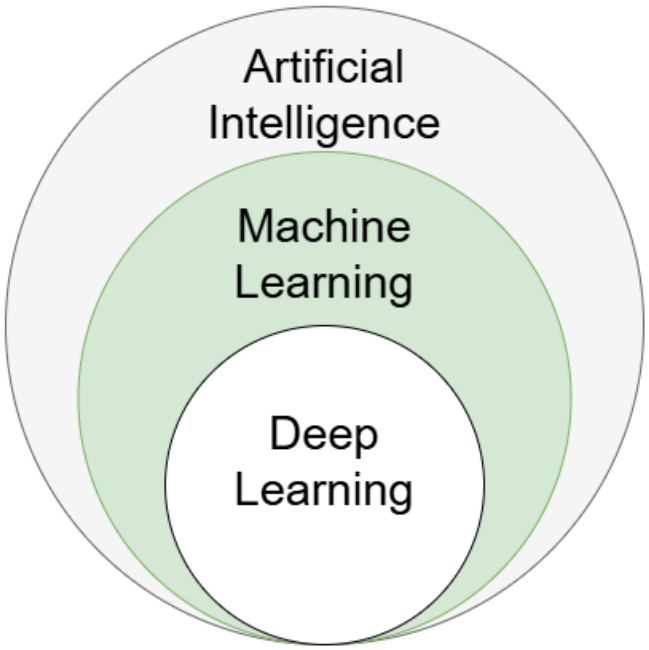

Figure I.1 The Hierarchical Relationship Between AI, Machine Learning, and Deep Learning [4]

2.4 Challenges in Artificial Intelligence

Even though AI is powerful, it has some problems:

It needs a lot of data to learn. [9]
It can make mistakes if the data is wrong.
It sometimes cannot explain how it made a decision. [10]
It raises ethical questions like privacy and job loss. [11]

3 Machine Learning Techniques

Machine Learning (ML) is a part of Artificial Intelligence (AI), Figure 1. It helps computers learn from data. The computer can make decisions without being programmed for each case. In sheep identification, ML helps detect or recognize sheep using data such as images or sounds. [12]

3.1 Types of Machine Learning

Machine Learning has three main types:

Supervised Learning, Unsupervised Learning, and Reinforcement Learning. Each type uses data in a different way.

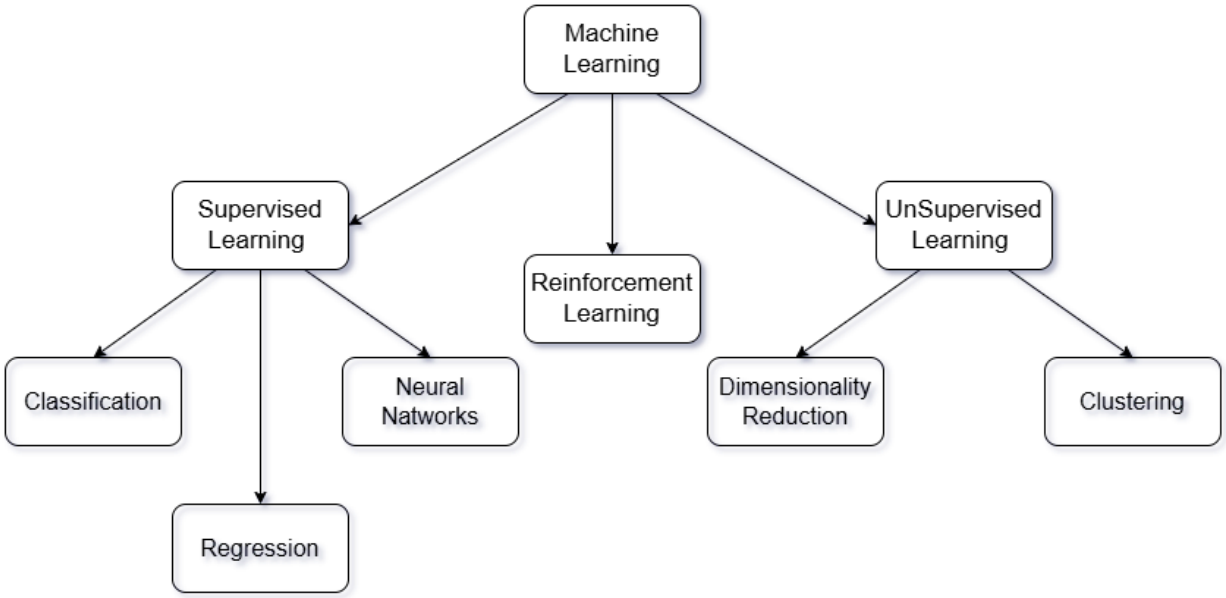

Figure I.2 Types of Machine Learning [13]

I. Supervised Learning

Supervised learning is the most common type of machine learning. It uses labeled data, which means the data has both input and correct output. The model learns from examples. After training, the model can predict the correct output for new inputs. [13]

Example: If we have pictures of sheep with labels like "Sheep A" or "Sheep B," the model learns to identify each sheep from these labels.

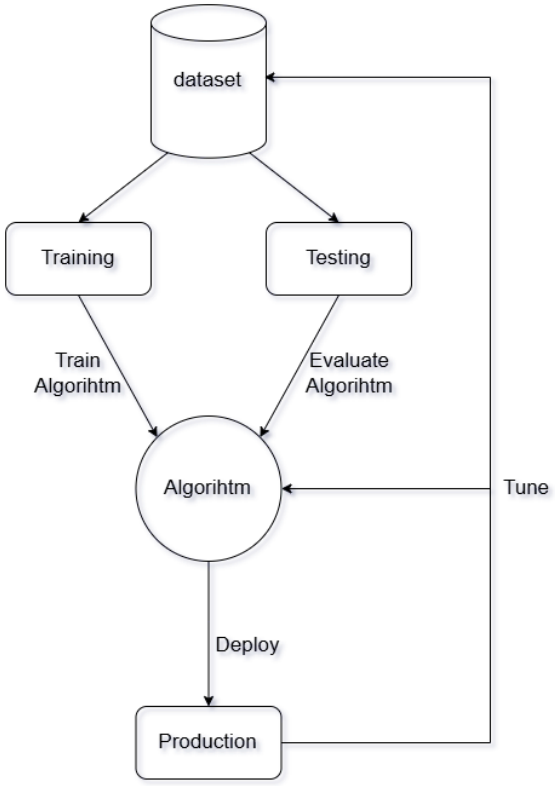

Figure I.3 Workflow of Supervised Learning [13]

Supervised learning has two main tasks:

- **Classification:** when the output is a category (e.g., type of sheep).
- **Regression:** when the output is a number (e.g., sheep weight).

A. Common Classification Algorithms

Support Vector Machine (SVM): SVM finds the best line (or hyperplane) that separates data into classes. It works well when the data has clear boundaries. [14]

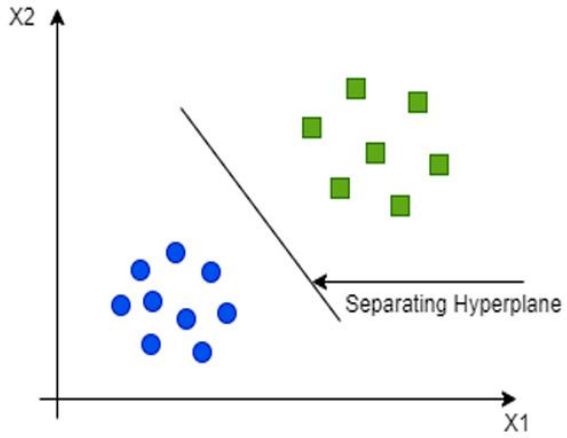

Figure I.4 Support Vector Machine [14]

k-Nearest Neighbors (k-NN): This algorithm looks at the 'k' closest data points and chooses the most common class among them. It is simple and good for small datasets.

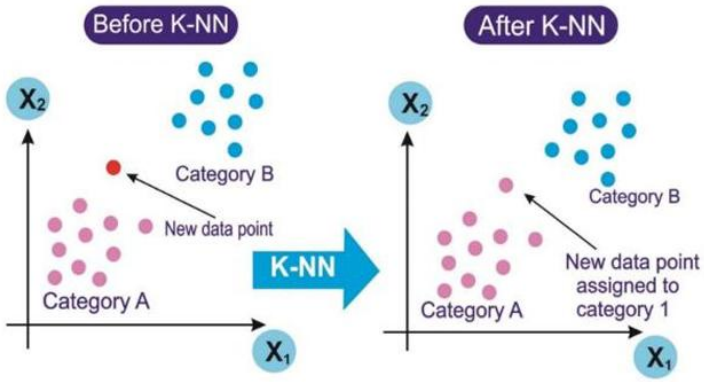

Figure I.5 K-Nearest Neighbors [14]

Decision Tree: A tree-like structure where each node checks a condition. It is easy to understand and visualize.

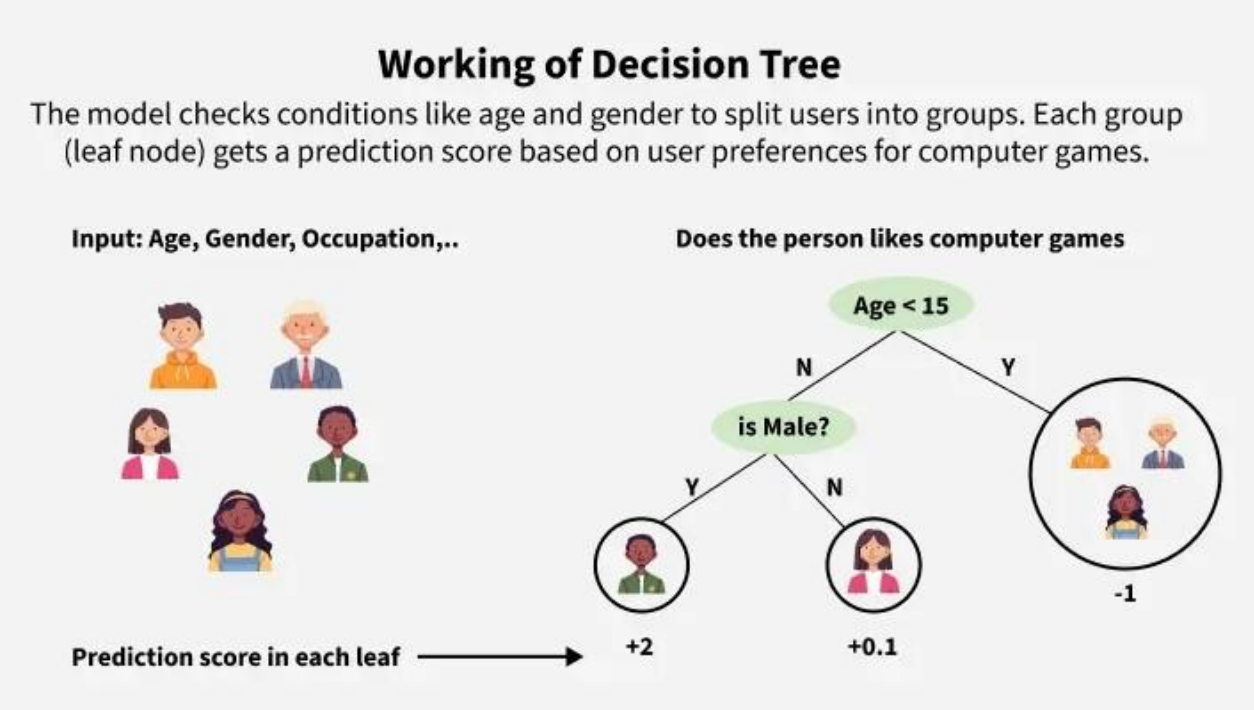

Figure I.6 Decision Tree [14]

Random Forest: A group of decision trees. It makes predictions by averaging the results of many trees. It reduces errors from individual trees.

Naive Bayes: Based on Bayes' Theorem. It works well when features are independent. Common in spam detection.

Logistic Regression: A statistical model that predicts the probability of a class. Useful when the result is binary (yes/no).

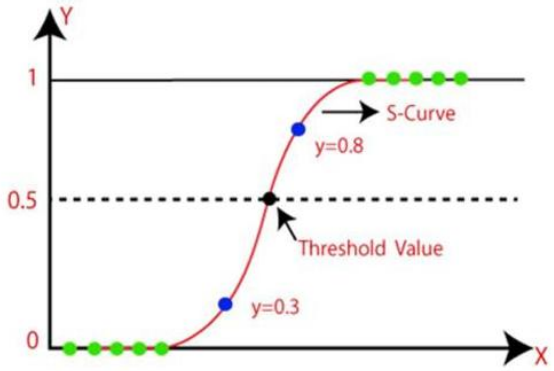

Figure I.7 Logistic Regression [14]

B. Common Regression Algorithms

Linear Regression: It models the relationship between input and output using a straight line. Simple and widely used. [14]

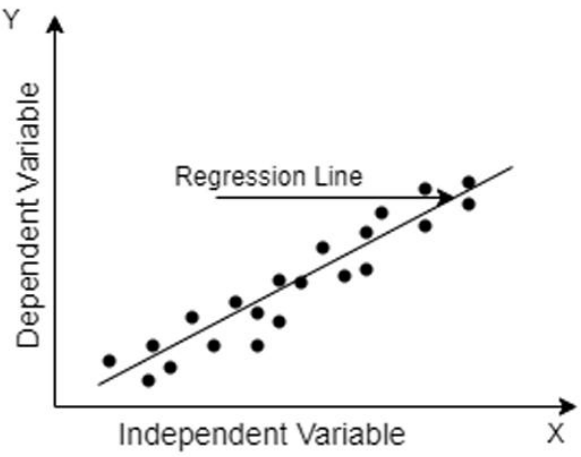

Figure I.8 Linear Regression [14]

Ridge Regression: Like linear regression but adds a penalty to reduce overfitting. Suitable for large datasets.

Lasso Regression: It also adds a penalty but can remove less important features. Helps in feature selection.

Support Vector Regression (SVR): SVR uses the same ideas as SVM but for predicting continuous values.

Decision Tree Regression: Uses tree-based structure to predict numbers instead of categories.

II. Unsupervised Learning

Unsupervised learning uses unlabeled data. The model tries to find hidden patterns or groupings in the data without knowing the output. It is useful when we don't have labeled examples. [13]

Example: If we have images of sheep without any names or IDs, the model can group similar sheep based on features like color or size.

A. Common Unsupervised Algorithms

K-Means Clustering: Divides data into 'k' clusters. Each point belongs to the nearest center. Simple and fast.

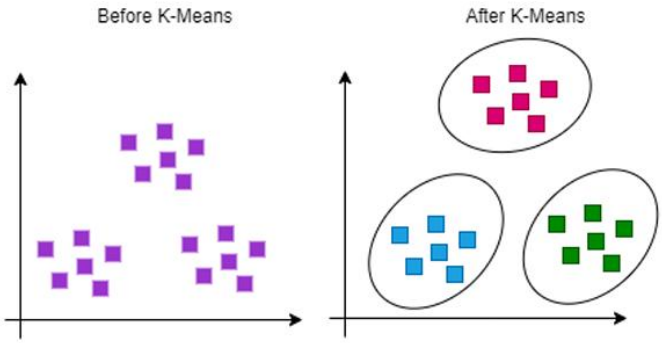

Figure I.9 K-Means Clustering [16]

Hierarchical Clustering: Builds a tree of clusters. Good for understanding relationships between data.

Principal Component Analysis (PCA): Reduces the number of features while keeping important information. Helps in data visualization.

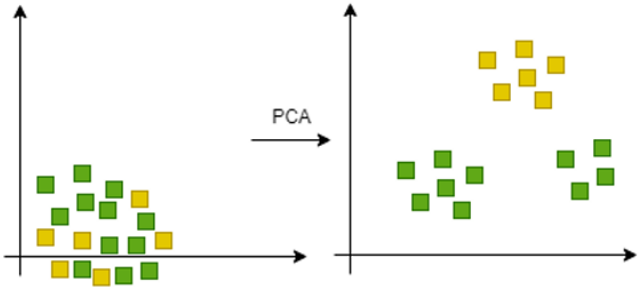

Figure I.10 Principal Component Analysis [4]

DBSCAN (Density-Based Spatial Clustering): Groups points close together in dense regions. Works well for data with noise or irregular shapes.

Autoencoders: Neural networks that learn to compress and reconstruct data. Useful for anomaly detection and image compression.

III. Reinforcement Learning

Reinforcement Learning is based on trial and error. The agent interacts with the environment and learns from rewards or punishments. The goal is to learn the best actions that give the most reward. [15]

Example: A robot sheepdog learns to move sheep into a pen. If it does well, it gets a reward. If not, it gets nothing or a penalty.

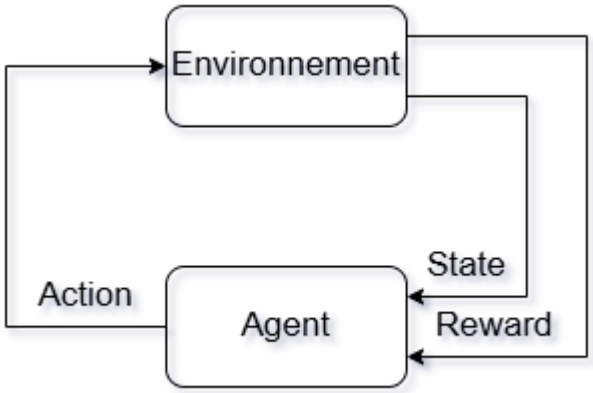

Figure I.11 Reinforcement Learning [14]

Positive Reinforcement

When the agent receives a reward for a good action. This increases the chance to do the same action again.

Example: Giving a reward when the model correctly identifies a sheep.

Negative Reinforcement

When the agent removes a negative condition after a good action. This also increases the chance to repeat the action.

Example: If the model makes a mistake, it loses points. When it makes the right decision, the penalty stops.

3.2 Machine Learning Steps

Data Collection

Collect images or data about the sheep. This can include pictures, sound, or RFID signals.

Data Preprocessing

Clean the data. Remove noise and fix missing values.

Feature Selection

Choose the most important features in the data. For example: size, color, or texture of the sheep.

Feature Transformation
Transform data into a format that the algorithm can understand. For example: scaling or encoding.

Selecting the Classification Algorithm
Choose a good algorithm based on the problem. In sheep identification, classification algorithms like CNN or SVM are common.

Model Training
Train the model using the training data.

Hyperparameter Tuning
Adjust the parameters of the model to improve accuracy.

Improving the Model
Use techniques like:
- Cross-validation
- More data
- Better features
- Advanced algorithms

Model Evaluation
Test the model using test data. Use metrics like:
- Accuracy
- Precision
- Recall
- F1 Score

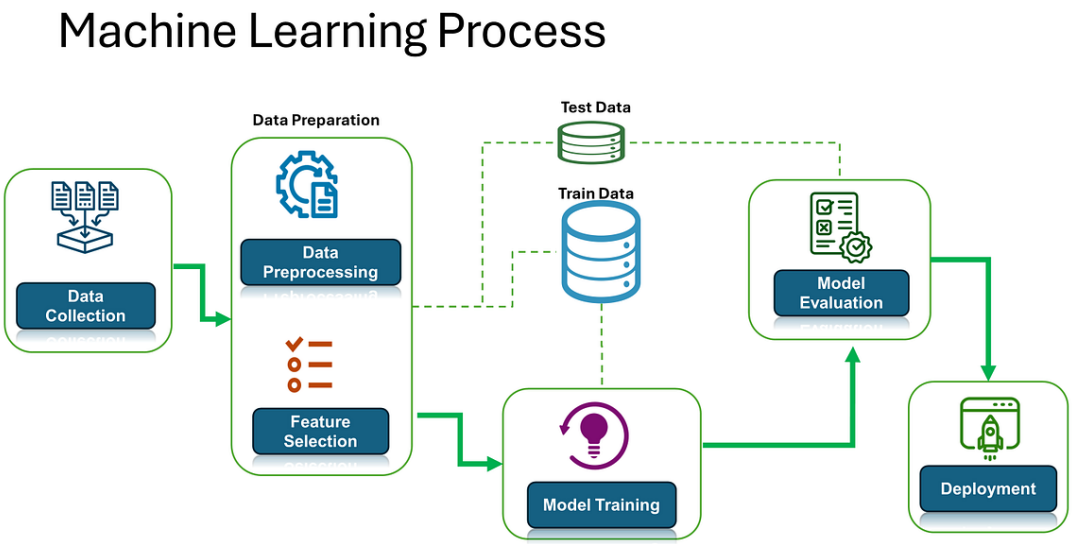

Figure I.12 Machine Learning Process [4]

3.3 Applications of Machine Learning

Machine Learning (ML) has many applications in different fields :

- Face recognition of sheep using images
- Voice recognition of sheep sounds
- Behavior monitoring
- Health detection

3.4 Challenges and Limitations

Machine Learning (ML) faces several challenges. It needs large and clean datasets for good results. Training models also requires strong hardware like GPUs. ML models may suffer from overfitting or poor generalization. Sometimes, it is hard to understand how the model makes decisions. Also, biased data can cause unfair results.

4 Traditional Machine Learning Techniques

Traditional machine learning (ML) techniques have long been used in various applications, such as image recognition, object detection, and speech processing. These techniques aim to find patterns in data and use them for classification or prediction tasks. In the context of object detection, such as identifying sheep in images or video footage, these methods rely on the extraction of relevant features and the subsequent classification of those features using different algorithms. [15]

4.1 Overview of Traditional Machine Learning

Traditional machine learning models are generally simpler than deep learning models, which require large datasets and complex architectures. These methods often perform well on smaller datasets and provide good interpretability, making them suitable for real-world applications like sheep identification. The most common machine learning models include Support Vector Machines (SVM), K-Nearest Neighbors (KNN), Decision Trees, and others, which all play an essential role in identifying and classifying images based on specific features. [16]

Figure I.13 Traditional Machine Learning Approach [4]

4.2 Feature Extraction and Classification

A. Feature Extraction

Feature extraction is a key step in machine learning pipelines, particularly for object detection tasks. The features extracted from an image or video frame are critical for distinguishing between different objects. In sheep identification, feature extraction typically focuses on shape-based, texture-based, and motion-based features.

I. Shape-Based Features

Shape-based features are important in object detection because they describe the geometry of the object. The shape can be 2D or 3D and includes elements like edges, curves, corners, and contours. In computer vision, 2D shapes are often represented using lines, junctions, and regions, while 3D shapes use surfaces, planes, or cylinders.

In model-based recognition, shape is a key factor. Using 2D models with 2D inputs is simple but may lead to confusion, as many 3D shapes can look similar in 2D. More accurate methods use 3D models with 3D inputs, but they are computationally heavy. A common compromise is to use 3D models with 2D images, estimating the pose (position and rotation) of the object.

In practical tasks like sheep identification, shape-based features are used to detect parts of the sheep like the body, head, or ears. Common techniques include:

- **Edge Detection**: Algorithms like Canny are used to detect object boundaries.
- **Contour Detection**: Finds the outlines of objects, useful for recognizing animal shapes.
- **Hu Moments**: These are mathematical values that describe shapes and stay the same even if the object is moved, rotated, or resized.

These shape features help create a unique and reliable description of each sheep. [15]

II. Texture-Based Features

Texture-based features are important in object detection because they describe the surface details and patterns of an object. Texture refers to how the surface of an object appears in terms of smoothness, roughness, or repeated patterns. In computer vision, texture is not defined by a single pixel but by the relationship between neighboring pixels in a region.

At different resolutions, texture appears differently. For example, when viewing a tiled floor from a distance, the tiles form a general texture. But when viewed closely, small patterns inside each tile become visible. This shows that texture depends on the scale and resolution of the image.

In practical applications like sheep identification, texture-based features help distinguish between animals by analyzing the wool patterns. These patterns are unique and consistent for each sheep, making texture a useful biometric feature.
Several methods are used to extract texture features:

- **Gray Level Co-occurrence Matrix (GLCM):** GLCM measures how often pairs of pixel values appear together in an image. It helps describe texture properties like smoothness, contrast, and entropy. [16]
- **Local Binary Patterns (LBP):** LBP compares each pixel to its neighbors and creates a binary pattern. This method is useful for recognizing fine textures such as wool, fur, or skin. [16]
- **Gabor Filters:** These filters analyze texture in different directions and frequencies. They are good for identifying complex or repetitive textures by simulating the way human vision detects patterns.

Texture can be described as fine**,** smooth**,** coarse**,** or grained. These characteristics are influenced by the size and arrangement of small texture elements, called texels. When texels are small and detailed, the texture looks smooth. When they are large, the texture appears coarse.
By combining texture features with shape and motion information, computer vision systems can more accurately identify and classify objects such as sheep. [15]

III. Motion-Based Features

Motion-based features are essential in object detection when analyzing video sequences or image frames over time. These features help recognize objects based on their movement, which is especially useful in identifying animals like sheep through their motion patterns, such as walking, turning, or grazing.
The first step in motion-based recognition is extracting motion information from a sequence of images. This can be done using two main techniques: motion correspondence and optical flow. Motion correspondence involves tracking key points (features) across multiple frames, forming motion trajectories. These trajectories represent the path of movement of a point, described as a sequence of

coordinates $(x_i,y_i)(x\_i, y\_i)(x_i,y_i)$ for each frame i. To make analysis easier, the trajectory can be represented in terms of speed, direction, or velocity components $(v_x,v_y)(v\_x, v\_y)(v_x,v_y)$, and changes like acceleration or direction shift can indicate important motion events.

One of the most common motion-based methods is optical flow, which calculates the apparent motion of objects between two consecutive frames. It identifies the direction and speed of each pixel, helping to detect and follow moving objects. For example, in sheep identification, optical flow can track body parts across time to recognize behavioral patterns. [16]

Other useful representations include motion histograms, which summarize motion vectors across a frame to highlight common actions, and trajectory analysis, where specific points on the sheep's body or head are followed to build a movement profile. These motion-based representations provide an additional layer of data that supports classification and identification when combined with other visual features like shape or texture. [15]

B. Classification Techniques

Once features are extracted, classification techniques are applied to make predictions about the class of the object, in this case, identifying whether an object is a sheep.

Support Vector Machines (SVM)

Support Vector Machines (SVM) are a powerful classification method used to separate data into distinct classes by finding a hyperplane that maximizes the margin between classes. SVM is effective in classifying sheep images by separating sheep from other objects in the feature space based on extracted features like texture and shape.

Decision Trees

Decision Trees are simple yet powerful models that recursively split the data based on the most informative feature. In sheep identification, decision trees can be used to classify sheep by analyzing features such as body shape, ear size, and wool texture.

Logistic Regression

Logistic regression is widely used for binary classification tasks, providing a probability output that can be used to classify data into two categories Logistic

regression can be applied in sheep identification to estimate the probability that an image belongs to the "sheep" class based on features such as wool texture and body shape.

Neural Networks

Neural Networks are powerful models that are particularly effective for capturing complex patterns in data. They consist of layers of interconnected nodes that process and learn features from the input data. Neural networks can be used in sheep identification to learn complex relationships between extracted features, such as the shape of the body and the wool texture, to accurately classify sheep in images.

Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a dimensionality reduction technique that projects data onto new axes, capturing the most significant variance in the data. PCA can reduce the dimensionality of feature sets extracted from sheep images, making the classification process more efficient by removing redundant information.

Singular Value Decomposition (SVD)

Singular Value Decomposition (SVD) is a matrix factorization technique that decomposes a matrix into its constituent parts, allowing for efficient processing. SVD is used to break down the feature matrix into components that are easier to classify, which helps in reducing computational complexity in large datasets.

K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) is a simple classification algorithm that classifies data based on the majority vote of its nearest neighbors. KNN can be applied to classify sheep by comparing the extracted features of a given image with those of nearby images in the feature space.

4.3 Features Used for Object Detection

A. Scale-Invariant Feature Transform (SIFT)

SIFT (Scale-Invariant Feature Transform) is a robust feature extraction technique used to detect and describe local features in images. It is invariant to changes in scale, rotation, and partial distortion. SIFT is used in sheep identification to detect and match keypoints in sheep images, enabling recognition across varying conditions.

B. Difference of Gaussian (DoG)

Difference of Gaussian (DoG) is used to highlight edges and keypoints in an image. DoG detects keypoints on the sheep's body or face, which can be used for matching and recognition across different images or video frames.

C. Histogram of Oriented Gradients (HOG)

HOG (Histogram of Oriented Gradients) is a feature descriptor used for object detection. It captures gradient information in localized regions of the image. HOG is used to extract the gradient patterns of a sheep's body, which helps in distinguishing sheep from other objects.

D. Local Binary Patterns (LBP)

Local Binary Patterns (LBP) is a texture descriptor that encodes local patterns around each pixel. LBP is effective in distinguishing the wool texture of sheep from other surfaces.

5 Deep Learning

Deep learning is a subfield of artificial Intelligence (AI) and machine learning (ML) that focuses on using artificial neural networks with many layers to learn from data. These models are inspired by how the human brain works. Deep learning has become very popular because it can solve complex problems in image recognition, speech processing, natural language understanding, and more. [4]

5.1 Difference Between Machine Learning and Deep Learning

Machine learning models usually need manual feature selection. This means humans decide which data features are important. In contrast, deep learning models automatically find these features from raw data. Also, deep learning works better with large datasets and complex tasks, while machine learning is more suitable for small to medium-sized data. [15]

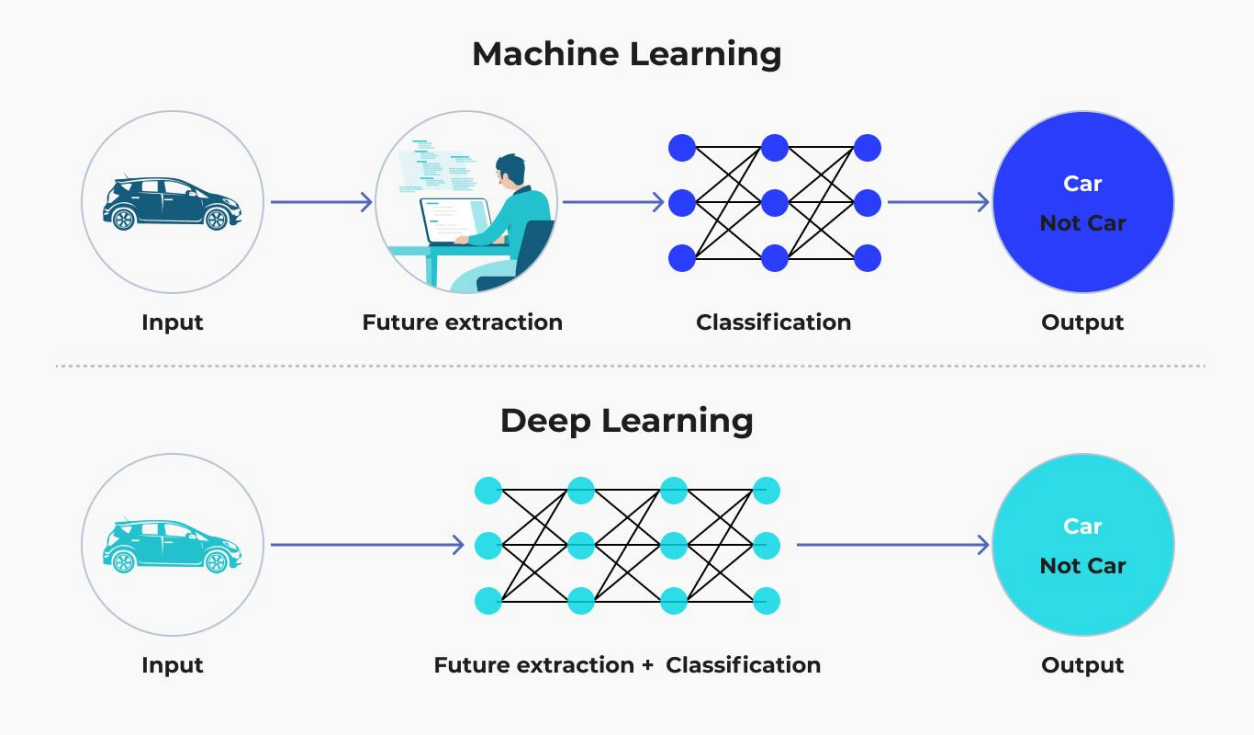

Figure I.14 Defrence Between Machine Learning and Deep Learning [4]

5.2 Artificial Neural Networks (ANNs)

Artificial neural networks are computing systems inspired by the biological brain. They contain layers of connected units called neurons. Each neuron takes input, processes it, and passes the output to the next layer. ANNs can be used for tasks like classification, prediction, and pattern recognition. [15]

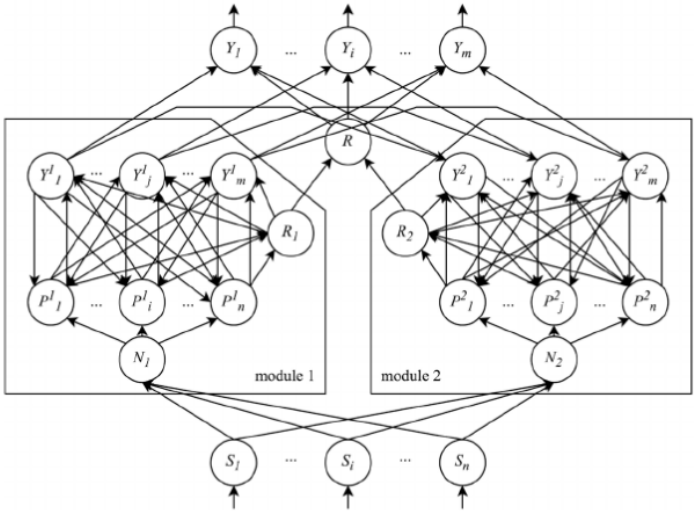

Figure I.15 Artificial Neural Network Architecture [4]

5.3 Deep Neural Networks (DNNs)

Deep neural networks are neural networks with many hidden layers. These deep layers help the model learn abstract features step by step. For example, in image analysis, early layers detect edges, while deeper layers identify shapes or objects. DNNs are powerful for tasks that need high-level understanding. [15]

5.4 How Deep Learning Works

Deep learning models learn from examples. During training, the model adjusts the weights of connections between neurons to reduce the error between predicted and actual outputs. This process is called backpropagation. The training continues until the model performs well on the training data and can generalize to new data.

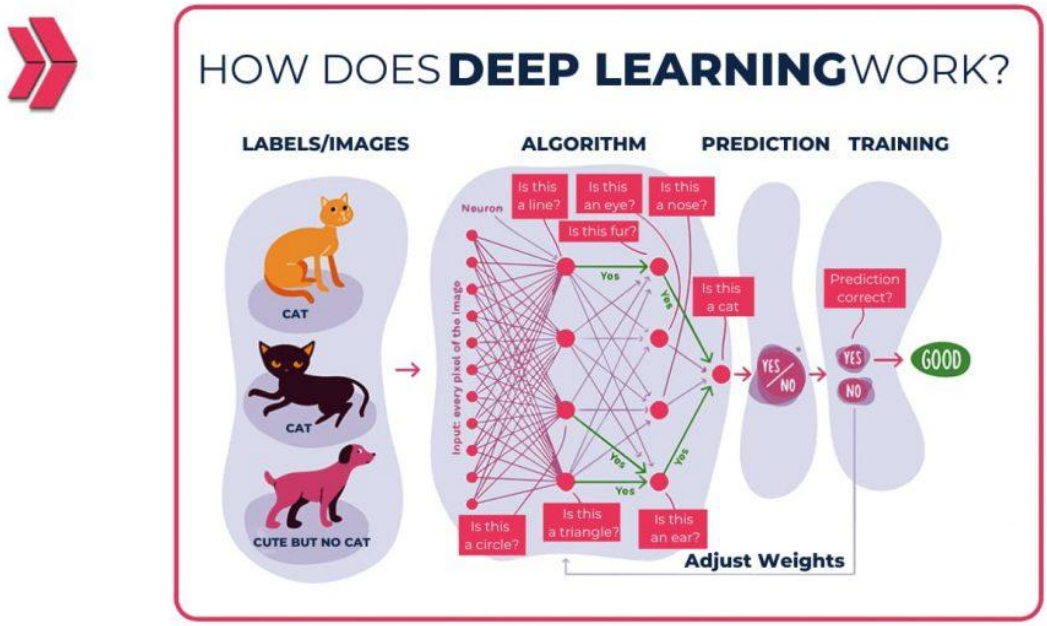

Figure I.16 How Does Deep Learning Work [4]

5.5 Deep Learning Architectures

A. Convolutional Neural Networks (CNNs)

CNNs are designed for image and video processing. They use filters to detect patterns like edges, textures, and shapes automatically. CNNs are used in face recognition, medical image analysis, and object detection. [15]

B. Recurrent Neural Networks (RNNs)

RNNs are used for sequence data like time series or text. They can remember previous inputs using loops in the network. This makes them useful for language translation, speech recognition, and text generation. [15]

C. Generative Adversarial Networks (GANs)

GANs consist of two networks: a generator and a discriminator. The generator creates fake data, while the discriminator tries to detect if the data is real or fake. This helps GANs learn to generate realistic data like images, audio, or text. [4]

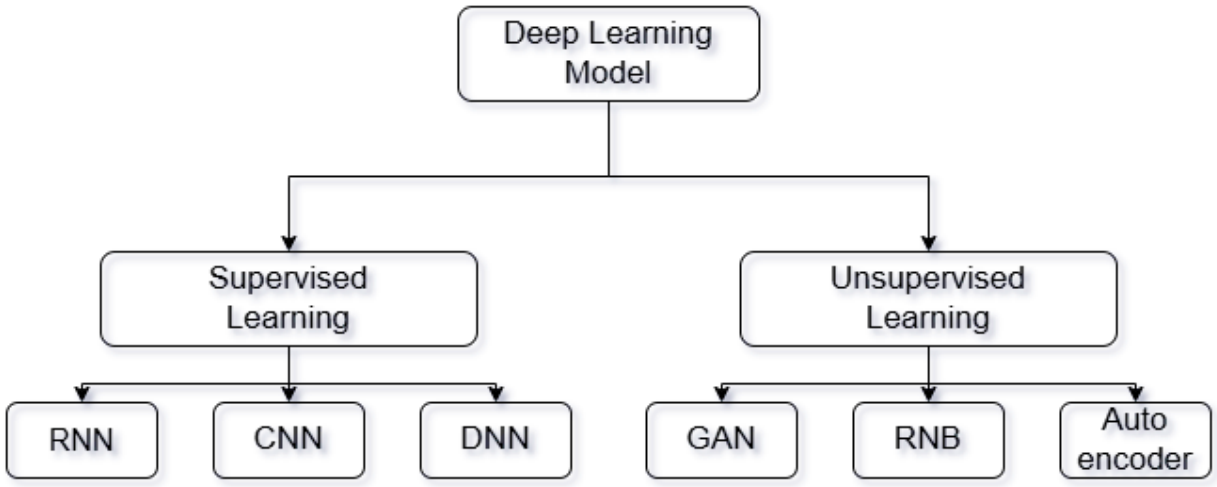

Figure I.17 Classification of Deep Learning Model [15]

5.6 Applications of Deep Learning

Deep learning is used in many real-world applications:

- Healthcare: Diagnosis from medical images, drug discovery. [6]
- Finance: Fraud detection, market prediction .[1]
- Transportation: Self-driving cars
- Security: Face and object recognition
- Natural Language Processing: Chatbots, translators, sentiment analysis .[8]

5.7 Advantages and Limitations

Deep learning has several advantages, such as high accuracy**,** automatic feature extraction**,** and the ability to handle large and complex datasets. It is also flexible and works well with images and videos.

However, it has limitations like the need for large labeled datasets**,** high computational cost**,** and being a black-box model that is hard to interpret. There is also a risk of overfitting when data is limited.

5.8 Deep Learning Tools and Frameworks

Several frameworks help developers build and train deep learning models:

- TensorFlow (by Google)
- PyTorch (by Meta/Facebook)
- Keras (easy-to-use interface for deep learning)
- Caffe (mainly used for image recognition)
- MXNet (used by Amazon)

5.9 Challenges in Deep Learning

Deep learning faces several challenges, such as the need for large and high-quality datasets**,** high computational cost**,** and the risk of overfitting**.** It also suffers from lack of interpretability and data bias**,** which may affect model performance and fairness.

5.10 Future of Deep Learning

The future of deep learning includes:

- More efficient models that need less data
- Better explanations of model decisions
- Integration with other fields like robotics and neuroscience
- Wider use in daily life, from smartphones to smart cities

6 Convolutional Neural Network (CNN or ConvNet)

Convolutional Neural Networks (CNNs), also called ConvNets, are a type of deep learning model designed for analyzing visual data. They are very useful in tasks such as image classification, face recognition, and object detection. CNNs work by automatically learning patterns from images using special layers.

6.1 Structure of CNN

Input Layer

This layer takes the image data. For example, a colored image with 100×100 pixels has 3 channels (red, green, blue), so the input is 100×100×3.

Convolutional Layer

This layer uses filters (also called kernels) to detect patterns like edges and textures. Each filter moves over the image and creates a new image called a feature map.

Activation Function (ReLU)

After convolution, the ReLU (Rectified Linear Unit) function makes the image non-linear. This helps the model learn more complex patterns.

Pooling Layer

This layer reduces the size of the feature maps to make computation faster and avoid overfitting. The most common pooling is max pooling.

Fully Connected Layer

At the end of the CNN, this layer connects all the neurons together. It uses the extracted features to make the final decision or classification. [15]

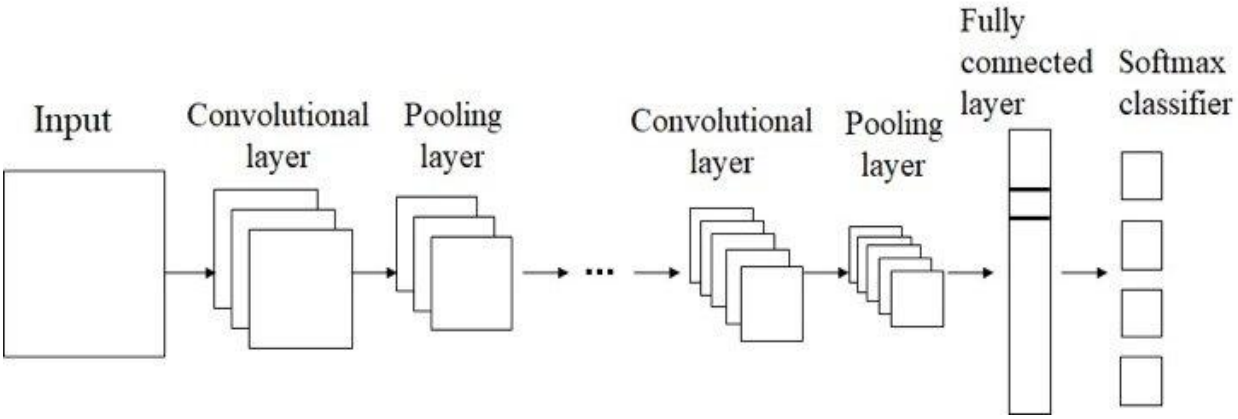

Figure I.18 CNN Architecture[15]

6.2 How CNN Works

1. The image is given to the input layer.
2. Filters in the convolutional layer scan the image to find basic patterns.
3. The activation function makes the output non-linear.
4. Pooling reduces the image size.
5. After several layers, the features are passed to the fully connected layer.
6. The output is a prediction (for example: "cat" or "dog").

6.3 Applications of CNN

- Medical Imaging: Detecting diseases in X-rays or MRIs
- Security: Face and object recognition
- Self-driving Cars: Identifying pedestrians, signs, and lanes
- Robotics: Visual navigation and object grasping
- Agriculture: Detecting plant diseases and monitoring crops [15]

6.4 Advantages of CNN

- Automatic Feature Learning: No need for manual feature selection
- High Accuracy: Performs better on complex visual tasks
- Scalability: Can handle large datasets efficiently
- Versatility: Works on images, videos, and even text (e.g., character recognition) [15]

6.5 Limitations of CNN

- Requires Large Data: Needs a lot of labeled images to train
- Computational Cost: Training needs GPUs and high memory
- Interpretability: It is often hard to understand how the model makes decisions
- Sensitivity to Noise: May misclassify if the image is unclear

6.6 Famous CNN Architectures for classification

- LeNet-5: One of the first CNNs, used for digit recognition [3]
- AlexNet: Won the ImageNet competition in 2012 and started the deep learning revolution [17]
- VGGNet: Known for its simplicity and deep layers
- GoogLeNet (Inception): Uses multiple filter sizes in the same layer
- ResNet: Introduced residual connections to solve the problem of vanishing gradients

7 Object Detection Techniques

Object detection is a computer vision task that identifies and locates objects in images or videos. It plays a major role in many fields, including surveillance, farming, self-driving cars, and medical imaging. This technology helps machines understand the world by recognizing objects and knowing where they are in a picture.

7.1 What is Object Detection?

Object detection is different from image classification. While classification tells what is in an image, detection shows where the object is using bounding boxes. For example, in an image of sheep in a field, object detection will not only say "sheep" but also show where each sheep is.

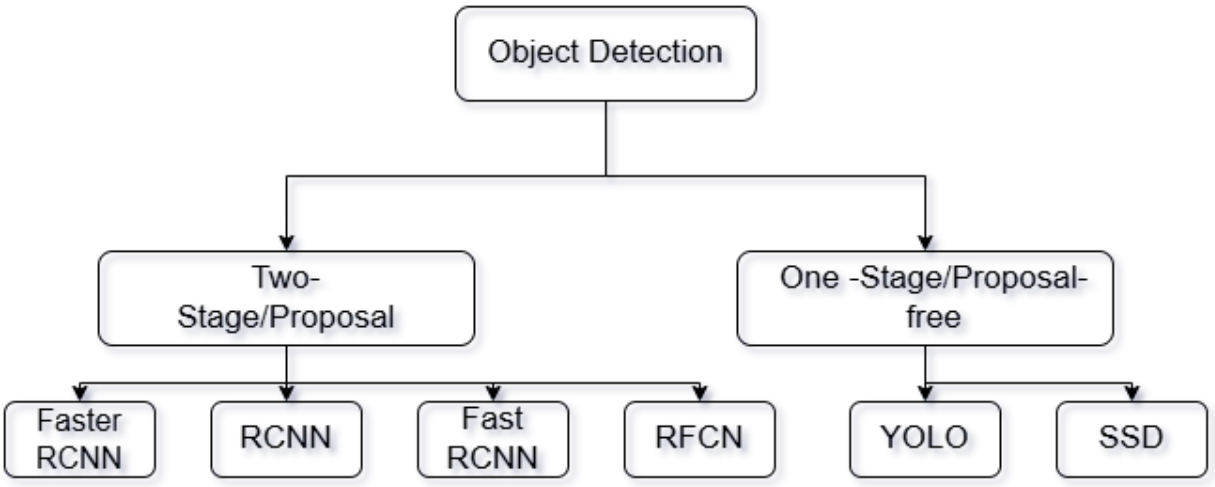

Figure I.19 Object Detection [13]

7.2 Main Steps of Object Detection

1. Input an image.
2. Use a model to scan the image.
3. Detect features like shapes, textures, or colors.
4. Draw bounding boxes around detected objects.
5. Classify the objects inside each box.

7.3 Popular CNN Architectures for Object Detection

Convolutional Neural Networks (CNNs) are widely used for object detection. The most popular CNN-based models include:

• R-CNN (Region-based CNN): Extracts regions from the image and classifies each one. [17]
• Fast R-CNN: Faster than R-CNN because it processes the image only once.
• Faster R-CNN: Adds a Region Proposal Network (RPN) to suggest regions of interest.
• YOLO (You Only Look Once): Detects all objects in a single pass, making it fast. [18]

• SSD (Single Shot MultiBox Detector): Detects objects in different sizes using multiple feature maps. [19]

7.4 Object Detection Datasets

Datasets are important for training and testing models. Some famous datasets include:

• COCO (Common Objects in Context): Contains images with 80 object categories. [1]

• PASCAL VOC: One of the first datasets, used widely in research . [3]

• ImageNet: Large dataset mainly used for classification, but it also has object detection data . [3]

• Open Images Dataset: A large dataset with bounding boxes and object labels .

7.5 Evaluation of Object Detection Models

To know how good a model is, we need to test it. Some important metrics are:

1.Precision: How many of the detected objects are correct.

2.Recall: How many of the real objects were detected.

3. Mean Average Precision (mAP): Combines precision and recall for different thresholds. A high mAP means the model is good at both finding and correctly labeling objects. [16]

8 Conclusion

In this chapter, We have provided a comprehensive overview of the evolution and key components of artificial intelligence (AI), from the foundations of traditional machine learning to the advanced capabilities of deep learning. We examined how deep learning, particularly convolutional neural networks (CNNs), has revolutionized feature extraction and pattern recognition, especially in image-related tasks. The discussion of modern AI frameworks highlighted the tools that facilitate the development and deployment of these models at scale. As AI systems advance toward data-efficient learning, few-shot learning emerges as a powerful paradigm, enabling models to generalize from minimal data. Central to this approach are embedding models, which transform inputs into meaningful vector representations that preserve semantic relationships. Finally, we explored FaceNet as a prime example of how embedding-based models and CNNs can be integrated to achieve highly accurate face recognition through metric learning. Collectively, these concepts illustrate the shift from handcrafted features to data-driven representation learning, marking a significant milestone in the journey of intelligent systems.

Chapter II

AI IN AGRICULTURE

1 Introduction

Agriculture is vital for providing food, fiber, and fuel, but it faces significant challenges such as climate change, population growth, limited resources, and labor shortages. These pressures make traditional farming methods inadequate. To ensure future food security, the sector is turning to advanced technologies like Artificial Intelligence (AI), the Internet of Things (IoT), and Big Data. This chapter focuses on how AI is transforming modern agriculture to enhance efficiency, productivity, and resilience. [2]

2 The Role of Artificial Intelligence in Agriculture

In agriculture, AI is being used to automate and enhance various farming operations. The adoption of AI technologies allows farmers to make data-driven decisions, monitor conditions in real-time, and respond more efficiently to changes in the environment . [1]

AI helps solve critical agricultural problems through advanced data analytics, machine learning models, and decision support systems. For example, farmers can use AI to predict the best planting times, identify disease outbreaks early, and optimize the use of inputs such as fertilizers and pesticides. AI also supports precision agriculture, where every square meter of a field is monitored and managed individually, reducing waste and increasing yield.

In livestock farming, AI tools can monitor the health, behavior, and productivity of animals. AI-based systems are used to track feeding patterns, detect illnesses, and even assist in breeding programs. These applications not only improve animal welfare but also contribute to the profitability and sustainability of livestock operations.

Figure II.1 Top Issues in Agriculture [2]

AI has the potential to close the knowledge gap in agriculture by providing expert advice to farmers in remote or underserved areas. Mobile applications powered by AI can offer recommendations based on local data, helping farmers who do not have access to agronomists or specialists. This democratization of agricultural knowledge is essential for global food security.

Figure II.2 Artificial Intelligence in Agriculture [1] .

3 Key AI Technologies Used in Farming

Several AI technologies are being integrated into agriculture, each with its specific functions and benefits. Among the most widely used are:

3.1 Machine Learning (ML): ML algorithms analyze historical and real-time data to identify trends and make predictions. These algorithms improve over time as they process more data. In agriculture, ML is used for crop yield prediction, soil analysis, pest detection, and resource optimization . [20]

3.2 Computer Vision: This technology enables machines to interpret and process visual information from the environment. Cameras mounted on drones, tractors, or robots can capture images of crops and animals. Computer vision systems then analyze these images to detect diseases, estimate growth stages, and classify plant species. [20]

3.3 Natural Language Processing (NLP): NLP helps machines understand and respond to human language. Agricultural chatbots and voice assistants use NLP to provide farmers with information on weather, prices, and best practices. [20]

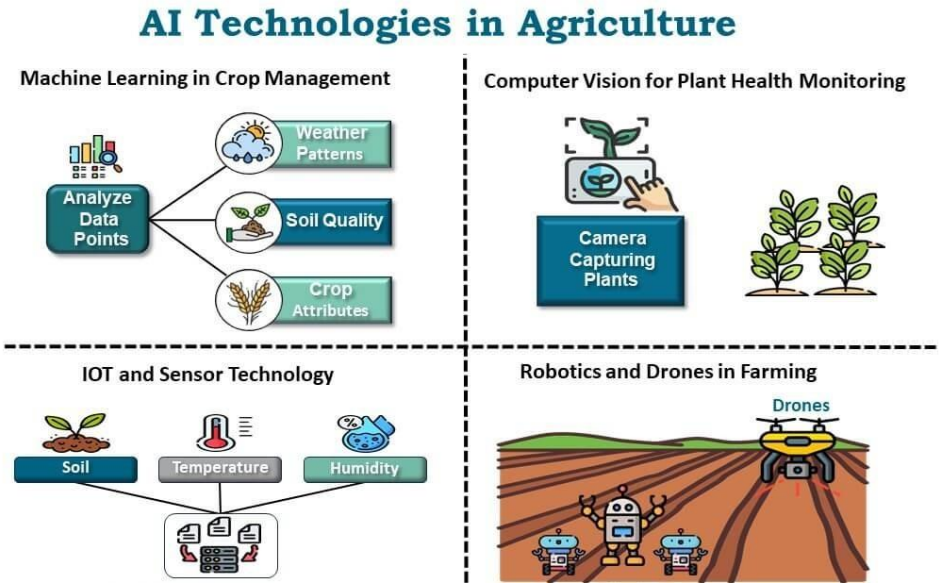

Figure II.3 AI Technologies [20].

3.4 Robotics and Automation: AI-driven robots perform tasks such as planting, weeding, and harvesting. These machines reduce labor costs and can operate continuously without fatigue. Autonomous tractors and robotic milkers are examples of how automation is transforming agriculture. [21]

3.5 Remote Sensing and Drones: Equipped with AI algorithms, drones collect data on crop health, soil conditions, and water distribution. This data supports precision farming by enabling timely interventions. [21]

3.6 Decision Support Systems (DSS): AI-based DSSs use data from multiple sources to offer recommendations for farm management. These systems help farmers decide when to irrigate, apply pesticides, or harvest.

Each of these technologies plays a vital role in building a more innovative, more efficient, and more resilient agricultural system. [21]

4 Benefits of AI in Agriculture

The application of AI in agriculture brings numerous benefits that address key issues in modern farming. Some of the most important advantages include:

4.1 Increased Productivity: AI enables better planning and management of farm activities. By using predictive analytics, farmers can make informed decisions that lead to higher crop yields and better livestock performance.

4.2 Resource Optimization: AI systems help monitor soil moisture, nutrient levels, and weather conditions. This information allows precise application of water, fertilizers, and pesticides, reducing costs and environmental impact.

4.3 Early Problem Detection: AI-powered sensors and imaging tools detect signs of disease, pests, or nutrient deficiencies at an early stage. Early detection allows farmers to take corrective action before the problem spreads. [21]

4.4 Cost Reduction: Automation and AI reduce the need for manual labor and minimize input waste. This leads to significant savings and higher profitability.

Figure II.4 Benefits of AI in Agriculture [21].

4.5 Sustainability: AI contributes to sustainable farming by promoting responsible use of resources and reducing greenhouse gas emissions through more efficient practices.

4.6 Decision Support: With access to real-time data and intelligent recommendations, farmers can make smarter decisions that align with both economic and environmental goals.

These benefits make AI a valuable asset for addressing the global challenge of feeding a growing population while protecting natural resources.

5 Challenges in Implementing AI in Agriculture

- Despite its promise, the implementation of AI in agriculture is not without obstacles. One of the major barriers is the lack of digital infrastructure, especially in developing countries. Many rural areas do not have stable internet access or reliable electricity, which limits the deployment of AI-powered systems . [22]
- Another challenge is the high initial cost of AI technology. Equipment such as drones, sensors, and robotics can be expensive, making it difficult for smallholder farmers to invest in them. Additionally, there is a need for specialized skills to operate and maintain these technologies. The shortage of trained personnel in rural areas can slow down the adoption of AI.
- Data privacy and security are also concerns. AI systems rely heavily on collecting and analyzing data, raising questions about who owns this data and how it is used. Farmers may be reluctant to share their information without clear policies on data governance.

Figure II.5 AI in Agriculture Challenges [22].

Furthermore, AI models are not perfect. They may provide inaccurate predictions if the input data is incomplete or of poor quality. This can lead to poor decisions and

loss of resources. Ensuring the reliability and transparency of AI models is crucial for building trust among farmers.

Finally, there is a risk of increased inequality. Wealthier farmers or large agribusinesses may adopt AI more quickly, gaining competitive advantages and leaving smaller farmers behind. Governments and organizations must work to ensure that AI benefits are distributed fairly and equitably.

Table 1 summarizes the key challenges and limitations of the application of AI in agriculture.

Table II.1 Key Challenges and Limitations

Challenge	Details	Impact
Data Availability	Need for large, quality datasets for AI training.	Lack of data limits model accuracy.
Technical Expertise	Farmers require knowledge to operate AI systems.	Training and education are necessary.
High Initial Costs	Equipment and software investment can be expensive.	Small farms may struggle to afford AI technologies.
Infrastructure Needs	Requires reliable internet and power supply.	Remote areas may face deployment issues.
Privacy and Ethics	Data collection raises security and ethical concerns.	It may affect farmers and animal rights.

Addressing these challenges requires coordinated efforts from governments, research institutions, and private sector stakeholders. Investments in infrastructure, education, and policy development are essential to unlocking the full potential of AI in agriculture.

6 Livestock Management Using AI

A. AI is becoming increasingly important in the field of livestock management. The use of AI tools in this area helps farmers improve animal health, increase production efficiency, and reduce operational costs. One of the key technologies used is sensor-based monitoring, which allows for real-time tracking of animal behavior, body temperature, feeding patterns, and movement. [23]

Figure II.6 Ranchers Using NFTs to Track and Share Cattle Data [24].

B. Machine learning algorithms analyze the data collected from these sensors to detect abnormal behaviors or signs of illness before they become serious problems. Early detection can help reduce mortality rates and improve overall herd health. For example, AI can identify when a cow is in heat, optimizing breeding programs and improving reproduction rates.

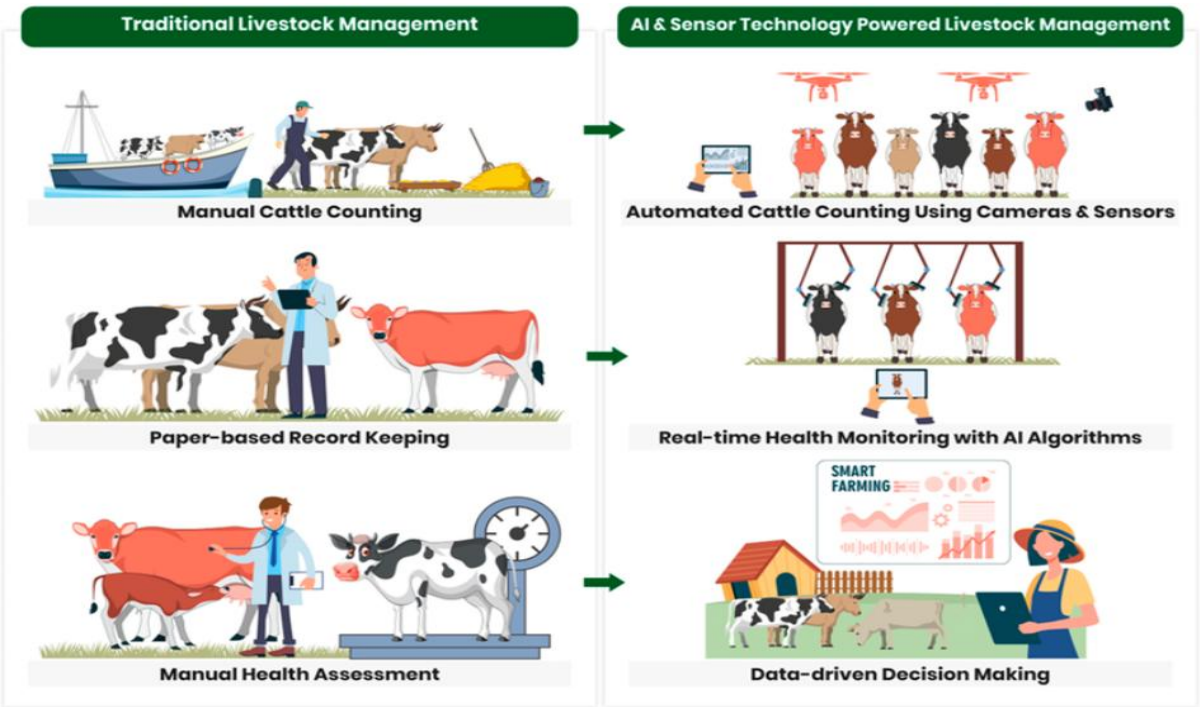

Figure II.7 Traditional method vs. AI and sensor technology powered [23].

C. Computer vision is also used to monitor livestock using cameras that can recognize specific animals and evaluate their physical condition. This reduces the need for manual inspections and improves accuracy. Additionally, AI systems can automate feeding schedules, milk production tracking, and weight gain analysis.

By integrating AI into livestock operations, farmers can enhance animal welfare, increase efficiency, and make more informed decisions. This contributes to more sustainable and profitable animal farming practices.

7 Sheep Identification with AI

Sheep farming presents unique challenges due to the difficulty in individually identifying and managing animals in large flocks. Traditional identification methods such as ear tags or branding can be invasive, prone to damage, and difficult to track manually. AI offers innovative solutions to address these issues through biometric identification techniques.

Facial recognition, powered by computer vision and deep learning algorithms, enables accurate and non-invasive identification of individual sheep based on unique facial features. This technology can be implemented using mobile devices or cameras installed at entry points, feeding stations, or enclosures. [3]

Once individual sheep are identified, AI systems can maintain detailed records for each animal, including health status, feeding history, breeding data, and medical treatments. This level of individual tracking improves herd management, enables targeted interventions, and supports traceability in food production.

AI-powered sheep identification also enhances security by preventing theft and unauthorized access to valuable livestock. The combination of accurate identification and comprehensive data management represents a major advancement in sheep farming.

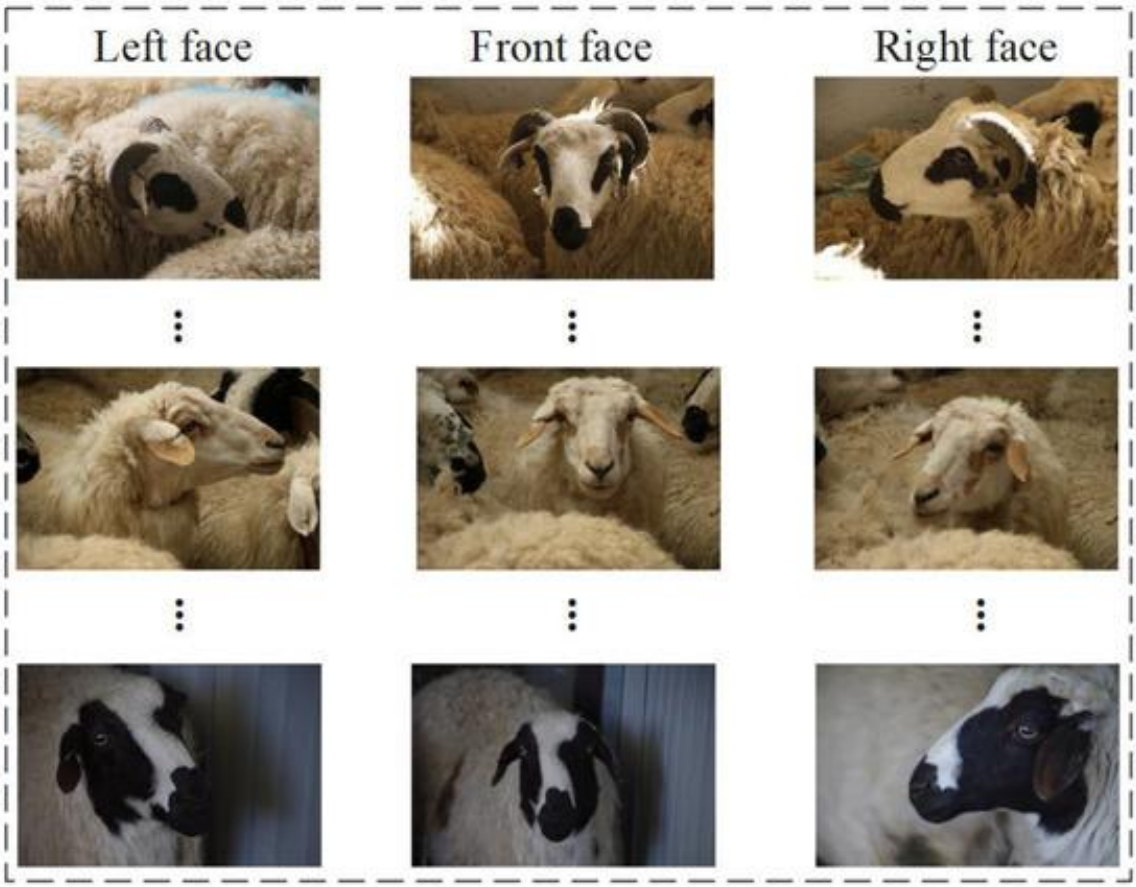

Figure II.8 Sheep face recognition method [3].

8 Future Trends in AI Agriculture

The future of AI in agriculture holds exciting possibilities. As technology advances, we can expect AI systems to become more accessible, affordable, and powerful. Several key trends are likely to shape the future landscape of smart farming:

- Edge Computing: Processing data closer to the source—on farms rather than in distant servers—will reduce latency and improve the speed of AI decision-making.
- Integration with IoT: As more sensors and devices are connected, AI will play a central role in analyzing vast data streams to guide agricultural practices.
- Autonomous Farming Equipment: Self-driving tractors, drones, and robots will take on more responsibilities, further reducing labor needs and enhancing precision.
- AI for Climate Resilience: Predictive models will help farmers anticipate and adapt to climate-related challenges.

- AI-driven Market Forecasting: Farmers will be able to make better economic decisions based on real-time price, supply, and demand predictions.

These trends suggest a future where AI not only increases productivity but also builds a more resilient and responsive agricultural sector.

9 Case Studies

Numerous real-world examples demonstrate the success of AI applications in agriculture:

9.1 Blue River Technology (USA): Developed a "see and spray" robot that uses computer vision and machine learning to identify and spray only the weeds, reducing herbicide use. [25]

Figure II.9 Blue River Technology [2].
Image from Blue River Technology (https://www.bluerivertechnology.com/).

9.2 Cainthus (Ireland): Uses facial recognition to monitor dairy cows, track their behavior, and provide insights to farmers for improving herd health.

9.3 Plantix (Germany): A mobile app that uses AI to diagnose plant diseases and recommend treatments by analyzing user-uploaded images. [26]

Figure II.10 Diagnosing Crop Diseases with Plantix App and AI [2]. Screenshot from Plantix app (https://plantix.net/en/).

9.4 AgriDigital (Australia): Implements AI to manage grain supply chains, improving transparency and efficiency for farmers and buyers.

These case studies illustrate the diversity and effectiveness of AI solutions in different agricultural contexts.

10 Conclusion

Artificial Intelligence is redefining the future of agriculture by addressing traditional challenges and unlocking new opportunities. From precision farming and livestock monitoring to market forecasting and environmental sustainability, AI is empowering farmers with the tools they need to thrive in a changing world.

While the benefits are numerous, it is essential to address the challenges related to cost, infrastructure, skills, and data privacy to ensure inclusive and equitable access to these technologies. With ongoing innovation, education, and policy support, AI has the potential to create a more efficient, productive, and sustainable agricultural system that can meet the needs of a growing global population.

Chapter III

METHODS, RESULTS AND DISCUSSION

Chapter III METHODS, RESULTS AND DISCUSSION

1 System Overview

This chapter presents a step-by-step explanation of the methodology, implementation, results, and discussion of the proposed sheep face recognition system. The system combines deep learning and computer vision techniques to automate the identification process. It is divided into several key phases, including dataset preparation, face detection, feature extraction, model training, and deployment. Each phase is described in detail, followed by an analysis of the results obtained, highlighting the entire process from data preparation to the final identification stage. The system architecture includes three major stages: face detection, embedding extraction, and face recognition. It supports both real-time detection using a webcam and recognition from static images, ensuring flexibility and accuracy in various operating conditions.

2 Tools and Libraries

We used the following tools and libraries to develop and implement our system:

- Python 3.10.
- OpenCV 4 for image processing.
- ONNX for converting the TensorFlow 2 model to ONNX.
- ONNXRuntime for loading and running the SSDMobileNet model.
- Scikit-learn for classification using SVM.
- NumPy for array processing.
- Pickle for saving and loading models.

3 Work environment

The work environment is a desktop computer with the following characteristics:

- Windows : 11 Famille
- Memory : 8 GB
- Processor : Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz (12 CPUs)
- Graphics : NVIDIA GeForce GTX 1060 with Max-Q Design
- Os type : 64-bit
- Disk : 500 GB

4 Installing libraries

4.1 OpenCV

OpenCV (Open Source Computer Vision Library) is an open-source library designed for real-time computer vision and image processing.
It was originally written in C++, but it also has bindings for Python**,** Java, and other languages.

OpenCV offers a wide range of capabilities for computer vision and image processing. It allows developers to perform tasks such as :

- Image and video reading
- Image processing
- Object and face detection
- Shape detection
- Camera calibration

OpenCV in Artificial Intelligence

OpenCV is widely used in AI and machine learning for:

- Face recognition
- Object detection
- Gesture recognition
- Smart surveillance systems

It can work with ML frameworks like:

- TensorFlow
- Scikit-learn
- ONNX

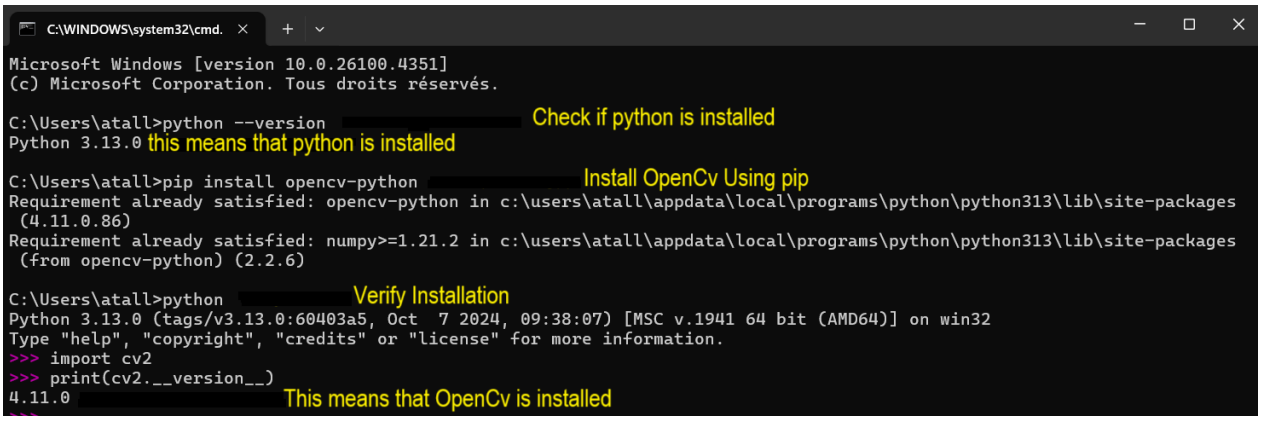

Figure III.1 OpenCV Installation

4.2 Numpy

NumPy (Numerical Python) is a fundamental Python library for numerical computing. It provides powerful support for:

- Arrays and matrices
- Mathematical operations
- Linear algebra
- Fourier transforms
- Random number generation

It is essential for data science, machine learning, image processing, and scientific computing.

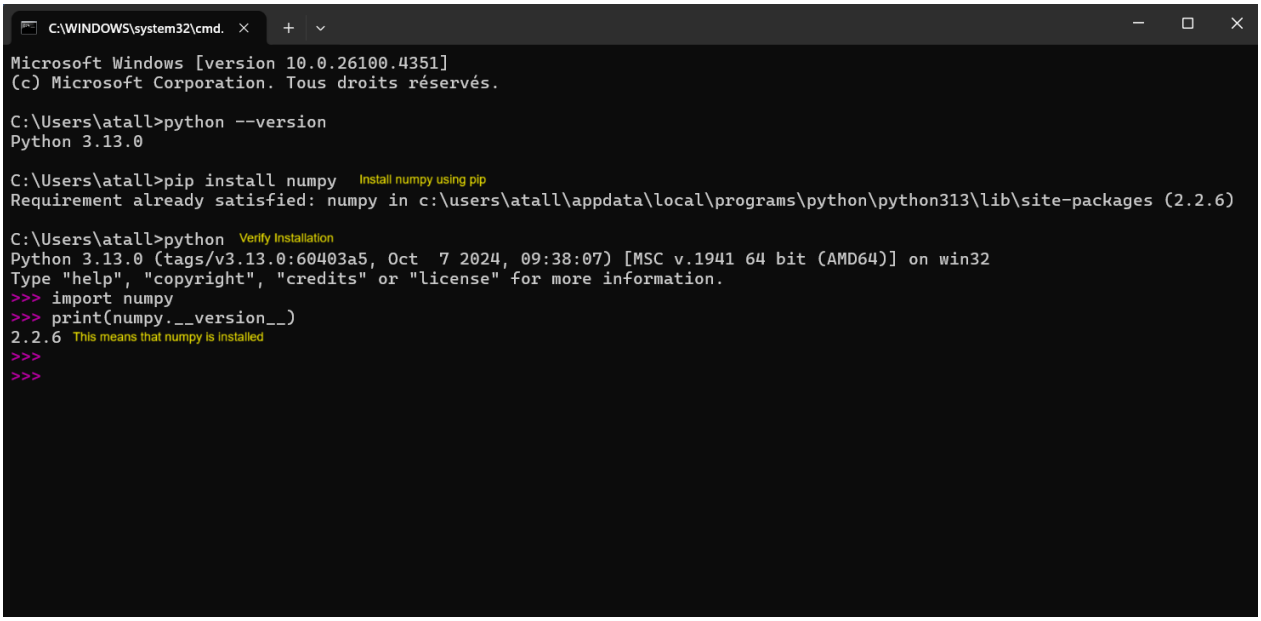

Figure III.2 Numpy Installation

4.3 Scikit-learn

Scikit-learn is a free and open-source Python library for traditional machine learning and data analysis. It provides simple and efficient tools for:

- Classification
- Regression
- Clustering
- Dimensionality reduction
- Model selection
- Preprocessing

Scikit-learn is built on:

- NumPy (for numerical operations)
- SciPy (for scientific computing)
- Matplotlib (for plotting)

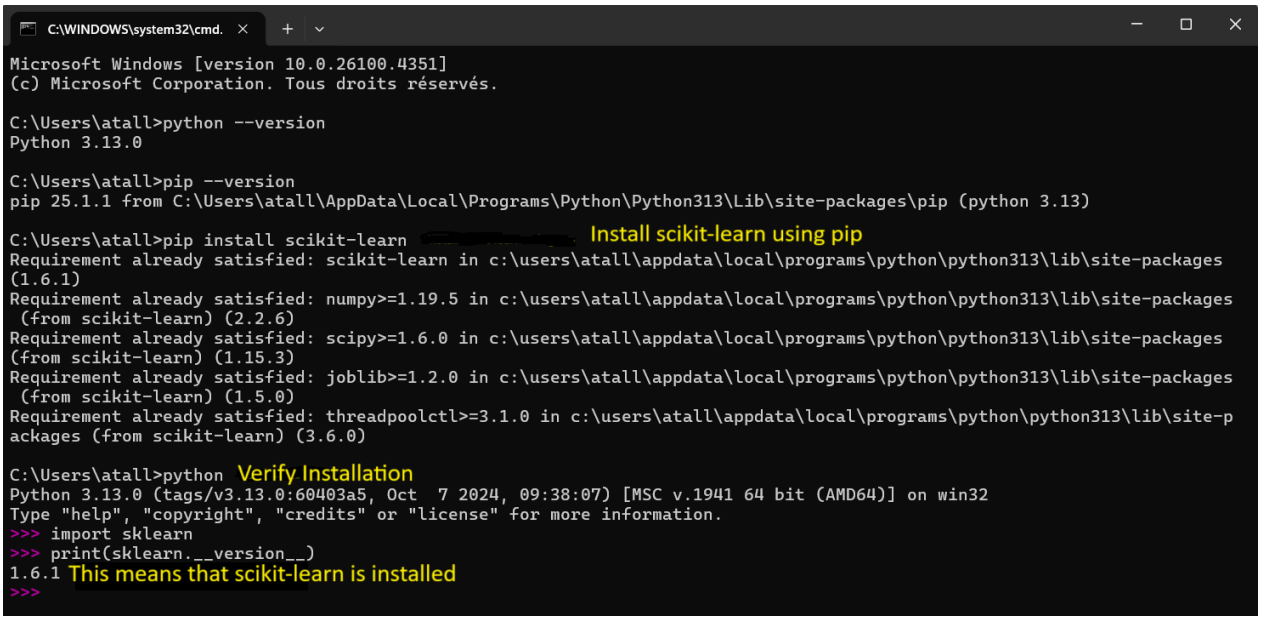

Figure III.3 Scikit-Learn Installation

5 Methodology

5.1 Dataset Preparation

The first step involves collecting and preparing a dataset of sheep face images. Images were captured using a standard camera in various lighting conditions and angles to ensure robustness. Each sheep was photographed multiple times to increase the diversity of the dataset. The images of each sheep were organized into separate folders, each labeled with the corresponding sheep's identity ID, forming the basis for supervised learning, where each folder represents a unique sheep.

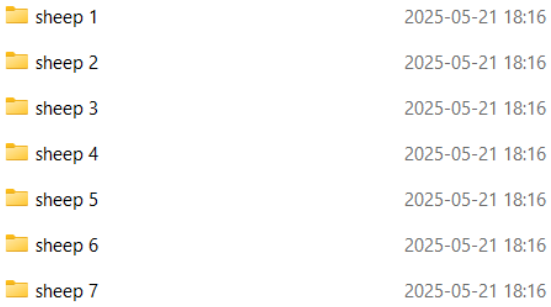

Figure III.4 The Dataset Folders

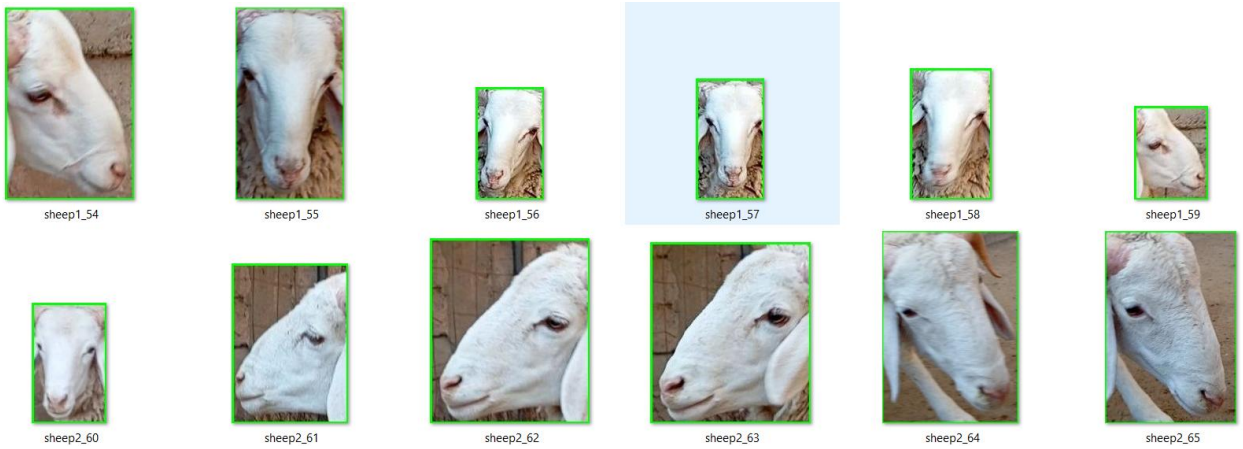

Figure III.5 Sheep image samples

5.2 Project structure

The project has folders organized as follows:

- **Dataset:** In a sheep recognition project, the dataset includes images of sheep faces, each labeled with the corresponding sheep's identity ID.

- **face_detection_model:** The face_detection_model is a project folder that contains a pre-trained object detection model used to detect sheep faces in images or real-time video streams. In this project, the main file inside this folder is based on the SSD MobileNet architecture trained on a local dataset using TensorFlow2.

- **Output:** In the sheep face recognition system, the output folder contains important data files used in the final steps of recognition. These files are saved using Pickle, a Python module that stores objects in binary format.

➢ **embeddings.pickle:** This file stores the face embeddings, which are numeric representations of each sheep face image. Each embedding is a vector that describes the unique features of the face. These vectors are later used by the classifier to recognize and compare different sheep.

➢ **le.pickle:** le file stands for Label Encoder. This file stores the mapping between the sheep names (or IDs) and numeric labels used during training. For example, it may convert 'sheep_01' to label 0, 'sheep_02' to label 1, etc. It is used during both training and prediction to identify sheep correctly.

➢ **recognizer.pickle:** This file contains the trained face recognition model**,** usually a Support Vector Machine (SVM) classifier. It uses the embeddings to predict which sheep is in a new face image. During recognition, this file is loaded to compare new embeddings with known ones and return the most likely identity.

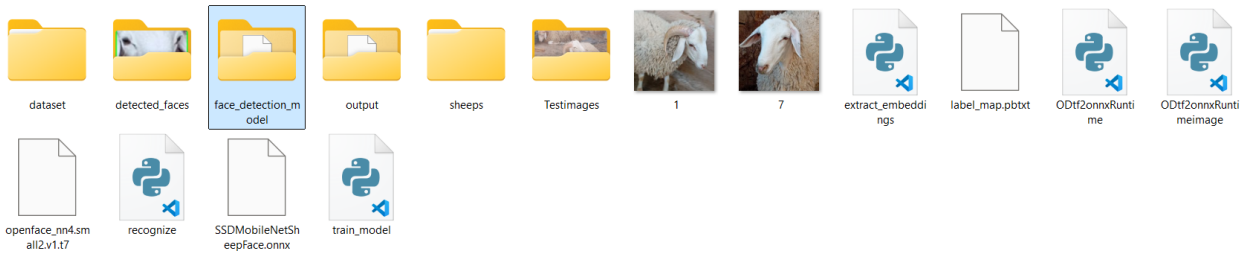

Figure III.6 Project structure

5.3 Explanation of the face detection algorithm

Face detection is a computer vision task that identifies the presence and position of human or animal faces in digital images. The algorithm draws a rectangle (bounding box) around each face.

I.Face Detection Using SSDMobileNet (ONNX)

Face detection is the first step in any face recognition system. It means finding and locating faces in an image or video. In this project, we use a pre-trained deep learning model called SSD MobileNet for this purpose. SSD MobileNet is a lightweight object detection model that combines the Single Shot Detector (SSD) framework with the MobileNet architecture. It provides fast and accurate detection, which is suitable for real-time applications such as sheep face identification.

In this project, we used the SSDMobileNet model to detect the faces of sheep. This model combines two important deep-learning technologies:

- The SSD (Single Shot MultiBox Detector) algorithm and the MobileNet convolutional neural network.
- SSD is a fast and accurate object detection method that detects multiple objects in a single step. It applies filters to the input image and returns bounding boxes around detected objects, making it suitable for real-time applications.
- MobileNet is a lightweight CNN designed to run efficiently on devices with low computing power, such as mobile phones or embedded systems. It uses depthwise separable convolutions to reduce the size and complexity of the model while keeping high accuracy.

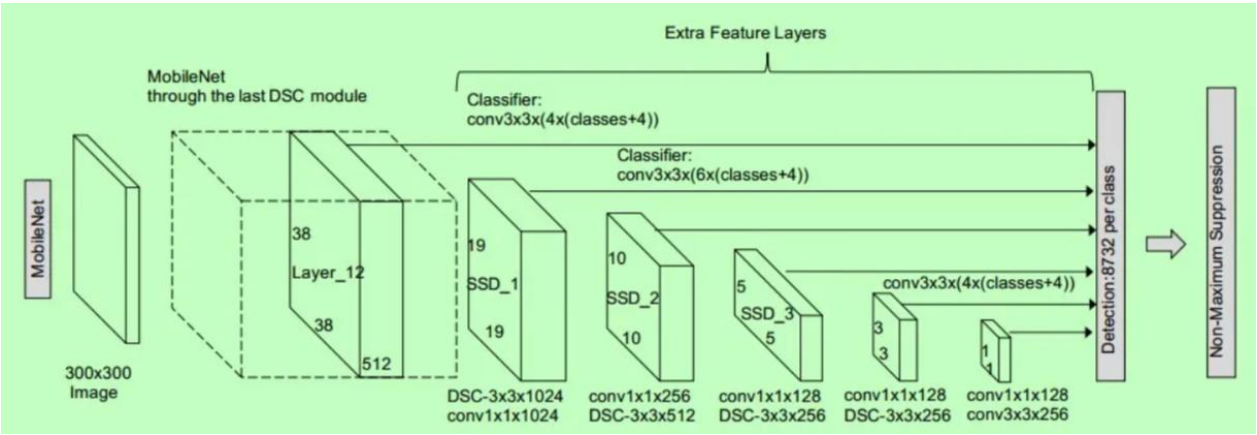

Figure III.7 SSD Mobile-Net

By combining SSD and MobileNet, the SSDMobileNet model provides a good balance between speed and accuracy. It can detect sheep faces quickly and runs well on standard computers. In our system, we used the SSDMobileNet model in ONNX format, which allows fast and efficient execution using the ONNX Runtime. We loaded and used the model in Python with OpenCV for real-time face detection.

a) How It Works

The face detection process begins when an input image or webcam frame is captured. The image is first resized to 300×300 pixels and converted to RGB format. It is then reshaped into a 4D array to match the input requirements of the SSD MobileNet model. This pre-trained ONNX model processes the image and returns outputs, including bounding box coordinates, class IDs, and confidence scores. If the confidence score is higher than a chosen threshold (e.g., 0.5), the bounding box is considered valid. The coordinates (x, y, width, height) are used to draw a rectangle around the detected face. Finally, the detected face is cropped and saved for the next phase: feature extraction and recognition.

Example Code Snippet:

```
input_image = cv2.resize(image, (300, 300))
input_image = cv2.cvtColor(input_image, cv2.COLOR_BGR2RGB)
input_image = np.expand_dims(input_image, axis=0)
outputs = detector.run(None, {input_name: input_image})
```

b) Advantages

- Fast and works in real-time.
- Light and suitable for mobile or low-resource devices**.**
- Good accuracy for animal faces such as sheep.

c) Steps to Convert a TF2 Model to ONNX

The conversion from a TensorFlow 2 (TF2) model to ONNX is necessary to allow the sheep face recognition system to work efficiently across different platforms. Since the detection part needs to run in real time using tools like OpenCV, converting the model to ONNX makes it faster and more compatible with various devices. This step ensures better performance during live sheep identification, especially in field conditions where TensorFlow support may be limited.

The following are the main steps to convert a TF2 model to ONNX:

Step 1: Install Required Dependencies
(for ONNX 1.16.1 and TensorFlow 2.5 install tf2onnx==1.9.3)

```
pip install tf2onnx==1.9.3
pip install onnx
pip install tensorflow==2.5
```

Step 2: Export the TensorFlow Model

Use the TensorFlow Object Detection API's exporter_main_v2.py script to export the model in SavedModel format:

```
python exporter_main_v2.py --input_type image_tensor \
    --pipeline_config_path path/to/pipeline.config \
    --trained_checkpoint_dir path/to/checkpoint/ \
    --output_directory exported-model
```

This will create a folder exported-model/saved_model/.

Step 3: Convert to ONNX Format
Use tf2onnx to convert the SavedModel into an ONNX model:

```
python -m tf2onnx.convert \
    --saved-model exported-model/saved_model \
    --opset 14 \
    --output SSDMobileNetSheepFace.onnx
```

II.Few-Shot Learning Methods

Few-shot learning (FSL) is a method in machine learning where a model learns to recognize new objects using only a few examples. This is useful when we do not have a large dataset. In many real-world tasks, collecting data is expensive or difficult. Few-shot learning helps AI systems learn faster and with less data.

a) Why Few-Shot Learning Is Important

In traditional machine learning, we often need thousands of labeled examples to train a good model. But few-shot learning tries to train models using just 1, 5, or 10 examples. This is important in areas like:

- Medical diagnosis (rare diseases)
- Security (new face recognition)
- Wildlife monitoring (rare species)
- Language translation (low-resource languages)

b) Key Concepts in Few-Shot Learning

- **Support Set**

This is the small set of labeled examples used to teach the model.

- **Query Set**

This set contains new examples that the model needs to classify using what it learned from the support set.

- **N-way K-shot**

This describes a task in FSL. 'N-way' means the number of classes. 'K-shot' means the number of examples per class. For example, a 5-way 1-shot means 5 classes with 1 example each.

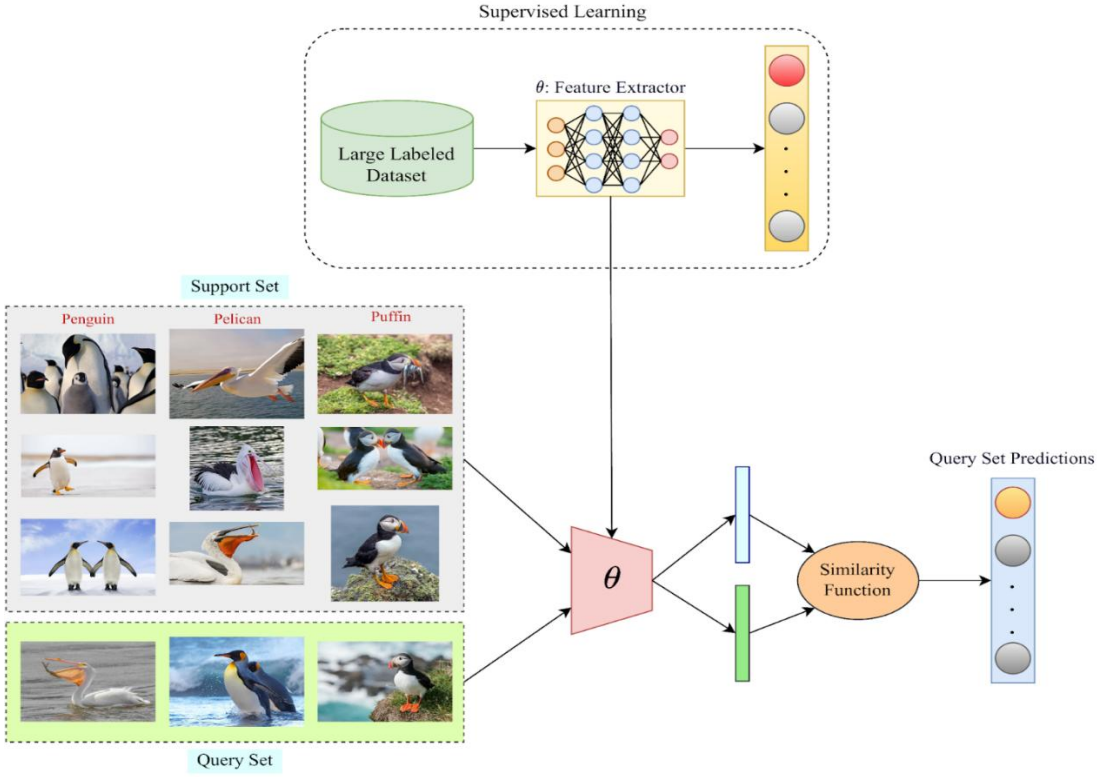

Figure III.8 Concepts in Few-Shot Learning [4]

c) Types of Few-Shot Learning Methods

- **Metric-Based Methods**

These methods learn how to compare new examples to known examples using a distance function.

- Prototypical Networks: Find an average (prototype) for each class and measure the distance to new examples .
- Matching Networks: Compare the similarity between new examples and support examples using embeddings .
- Siamese Networks: Use two identical neural networks to compare image pairs and decide if they belong to the same class . [3]

- **Optimization-Based Methods**

These methods learn how to adapt quickly to new tasks by changing their internal parameters. [3]

- Model-Agnostic Meta-Learning (MAML): Trains a model in a way that it can learn a new task with only a few steps . [14]
- Reptile: A simpler version of MAML that also focuses on fast learning . [14]

- **Generative Methods**

These models generate synthetic examples from the few available samples. [3]

- GAN-based methods: Use generative adversarial networks to create new data samples .
- Variational Autoencoders (VAEs): Help generate additional feature-rich data .

d) Challenges in Few-Shot Learning

Although Few-Shot Learning (FSL) has seen notable progress in recent years, it continues to encounter several persistent challenges.

- **Overfitting:** The model may memorize a few examples instead of learning useful patterns.
- **Generalization:** It is hard for the model to perform well on new, unseen tasks.
- **Dataset Bias:** If the support set is not diverse, the model can be biased.
- **Task Design:** Creating balanced few-shot tasks is not always easy.

e) Recent Advances shows that

- **Few-shot object detection:** Helps detect new objects with few annotated images .
- **Few-shot learning with transformers:** Uses attention-based models like BERT or ViT for better results .
- **Cross-domain few-shot learning:** Learns from one domain and applies it to another (e.g., from cartoon images to real photos).

III.Embedding Models

Embedding models are important tools in machine learning and deep learning. They help us change words, images, or objects into numbers so that computers can understand and work with them. These models make it easier to compare things, find patterns, and do many AI tasks.

a) What is an Embedding

An embedding is a way to turn something like a word or image into a vector, which is a list of numbers. The vector keeps the meaning or important features of the original data. For example, in language, the words "king" and "queen" will have similar vectors.

b) Why Use Embedding Models

- They reduce the size of data
- They keep important information
- They make it easier to find similarities
- They help in tasks like classification, clustering, and recommendation

c) Types of Embedding Models

We have many types of embedding models, we can classify them to:

1. Word Embeddings

These models change words into vectors. Some popular models are:

- Word2Vec: Learns from context and gives each word a vector. [4]
- GloVe (Global Vectors): Uses word co-occurrence to build embeddings [4]
- FastText: Similar to Word2Vec but includes sub-word information .

2. Sentence and Document Embeddings

These models work with full sentences or documents.

- Sentence-BERT: Uses transformer models to create sentence vectors [8]
- Universal Sentence Encoder: Makes sentence embeddings for many tasks [8]

3. Image Embeddings

Image embeddings turn images into vectors. Deep learning models like CNNs are used. Pre-trained models such as ResNet or VGG are common . [15]

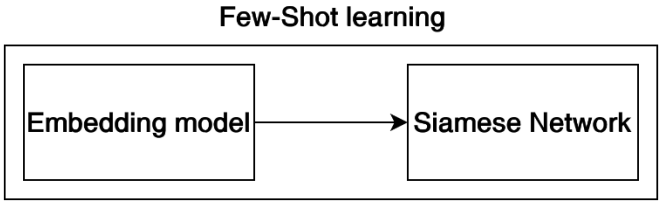

Figure III.9 Few-Shot learning (embedding model + Siamese Network) [13]

4. Graph Embeddings

Graphs have nodes and edges. Graph embeddings turn nodes into vectors.

- Node2Vec: A popular method for graph embeddings
- Graph Neural Networks (GNNs): Learn node representations from the graph structure

d) How Embedding Models Work

Most embedding models are trained using neural networks. They learn to represent similar items with similar vectors by predicting context or relationships. After training, the model can embed new data by passing it through the network.

e) Applications of Embedding Models

- NLP: Translate text, classify documents, or answer questions
- Recommendation Systems: Suggest items based on embeddings
- Computer Vision: Compare image embeddings for recognition
- Fraud Detection: Use transaction embeddings to find strange patterns

f) Challenges of Embedding Models

- Bias: Embeddings may reflect social bias in data
- Interpretability: Vectors are hard to explain
- Resource Intensive: Some models need a lot of time and power

IV.FaceNet Model

FaceNet is a deep learning model developed by Google to perform face recognition, verification, and clustering. It has achieved significant improvements in accuracy and efficiency compared to traditional face recognition systems. FaceNet transforms faces into compact representations known as embeddings, which can then be used for various tasks such as identifying, verifying, or clustering faces.

a) What is FaceNet

FaceNet is a convolutional neural network (CNN) based system that directly learns a mapping from face images to a Euclidean space, where the distances between points (embeddings) represent the similarity between faces. By minimizing a triplet loss function, FaceNet ensures that similar faces have closer embeddings, while dissimilar faces are farther apart. [3]

b) How FaceNet Works

FaceNet works by training on large datasets of labeled face images. The model takes a face image as input and processes it through several layers of convolutional networks. The output is a 128-dimensional vector, representing the facial features. The model uses a triplet loss function, which requires three images: an anchor image, a positive image (same identity), and a negative image (different identity). The goal of training is to minimize the distance between the anchor and positive embeddings and maximize the distance between the anchor and negative embeddings.

c) FaceNet Architecture

FaceNet's architecture can be broken down into the following steps:

- **Input Layer**: A face image is resized to a fixed size (typically 160x160 pixels).
- **Convolutional Layers**: Several convolutional layers are used to extract features from the image.
- **Fully Connected Layers**: After the convolutional layers, fully connected layers map the extracted features to a 128-dimensional vector.
- **Embedding**: The final 128-dimensional vector represents the unique features of the face.

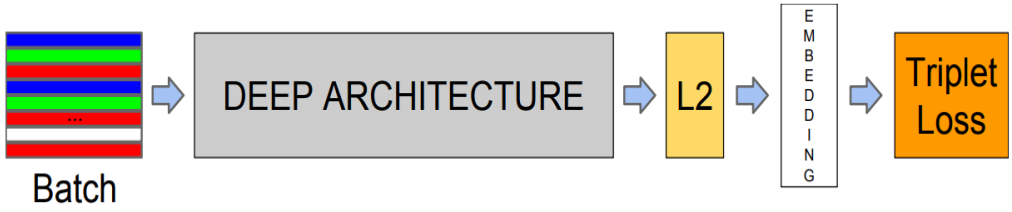

Figure III.10 Deep Learning Architecture [13]

d) Triplet Loss Function

The triplet loss function used in FaceNet is designed to minimize the intra-class variance and maximize the inter-class variance. It is defined as:

$$\left[f(x_{anchor}) - f(x_{positive}) \|_2^2 - \| f(x_{anchor}) - f(x_{negative}) \|_2^2 + \alpha \| \right] \sum_i = L$$

Where:

- f(x)represents the embedding of the input image x,
- 2‖·‖is the Euclidean distance,
- α is the margin that defines how much farther the negative samples should be from the anchor.

This loss function ensures that the network learns to distinguish between different identities by pushing away negative samples while pulling in positive samples. [15]

e) Applications of FaceNet

FaceNet is widely used in various face recognition applications, including:

- **Face Recognition**: Identifying individuals based on their face images.
- **Face Verification**: Verifying if two images belong to the same person.
- **Face Clustering**: Grouping similar faces into clusters based on their embeddings.
- **Face Search**: Searching for faces in a large database using face embeddings.

f) Advantages of FaceNet

- **High Accuracy**: FaceNet provides state-of-the-art accuracy in face recognition tasks.

- **Compact Embeddings**: The 128-dimensional vector representation of faces is compact, making it suitable for large-scale face databases.
- **End-to-End Training**: FaceNet can be trained end-to-end, which eliminates the need for hand-crafted features and manual tuning.

g) Challenges and Limitations

Despite its success, FaceNet faces some challenges:

- **Bias in Training Data**: FaceNet may show biases if the training data is not diverse enough.
- **Occlusion and Lighting Variations**: Variations in lighting and occlusion can affect the model's performance.
- **Privacy Concerns**: The use of facial recognition technology raises concerns regarding privacy and surveillance.

V.Feature extraction phase for face recognition

Feature extraction phase for face recognition goes through serval steps:

a) Feature Extraction Using OpenFace (FaceNet Model)

For each detected sheep face, we extract a 128-dimensional feature vector (embedding) using the OpenFace model based on FaceNet.

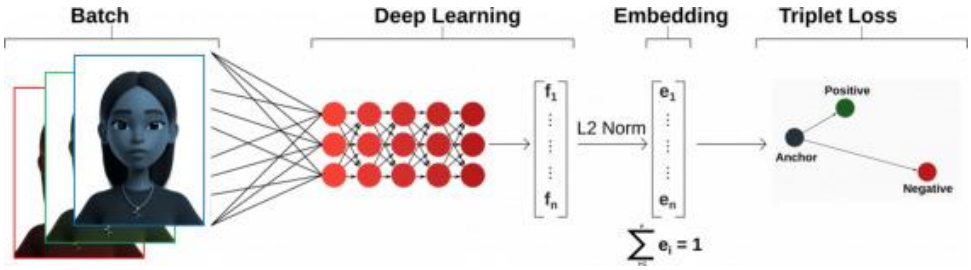

Figure III.11 FaceNet Architecture

b) How It Works:

1. The detected face is resized to 96x96 pixels.
2. A blob is created and passed to the embedding model.
3. The model outputs a 128-d vector representing the face features.

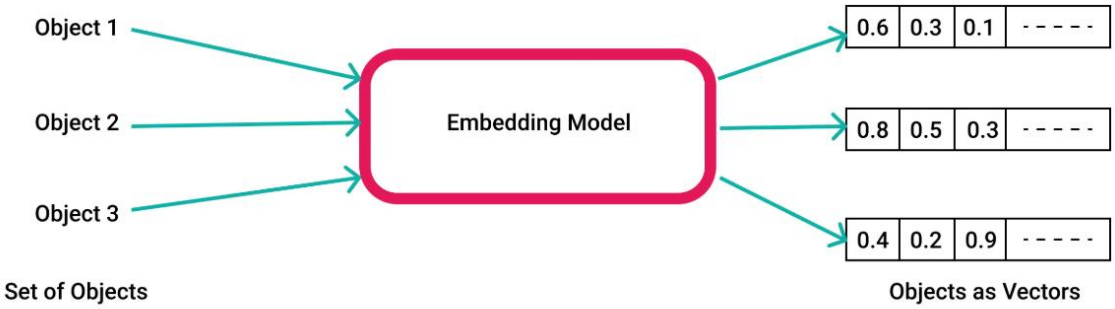

Figure III.12 Embedding Model

c) Face Recognition Using SVM

The embeddings extracted are labeled and used to train a Support Vector Machine (SVM) classifier. This classifier is responsible for identifying the sheep from the feature vectors.

As mentioned in Chapter 1, Support Vector Machine (SVM) is a supervised machine learning algorithm used for classification and regression. In our project, SVM is used to classify the 128-d feature vectors into known sheep identities. SVM tries to find a hyperplane that best separates the data into different classes. We use a linear kernel and enable probability estimates for confidence scores.

- **Steps:**

1. Load all embeddings and their labels.
2. Encode the labels using LabelEncoder.
3. Train an SVM with a linear kernel.
4. Save the trained model and label encoder.

Example Code Snippet:

```
le = LabelEncoder()
labels = le.fit_transform(data["names"])
recognizer = SVC(C=1.0, kernel="linear", probability=True)
recognizer.fit(data["embeddings"], labels)
```

VI. Training and Testing Phases

In this project, the process of recognizing sheep faces is divided into two main phases: **Training** and **Testing**. These two phases are essential to ensure that the system learns from the existing data and performs well on new images.

- **Training Phase**

The training phase is where the system learns to recognize different sheep based on their face images. The training goes through several steps:

1. **Load Dataset**: Read all images of sheep from organized folders (one folder per sheep).
2. **Face Detection**: Use SSDMobileNet (ONNX) to detect the face in each image.
3. **Embedding Extraction**: Use the OpenFace (FaceNet) model to extract a 128-dimensional vector (embedding) from each detected face.
4. **Label Encoding**: Each embedding is associated with the sheep's ID (label), which is encoded using LabelEncoder.
5. **Train Classifier**: Use these embeddings and labels to train a Support Vector Machine (SVM) classifier with a linear kernel.
6. **Save Models**: Save the trained SVM model and the label encoder using Pickle for use during testing and real-time recognition.

- **Inference (Testing) Phase**

In the testing phase, we evaluate the system's performance using new images that were not used during training. The inference phase goes through several steps, figure 10:

1. **Load Models**: Load the saved SSDMobileNet detector, FaceNet embedding model, and SVM classifier.
2. **Read Test Image or Video**: Get input from a webcam or an image file.
3. **Face Detection**: Detect sheep faces using the SSDMobileNet model.
4. **Embedding Extraction**: Extract 128-d embeddings using OpenFace.
5. **Prediction**: Use the trained SVM classifier to predict the identity of the sheep.
6. **Display Results**: Show the result on the screen with a name or "Unknown" if the face is not recognized.
7. draw boxes and display names.

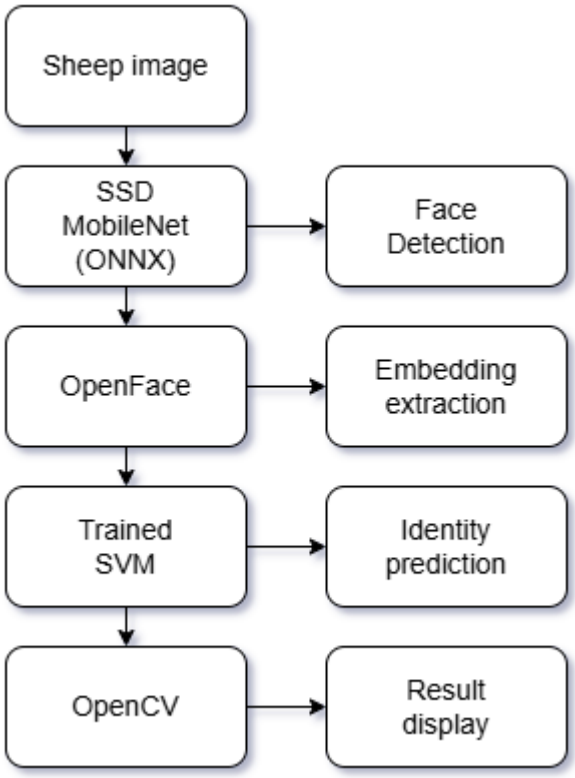

Figure III.13 Inference running flowchart

Using the few-shot-based learning method has several advantages:

- The training phase is offline and happens once using labeled data.
- The testing phase is online, using the trained model to recognize faces in real-time or static images.
- Good results depend on precise training data and a strong embedding model like FaceNet.

The entire recognition pipeline includes the following:

1. Importing Libraries: Load all required Python packages.
2. Load Models: Load ONNX detector, embedding model, and trained SVM.
3. Read Input (Image or Video): Capture frames or read image files.
4. Face Detection: Detect sheep faces using SSDMobileNet.
5. Embedding Extraction: Extract 128-d vectors using FaceNet (OpenFace).
6. Classification: Predict identity using the trained SVM model.
7. Display Results: Draw the bounding box

6 Discussion of Results

The system worked effectively for our dataset. SSDMobileNet provides fast detections that are suitable for real-time use. OpenFace produced discriminative embeddings, and SVM achieved high recognition accuracy.

The name of the sheep is found by using a Support Vector Machine (SVM) classifier. This classifier learns to recognize the sheep by finding the closest embedding vector to the input image from the embedding vectors of the face images saved in the database.

Figure III.15 Unknown sheep – correctly labeled as unknown.

Figure III.14 Sheep 3 – correctly identified (78% confidence).

Figure III.17 Sheep 7 – correctly identified (81% confidence).

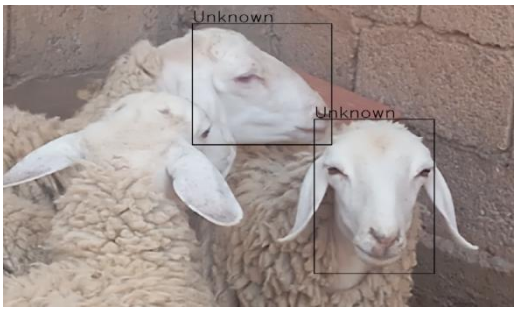

Figure III.16 Sheep 4 not recognized; unknown sheep correctly labeled.

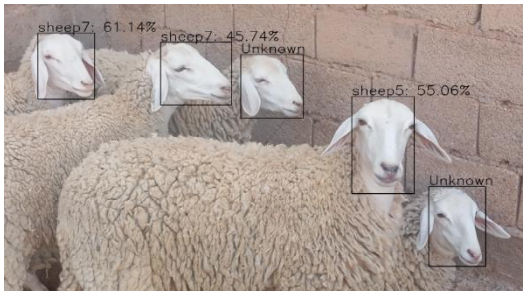

Figure III.19 Sheep 5 and 7 recognized; sheep 4, 6 not recognized; unknown misclassified.

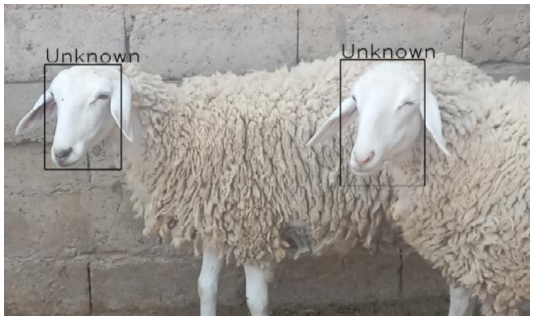

Figure III.18 Sheep 5 and 6 – both not recognized (unknown).

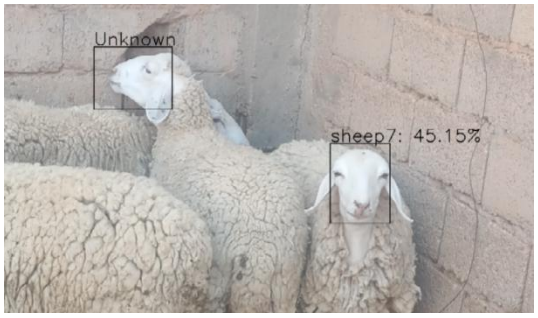

Figure III.21 Sheep 4 – misclassified.

Figure III.20 Sheep 6 – misclassified as sheep 7.

Figure III.23 Sheep 2 – not recognized (unknown).

Figure III.22 Unknown sheep – correctly labeled as unknown.

Figure III.24 Sheep 7 – correctly identified (83% confidence).

Figure III.25 Sheep 6 – misclassified as sheep 7.

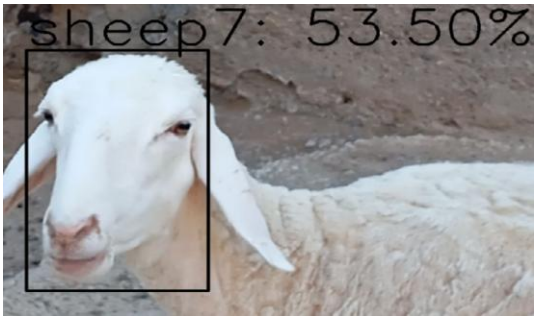

Figure III.11 Sheep 6 – misclassified as sheep 7.

Figure III.26 Sheep 6 – not recognized (unknown).

Figure III.12 Sheep 7 recognized; sheep 6 not recognized.

Figure III.13 Sheep 5 – correctly identified.

6.1 Evaluation of Image-Based Recognition Code

The image-based recognition script was tested using 16 different images. Each image contained one or more sheep faces, and the goal was to check if the system correctly identified the sheep or returned "unknown" when the face was not in the training database.

Table III.1 Recognition Results

Image No.	True Identity	System Output
1	Sheep not in database	Unknown (✓ correct)
2	Sheep 3	Sheep 3 - 78% (✓ correct)
3	Sheep 7	Sheep 7 - 81% (✓ correct)
4	Sheep 4 + unknown	Sheep 4: Unknown ✗Unknown: Unknown ✓
5	Sheep 4, 5, 6, 7 + unknown	Sheep 5, 7: Correct ✓Sheep 4, 6: Unknown ✗Unknown: Misclassified as 7 ✗
6	Sheep 5 and 6	Both: Unknown ✗
7	Sheep 4	Misclassified ✗
8	Sheep 6	Misclassified as 7 ✗
9	Sheep 2	Unknown ✗
10	Sheep not in database	Unknown ✓
11	Sheep 7	Sheep 7 - 83% ✓
12	Sheep 6	Misclassified as 7 ✗
13	Sheep 6	Unknown ✗
14	Sheep 6	Misclassified as 7 ✗
15	Sheep 6 and 7	Sheep 7: Correct ✓Sheep 6: Unknown ✗
16	Sheep 5	Correct ✓

Summary of Performance

Total Images	16
Correct Recognitions	6
Misclassifications	6
Missed Known Sheep (False Unknowns)	6
Correct Unknown Detections	2
Wrongly Classified Unknowns	1
Average Confidence (correct matches)	~80%

Observations

Frequent Misclassification as Sheep 7:Sheep number 6 was often misclassified as sheep 7 (in images 8, 12, 14), suggesting the model has difficulty distinguishing similar sheep.

Good Unknown Detection:The system correctly identified out-of-database sheep in images 1 and 10, but failed in image 5 (unknown sheep misclassified as sheep 7).

Weaker Performance with Multiple Faces:In multi-face images like image 5 and 15, the system was only able to recognize some sheep. Others were missed or misidentified.

Inconsistent Accuracy for Sheep 6 and 4:Sheep number 6 had a poor recognition rate across all images (0 correct out of 5 attempts). Sheep 4 was never correctly identified either.

The system demonstrates acceptable recognition performance for certain individuals, such as Sheep 5 and Sheep 7. However, its recognition accuracy remains inconsistent for others due to several factors, including:

- Confusion caused by visually similar sheep, particularly between Sheep 6 and Sheep 7.
- Weak detection performance in group settings.
- Sheep faces not being directly oriented toward the camera.
- Challenging conditions such as poor lighting or extreme viewing angles.

Addressing these issues requires a significantly larger and higher-quality dataset, with well-distributed and diverse training samples for each individual sheep.

7 Conclusion

In summary, this chapter presented the whole workflow of our sheep face recognition system, encompassing face detection using SSD MobileNet, feature embedding with FaceNet (OpenFace), and classification via Support Vector Machines (SVM). We also developed and tested both real-time and image-based recognition modules. The implementation and evaluation results demonstrate the feasibility and effectiveness of the proposed approach for accurate sheep identification.

CONCLUSION

General Conclusion

At the end of this thesis, we successfully developed a prototype system for recognizing sheep faces using artificial intelligence techniques. The system was designed to work in real-time and was tested using a webcam and a set of sheep face images. The results showed that our method could accurately detect and recognize individual sheep, proving the effectiveness of using AI in animal identification tasks.

Our approach combined object detection using SSD MobileNet and face recognition using FaceNet embeddings and an SVM classifier. This combination allowed us to detect sheep faces quickly and recognize them with high accuracy. The use of ONNX format helped to make the model run efficiently in real-time applications. We also developed Python scripts for different stages of the process: real-time detection, image detection, feature extraction, model training, and recognition.

However, we also faced some challenges during the project. One of the main difficulties was the limited number of training images, which made it harder to achieve high generalization. In addition, capturing clear images of sheep faces in different angles and lighting conditions was not always easy. These limitations highlight the need for more data and the possibility of using advanced learning techniques like few-shot learning to train models with fewer examples.

In conclusion, this thesis demonstrates the potential of artificial intelligence in improving agricultural practices, especially in livestock identification and management. Although the system is still in its early stages, it offers a solid foundation for future improvements. With more data and better hardware, the system can be expanded and used in real farm environments. We hope this work contributes to the development of smart agriculture and encourages further research in this promising area.

REFERANCES

[1] J. Kamilaris and F. X. Prenafeta-Boldú, "Deep learning in agriculture: A survey," *Computers and Electronics in Agriculture*, vol. 147, pp. 70–90, 2018. [Online]. Available: https://doi.org/ 10.1016/j.compag.2018.02.016

[2] W. Liu et al., "SSD: Single shot multibox detector," in *Proc. ECCV*, 2016, pp. 21–37.
[Online]. Available: https://doi.org/ 10.1007/978-3-319-46448-0_2

[3] H. Meng et al., "Livestock Biometrics Identification Using Computer Vision Approaches: A Review," *Agriculture*, vol. 15, no. 1, pp. 102, 2025.

[4] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 4th ed.,

[5] J. Shabbir and T. Anwer, "Artificial Intelligence and its Role in Near Future," Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol., vol. Apr. 2018.

[6] G. Litjens, T. Kooi, B. Ehteshami Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. W. M. van der Laak, B. van Ginneken, and C. I. Sánchez, "A survey on deep learning in medical image analysis," *Medical Image Analysis*, vol. 42, pp. 60–88, Dec. 2017. doi: 10.1016/j.media.2017.07.005.

[7] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, "End to End Learning for Self-Driving Cars," *arXiv preprint arXiv:1604.07316*, Apr. 2016. doi: 10.48550/arXiv.1604.07316.

[8] D. Jurafsky and J. H. Martin, *Speech and Language Processing*, 3rd ed. Upper Saddle River, NJ, USA: Pearson, 2023.

[9] F. Doshi-Velez and B. Kim, *"Towards a rigorous science of interpretable machine learning,"* .

[10] B. D. Mittelstadt, P. Allo, M. Taddeo, S. Wachter, and L. Floridi, "The Ethics of Algorithms: Mapping the Debate," *Big Data & Society*, vol. 3, no. 2, pp. 1–21, Nov. 2016 .

[11] A. Azaria et al., 2020.

[12] T. Mitchell, *Machine Learning*, 1997.

REFERANCES

[13] S. Marsland, *Machine Learning: An Algorithmic Perspective*, 2015.

[14] R. Sutton & A. Barto, *Reinforcement Learning*, 2018.

[15] C. M. Bishop, *Pattern Recognition and Machine Learning*, 2006.

[16] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*, 2012.

[17] R. Girshick, J. Donahue, T. Darrell, and J. Malik,"Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Columbus, OH, USA, Jun. 2014, pp. 580–587. doi: 10.1109/CVPR.2014.81

[18] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," *in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 2016, pp. 779–788. doi: 10.1109/CVPR.2016.91.

[19] R. Wolfert, L. Ge, C. Verdouw, and M.-J. Bogaardt, "Big Data in Smart Farming – A review," Agricultural Systems, vol. 153, pp. 69–80, 2017.

[20] K. G. Liakos, P. Busato, D. Moshou, S. Pearson, and D. Bochtis, "Machine learning in agriculture: A review," Sensors, vol. 18, no. 8, p. 2674, 2018.

[21] K. Zhang, L. Chen, Y. Liu, and Y. Gong, "Applications of artificial intelligence in agriculture: Current trends and future perspectives," Computers and Electronics in Agriculture, vol. 180, 2021.

[22] S. Fountas, D. G. Mandolesi, M. G. Sørensen, and A. G. Tsiropoulos, "Adoption of precision agriculture technologies in Europe," Computers and Electronics in Agriculture, vol. 172, 2020.

[23] C. J. Rutten, A. Velthuis, M. J. M. Huirne, and R. B. M. Huirne, "Sensors to support health management on dairy farms," Journal of Dairy Science, vol. 96, no. 4, pp. 1928–1952, 2013.

[24] NFTGlee, "Ranching NFTs and Cattle Data,"
[Online]. Available: https://nftglee.com/ ranching-nfts-and-cattle-data

REFERANCES

[25] A. Sharma, R. Choudhury, and M. Rani, "Future Trends in Precision Agriculture using AI and Robotics," Journal of Intelligent & Fuzzy Systems, vol. 43, no. 4, pp. 5391–5402, 2022.

[26] Blue River Technology, “See & Spray,”
[Online]. Available: https://www.bluerivertechnology.com/

[27] Plantix, “AI-powered plant disease diagnosis app,”
[Online]. Available: https://plantix.net/en/

[28] OpenAI,“ChatGPT”