PEOPLE'S DECMOCRATIC REPUBLIC OF ALGERIA
Ministery og higher Education and Scientific Research
University Mohamed Khider – BISKRA
Faculty of Exact Sciences, Natural and Life Sciences
Department of Computer science

Order N°: IA_StartUp 16/M2/2023

Thesis

Presented to obtain the academic master's degree in

Computer Science

Option : **Artificial Intelligence (AI)**

Design of a new Intelligent System for Early Prediction of Date palm spider mite (Boufaroua)

---

by: Abdelhak HEROUCAH

Defended in /06/2023 before the members of the jury composed of :

Sihem SLATNIA	Prof.	President
Salim BITAM	Prof.	Supervisor
Okba HOUHOU	Dr.	Examiner

Academic year 2022-2023

Acknowledgments

In the name of Allah, the Most Gracious, the Most Merciful

It is wonderful to hear about your gratitude and appreciation for the blessings and support you have received throughout your academic journey. Expressing gratitude is an important aspect of acknowledging the people and forces that have positively influenced our lives.

Indeed, having a supportive supervisor like BITAM Salim can make a significant difference in one's academic and research endeavors. Their guidance, expertise, and patience can shape our work and help us grow.

Additionally, the encouragement, and understanding of family and friends play a crucial role in our lives. Their unwavering support and belief in our abilities provide the motivation and inspiration needed to overcome challenges and achieve our goals.

HEROUCHA Abdelhak

ملخص:

هذا العمل هو إعداد و يطوير لنظام تنبؤ مبكر مصمم خصيصًا لآفة البوفروة التي تصيب النخيل. الهدف من هذا النظام هو تزويد المزارعين بقدرات تنبؤ متقدمة لدعم اتخاذ القرار وتحسين ممارسات إدارة المحاصيل لديهم. باستخدام خوارزميات التعلم العميق وانترنت الأشياء، قمنا ببناء الشبكة العصبية الاصطناعية الخاصة بنا وحصلنا على دقة جيدة (98٪). بعد ذلك، ننشر النماذج في Raspberry ونحصل على بيانات حقيقية من lilygo / t-a7670sa بواسطة الجيل الرابع للشبكات الخلوية. تم تطوير واجهة سهلة الاستخدام كجزء من تنفيذ النظام، تمكن المزارعين من الوصول بسهولة إلى التنبؤات الناتجة عن النظام وتفسيرها، تم دمج المرئيات والتنبيهات والميزات التفاعلية لتحسين تجربة المستخدم وتسهيل اتخاذ القرار الصائب. نظام التنبؤ المبكر لديه القدرة على إحداث ثورة في الممارسات الزراعية من خلال تزويد المزارعين بمعلومات دقيقة وفي الوقت المناسب.

في الختام، فإن تطبيق نظام التنبؤ المبكر للأغراض الزراعية يمثل تقدمًا كبيرًا في مجال الزراعة الدقيقة. من خلال دمج التقنيات المتقدمة وتقنيات تحليل البيانات، يوفر النظام للمزارعين أداة قوية لتحسين ممارساتهم الزراعية وتحقيق نتائج زراعية مستدامة وفعالة. يمتد التأثير المحتمل للنظام إلى ما وراء المزارع الفردية، حيث يساهم في الهدف الأكبر المتمثل في ضمان الأمن الغذائي، والإشراف البيئي، والاستدامة الاقتصادية في القطاع الزراعي.

الكلمات المفتاحية: انترنت الأشياء، التعلم العميق، الشبكة العصبية الاصطناعية، التنبؤ المبكر، التنبؤ، آفة البوفروة ، النخيل.

Abstract:

This work is considered as design and implementation of an early prediction system specifically designed for Oligonychus afrasiaticus (Boufaroua) affecting date palms. The objective of this system is to provide farmers with advanced forecasting capabilities to support decision-making and optimize their crop management practices. By using deep learning algorithms and IoT. We built our own architecture and got a good accuracy (98%). Next, we deploy the models in Raspberry and get real data from lilygo/t-a7670sa by 4G. To ensure usability and accessibility, a user-friendly interface is developed as part of the system implementation. The interface allows farmers to easily access and interpret the predictions generated by the system. Visualizations, alerts, and interactive features are incorporated to enhance user experience and facilitate informed decision-making. The early prediction system has the potential to revolutionize agricultural practices by empowering farmers with timely and accurate information. In conclusion, the implementation of the early prediction system for agricultural purposes represents a significant advancement in the field of precision agriculture. By integrating advanced technologies and data analysis techniques, the system offers farmers a powerful tool to optimize their farming practices and achieve sustainable and efficient agricultural outcomes. The system's potential impact extends beyond individual farms, as it contributes to the larger goal of ensuring food security, environmental stewardship, and economic sustainability in the agricultural sector.

Keywords: IoT, deep learning, neural network, early predictor, prediction, oligonychus afrasiaticus, DPM, Boufaroua.

Contents:

Chapter 3: Design of the suggested system

Figure list:

List of tables:

General Introduction

General Introduction

1. Context:

Dates fruits are one of the most important agricultural products in the country, as Algeria owns 18 million palm trees of various types and classifications, and their annual production volume reaches 400,000 tons annually **[1]**. There are large areas dedicated to palm cultivation in Algeria. According to agricultural data, the area of land cultivated with palm trees in Algeria is estimated at 172,000 hectares **[2]**.

Palm cultivation in Algeria suffers from several diseases that affect the productivity of palm trees, such as black blight. Boufaroua (named Oligonychus Afrasiaticus by the scientists) disease is considered one of the most dangerous pests that affect dates. The infection increases with a lack of irrigation water and neglect of service, and the percentage of damage and loss in the crop may reach more than 80%, especially during dry years and warm.

2. Studied Problem:

Boufaroua is one of the most common pests in Algeria. Boufaroua disease is difficult to detect in the early stages with the eyes, and it is also extremely difficult to control it in the advanced stages because it is rapidly spreading. Boufaroua destroys a large number of dates in a short period if it is not treatment in time. The problem here is that the farmer does not know the appropriate time to use chemicals or any other methods that eliminate the insect.

3. Objective:

Our goal is to create an intelligent system predicts the appropriate time to use chemicals or other pest treatment methods to help the farmer.

Our idea is to use collected data to build a prediction model with our architecture, and then use the real-time data by using sensors to predict, then notify the result for the farmer. Based on deep learning and IoT Technologies.

General Introduction

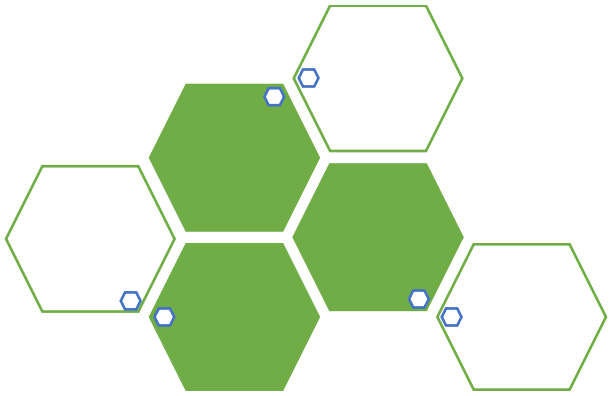
4. Manuscript organization:

Chapter one: we will discuss two topics Internet of Things (IoT) and Smart Farming. For IoT, we will cover its definition, characteristics, architecture, protocols, and applications. Regarding Smart Farming, we will define it as the application of modern technologies in agriculture, with a focus on IoT.

Second chapter deep learning and related work: neural networks description and discussion of related work.

Third chapter system design: in this chapter we discuss the design aspects of our system, showing the different design diagrams, which describe the architecture and operation of our system.

Fourth Chapter system Implementation: This chapter will present the software/hardware environment as well as the programming languages and the tools used in this study. Also, the results reached are presented in this chapter.



Chapter 1:

IoT and Agriculture

Chapter 1 IoT and Agriculture

1.1. Internet of Things

1.1.1. The IoT definition

Definition 1: IoT is defined as a huge network of interconnected things; things may be small devices, big machines and also includes people. Via this interconnected network communication can occur between things-things, things-people, and people-people **[3]**.

Definition 2: IoT is known as a set of smart things/objects such as home devices, mobile, laptop, etc., addressed by a unique addressing scheme and connected to the Internet through a unified framework this framework may be cloud computing **[4]**.

Definition 3: define IOT into three categories as below: Internet of things is an internet of three things: (1). People to people, (2) People to machine /things, (3) Things /machine to things /machine, Interacting through internet **[5]**.

Definition 4: IoT enables anything to communicate with anything, the "thing" could be a living or non-living, a machine or non-machine things after digitizing these things **[6]**.

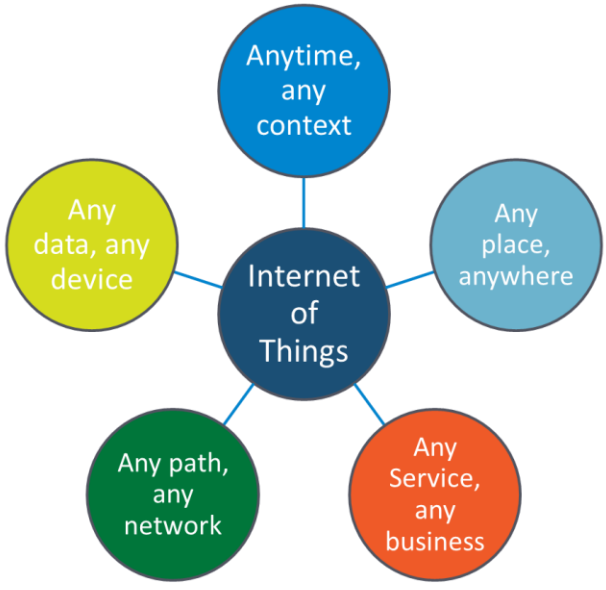


Figure 1.1: *IoT Definition [6]*

1.1.2. Characteristics of IoT

The Internet of Things offers a variety of features that allow heterogeneous objects to dynamically link via a secure connection The following figure (Figure 2) depicts some of these characteristics:

Interconnectivity:

In terms of the Internet of Things, anything can be linked to the global information and communication infrastructure **[5]**.

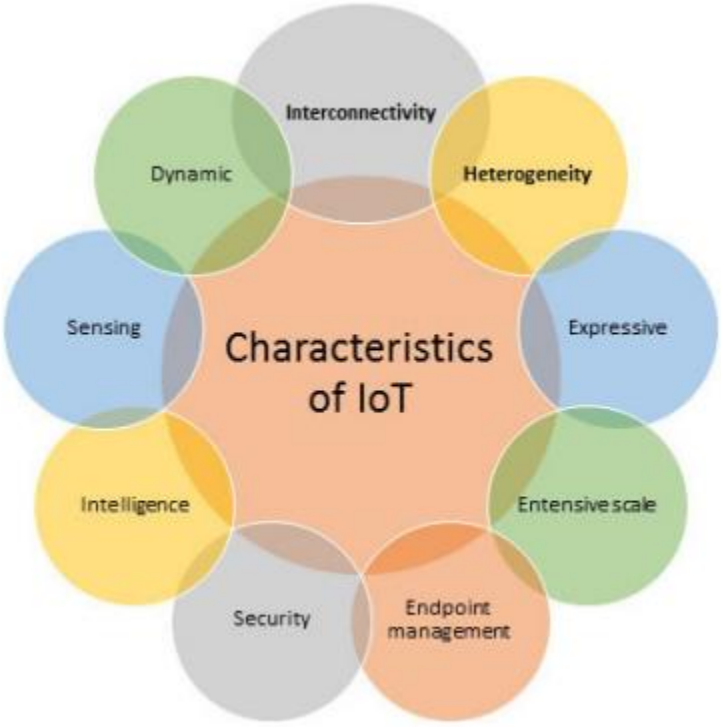


Figure 1.2: *Characteristics of IoT [3].*

Heterogeneity:

The devices in the Internet of Things are heterogeneous since they are based on multiple hardware platforms and networks. They can communicate with other devices or service platforms via various networks **[5]**.

Dynamic changes:

The state of devices, such as sleeping and waking up, being connected and/or disconnected, as well as the context of devices, such as location and speed, modify dynamically. Furthermore, the number of devices might change dynamically **[5]**.

Sensing:

Things must check frequently for any changes in their status or physical environment for example changes in temperature, humidity, motion, air quality, speed, and pressure **[3]**.

Intelligence:

Things recognize themselves, and they gain intelligence through context-related decisions enabled by their ability to relay information about themselves. They can obtain data that has been gathered from other items **[3]**.

Expressing:

Working on the command lines is sometimes required by the developer. It even writes code in a high-level language. But the IoT system expresses the output to the end-user either in the form of a Graphical User Interface (GUI) or in terms of actions performed on things **[3]**.

Extensive scale:

The number of devices that must be controlled and communicated with one another will be at least an order of magnitude greater than the number of devices currently linked to the Internet **[3]**.

Endpoint management:

Endpoint management is required in an IoT system since the entire system can fail. For example, in a smart IoT-based Fruit juice machine that orders Fruits from the nearest retailer when it is finished, failure can be faced in case of unavailability of the recipient to receive the order at the time of delivery **[3]**.

Security:

As we enjoy the advantages of IoT, we must not forget about safety. As both the creators and recipients of the IoT, we must design for safety. This encompasses the security of our personal information as well as the security of our physical well-being. Securing endpoints, networks, and the data flowing through them **[5]**.

1.1.3. IoT architecture and components

The Internet of Things should be able to link and transport data between billions and trillions of devices. A well-structured and organized architecture is required for this. This architecture is designed to support a diverse set of components and technologies that comprise the IoT ecosystem **[3]**.

1.1.3.1. Three-layer architecture of IoT

The architecture consists of three layers, namely, the perception, network, and application layers (Figure 1).

The perception layer (devices layer): The perception layer is the outer layer, which contains sensors for perceiving and collecting environmental data for example (DHT 22 is a temperature and humidity sensor) and light sensors etc..**[3]**. The sensors link between the digital and physical environments, allowing for real-time data collection and processing **[5]**.

The network layer: These tiny sensors will generate massive amounts of data, which requires the development of a strong and high-performance wired or wireless network infrastructure as a transport medium **[5]**. The network layer connects network devices and servers to create IoT or smart devices. Its capabilities are also employed to transmit sensor data. the following technologies can use to transmit data: RFID, 3G, GSM, UMTS, Wi-Fi, Bluetooth low energy, Infrared, and ZigBee **[3]**.

The application layer: The application layer enables consumers to receive specialized services. It is the element of the IoT architecture that defines many IoT applications that will be deployed, such as smart homes and smart health **[3]**.

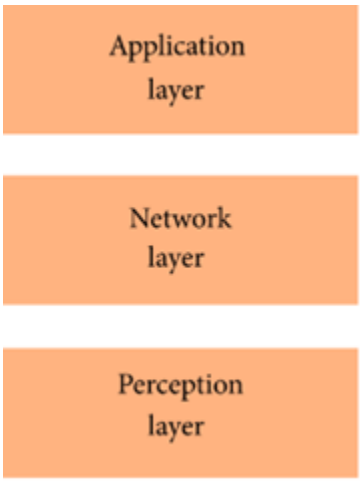


Figure 1.3: Architecture of IoT [3].

1.1.3.2. IoT protocols

IoT protocols are a set of rules for transferring data between electronic devices under a predetermined agreement regarding information structure and how each side will send and receive data **[7]**. IoT protocols can be divided into two categories: IoT network protocols and IoT data protocols.

- **IoT Network Protocols:**
  - Wi-Fi
  - Bluetooth
  - ZigBee
  - LoRaWAN
- **IoT Data Protocols:**
  - AMQP
  - MQTT
  - HTTP
  - CoAP
  - DDS
  - LwM2M

1.1.4. IoT applications

- Weather and water systems
- Vehicular traffic
- Smart parking lots
- Surveillance systems
- Environmental pollution
- Smart energy and smart grids
- Remote patient monitoring
- Smart homes:

Smart homes could be monitored by using the data that are generated by the sensors. For instance, innovative demand response functions can be implemented or by monitoring the pollution, it will be possible to alert customers if the pollution exceeds its marginal limit **[8]**.

1.2. Smart Farming

1.2.1. Smart Farming definition

The application of modern technologies into agricultural activities is the essence of smart farming. UAVs, artificial intelligence, big data, IoT, satellites, and other technologies are making farming and agriculture "smart," allowing growers to optimize their labor and achieve greater outcomes. All of this minimizes the amount of physical labor required, lowers financial expenses, and boosts production volumes, making agriculture more cost-effective **[9]**.

1.2.2. Internet of Things (IoT) in agriculture

The IoT adoption in agriculture revolves around a system that is built for monitoring crop field with the help of sensors capturing data on light, humidity, temperature, soil moisture, and more, and automating the irrigation system. It is thus a system of smart farming which is a capital intensive and hi-tech system geared towards growing cleaner and sustainable food for the masses. It is basically the application of modern ICT (Information and Communication Technologies) into agriculture. IoT based technology of farming such as smart farming creates

opportunities for farmers to improve productivity and at the same time, help reduce waste on the farm. For example, the number of farm trips (by farm vehicles) are significantly reduced as some activities are pre-programmed and continuous feedback to farmers sent automatically. The focus of adoption of IoT-based farming is not only on the traditional, large farm operations but also strategically towards approaches that tend to lift other emerging techniques and trends in agriculture such as organic farming, family farming as well as both complex and simple small spaces farming of cattle culture, preservation of high-quality seeds and subsequently enhancing transparent farming. The IoT-based agricultural environment positive impacts are enormous in terms of ensuring food for all for the ever-growing population of the world. For example, IoT-based smart farming can provide great benefits, including more efficient water usage or the optimization of inputs and treatments. Some of the major applications of IoT-based smart farming include the points explained in the following sections **[3]**:

IoT and precision farming:

Precision farming allows farmers better monitor the specific needs of various plants and animals and modify their nutrition accordingly, reducing disease and improving herd health **[3]**.

IoT and agricultural drones:

Drones equipped with IoT devices are now being used to augment and improve agricultural management practices. Some advantages of both ground and aerial drones include crop and livestock monitoring, crop health assessment, crop spraying and planting, and soil field analysis. Consequently, the use of drones provides real-time data, allowing for timely decision-making and potential yield increases **[3]**.

1.2.3. Smart Farming Benefits

- speeding up data collection and processing
- increasing accuracy and precision level
- enhancing production efficiency
- reducing production costs
- lowering the need for manual labor

- increasing crop yield
- minimizing driver stress
- simplifying risk forecasting
- streamlining task recording and reporting
- boosting sustainability

1.2.4. Smart Farming Limits

Despite all of the benefits of smart farming and the use of advanced technologies, this concept is fraught with difficulties:

Lack of Internet: A stable connection to the Internet is required to implement smart agriculture technologies. Regrettably, it is not available in every country **[10]**.

Low awareness: Modern systems necessitate fine-tuning and knowledge of their operational features. Some farmers are unaware of the benefits of smart farming technologies or are unsure of how to work effectively with them **[10]**.

Data Privacy and Security: Smart farming involves the collection and analysis of large amounts of data, including sensitive information about crops, soil conditions, and farm operations. Ensuring the privacy and security of this data is crucial. Farmers need to implement robust cybersecurity measures to protect against data breaches, unauthorized access, and potential misuse of sensitive information **[10]**.

Limited Tailoring: Smart farming technologies often provide generalized solutions that may not fully address the specific needs and conditions of individual farms. Farmers may require more customized approaches or adaptations to suit their unique circumstances, which may not be readily available or require additional development **[10]**.

Cost: Implementing smart farming technologies can involve significant upfront costs. The installation of sensors, drones, precision equipment, and data management systems can be expensive, making it challenging for small-scale farmers or those with limited resources to adopt these technologies **[10]**.

1.3. Oligonychus afrasiaticus

1.3.1. Oligonychus afrasiaticus (McGregor 1939) – Boufaroua -

It is a date palm mite **(DPM),** which is a spider mite in the family Acari:Tetranychidae a major Disease Of the date palm in North Africa and the Middle East. It is characterized by rapid spread and difficulty in detecting it in the early stages. It damages the crop and causes huge economic losses [**11]**.

The date mite has different names in different countries, as follows:

- Boufaroua in Algeria.
- Boufazroua in Tunisia.
- Goubar in Iraq.
- Goubash in Libya.

Countries are infested by DPM are: Algeria, Palestine, Iran, Iraq, Saudi Arabia, United Arab Emirates, Bahrain, Oman, Kuwait, Yemen, Jordan, Libya, Tunisia, Morocco, Egypt, Sudan, Chad, Mauritania, Mali, and Nigeria **[11]**.

Oligonychus afrasiaticus attacks mainly plants of the family Arecaceae and Poaceae **[11]**. The Arecaceae (Palmae), also known as the Palm family, consists of perennial trees. They have a rhizomatous, lianas or usually arborescent stem, with large, sheathing, plicate leaves, a fleshy, usually drupaceous fruit, and seeds lacking starch. The plicate leaf posture and drupaceous fruit are likely apomorphies for the family (Figure 4) **[12]**. Additionally, the Poaceae (Gramineae), or the Grass family, derived its name from "poa," the Greek name for grass. It encompasses 668 genera and 9,500 species **[12].**

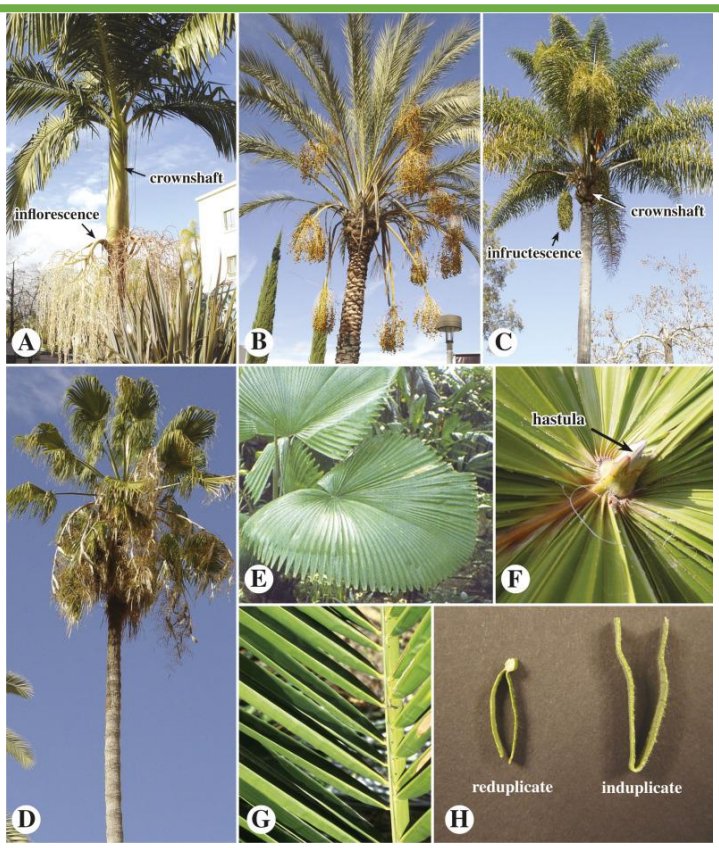


Figure 1.4: *A. king palm. B. Phoenix dactylifera, date palm. C. queen palm. D. Washingtonia robusta. E. Licuala peltata, with palmately lobed leaves. F. Livistona drudei leaf close-up. G. Jubaea chilensis leaf close-up. H. Reduplicate and in induplicate [12].*

1.3.2 The DPM environment, behavior, and biology

The Oligonychus afrasiaticus develops through four developmental stages the egg, larva, nymph, and adult. Adult measurement is between 0.17 mm and 0.2 mm in width and 0.2 mm to 0.5 mm in length. The egg is round and about 0.1 millimeters in diameter, and it has colors yellow, pink, or red (Figure 5). A single female can lay between 50 and 100 eggs, usually in groups on fruits and fronds, depending on the temperature and host plants. The pre-oviposition lasts about three days (the period between the emergence of an adult female insect and the start of its egg-laying). The female can survive approximately for three to four weeks in the summer and a few months in the winter. No matter the food supply or host, the gender percentage of Oligonychus afrasiaticus is always female-biased (85%). The eggs hatch into a small oval larva (0.2 mm) with three pairs of legs in two to three days. The larva feeds for two days before entering a dormant state (larval chrysalis) for one day before moulting into a protonymph with

four pairs of appendages. The protonymph feeds for 1-2 days before entering a 1-day quiescent phase (protochrysalis) and molting into a deutonymph that feeds for 1 day before entering a 1-day quiescent period (deutochrysalis) and molting into an adult. When the temperature rises, the duration of all DPM phases decreases. The larval phase of DPM is 2-3 days, the nymphal period is 4-7 days, and the generation time is 8-14 days (Figure 6). At temperatures of 35°C and 20°C, the typical generation time was 6 and 14 days, respectively. Thus, the number of generations per season is heavily influenced by the weather and fruit chemistry. This will affect the severity of the mite's harm. The mites' dusty silken webbings around the fruit create a microclimate in which the mites feed and reproduce, creating colonies away from predators and unfavorable environmental conditions. The DPM favors hot, dry weather, and generation time may be as short as two weeks depending on the temperature. As a result, 10-12 generations of mites may be produced yearly, depending on temperature, mortality factors (pathogens and predators), and host type. The last generation can live for 5 months during the wintertime. Oligonychus afrasiaticus typically colonizes date palms at the adaxial leaf surface and fruit bases. The optimal temperature for O. afrasiaticus development is between 30.5 and 32.7°C, and reproduction slowed as the temperature decreased from 35 to 20°C. During off-fruiting times, the mites migrate to young fronds and grasses in the vicinity of infested palms to overwinter. The mite's dense webbing on the fruits is supposed to protect the eggs and quiescent stages from predators and acaricides **[11]**.

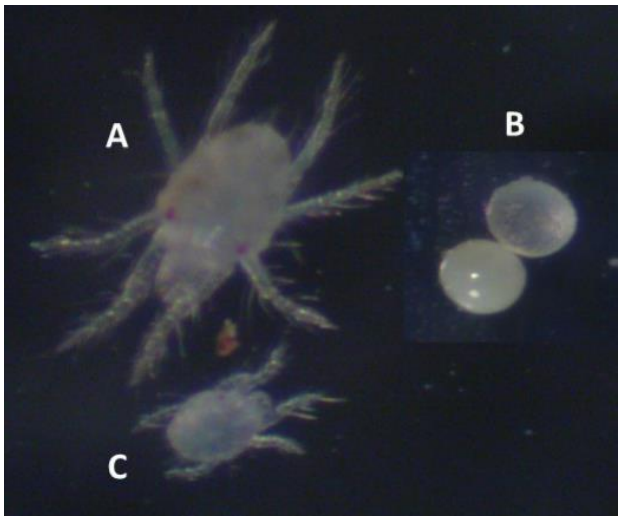


Figure 1.5: *Old world date palm dust mite, Oligonychus afrasiaticus, different life stages; (A) adult, (B) egg and (C) larva.[11]*

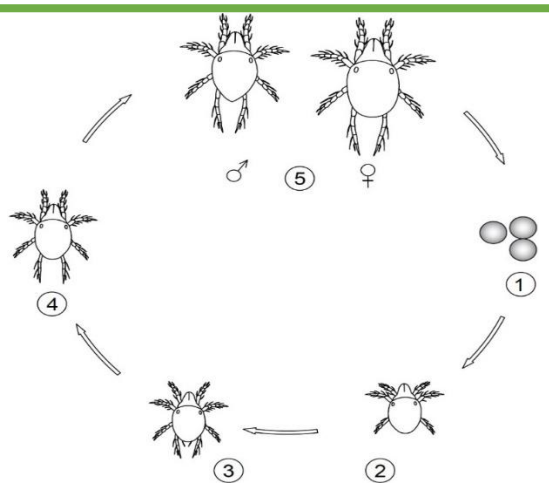


Figure 1.6: *Life cycle of the date dust mite, Oligonychus afrasiaticus; (1) egg, (2) larva, (3) protonymph, (4) deutonymph and (5) adult,[11]*

1.3.3. DPM dispersal and phenology

The date dust mite generally starts its colony on the date palm at the adaxial leaf surface. The population of O. afrasiaticus grew during hot weather, and as many as 10-12 generations can be produced yearly, with the last generation living for 5 months during the winter. Fruit chemistry has a significant impact on the date dust mite's reproduction. Winds, insects, and fruit-eating birds are responsible for mite spread. During the summer months, the accumulation of dust on infested fruit bunches is exacerbated by a turbulent breeze carrying dust. The crawling causes mites to proliferate within fruit bunches and migrate to young fronds on the same palm. The mite's dispersal phase appears to begin in reaction to a lack of food sources. O. afrasiaticus migrates to fibers, infertile date trees, and grasses in the dearth of fruits. DPM phenology is the timing of period life cycles activities such as the yearly appearance of date palm fruits or infestation. Many factors affect this phenology, the most significant of which are photoperiod, predator presence, alternative hosts, date palm flowering phenology, and acaricide application. For example, during the date palm fruiting period, the kimri fruit stage is a key regulating factor of O. afrasiaticus phenology and abundance. During the fruiting off-season, the young fronds provide refuge and survival places for mites in overwintering **[11]**.

1.3.4. Economic losses

Mite infestation can result in total losses, and 100% yield losses have been recorded in Iran. In Iraq, casualties ranged from 50% to 80%. In Sudan, an estimated 96.6% loss in palm productivity was recorded for the popular 'Barakawi' cultivar. In 1981, approximately 30%-70% of dates were lost in Algeria, while an annual loss of 70% was recorded in Mauritania **[11]**.

1.3.5. Infestation

DPM reproduces primarily on growing or unripe fruits (kimri stage), which have low sugar, high acidity, and high moisture content. Date fruits are classified into five phases (in Arabic), which are hababook, kimri, khalal, rutab, and tamr (Figure 7). Web spinning and egg laying are at their peak during the kimri period **[11]**.

The larvae and nymphs of O. afrasiaticus, as well as the adults, feed primarily on date fruit in the kimri and pre-khalal stages. Because of mites eat date fruit, the fruits become silvery-gray and then turn reddish brown (Figure 8). infested fruits become scarred and coated in the mites' dense silken webs. These webbings collect dust particles, giving the fruits a dusty look. (Figure 9) **[11]**.

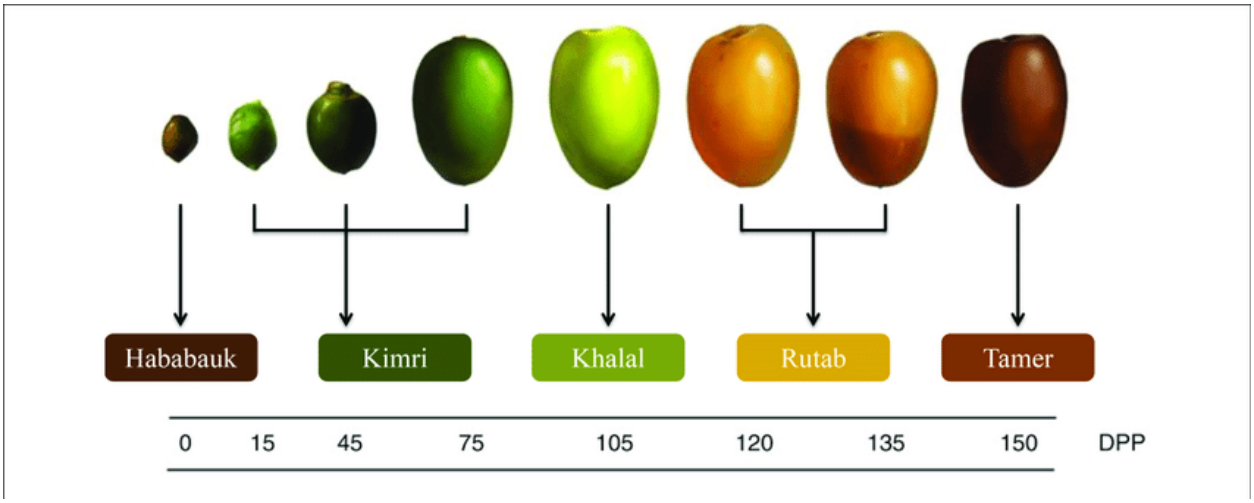


Figure 1.7: *The five growth stages of a date fruit by days post pollination (DPP) [13]*

Figure 1.8: Dust mite infestation on immature date crops with colour variations ranging from pale green to reddish brown.[11]

Figure 1.9: Clumped dust mite infestation with highly infested bunch next to uninfested one. [11]

1.3.6. Sensitivity of different date palm types to infestation

The Iraqi type 'Sayer' has been found to be less susceptible to O. afrasiaticus than other types. 'Deglet Noor' is the most vulnerable to dust mite infestation to date. Seasonal pests, such as date mites, typically coincide with the time when the majority of date fruits are in the kimri stage. The asynchrony between the pest and host plant allows the latter to avoid attack by finishing the development of its susceptible stage before or after the pest appears. In this regard,

late or early-maturing date palm cultivars may avoid mite assaults and thus exhibit some resistance **[11]**.

1.3.7. Treatment Techniques for DPM

Date dust mite control is challenging due to its rapid and high reproductive capacity. The mite's thick webbing protects it from predators and acaricides. Cultural, biological, and chemical measures are successful tactics**:**

1.3.6.1. Chemical Treatment of DPM

In date palm farms, a variety of acaricides have been used to control O. afrasiaticus. Recently, the management of O. afrasiaticus in date palm groves has relied mainly on the application of synthetic acaricides regularly. Acaricide application is not sustainable, and excessive and frequent use of a restricted number of active ingredients may result in resistance development and mite population resurgence.

1.3.6.2. Biological Treatment of DPM

Pathogens and predators are two significant mortality factors that play a big part in limiting O. afrasiaticus population development in the field. A lot of predatory mites and coccinellid beetles are now useful for date dust mite management in the field (Predatory mite, Lady bird beetle, Predatory bug). Stethorus gilvifrons, a ladybird, was released at a rate of three bugs per square meter on fruit bunches. This coccinellid predator has been observed to consume both adults and juveniles of the date mite. Natural foes of O. afrasiaticus include predatory mites Cydoneius negevi and Neoseiulus barkeri. Cydoneius negevi has the ability to get into the dense webbing produced by O. afrasiaticus, making it more appropriate for date dust mite control than other predators.

1.3.6.3. Cultural Treatment of DPM

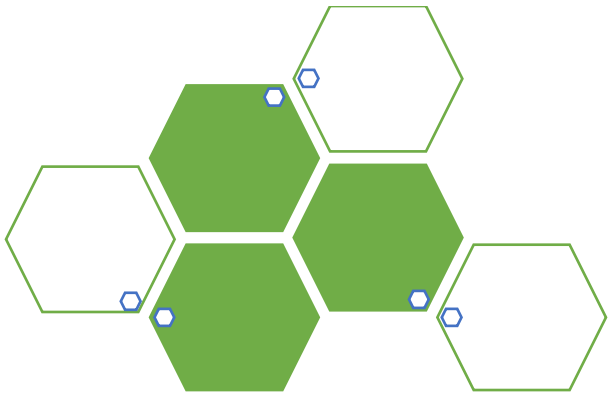
Many cultural practices have been implemented, such as the removal of grasses around date palms, which may serve as overwintering sites for females during the off-fruiting season. Removal of old fronds and fibers may be used to combat date palm dust mites and reduce their negative effect on fruit quality. Spraying infested bunches with a strong stream of water to

remove the webbings and dislodge the mites at various stages of development, especially the eggs, which typically cling to the webbings.

1.4. Conclusion

In conclusion, IoT is a vast network of interconnected devices enabling communication between people, machines, and things. It offers interconnectivity, heterogeneity, sensing capabilities, and security. Smart farming, a key application of IoT, improves agricultural practices through sensors, drones, and precision farming. However, Oligonychus afrasiaticus, a date palm mite, poses a major threat to date palms in North Africa and the Middle East. It reproduces rapidly, infests plants, and causes economic losses.

Understanding and utilizing IoT and smart farming technologies have the potential to revolutionize various industries, including agriculture, by improving efficiency, reducing costs, and enhancing sustainability. Controlling pests like Oligonychus afrasiaticus is important for maintaining crop health and minimizing economic losses in affected regions. Therefore, finding a solution using artificial intelligence and the Internet of Things to predict this disease is important for the farmer and the economy.



Chapter 2:
Deep Learning and Related Work

Chapter 2 Deep Learning and Related Work

2.1. Deep learning basic concepts

2.1.1. Deep learning definition

There isn't a general definition of deep learning. Here are some initiatives:

Definition 1:

Deep learning is a subset of machine learning approaches based on artificial neural networks and feature learning. Learning could be supervised, semi-supervised, or unsupervised **[14]**.

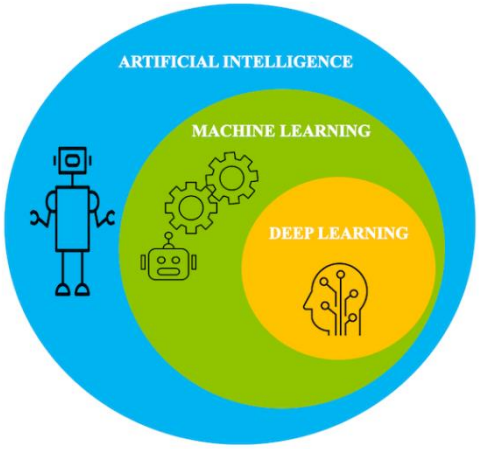


Figure 2.1: *AI, ML and DL [15].*

Definition 2:

Deep learning is a specific class of machine learning technique that: uses a cascade of multiple layers of nonlinear processing units for feature extraction. Each subsequent layer takes the preceding layer's output as input **[16]**.

Definition 3:

Deep learning is a process not only to learn the relation among two or more variables but also the knowledge that governs the relation as well as the knowledge that makes sense of the relation **[17]**.

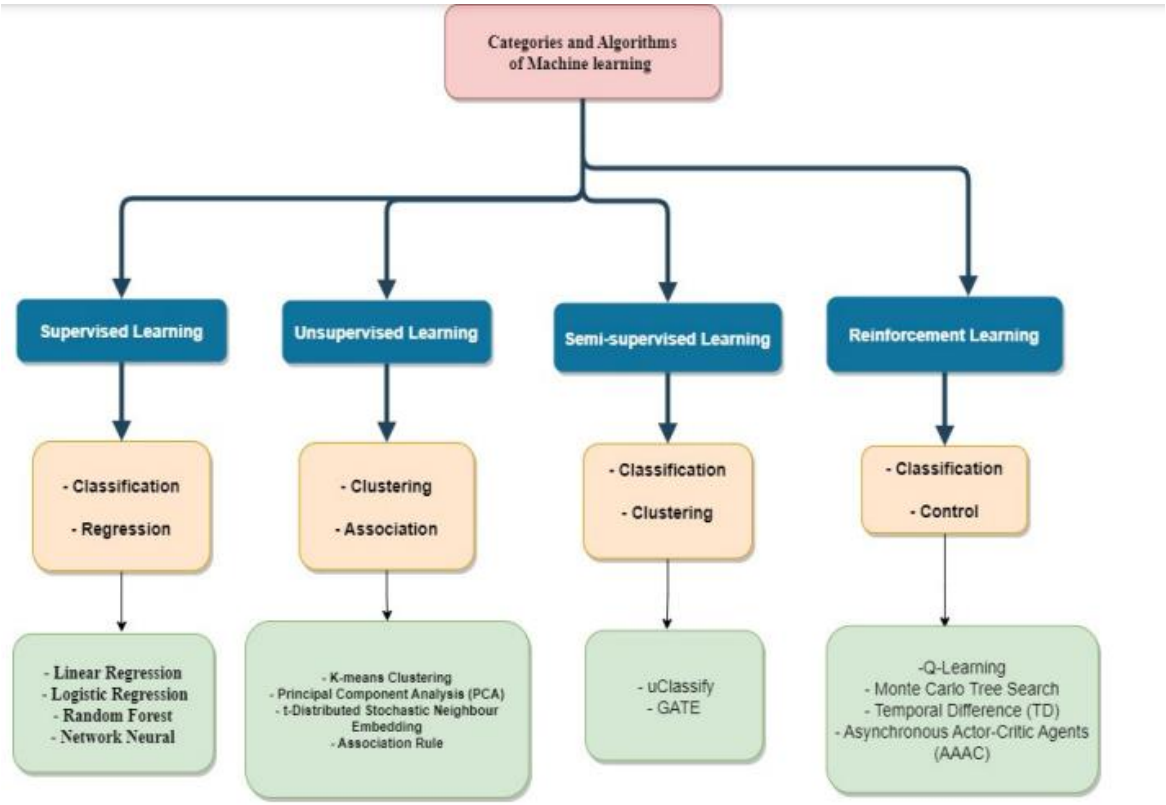


Figure 2.2: *machine learning categories and algorithms [18]*

2.1.2. Supervised learning vs Unsupervised learning

Supervised learning occurs when both input and output variables are provided, as well as an algorithm to assist the machine in mapping the transition from input to output. Two main types of supervised learning are classification and regression for example image recognition systems and spam filters. When we just have input data but no output data, we say unsupervised learning. Two main types of unsupervised learning clustering and association rules, for example, Anomaly detection to detect bot activity and Pattern recognition **[19]**.

2.1.3. Semi-supervised learning

Semi-supervised learning is a hybrid of supervised and unsupervised learning methods. It combines a small amount of manually labeled data (a supervised learning part) to define a big amount of unlabeled data (an unsupervised part) autonomously. This method uses data clustering to train a machine learning algorithm on data labeling without having to manually label all of the sample data beforehand, thereby increasing efficiency without losing quality or accuracy. This combines the advantages of supervised and unsupervised learning by guiding the algorithm to produce powerful autonomous predictions with less initial human control **[20]**.

2.1.4. Reinforcement Learning

Reinforcement Learning is a feedback-based algorithm for machine learning in which an agent learns to behave in an environment by performing actions and observing the outcomes of those actions. For every successful action, the agent receives positive feedback, and for each incorrect action, the agent receives negative feedback or a penalty.

2.1.5. Machine Learning Vs Deep Learning

Machine Learning is a set of algorithms that can analyse data, gain knowledge from it, and use what it has learned to make intelligent predictions. Machine learning may require human intervention at times, and it tends to execute limited tasks for what it has been designed to do because it does not think beyond the confines of what it has been programmed for and subsequently learned to do within those domains. Deep learning contains a complicated hierarchy of concepts, each layer defined and related to the others in some way. Deep learning technique development entails processing data across many layers in a step-by-step. Deep learning is the next evolution of machine learning. The technology empowers the machine to learn from previous decisions and make its own decisions at any level and in any form. The main advantages of learning are scalability and feature learning is fully automated feature extraction from raw data **[19]**.

2.1.6. How does deep learning work?

Deep learning networks learn by detecting intricate patterns in the data they encounter. The networks may represent data at different levels of abstraction by constructing computational models built of several processing layers.

A deep learning model known as convolutional neural network (CNN), for example, can be trained using large numbers (as in millions) of images, such as those containing apples. This type of neural network typically learns from the pixels contained in the images it acquires. It can classify groups of pixels that are representative of an apple's features, with groups of features such as color shape indicating the presence of an apple in an image.

Deep learning differs fundamentally from traditional machine learning. In this case, an expert in the field would need to spend a significant amount of time developing a traditional machine-learning system to recognize the attributes that describe an apple. With deep learning, all that is required is to feed the system a huge number of apple photos, and the system will learn the features that represent an apple on its own.

2.1.7. Artificial Neural Network (ANN)

Modern artificial neural network architecture is inspired by the natural architecture of the brain, which is a complex network of input-output wirings between neurons that cascade decision signals across a network. A deep neural network (DNN) is an artificial neural network with numerous layers between the input and output layers. The layers between the input and output layers are known as hidden layers. Neural networks must have neurons, inputs, weights, biases, and functions to function. These components interact in the same way that the human brain does. The word "deep" in deep learning means the network's utilisation of numerous layers **[21]**.

Chapter 2 Deep Learning and Related Work

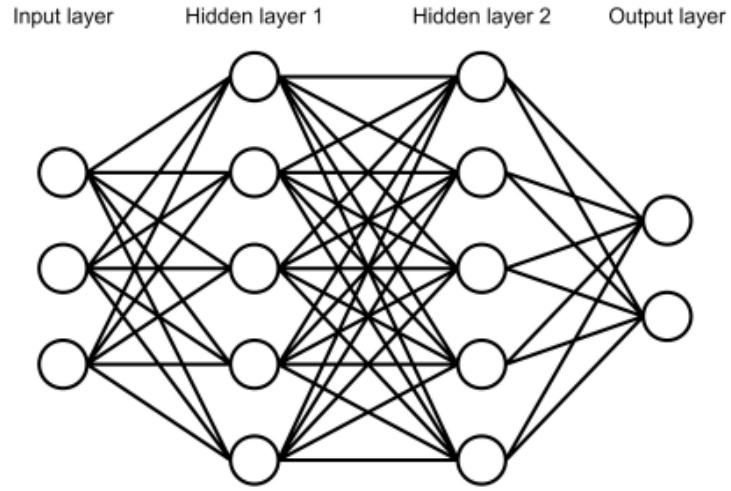


Figure 2.3: *Structure of a Multilayer Neural Network [21]*

An artificial neural network is made up of many connected nodes known as perceptrons or neurons. A perceptron (shown below) takes real-valued numerical inputs of a feature (x) and multiplies each input by the weights (w) and bias (b) associated with it to compute the net input, which is the total of all connected neurons. The net input result is handed on to the activation function, which mathematically turns the output into the -1, 1 range defined by the hyperplane at the origin. These distinct planes translate to the predicted class label of the input data points. During the learning phase, this output is utilised to determine the prediction error and to update the weights and bias unit **[21]**.

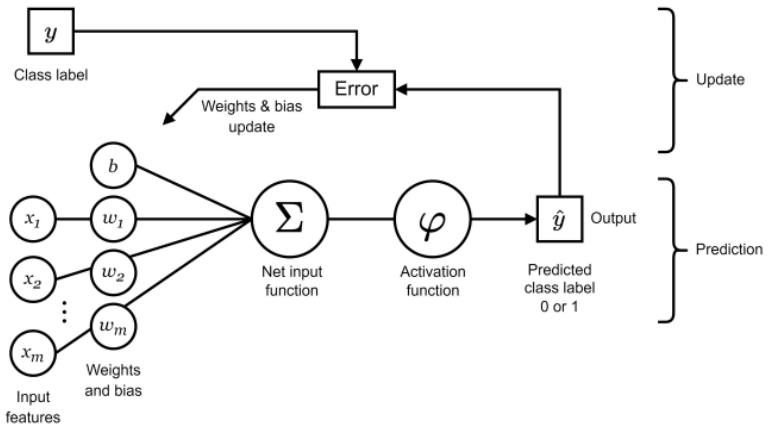


Figure 2.4: *The Perceptron [21]*

2.1.8. Convolutional Neural Network (CNN)

A Convolutional Neural Network (CNN) is a type of neural network that specialises in processing data with a grid-like architecture, such as an image. A CNN typically includes three layers: a convolutional layer, a pooling layer, and a fully connected layer. An RGB image is nothing more than a pixel value matrix with three planes, whereas a grayscale image is the same but only contains one plane.

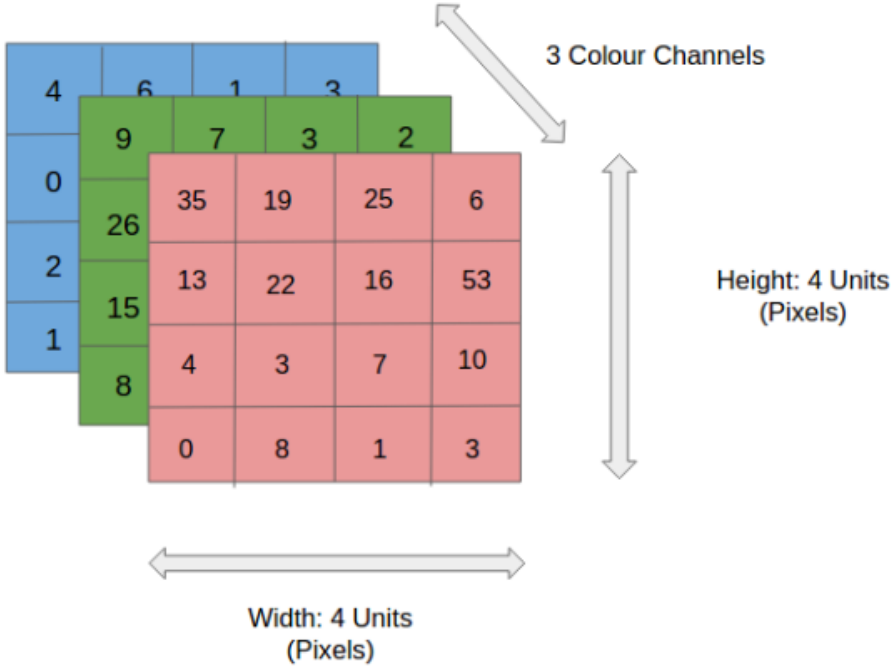


Figure 2.5: *An RGB image*

Convolutional layers create a feature map by applying a filter over an input image and recognising patterns in images. Pooling layers: These layers downsample the feature map to introduce Translation invariance, reducing the CNN model's overfitting.

Chapter 2 Deep Learning and Related Work

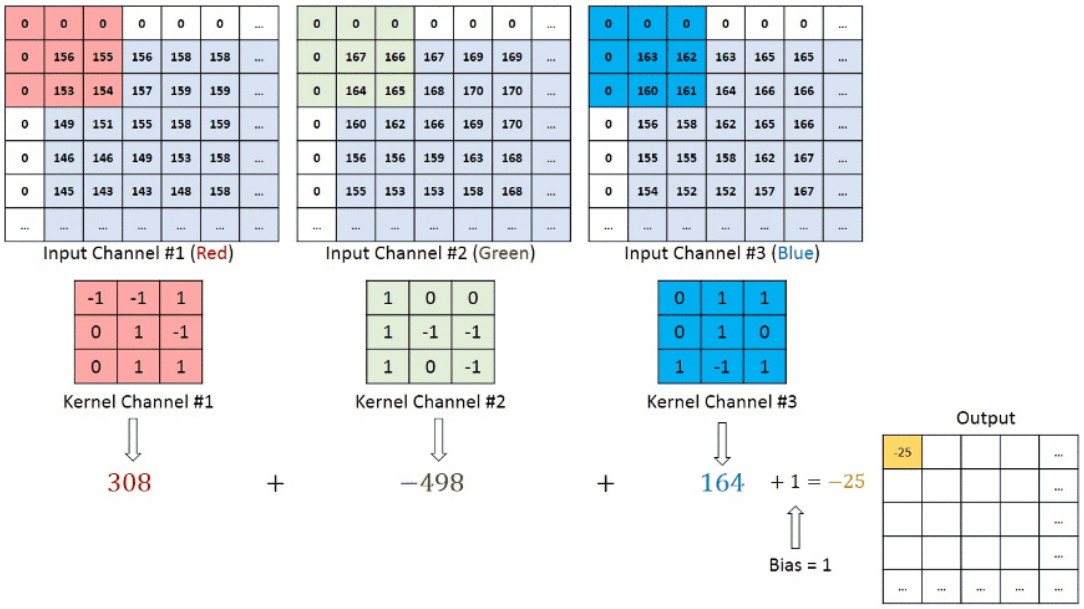


Figure 2.6: Applying a filter over in an image

2.1.9. Recurrent Neural Networks (RNN)

RNNs are a type of NN that has hidden states and allows past outputs to be used as inputs.

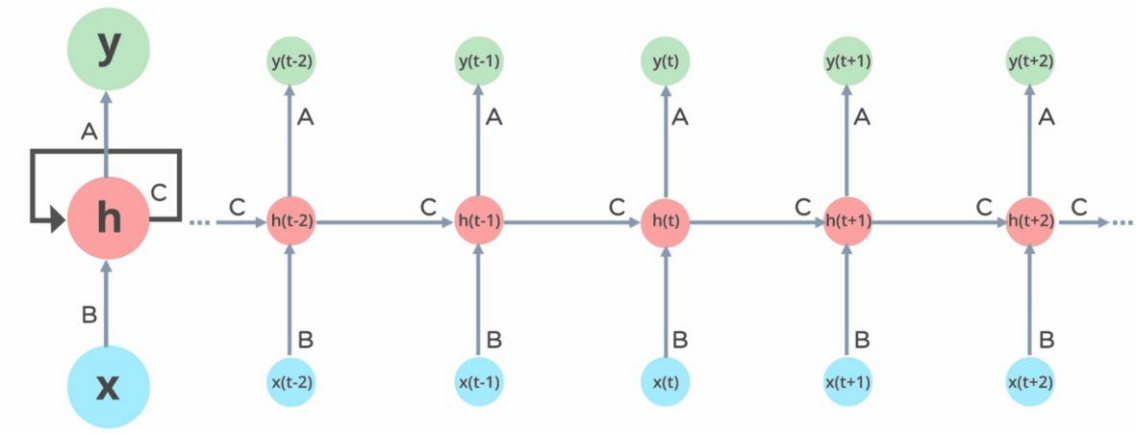


Figure 2.7: Recurrent neural networks

In recurrent neural networks, information is cycled through a loop to the middle hidden layer. The neural network's input is received and processed by the input layer x before being passed on to the middle layer. In the middle layer h, there are several hidden layers, each with its activation functions, weights, and biases. A recurrent neural network can be used if the various parameters of different hidden layers are not influenced by the preceding layer, the neural

network has no memory. The Recurrent Neural Network will standardise the various activation functions, weights, and biases, guaranteeing that each hidden layer has the same properties. Rather than creating multiple hidden layers, it will simply generate one and loop over it as many times as needed **[22]**.

When we apply a Backpropagation algorithm to a Recurrent Neural Network with time series data as its input, we call it backpropagation through time. A single input is delivered into the network at a time in a conventional RNN, and a single output is obtained. Backpropagation, on the other hand, takes both the current and earlier inputs as input. This is referred to as a timestep, and one timestep will consist of numerous time series data points entering the RNN at the same time. The neural network's output is used to calculate and gather error after it has trained on a time set and given you an output. The network is subsequently rolled back up, and weights are recalculated and changed to account for the faults **[22]**.

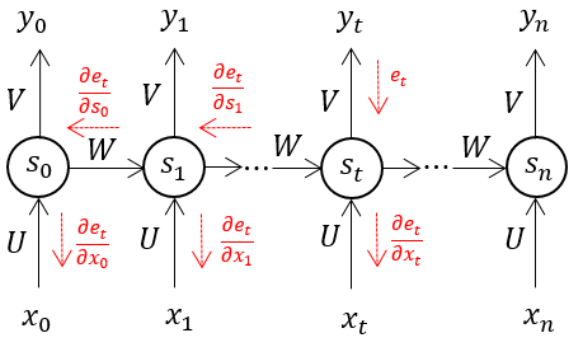


Figure 2.8: *RNN backpropagation through time*

Recurrent Neural Networks are utilised to solve a wide range of sequence data challenges. There are numerous sequence data types, but the following are the most common: Audio (Speech recognition), text (Automated Translations), video (Analysis of video action), and biological sequences are all possible.

Vanishing gradients are one of the drawbacks of RNN. They arise when the gradient values are too small, causing the model to stop learning or take far too long. Fortunately, the LSTM concept developed by Sepp Hochreiter and Juergen Schmidhuber solved the problem.

2.1.10. Applications of Deep Learning

- Smart Agriculture
- Healthcare
- Personalized Marketing
- Financial Fraud Detection
- Natural Language Processing
- Autonomous Vehicles
- Facial Recognition
- Recommendation Systems
- Space Travel
- Fake News Detection

2.1.11. Deep learning use case examples

Robotics:

Many recent advancements in robotics have been powered by AI and deep learning capabilities **[23]**. Robots, for example, can sense and respond to their surroundings thanks to artificial intelligence. This skill broadens their variety of functions, from traveling warehouse floors to sorting and managing goods that are uneven, fragile, or tangled together. Picking up a strawberry is a simple activity for humans, but it has proven to be quite challenging for robots. The capabilities of robots will be enhanced as AI advances. Collaborative robots already exist today, with humans and robots conducting separate activities that fit most closely to their skills for example Robot welding or arc welding One of the motivations for the shift to robot welding is the desire to protect workers from arc burns and dangerous gases.

Medical imaging and healthcare

Because of the availability of high-quality data and the ability of convolutional neural networks to categorise images, deep learning has been particularly useful in medical imaging. Deep learning, for example, can be as effective as, if not more effective than, a dermatologist in classifying skin malignancies **[24]**. Several vendors have previously been approved by the FDA for deep learning algorithms for diagnostic purposes, such as image analysis for oncology and

retina illnesses. Deep learning is also helping to improve healthcare quality by predicting medical events from electronic health record data.

Deep Learning and Cybersecurity:

Deep learning is also used in cybersecurity to improve threat detection and response. Cybersecurity threats have become more sophisticated, and old rule-based techniques for detection and prevention are ineffective. Deep learning algorithms can analyse massive amounts of data to spot patterns and abnormalities that could suggest a security breach or assault.

Deep learning is used in cybersecurity for intrusion detection. Intrusion detection systems (IDS) allow us to monitor network traffic and spot unusual behaviour that could indicate an attempted attack. Deep learning algorithms trained on massive network traffic datasets for recognising patterns and anomalies that may suggest an attack allow us to detect and respond to attacks faster and more effectively than traditional rule-based approaches.

Deep learning limitations:

While deep learning undoubtedly changed several fields, it is equally vital to recognise its potential mistakes and weak points for improvement. One of the most significant constraints is deep learning's inability to generalise beyond their training data. This means that, while DL may perform well on the tasks for which it was trained, it may be unable to deal with new or unexpected inputs that differ greatly from its training data. This is especially problematic in real-world applications because the input data may vary significantly **[25]**.

Another limitation is the difficulty in interpreting and explaining the decisions made by neural networks. Unlike traditional rule-based systems, which can provide clear explanations for their decisions, neural networks are often seen as "black boxes" whose decision-making processes are difficult to understand. This can make it challenging to identify and correct errors or biases in the model **[25]**.

A third limitation is the need for large amounts of labeled data to train deep learning models effectively. While this may not be an issue in some domains where large amounts of labeled data are readily available (such as image recognition), it can be a significant barrier in other domains where labeled data is scarce or expensive to obtain **[25]**.

These limitations can have significant implications for the reliability and trustworthiness of deep learning models in real-world applications. For example, if a self-driving car relies on a deep learning model that cannot generalize beyond its training data, it may not be able to handle unexpected situations on the road. Similarly, if a medical diagnosis system relies on a black-box deep learning model, doctors may not be able to understand or explain why certain diagnoses were made **[25]**.

While deep learning has shown significant promise in many sectors, it is equally necessary to acknowledge its limitations and strive towards addressing them through additional study and development.

The next section illustrates the most important studies that applied ML/DL to deal with Palm tree diseases. Overall, these articles highlight different aspects of palm tree diseases, insect infestation, and machine-learning approaches for identification and prediction. Each study contributes to understanding and addressing challenges related to palm tree health and pest management.

2.2. Related work

2.2.1. Classification of Palm Trees Diseases using Convolution Neural Network

The authors of this article **[26]** proposed a CNN model for identifying and categorizing four common ailments affecting palms today: bacterial leaf blight, brown spots, leaf smut, white scale, and healthy leaves. In general, common palm diseases are Bacterial leaf blight, Brown spots, Leaf smut, and white scale. Because the nature and symptoms of these diseases differ in their form, area of appearance, and distribution on palm trees, they need new techniques that contribute greatly to discovering them before they cause problems to palm trees **[26]**.

Chapter 2 Deep Learning and Related Work

Figure 2.9: *four common diseases and healthy leaves [26].*

Materials and Methods:

First, in Data Preprocessing, the authors resized all images to 60 x 60 resolution to be consistent. They use image whitening based on Zero Component Analysis (ZCA) to reduce data correlation and make essential features such as edges and curvatures more visible, making them easier for CNN to detect. Next, all pixel values in any image are normalized to be between 0 and 1. The proposed model consists of four convolutional layers, a fully connected layer, and softmax as the output layer. This model achieved 99.10% accuracy (MobileNet achieved 99.56%, and VGG16 achieved 99.35% accuracy) **[26].**

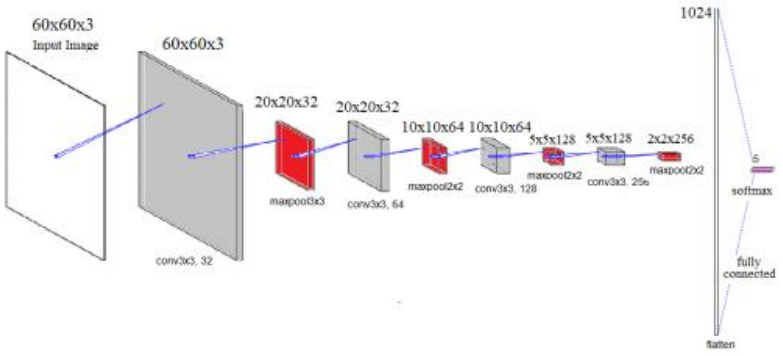


Figure 2.10: *The Architecture of Proposed CNN [26]*

Chapter 2 Deep Learning and Related Work

This study showed an effective CNN model for identifying and classifying common palm trees diseases such as bacterial leaf blight, brown spots, leaf smut, and white scale. This solution provides high accuracy and is close to VGG-16 and MobileNet, and has simplicity in structure. It also has a low computational training cost (20.048s/epoch) compared to others (343.85s/epoch and 2559s/epoch for MobileNet and VGG-16, respectively). But this model doesn't accurately identify the affected parts of palm trees **[26]**.

Table 2.1: The performance of each model based on accuracy [26].

Diseases	Proposed model	MobileNet model	Vgg16 model
Brown Spots	99.55%	100.00%	100.00%
white Scale	99.35%	99.75%	99.35%
Bacterial Leaf Blight	98.79%	98.98%	98.99%
Leaf Smut	99.20%	98.88%	99.90%
Healthy	98.90%	99.90%	99.90%
Average	**99.10%**	**99.56%**	**99.35%**

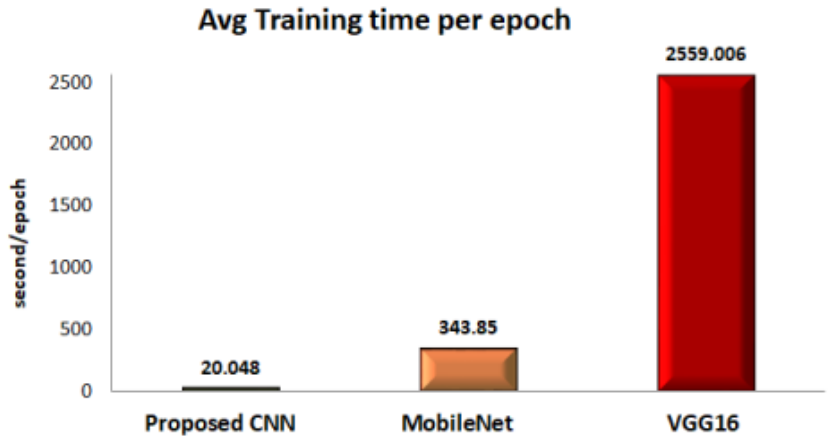


Figure 2.11: Avg Training Time per Epoch for each Model [26].

2.2.2. Relationship of Date Palm Tree Density to Dubas Bug Ommatissus lybicus Infestation in Omani Orchards

The objective of this study **[27]** is to determine the number of dates palm trees in traditional agricultural locations to find the relationship between date palm tree density and O. lybicus infestation. The Dubas insect Ommatissus lybicus is the major pest damaging date palm harvests in Omasn **[27].**

Materials and Methods:

The study area of this research was in the north of Oman, Ad'Dakhiliyah Governorate, in Wilayat Samail (23◦140–23◦050 N, 57◦520–57◦510 E) **[27].**

First, they use Local maxima methods that rely on detecting the brightest pixel inside a specific window to identify the trees in the image, the pixel with the highest reflectance compared to the other pixels in the same window is evaluated as a probable tree location. To determine the best windows, three window scales were tested (Figure 2. 13): 3 m (6× 6 pixels), 5 m (10 × 10 pixels), and 7 m (14 × 14 pixels). Normalized Difference Vegetation Index (NDVI) was calculated from the picture to separate non-vegetation. Following that Maximum Likelihood classification algorithm was employed for separating the date palm trees from other kinds of plants**[27]**.

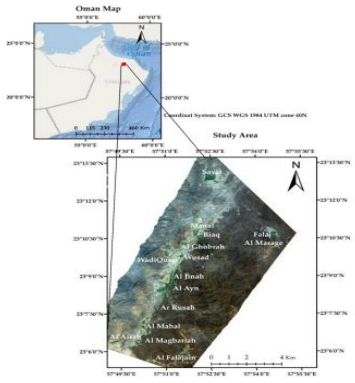


Figure 2.12: *Study area location in the north of Oman [27].*

Chapter 2 Deep Learning and Related Work

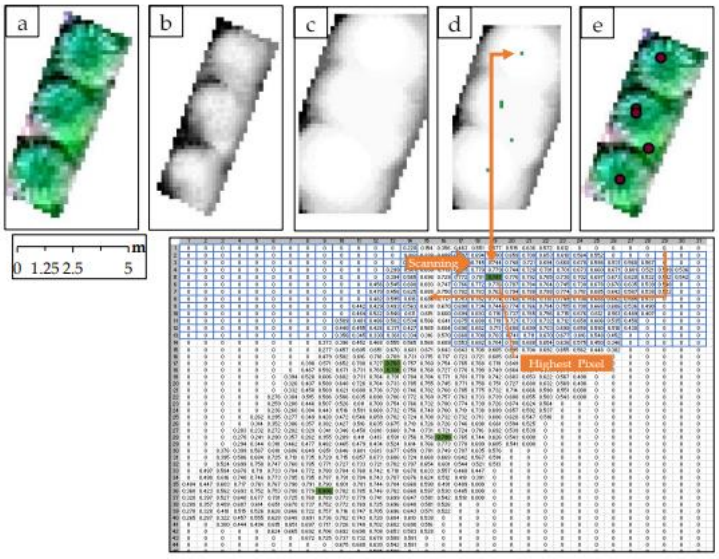


Figure 2.13: *Local Maxima illustration process figure of windows 7 [27]*

Next, the infestation data was prepared in an Excel spreadsheet, including the tree coordinate location and infestation levels of each tree, and saved as a comma delimited (CSV) file and imported into ArcMap as a point layer. The ordinary least square (OLS) regression was used to test the global correlation and Geographic Weight Regression (GWR) was used to find the local spatial relationship **[27].**

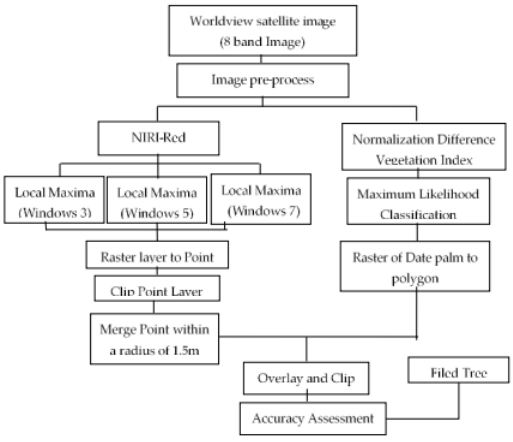


Figure 2.14: *The image processing steps [27].*

The tree counting results from different windows by local maxima, we can see in the table 7m radius window was the most suitable window (error of 11.8%) overall accuracy 88.2%. The problem of local maxima is error increased with very dense and unsystematic planting patterns.

The current study indicates that density is one factor that contributes to the severity of the O. lybicus infestation. but this is not enough to study the O. lybicus infestation. They need to add other factors.

Table 2.2: Counting Date Palm Trees by Local Maxima Accuracy [27].

Village	Field Count	Local Maxima Prediction			Estimation Error		
		Window 3	Window 5	Window 7	Window 3	Window 5	Window 7
Al'Afiah	282	529	371	244	87.6%	31.6%	−13.5%
Al Ayn	298	484	380	323	62.4%	27.5%	8.4%
Al Falajain	302	400	400	268	32.5%	32.5%	−11.3%
Al Ghobrah	179	984	553	180	449.7%	208.9%	0.6%
Al Jinah	271	457	347	240	68.6%	28.0%	−11.4%
Al Mahal	550	852	667	493	54.9%	21.3%	−10.4%
Al Magbariah	382	634	420	295	66.0%	9.9%	−22.8%
Falaj Al Maraga	338	578	397	275	71.0%	17.5%	−18.6%
Biaq	376	786	567	365	109.0%	50.8%	−2.9%
Manal	249	342	248	178	37.3%	−0.4%	−28.5%
Sayja	354	591	370	273	66.9%	4.5%	−22.9%
WadiQurai	582	760	591	458	30.6%	1.5%	−21.3%
Wusad	474	842	577	450	77.6%	21.7%	−5.1%
Total	**4637**	**8327**	**5957**	**4091%**	**79.6%**	**28.5%**	**−11.8%**

2.2.3. Development and Validation of Innovative Machine Learning Models for Predicting Date Palm Mite Infestation on Fruits

Authors **[28]** developed an accurate decision-making approach for monitoring and predicting the infestation of DPM.

Materials and Methods:

This study was performed in an arid environment at the experimental farm of Date Palm Research Center of Excellence (latitude: 25◦16004.300 N, longitude: 49◦42030.100 E) at the Agricultural Training and Research Station, King Faisal University (KFU), Saudi Arabia. The

experiment was performed during two successive seasons (2021 and 2022). The study began on 1 April (one month after fruit set) and lasted until the end of August in each experimental season. The meteorological variables data and fruit properties at various development stages, i.e., hababook, kimri, khalal, rutab, and tamr, of two fully grown date palm cultivars (Khalas and Barhee), were collected from the study area for training and testing the machine learning regression models for predicting DPM infestation on date palm fruits. Furthermore, the main meteorological variables data, i.e., the maximum temperature, minimum temperature, average temperature, maximum relative humidity, minimum relative humidity, average relative humidity, average wind speed, max solar radiation, and average daily solar radiation, were collected by an IoT-based weather station installed at the study area **[28]**.

In this study, two regression algorithms, i.e., linear regression (LR) and decision forest regression (DFR), were developed, evaluated, and validated using Microsoft Azure Machine Learning (ML) Studio to determine the best model for predicting the DPM on date palm fruits based on three input variables: (1) The meteorological variables of the study area, i.e., the maximum temperature (TMax), minimum temperature (TMin), average temperature (TAvg), maximum relative humidity (RHMax), minimum relative humidity (RHMin), average relative humidity (RHAvg), average wind speed (WSAvg), max solar radiation (SRMax), average daily solar radiation (DSRAvg); (2) The physicochemical properties data of the date fruits during the development stages, i.e., fruit weight (FW), fruit firmness (FF), fruit moisture content (FMC), total soluble solids (TSS), total sugar (TS), and tannin content (TC); and (3) The combination data of meteorological variables and physicochemical properties. The following is a description of the regression algorithms used within Azure ML in this study **[28]**.

They use Microsoft Azure Machine Learning to Building Predictive Models (Data Acquisition, Data Analysis, Models Training, and Models Performance) **[28]**.

After they created and evaluated a predictive model suitable for predicting DPM infestations, we deployed it using Azure ML to make it easier to use as a predictive Azure ML web service on the Azure cloud platform to make it easier to use as a web service. The predictive experiment was automatically obtained after selecting the train model module for the best-

developed model. Figure shows the predictive experiment of the web service that was created automatically for the prediction in the Azure ML studio platform web service **[28]**.

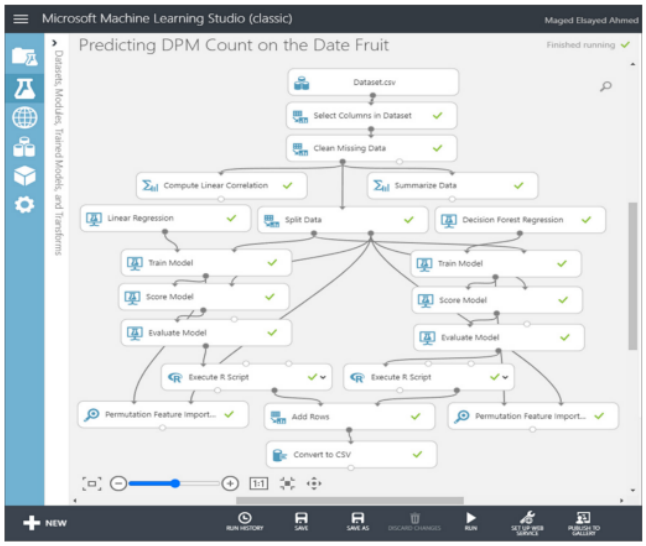


Figure 2.15: *A screenshot for an experiment of the data analysis and building the predictive models for predicting DPM infestation on the date fruit using Microsoft Azure Machine Learning [28].*

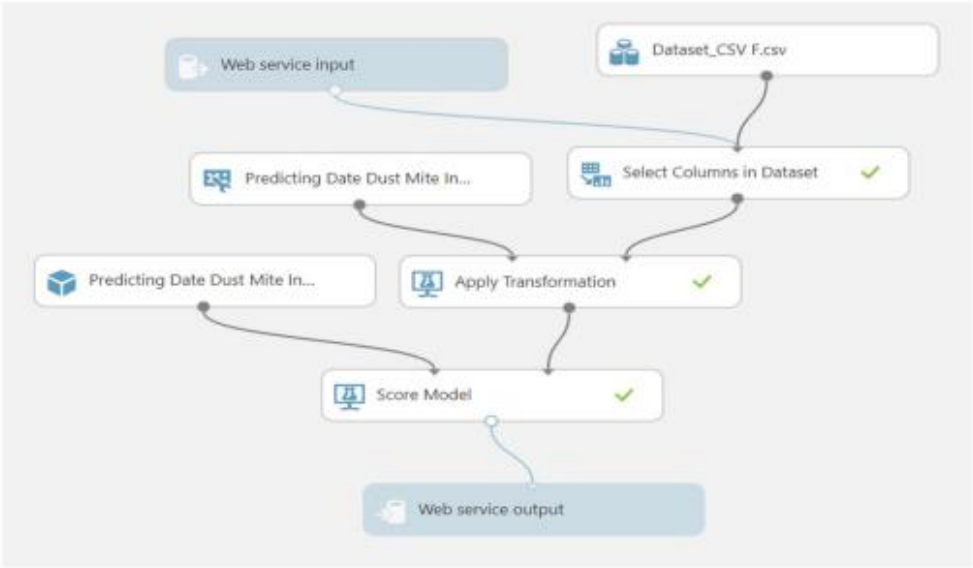


Figure 2.16: *A screenshot for the experiment of the web service that was created for the prediction model in Azure Machine Learning [28].*

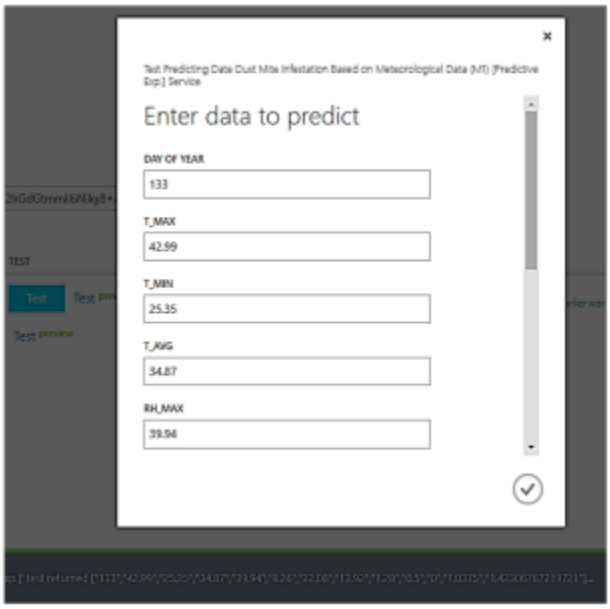


Figure 2.17: *A web service tab in the Azure ML Studio platform with an input field for test predicting date palm mite infestation based on the input variables [28].*

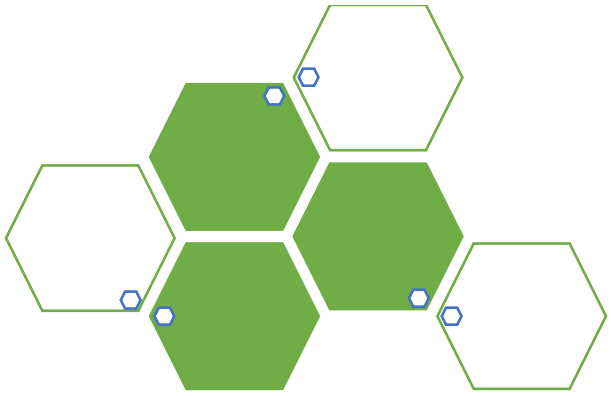
In this research activity, the authors said that the decision forest regression model has more Accuracy than ml, but they Did not mention the Accuracy of both models. In my opinion in that research, they predict the count of the date palm mite, but the count of the date palm mite is not enough to manage the infestation. We need DPM early prediction.

2.3. Conclusion

In conclusion, the first article focused on the development of a Convolutional Neural Network (CNN) model for identifying and categorizing common diseases affecting palm trees. The proposed model showed high accuracy and simplicity in structure, although it didn't accurately identify the affected parts of the trees. The second article investigated the relationship between the density of date palm trees and the infestation of Dubas bug Ommatissus lybicus in Omani orchards. The study found that tree density contributes to the severity of infestation but emphasized the need to consider other factors as well. The third article aimed to classify the infestation of Date Palm Mites (DPM) on fruits using machine learning models. The study utilized meteorological and fruit properties data to develop regression models, but the accuracy

comparison between the models was not mentioned. The researchers highlighted the importance of early prediction of DPM infestation rather than just counting the mites.

Our work is to predict the infestation of Date Palm Mite (DPM) using Deep learning, in the next chapter, we start designing the system.



Chapter 3:

Design of the Suggested System

Chapter 3 Design of the Suggested System

3.1. Introduction

In this chapter, we illustrate the design of our proposed system, showing the different design diagrams, which describe the operation of our system. We present an overview of the general architecture and also the methodology of our system, then we detail each part. Finally, we conclude the chapter.

3.2. Context and objective

Date palm diseases, especially insect-caused diseases (DPM), and how quickly they spread in crops are a real concern that can lead directly to stunted growth and catastrophic effects on yields.

Moreover, traditional methods mainly rely on specialists and manuals, but the majority of them are expensive and was late. They also use a lot of chemicals to make up for the lost time.

In this study, we suggest a new system, which is based on the principles of combining IoT and deep learning to intelligently monitor farms. This system is able to perform an expert task automatically to solve this problem and predict the disease before it appears.

The proposed idea is the prediction period of DPM and risk level to Help the farmer intervene in a timely manner.

3.3. Overall system architecture

Our architecture is composed of two entities:

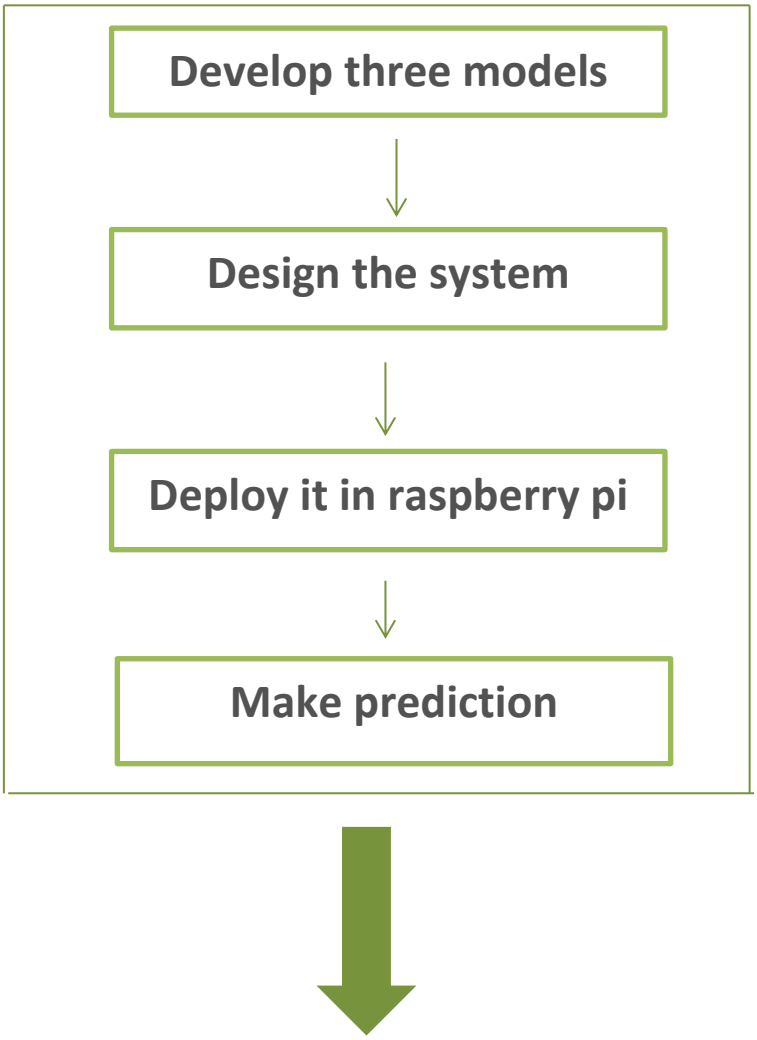
In the first entity (deep learning): we will prepare data, build our models, train and test them (the learning phase) and save them.

The second entity (IOT): this section is reserved for the prediction in two elements, part put in a palm tree composed of sensors and a communication module that sends the gathered data to Raspberry Pi 4. This part can use one or many depending on the size of the farm. Part two is Raspberry Pi 4 and screen to show results. Next, we will put models in Raspberry Pi 4. So, that it

automatically makes a prediction. The figure below represents the overall architecture of our system.

3.4. Overall system operation

The working principle of our system is to use Deep Learning and IoT to predict the problem of DPM. As shown in the figure below, the user will launch the program that is in the Raspberry Pi, next it collects data for sensors and makes predictions.



Chapter 3 Design of the Suggested System

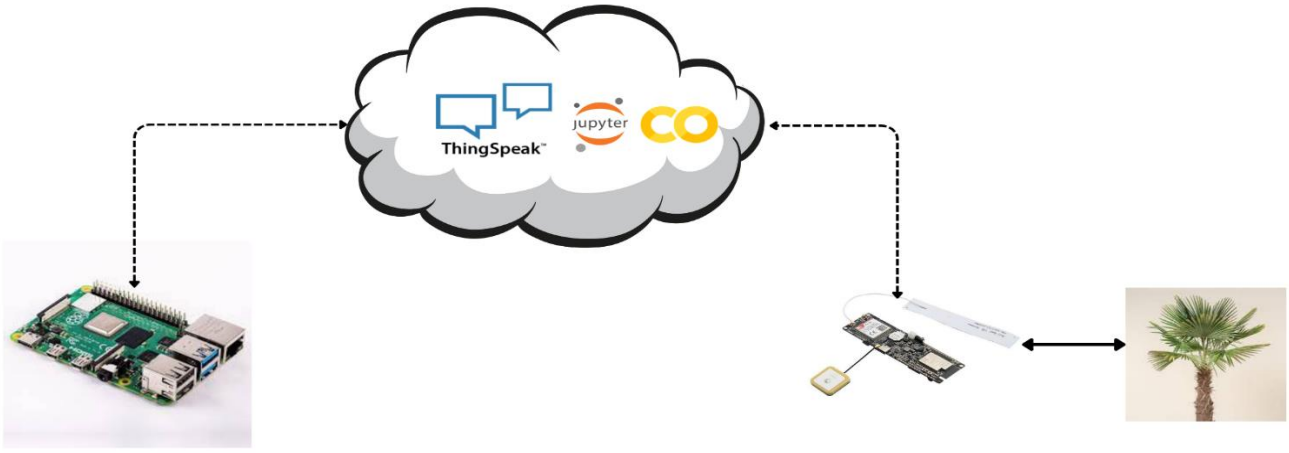


Figure 3.1: *Global architecture of our system.*

A farmer utilizes a Raspberry Pi and installs a Microcontroller equipped with various sensors onto a tree. The Microcontroller collects data from the sensors and sends it to the cloud. Subsequently, the Raspberry Pi receives the data from the cloud and utilizes it to make predictions. After a certain period (1 year), we will update the prediction system in Colab and forward it to Raspberry.

3.5. Methodology

In our work, we expose the methodologies to realize our system. First, we build early prediction models for DPM. Then, we present the deployment of models in the Raspberry Pi.

To better understand our approach, we explain each part separately.

Chapter 3 Design of the Suggested System

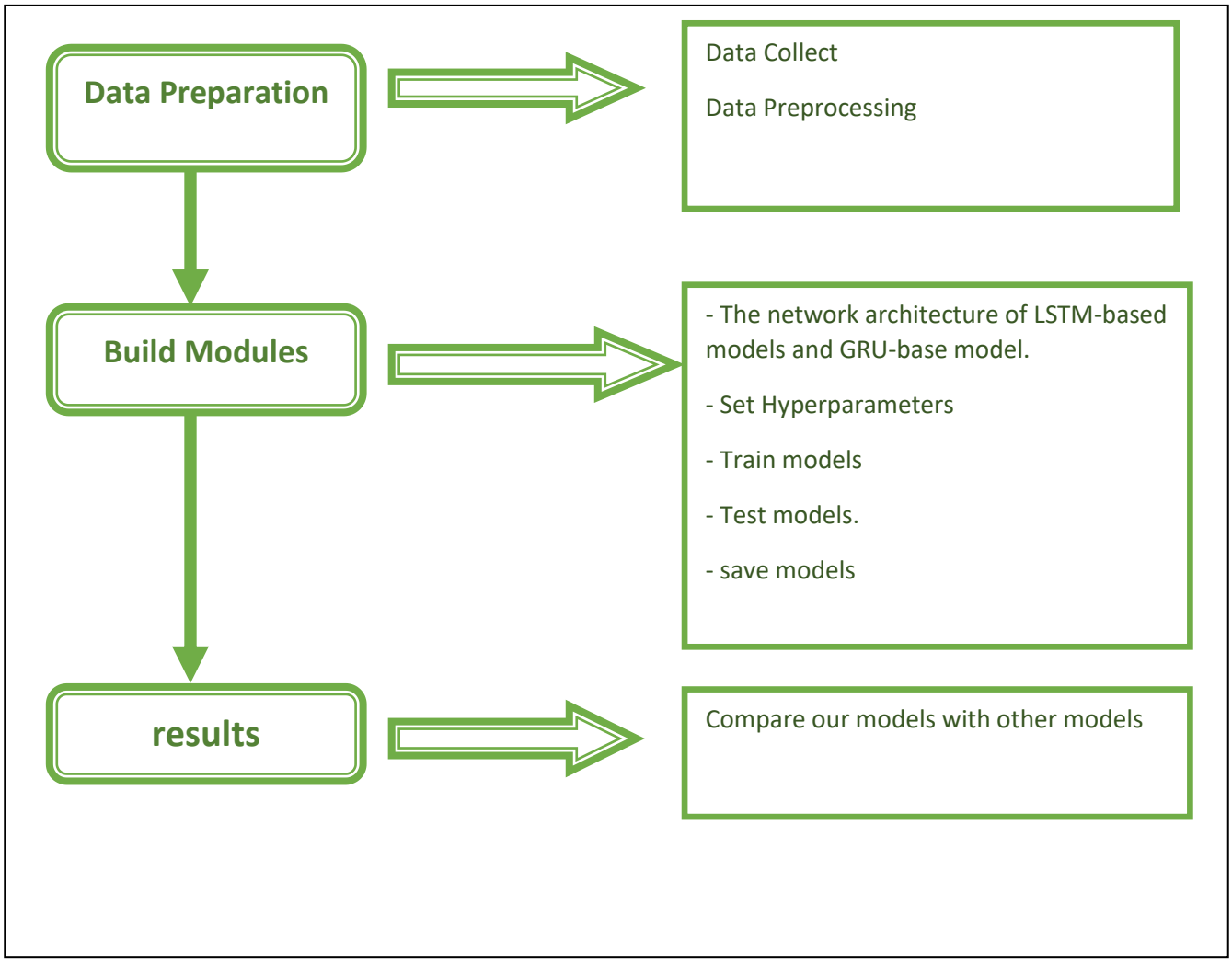


Figure 3.2: *steps of develop the models.*

3.5.1. Data Preparation

Why should prepare data? Why not just take it as it comes? The answer is that preparing data also prepares the learning so that when using prepared data, the learning produces better models, faster.

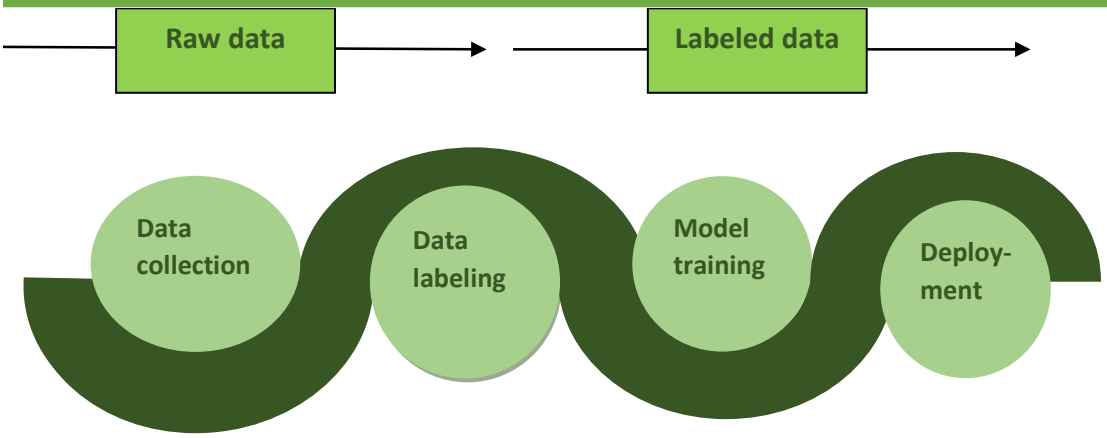


Figure 3.3: raw data to labeled data.

3.5.2 Data Collect

We got data from the Scientific and Technical Research Center on Arid Regions in Biskra **[29]**. They gather it by temperature and humidity logger (USB KEY FORMAT FI84ED in the figure below). In the period (two years 2021 – 2022), from Hadjeb, Leghrous and Mlili cities.

Figure 3.4: temperature and humidity logger.

3.5.3. Data labelling

Data labeling is the method through which data models are taught by tagging data samples. This stage of supervised learning enables the model to serve as the foundation for future learning.

3.5.4. Data aggregation

This is usually a single attribute grouping. For example, if one has a data collection with the attribute time organised in days over a specific time series, the data (temperature, humidity, wind speed, wind direction, daily solar radiation, fruit weight, fruit moisture content, total sugar, pressure, soil moisture…) can be aggregated into monthly groups to make dealing with the time attribute easier.

3.5.5. Data Preprocessing

Data preprocessing is a critical step in data analysis that involves cleaning, transforming, and preparing raw data into a format that can use. The data preprocessing goal ensures that the data (temperature, humidity, wind speed, wind direction, daily solar radiation, fruit weight, fruit moisture content, total sugar, pressure, soil moisture…) is accurate, complete, consistent, and can easily use in deep learning. Some common techniques used in data preprocessing: Data Cleaning (missing and duplicate values, Normalization) **[30]**.

3.6. Network architectures

We have made 3 network architectures based on the previous 30 days, another on the 60, and another on the 90 days. The reason for this is 3 architectures on the life cycle of this insect (Get help from experts in this field), as it can be predicted based on the previous 90 days. However, this data may not be available for some farms who using late this device. Therefore, we added two models based on the previous 60 and 30 days.

We use LSTM in the model based on their previous 60 and 30 days because LSTM can store information for long periods. But for the model based on previous 90 days, the data is more complex this is why we use GRU because GRU networks are computationally more efficient than LSTM networks.

The first model of 30-day time frame based on LSTM:

Table 3.1: layers of our model.

Layer type	Output Shape	Activation
Normalization	**14*30**	
LSTM	**20*30**	**Relu**
Dropout	**20*30**	**-**
LSTM	**20*30**	**Relu**
Dropout	**20*30**	**-**
LSTM	**20*30**	**Relu**
Dropout	**20*30**	**-**
LSTM	**20*30**	**Relu**
Dropout	**20*30**	**-**
LSTM	**20**	**Relu**
Dropout	**20**	**-**
Dense	**4**	**softmax**

The first model of 60-day time frame based on LSTM:

Table 3.2: layers of our model.

Layer type	Output Shape	Activation
Normalization	**14*60**	
LSTM	**40*60**	**Relu**
Dropout	**40*60**	**-**
LSTM	**40*60**	**Tanh**
Dropout	**40*60**	**-**
LSTM	**40*60**	**Relu**
Dropout	**40*60**	**-**
LSTM	**40*60**	**Tanh**
Dropout	**40*60**	**-**

LSTM	**40*60**	**Relu**
Dropout	**40*60**	**-**
LSTM	**40*60**	**Tanh**
Dropout	**40*60**	**-**
LSTM	**40*60**	**Relu**
Dropout	**40*60**	**-**
LSTM	**40*60**	**Tanh**
Dropout	**40*60**	**-**
LSTM	**40**	**Relu**
Dropout	**40**	**-**
Dense	**4**	**softmax**

The first model of 90-day time frame based on GRU:

Table 3.3: *layers of our model.*

Layer type	**Output Shape**	**Activation**
Normalization	**14*90**	**-**
GRU	**128*90**	**Relu**
Dropout	**128*90**	**-**
GRU	**128*90**	**Relu**
Dropout	**128*90**	**-**
GRU	**128*90**	**Relu**
Dropout	**128*90**	**-**
GRU	**128**	**Relu**
Dense	**4**	**Softmax**

3.6.1. What is LSTM?

LSTM (long short-term memory) functions similarly to an RNN cell:

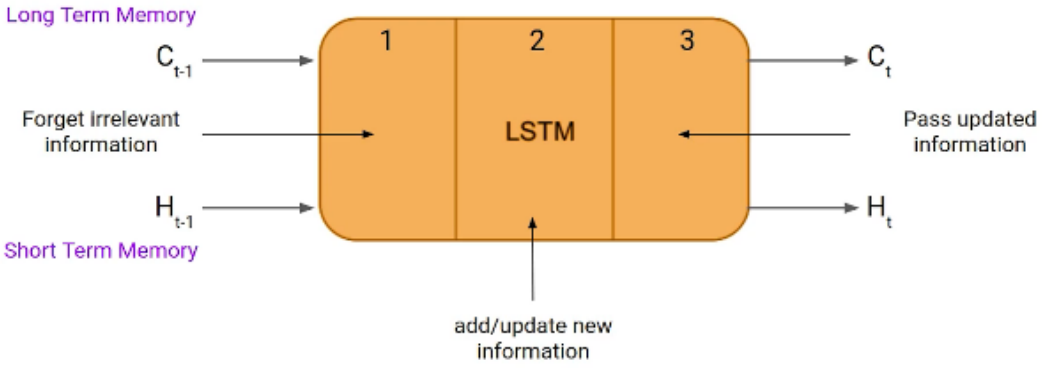


Figure 3.5: *long short-term memory*

The initial step in an LSTM neural network cell is to select whether to keep or discard the information from the previous time step. Here is the forget gate equation:

Forget Gate:

- $f_t = \sigma (x_t * U_f + H_{t-1} * W_f)$

Xt: the current timestamp's input (t).

Uf: the related weight with the input

Ht-1: The preceding timestamp's hidden state

Wf: This is the weight matrix for the hidden state.

$C_{t-1} * f_t = 0$ ...if $f_t = 0$ (forget everything)

$C_{t-1} * f_t = C_{t-1}$ ...if $f_t = 1$ (forget nothing)

Input Gate:

- $i_t = \sigma (x_t * U_i + H_{t-1} * W_i)$

Ui: input weight matrix

Wi: Input weight matrix of the hidden state

The input gate calculates the significance of the new information carried by the input. Here is the input gate equation.

- $N_t = \tanh(x_t * U_c + H_{t-1} * W_c)$ (new information)

$C_t = f_t * C_{t-1} + i_t * N_t$ (updating cell state)

Now, the new information that needed to be passed to the cell state is a function of a hidden state at the previous timestamp t-1 and input x at timestamp t. The activation function here is tanh. Due to the tanh function, the value of new information will be between -1 and 1. If the value of Nt is negative, the information is subtracted from the cell state, and if the value is positive, the information is added to the cell state at the current timestamp. However, the Nt will not be directly added to the cell state. The modified equation is as follows:

Output Gate:

- $o_t = \sigma (x_t * U_o + H_{t-1} * W_o)$

$H_t = o_t * \tanh(C_t)$

Output = Softmax(H_t)

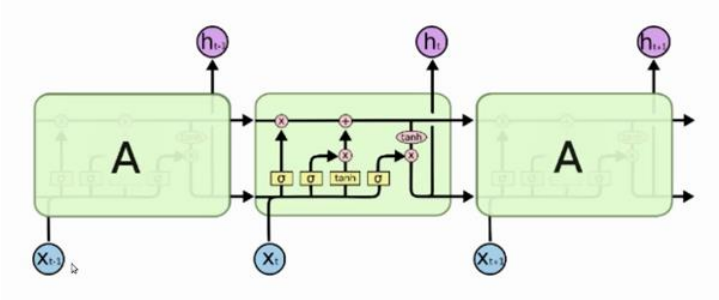


Figure 3.6: LSTM neural network cells [31]

3.6.2. What is GRU?

GRU (Gated Recurrent Unit) functions similarly to an LSTM cell:

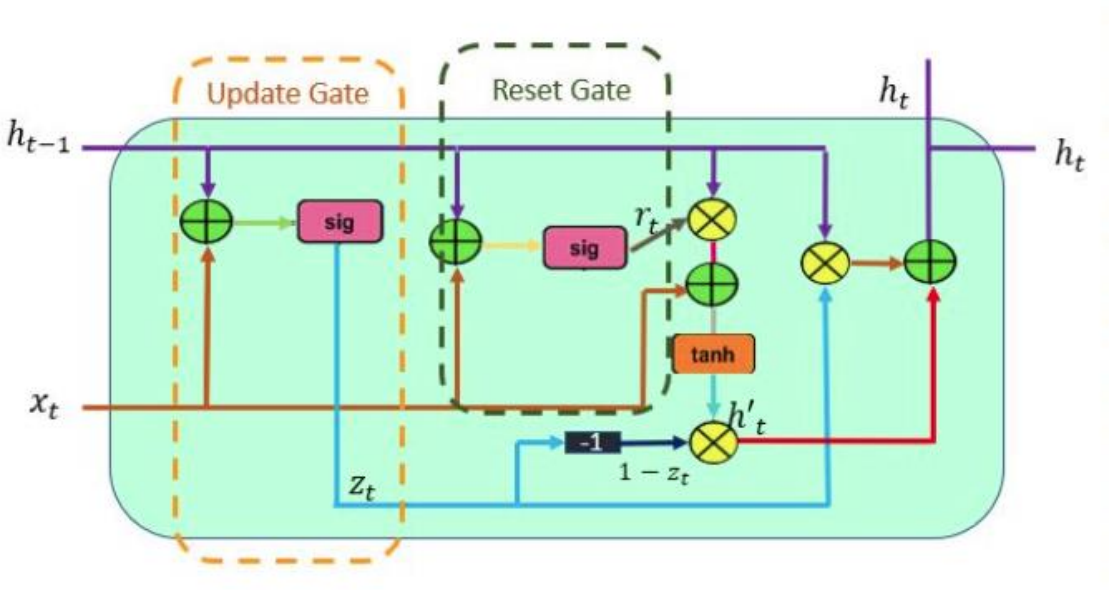


Figure 3.7: Gated Recurrent Unit [32].

Gate for updating the update gate is in charge of determining how much prior information must be passed along to the next state. This is quite powerful since the model may choose to duplicate all previous information and remove the possibility of disappearing gradients **[33]**.

The reset gate works in the model to determine how much prior information may be ignored; in other words, it determines whether the previous cell state is significant or not **[33]**.

First, the reset gate is activated, and pertinent information from the previous time step is stored in new memory content. The weights of the input vector and hidden state are then multiplied. It then performs element-wise multiplication on the reset gate and previously hidden state multiple. Following the completion of the preceding steps **[33]**.

3.7. Class diagram

This class diagram of our models is as follows:

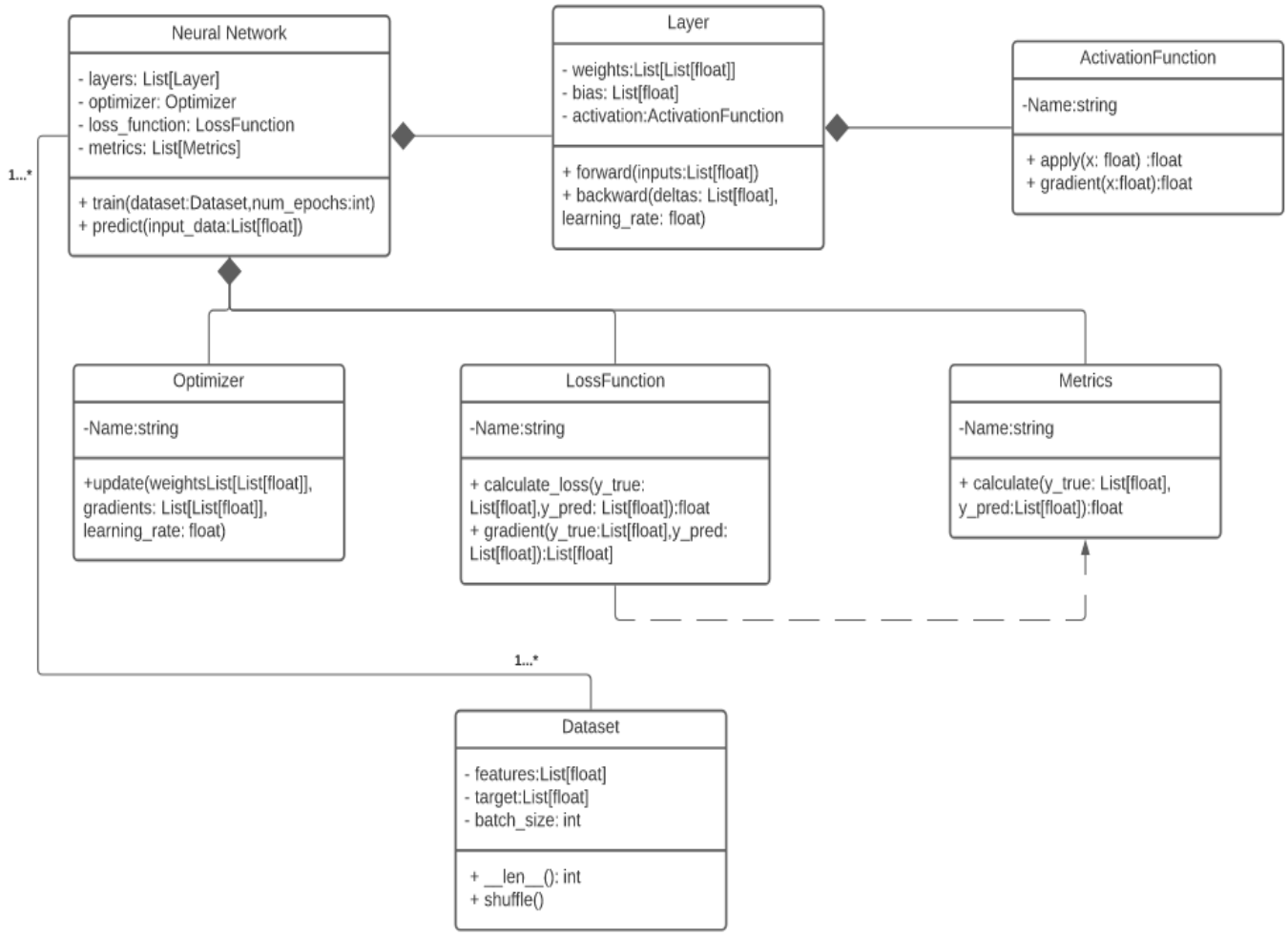


Figure 3.8: *class diagram of our models*

Train and validation of all modules will be 80% of the dataset for the train and 20% for validation:

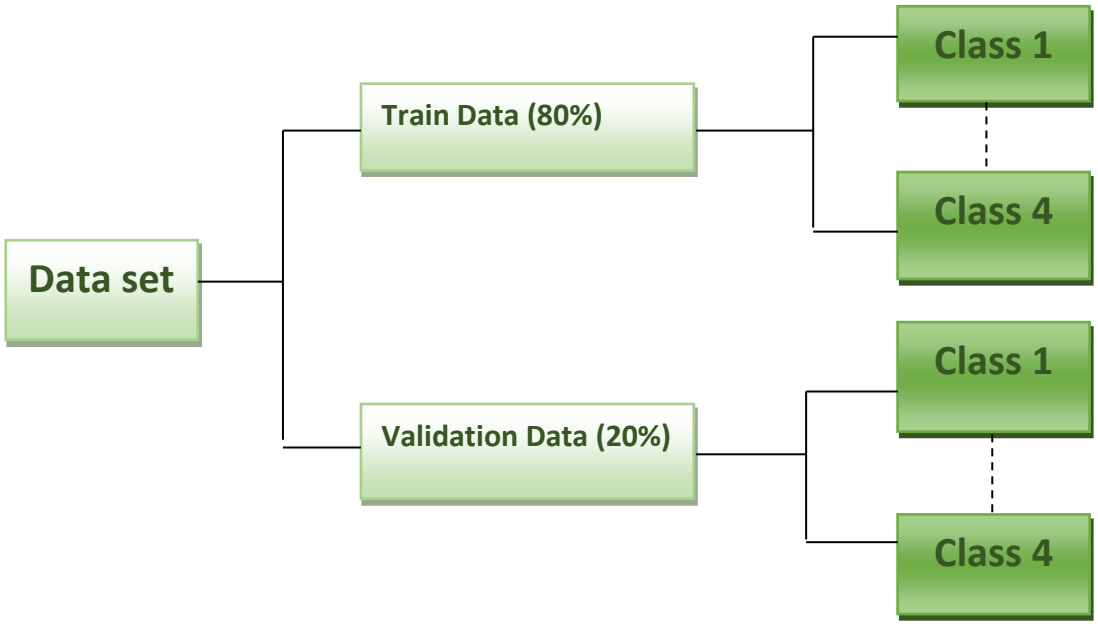


Figure 3.9: Data splitting

3.8. Design of the second section

First, we get data from the sensor to Raspberry Pi next, we make a prediction by using 3 modules. The figure below represents the flowchart that summarizes the work stages for this part.

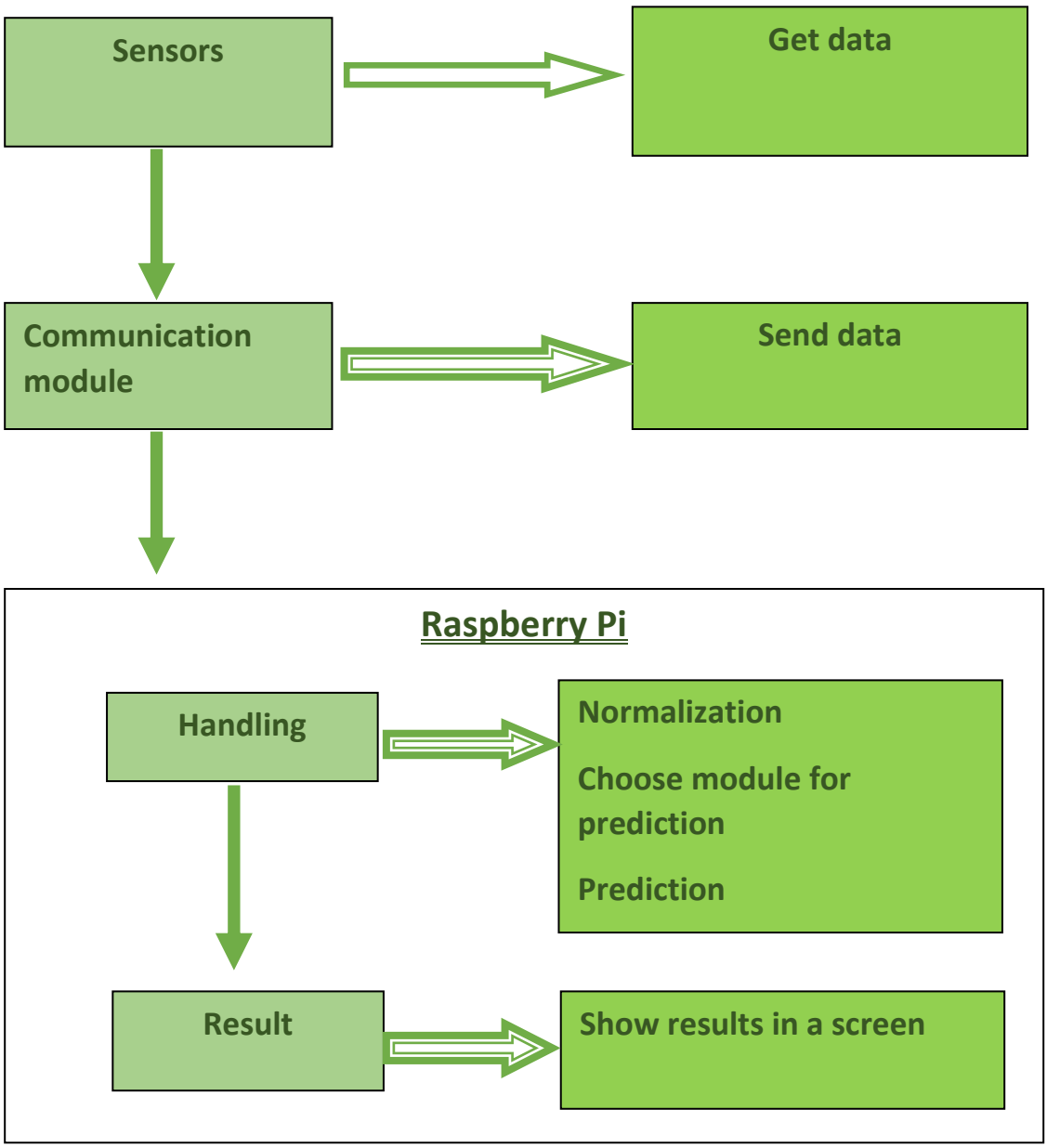


Figure 3.10: *steps of develop the device.*

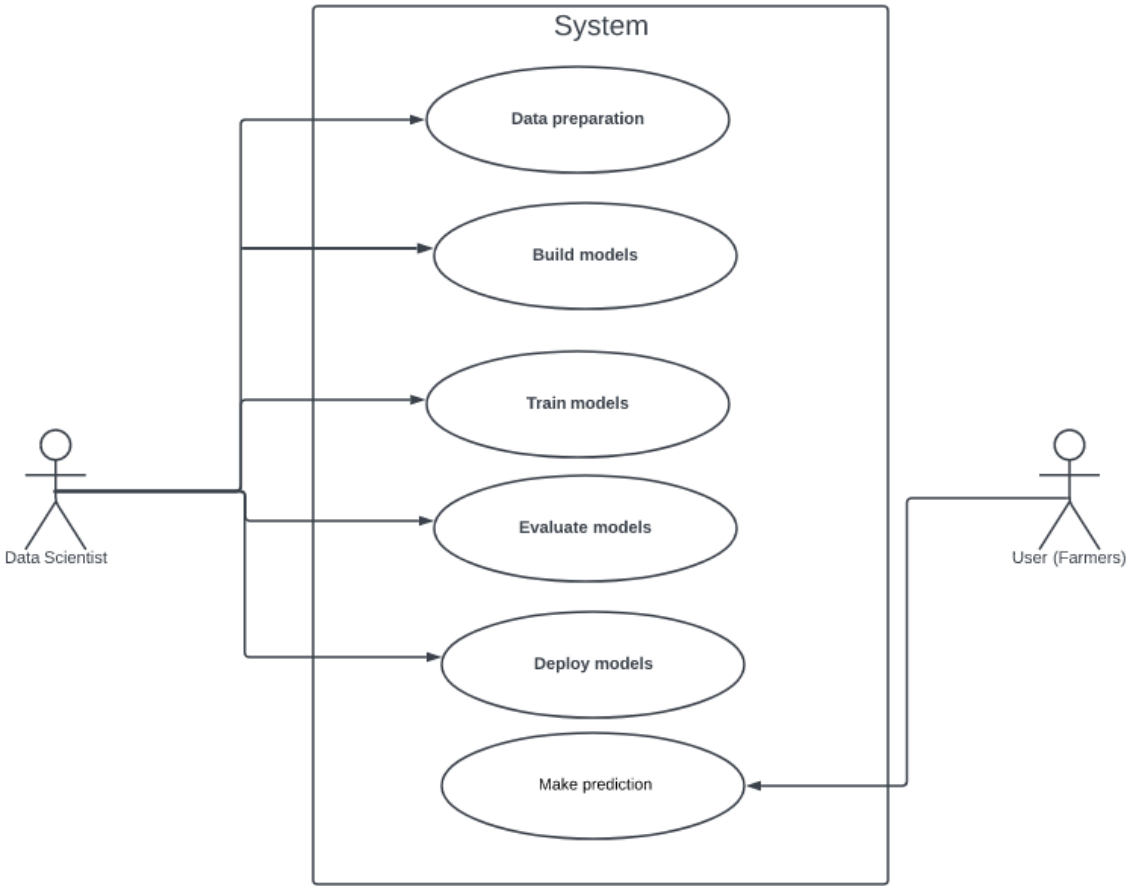


Figure 3.11: *Use case diagram.*

3.9. Sequence diagram

We will present the sequence diagram which is a graphical representation of the chronology of message exchanges between the actor and the system.

1- User: runs application after installation of the device.
2- Raspberry Pi: gets data from the sensor by communication module,
3- Communication module: send data from the sensor to Raspberry Pi.
4- Sensors: measures temperature, humidity, wind speed, wind direction, daily solar radiation, fruit weight, fruit moisture content, total sugar, pressure, soil moisture.

5- Next, Raspberry Pi makes a prediction and displays it in GUI.

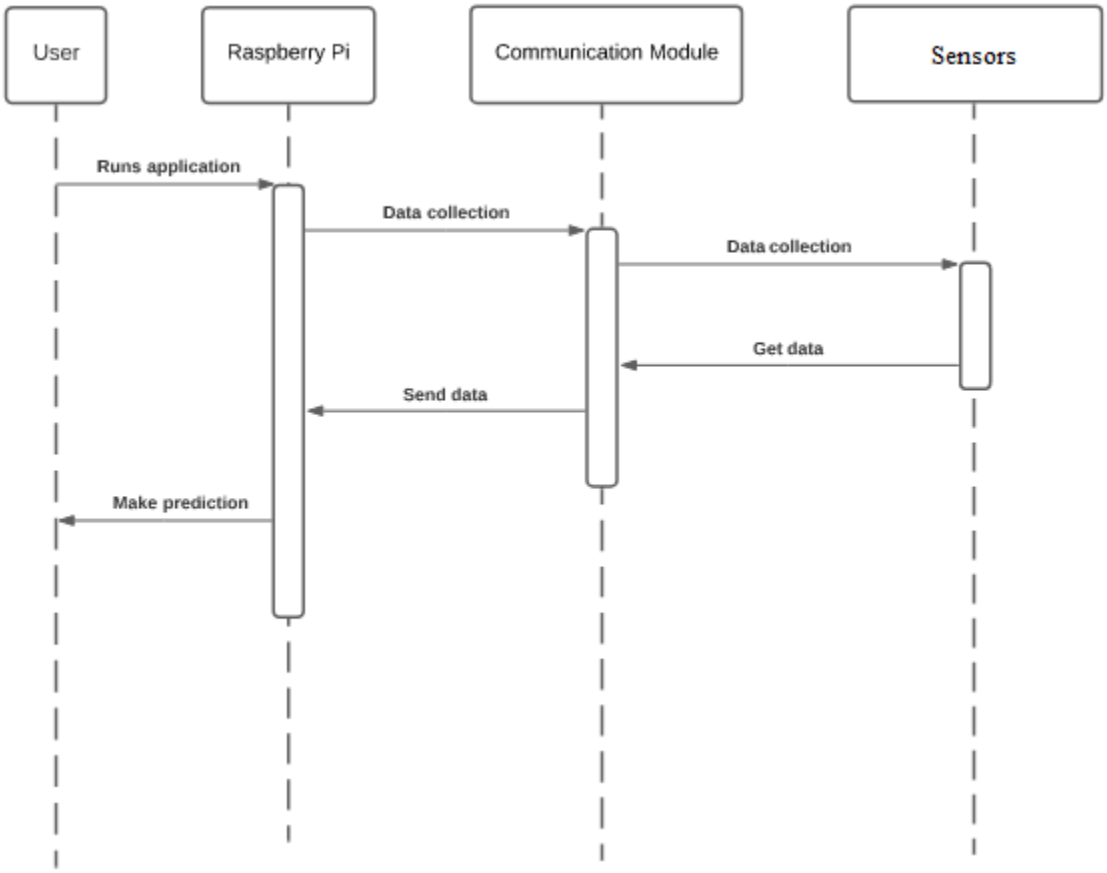
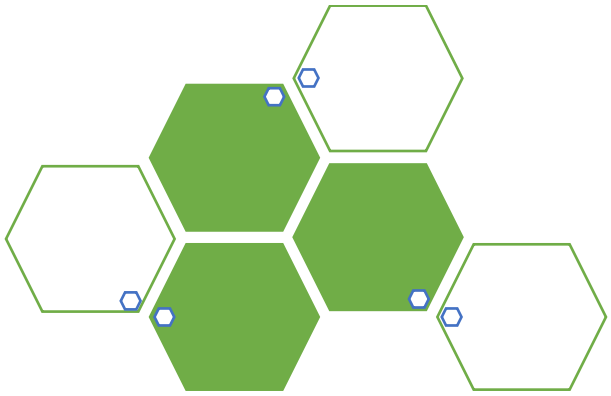


Figure 3.12: Sequence Diagram.

3.10. Conclusion

In this chapter, we have designed our idea of our system, starting with its global architecture and its operation, then we have defined the methodology adopted for the two parts that make up our system as well as the use case diagram, sequence diagram, and class diagram.

The next chapter will present the implementation of our system. The software-hardware environment, as well as the implementation techniques used, will be detailed.



Chapter 4:
Implementation and Results

Chapter 4 Implementation and Results

4.1. Introduction

The goal of this chapter is to demonstrate how we implemented our early prediction system. First, we will describe the hardware/software environment of the constructed system, followed by a description of the data set and the building of our models using LSTM and GRU.

In the second section will be deployed models in Raspberry Pi and we make real predictions with sensor humidity and temperature.

At the end of this chapter, we will show results obtained from the execution of the system and compare them to other systems, and then a conclusion.

4.2. The hardware/software environment

4.2.1. Software

For the environment, we chose Google Colab for the characteristics that we will mention later. Google Colab is a web-based document that allows you to develop, run, and share Python code. It is a Google suite of tools version of the popular Jupyter Notebook. Jupyter Notebooks (and hence Google Colab) enable you to create a document including executable code as well as text, photos, HTML, LaTeX, and so on, which is then saved in your Google Drive and shared with peers and colleagues for editing, commenting, and viewing. Colab is a hosted Jupyter Notebook service that requires no setup and provides free access to processing resources such as GPUs and TPUs **[33]**. We use TPUs for faster training.

Programming Language:

The Python programming language is establishing itself as one of the most popular languages for scientific computing. Thanks to its high-level interactive nature and its maturing ecosystem of scientific libraries **[34]**.

we use Python because contains, TensorFlow a modular machine-learning library that provides simple algorithms for use in machine-learning tasks and other libraries:

Keras: Keras is an open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library **[35]**.

Scikit-learn: scikit-learn is an open source machine learning library written in Python. It allows the easy and fast integration of machine learning methods into Python code. The scikit-learn library comprises a wide bandwidth of methods for classification, regression, covariance matrix estimation, dimensionality reduction, data pre-processing, and benchmark problem generation. **[36]**.

Numpy: NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting **[37]**.

Pandas: pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language **[38]**.

4.2.2. Hardware

Lilygo T-SIM:

The Lilygo T-SIM is a product of SIM card-based communication modules developed by Lilygo. These modules are designed to provide connectivity to various networks, such as GSM, GPRS, and NB-IoT (Narrowband Internet of Things). They allow devices to connect to cellular networks and enable communication capabilities like sending and receiving data.

Chapter 4 Implementation and Results

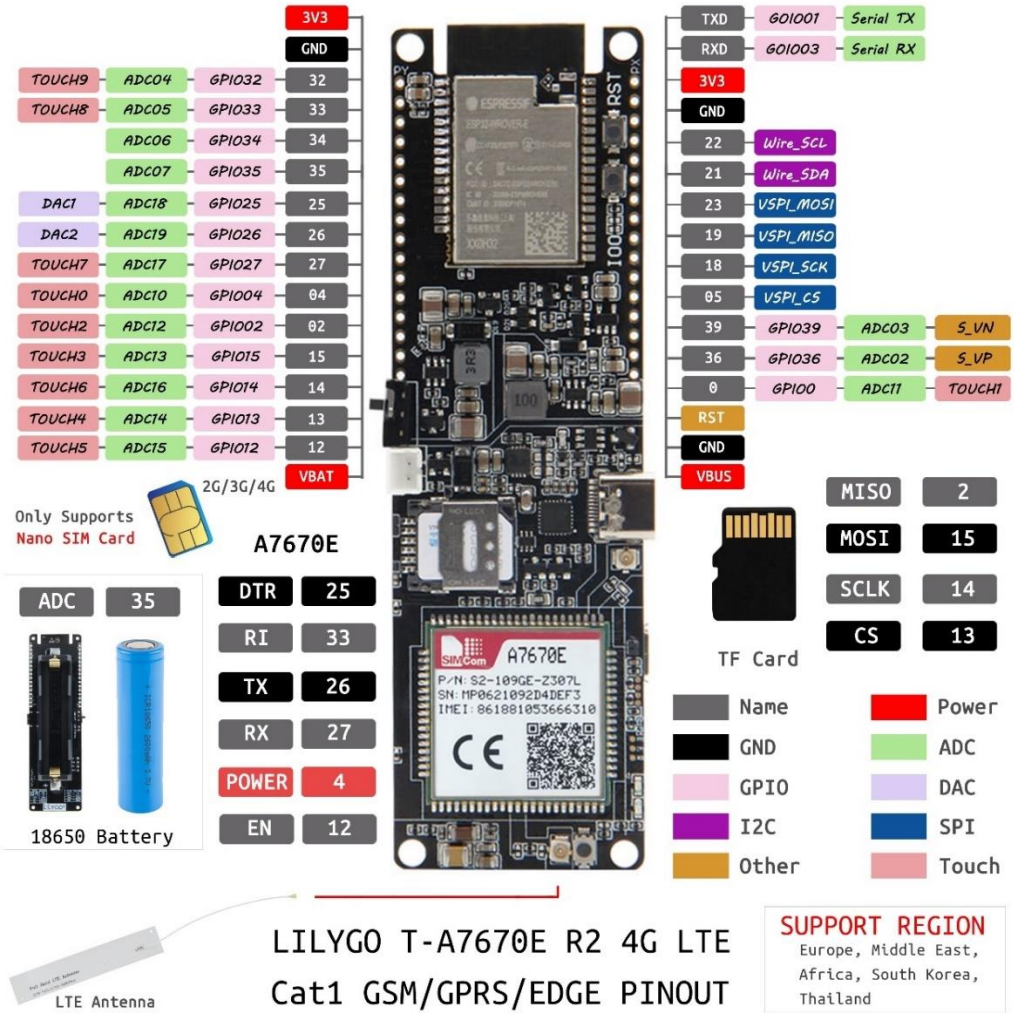


Figure 4.1: Lilygo T-SIM

LILYGO T-SIMA7670SA R2 ESP32 board features:

- Supply voltage: 3.3V DC or 5V DC
- ESP32 chip (WROVER-B Module) (240MHz dual-core processor)
- Flash memory: 4MB
- PSRAM: 8MB
- SRAM: 520KB
- Built-in Wi-Fi
- Built-in Bluetooth
- USB to serial converter: CP2104 or CH9102 (drivers)

- Built-in SIMA7670SA module
- Built-in nano SIM card slot
- Built-in SIM antenna slot
- Built-in GPS antenna slot
- Built-in Li-ion/Li-Po battery charging circuit:
    - DW01A battery protection IC
    - CN3065 solar energy charging interface for 4.4-6.8V solar panel
    - Built-in 1x 18650 battery holder
    - Built-in solar panel connector 2p JST-PH
- Built-in Micro SD card slot
- Built-in on/off switch

The Raspberry Pi 4 Model B with 8GB RAM:

Figure 4.2: The Raspberry Pi 4

The Raspberry Pi 4 Model B with 8GB RAM is a single-board computer released by the Raspberry Pi Foundation. It is an upgraded version of the Raspberry Pi 4 Model B with increased memory capacity. Here are some key features and specifications of the Raspberry Pi 4 Model B 8GB variant:

Processor: It is powered by a Broadcom BCM2711 quad-core Cortex-A72 (ARMv8) 64-bit system-on-a-chip (SoC) running at 1.5 GHz.

Memory: The 8GB variant offers 8 gigabytes of LPDDR4-3200 SDRAM, providing improved multitasking and memory-intensive application performance compared to lower RAM configurations.

Connectivity: It has dual-band 2.4 GHz and 5 GHz IEEE 802.11ac wireless LAN (Wi-Fi) and Bluetooth 5.0. Additionally, it features Gigabit Ethernet for wired network connectivity.

USB Ports: The Raspberry Pi 4 Model B 8GB has two USB 3.0 ports and two USB 2.0 ports, allowing you to connect various USB devices like keyboards, mice, external storage, etc.

Video Output: It supports dual monitor setups with two micro-HDMI ports, capable of outputting 4K resolution at 60Hz or lower resolutions.

Storage: The board offers microSD card slot for primary storage and also has USB 3.0 ports for connecting external storage devices like SSDs.

GPIO: It includes a 40-pin GPIO header that allows for connecting various sensors, modules, and other external components.

Operating System: The Raspberry Pi 4 supports a variety of operating systems, including Raspberry Pi OS (formerly Raspbian), Ubuntu, and several other Linux distributions.

DHT22 sensor:

Figure 4.3: DHT22 sensor

The DHT22 sensor, also known as the AM2302, is a temperature and humidity sensor commonly used in electronics projects and environmental monitoring applications.Temperature Measurement: The DHT22 sensor can measure temperature with a range of -40°C to 80°C (-40°F to 176°F). The temperature readings provided by the sensor are relatively accurate, typically within ±0.5°C. Humidity Measurement: The DHT22 sensor is also capable of measuring relative humidity (RH) with a range of 0% to 100% RH. The humidity readings it provides are generally accurate within ±2% RH.

Figure 4.4: 3.7 V 2200 mAh rechargeable lithium-ion battery.

Figure 4.5: Solar panel 6v 0.8w.

4.3. Implementation

4.3.1. Implementation of first section (models)

4.3.1.1. Database Description

We got data from the Scientific and Technical Research Center on Arid Regions in Biskra **[29]**. They gather it by temperature and humidity logger In the period 2021 - 2022 from Hadjeb, Leghrous and Mlili. Contains four records of temperature and humidity each day.

	Date	T°C	H%
0	10.02.2021	20.3	40.0
1	10.02.2021	20.9	40.0
2	10.02.2021	17.9	54.2
3	11.02.2021	16.0	50.3
4	11.02.2021	21.2	40.3
...	...	...	...
2018	29.09.2022	22.6	34.9
2019	30.09.2022	24.4	36.9
2020	30.09.2022	27.1	34.0
2021	30.09.2022	24.3	41.0
2022	30.09.2022	21.1	52.6

Figure 4.6: Raw data.

4.3.1.2. Data preparation

First we checked the missing data:

```
print(Data.isnull().sum())
```

```
Date    0
T°C     0
H%      0
dtype: int64
```

Figure 4.7: missing data

We restructured the data Every row is a day and we add the location by one hot encoder (p1= Leghrous, p2=Hadjeb, p3 =Mlili):

min_max_meanT = (min temp in the day + max temp in the day)/2.

min_max_meanH = (min humidity in the day + max humidity in the day)/2.

	meanT	maxT	minT	meanH	maxH	minH	p1	p2	p3	year	month	day	min_max_meanT	min_max_meanH
0	12.533333	21.1	5.4	51.966667	65.8	36.0	0	0	1	2021	2	4	13.25	50.90
1	14.025000	22.3	8.9	48.000000	56.1	32.4	0	0	1	2021	2	5	15.60	44.25
2	16.500000	19.1	13.7	36.875000	49.2	31.1	0	0	1	2021	2	6	16.40	40.15
3	15.100000	19.4	11.5	45.925000	52.0	35.9	0	0	1	2021	2	7	15.45	43.95
4	14.800000	21.2	10.1	51.125000	65.5	38.4	0	0	1	2021	2	8	15.65	51.95
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

Figure 4.8: restructured data.

These are ratings of the amount of danger from this insect (class Zero is no risk, and class four is very dangerous).

After that, we divided the data equally among the class, as well as randomly:

```
xtrain,xtest,ytrain,ytest = train_test_split(all,yc,test_size=0.2,stratify=yc)
```

This is the data distribution and training and test value for each model (30 days, 60 days, 90 days):

Table 4.1: The data distribution

class	All data 30 days	train data 30 days	test data 30 days	All data 60 days	train data 60 days	test data 60 days	All data 90 days	Train data 90 days	Test data 90 days
0	1071	857	214	981	785	196	906	725	181
1	252	201	51	252	201	51	237	189	48
2	60	48	12	60	48	12	60	48	12
3	66	53	13	66	53	13	66	53	13
Total	1449	1159	290	1359	1087	272	1269	1015	254

we use one hot encoder to code the classes. At the final step of preprocessing is the normalization we added a layer in each Network.

4.3.1.3. The models architecture

Our architecture of the models we build from scratch, this is the code of three models.

```
model=keras.Sequential([
    keras.layers.LayerNormalization(axis=1 , center=True , scale=True),
    keras.layers.LSTM(20,activation="relu",return_sequences=True,input_shape=(30,14)),
    keras.layers.Dropout(0.2),
    keras.layers.LSTM(20,activation="relu",return_sequences=True),
    keras.layers.Dropout(0.2),
    keras.layers.LSTM(20,activation="relu",return_sequences=True),
   keras.layers.Dropout(0.2),
    keras.layers.LSTM(20,activation="relu",return_sequences=True),
   keras.layers.Dropout(0.2),
    keras.layers.LSTM(20,activation="relu"),
   keras.layers.Dropout(0.2),
    keras.layers.Dense(4,activation="softmax")
])
opt = keras.optimizers.Adam(learning_rate=0.0001)
model.compile(
    optimizer= opt,
    loss="categorical_crossentropy",
    metrics=["accuracy"]
)
```

Figure 4.9: 30 days model.

```
model=keras.Sequential([
    keras.layers.LayerNormalization(axis=1 , center=True , scale=True),
    keras.layers.LSTM(40,activation="relu",return_sequences=True,input_shape=(60,14)),
    keras.layers.Dropout(0.2),
    keras.layers.LSTM(40,activation="tanh",return_sequences=True),
    keras.layers.Dropout(0.2),
    keras.layers.LSTM(40,activation="relu",return_sequences=True),
    keras.layers.Dropout(0.2),
    keras.layers.LSTM(40,activation="tanh",return_sequences=True),
    keras.layers.Dropout(0.2),
    keras.layers.LSTM(40,activation="relu",return_sequences=True),
    keras.layers.Dropout(0.2),
    keras.layers.LSTM(40,activation="tanh",return_sequences=True),
    keras.layers.Dropout(0.2),
    keras.layers.LSTM(40,activation="relu",return_sequences=True),
    keras.layers.Dropout(0.2),
    keras.layers.LSTM(40,activation="tanh",return_sequences=True),
    keras.layers.Dropout(0.2),
    keras.layers.LSTM(40,activation="relu"),
    keras.layers.Dropout(0.2),
    keras.layers.Dense(4,activation="softmax")
])
opt = keras.optimizers.Adam(learning_rate=0.0001)
model.compile(
    optimizer= opt,
    loss="categorical_crossentropy",
    metrics=["accuracy"]
)
```

Figure 4.10: 60 days model.

```
model=keras.Sequential([
    keras.layers.LayerNormalization(axis=1 , center=True , scale=True),
    keras.layers.GRU(128,activation="relu",return_sequences=True,input_shape=(90,14)),
    keras.layers.Dropout(0.2),
    keras.layers.GRU(128,activation="relu",return_sequences=True),
    keras.layers.Dropout(0.2),
     keras.layers.GRU(128,activation="relu",return_sequences=True),
     keras.layers.Dropout(0.2),
    keras.layers.GRU(128,activation="relu"),
    keras.layers.Dense(4,activation="softmax")
])
opt = keras.optimizers.Adam(learning_rate=0.0001)
model.compile(
    optimizer= opt,
    loss="categorical_crossentropy",
    metrics=["accuracy"]
)
```

Figure 4.11: 90 days model.

This is the end of training three models:

30 days model:

```
Epoch 146/150
73/73 [==============================] - 5s 64ms/step - loss: 0.0274 - accuracy: 0.9922 - val_loss: 0.0134 - val_accuracy: 0.9931
Epoch 147/150
73/73 [==============================] - 6s 77ms/step - loss: 0.0448 - accuracy: 0.9862 - val_loss: 0.0329 - val_accuracy: 0.9897
Epoch 148/150
73/73 [==============================] - 5s 64ms/step - loss: 0.0467 - accuracy: 0.9862 - val_loss: 0.0346 - val_accuracy: 0.9828
Epoch 149/150
73/73 [==============================] - 5s 65ms/step - loss: 0.0312 - accuracy: 0.9896 - val_loss: 0.0298 - val_accuracy: 0.9862
Epoch 150/150
73/73 [==============================] - 6s 79ms/step - loss: 0.0434 - accuracy: 0.9871 - val_loss: 0.0204 - val_accuracy: 0.9897
```

60 days model:

```
17/17 [==============================] - 6s 358ms/step - loss: 0.0629 - accuracy: 0.9798 - val_loss: 0.0446 - val_accuracy: 0.9779
Epoch 97/100
17/17 [==============================] - 8s 477ms/step - loss: 0.0505 - accuracy: 0.9816 - val_loss: 0.0196 - val_accuracy: 0.9963
Epoch 98/100
17/17 [==============================] - 6s 360ms/step - loss: 0.0765 - accuracy: 0.9752 - val_loss: 0.0195 - val_accuracy: 1.0000
Epoch 99/100
17/17 [==============================] - 8s 479ms/step - loss: 0.0548 - accuracy: 0.9807 - val_loss: 0.0574 - val_accuracy: 0.9706
Epoch 100/100
17/17 [==============================] - 6s 370ms/step - loss: 0.0509 - accuracy: 0.9825 - val_loss: 0.0124 - val_accuracy: 0.9963
```

90 days model:

```
32/32 [==============================] - 19s 580ms/step - loss: 0.0404 - accuracy: 0.9842 - val_loss: 0.0357 - val_accuracy: 0.9843
Epoch 26/50
32/32 [==============================] - 17s 546ms/step - loss: 0.0317 - accuracy: 0.9842 - val_loss: 0.0493 - val_accuracy: 0.9764
Epoch 27/50
32/32 [==============================] - 19s 600ms/step - loss: 0.0306 - accuracy: 0.9882 - val_loss: 0.0290 - val_accuracy: 0.9921
Epoch 28/50
32/32 [==============================] - 18s 560ms/step - loss: 0.0489 - accuracy: 0.9833 - val_loss: 0.0315 - val_accuracy: 0.9882
Epoch 29/50
32/32 [==============================] - 18s 578ms/step - loss: 0.0285 - accuracy: 0.9911 - val_loss: 0.0326 - val_accuracy: 0.9843
```

Chapter 4

Implementation and Results

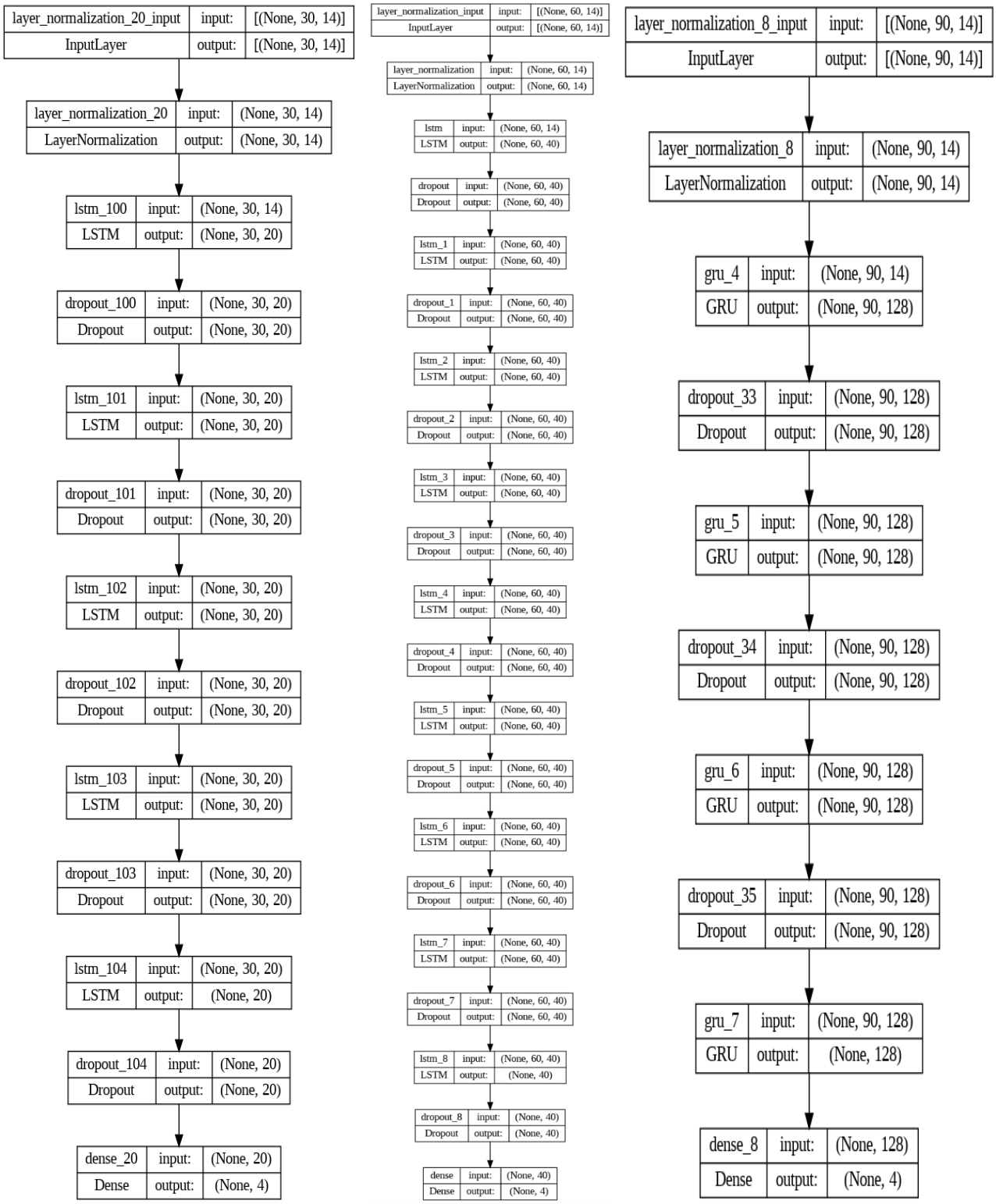


Figure 4.12: Models 30,60,90 days after completion.

4.3.1.4. Results and discussion

Before presenting the results and the comparison, there are some measures that we used, which we will explain in this section:

The confusion matrix is a table that summarises how well the classification model predicts examples from different classes. The confusion matrix has two axes: one for the predicted label and one for the actual label:

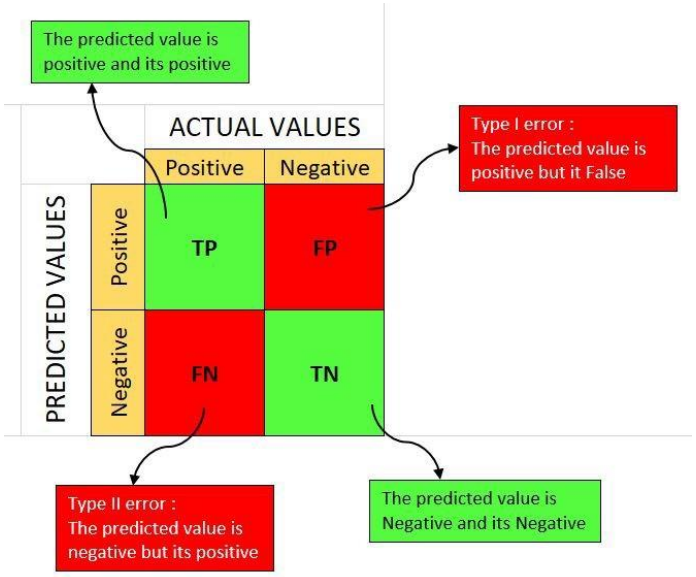


Figure 4.13: The confusion matrix

Precision: In definition it is define as the actual correct prediction divided by total prediction made by model: Precision = TP / (TP+FP)

Recall:

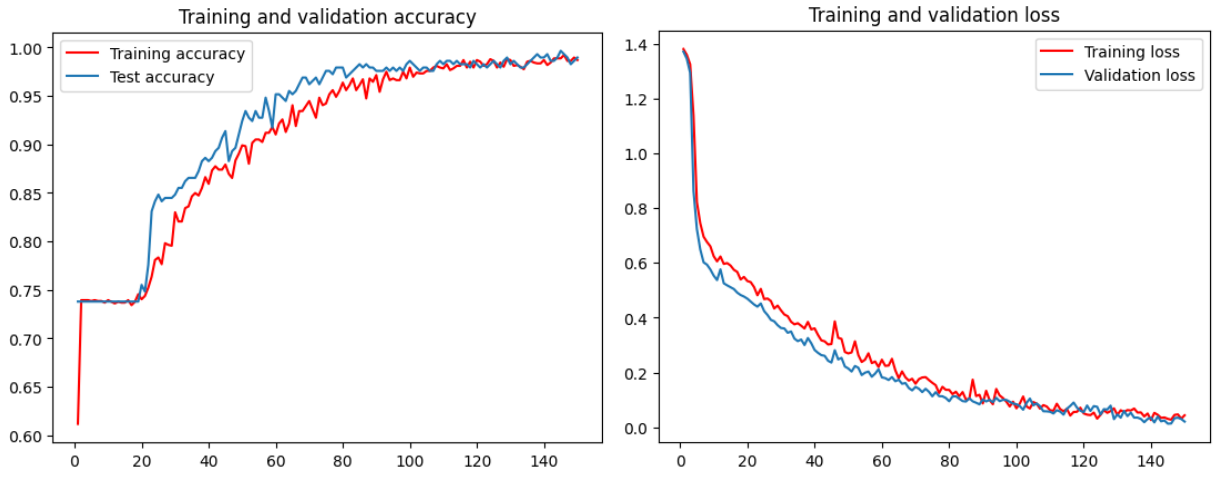
Recall = TP / (TP+FN)

It is defined as total correctly classified example divided by the total number of classified examples: (TP +TN) / (TP +TN+ FP +FN)

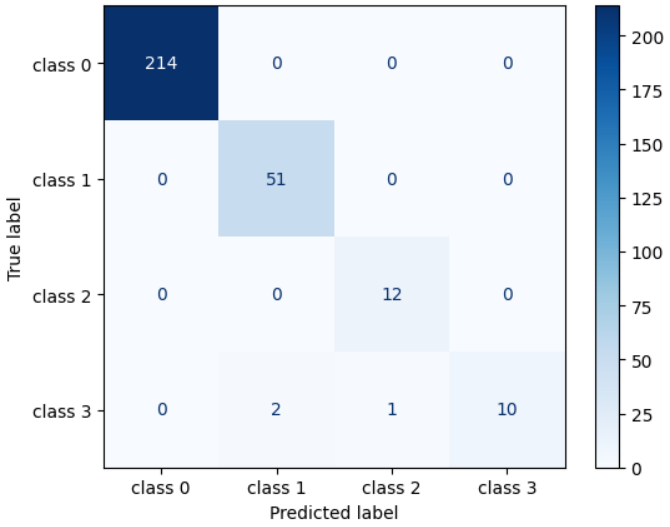
F1 score : is a weighted average of precision and recall:

F1 Score = 2*(Recall * Precision) / (Recall + Precision)

4.3.1.4.1. 30-days timeframe:



The first graph represents the accuracy of the training and validation data with a batch size=16 and epochs number = 150. We see an accuracy of 98% precision. The second graph represents loss of training and validation, we notice that these percentages quickly decrease as soon as the accuracy of the model increases and that is completely normal since the model is in training and as soon as the accuracy improves then the classification error is quickly reduced, the accuracy of loss is 2% (Train loss: 0.0434, Train accuracy: 0.9871, Validation loss: 0.0204, Validation accuracy: 0.9897).



This confusion matrix show that all class are predicted well, with some errors.

Compare with other models:

Table 4.2: Compare 30 day with other models

	accuracy	recall class0	recall class1	recall class2	recall class3	precision class0	precision class1	precision class2	precision class3	F1 class0	F1 class1	F1 class2	F1 class3
Our model	**98.97%**	**100%**	**100%**	**100%**	77%	**100%**	96%	**92%**	**100%**	**100%**	**98%**	**96%**	**87%**
RNN	94.83%	99%	98%	25%	**85%**	**100%**	**98%**	38%	55%	99%	**98%**	30%	67%
Ann	73.79%	**100%**	0%	0%	0%	74%	0%	0%	0%	85%	0%	0%	0%

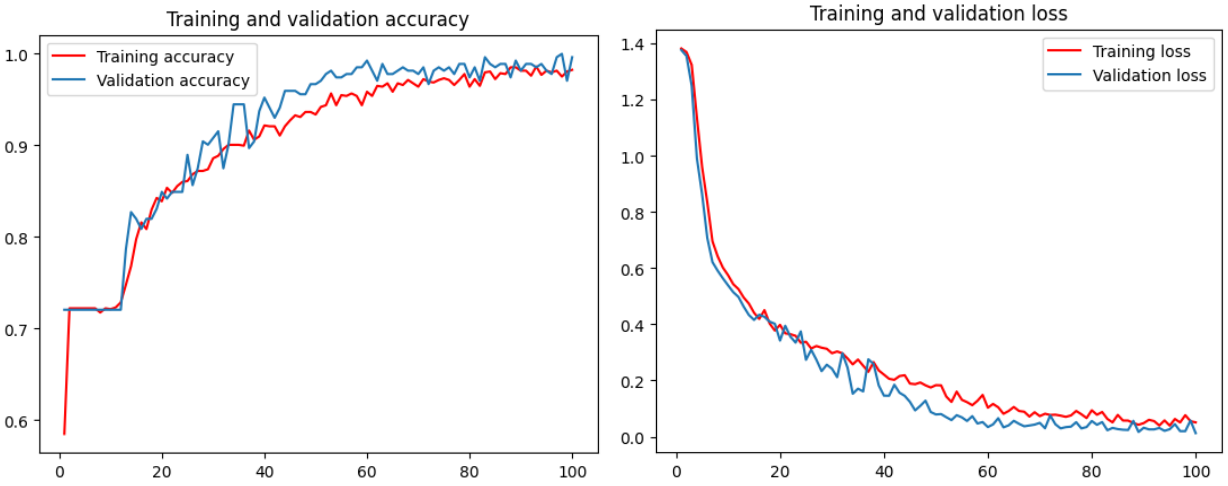
This table shows the results (accuracy, recall, and precision) of our model compared with other models: RNN and ANN. Our model achieved accuracy of 98.97%, with perfect recall for class 0, class 1, and class 2, while achieving 77% recall for class 3. It also demonstrates high precision for most classes. The F1 scores indicate a good balance between precision and recall for class 0, class 1, and class 2, with a slightly lower F1 score for class 3.

The RNN model achieved an accuracy of 94.83%. It demonstrates high recall and precision for class 0 and class 1 but has lower recall and precision for class 2 and class 3. The F1 scores reflect this imbalance, with significantly lower scores for class 2 and class 3.

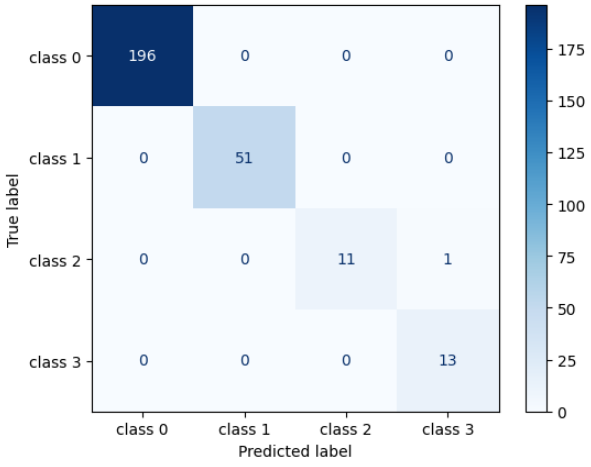
The ANN model achieved a lower accuracy of 73.79%. It shows perfect recall for class 0 but fails to recall any instances for class 1, class 2, and class 3. The precision values are also quite low. Consequently, the F1 scores for all classes are 0%, indicating poor performance.

In summary, our model performs well in terms of accuracy, recall, precision, and F1 scores compared to the RNN and ANN models. It achieves high accuracy and balanced recall and precision across most classes, providing a more reliable and accurate classification.

4.3.1.4.2. 60-days timeframe



The first graph represents the accuracy of the training and validation data with a batch size=64 and epochs number = 100. We see an accuracy of 99% precision. The second graph represents loss of training and validation, we notice that these percentages quickly decrease as soon as the accuracy of the model increases and that is completely normal since the model is in training and as soon as the accuracy improves then the classification error is quickly reduced, the accuracy of loss is 1.2% (Train loss: 0.0509, Train accuracy: 0.9825, Validation loss: 0.0124, Validation accuracy: 0.9963).



This confusion matrix show that all class are predicted well.

Compare with other models:

Table 4.3: Compare 60 day with other models

	accuracy	recall class0	recall class1	recall class2	recall class3	precision class0	precision class1	precision class2	precision class3	F1 class0	F1 class1	F1 class2	F1 class3
Our model	**99.63%**	**100%**	**100%**	**92%**	**100%**	**100%**	**100%**	**100%**	93%	**100%**	**100%**	**96%**	**96%**
RNN	95.22%	**100%**	96%	**100%**	15%	98%	**100%**	55%	**100%**	99%	98%	71%	27%
Ann	88.97%	92%	86%	33%	**100%**	96%	75%	**100%**	62%	94%	80%	50%	76%

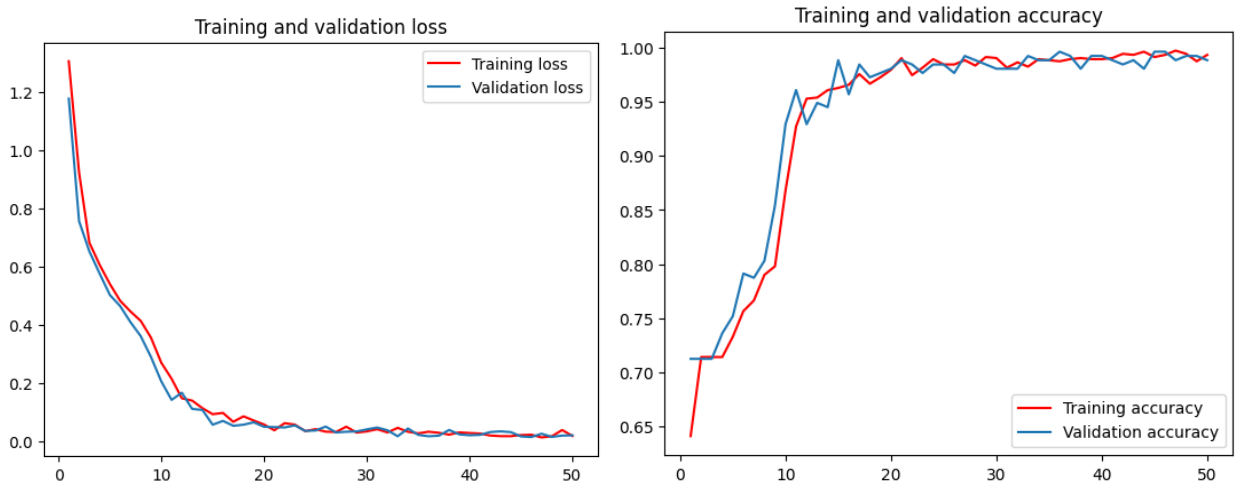
This table shows results (accuracy, recall, and precision) of our model compared to other models: RNN and ANN. It appears that our model achieved the highest accuracy (99.63%) among the three models. It also shows perfect recall and precision for class 0 and class 1. The F1 scores indicate a balance between precision and recall for each class.

The ANN model achieved the lowest accuracy (88.97%) among the three models. It has varying recall and precision values across different classes, with class 2 having the lowest recall (33%). The F1 scores for class 2 and class 3 are also comparatively lower for this model.

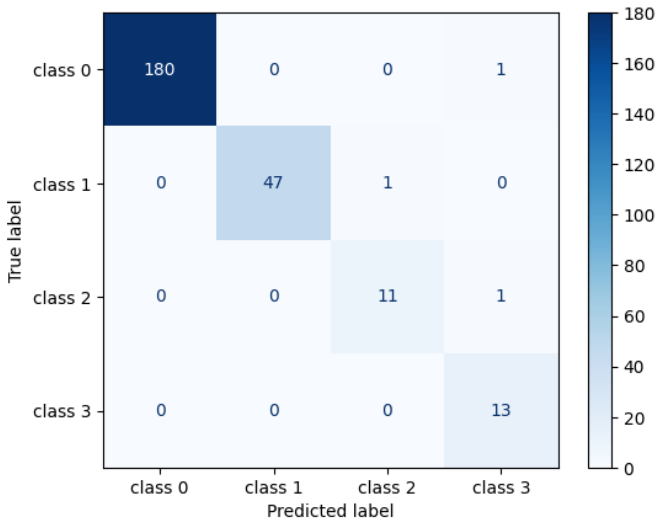
On the other hand, the RNN model achieved a lower accuracy (95.22%) compared to our model. It exhibits high recall and precision for class 0 and class 1 but struggles with class 2, having only a 15% recall. This model also has relatively lower F1 scores for class 2 and class 3.

Overall, our model outperforms the RNN and ANN models in terms of accuracy, recall, and precision for most classes.

4.3.1.4.3. 90-days timeframe:



The first graph represents the accuracy of the training and validation data with a batch size=32 and epochs number = 50. We see an accuracy of 98% precision. The second graph represents loss of training and validation, we notice that these percentages quickly decrease as soon as the accuracy of the model increases and that is completely normal since the model is in training and as soon as the accuracy improves then the classification error is quickly reduced, the accuracy of loss is 1.9% (Train loss: 0.0169 Train accuracy: 0.9931 Validation loss: 0.0191 Validation accuracy: 0.9882).



This confusion matrix show that all class are predicted well.

Compare with other models:

Table 4.4: Compare 90 day with other models

	accuracy	recall class0	recall class1	recall class2	recall class3	precision class0	precision class1	precision class2	precision class3	F1 class0	F1 class1	F1 class2	F1 class3
Our model	**98.82%**	99%	**98%**	**92%**	**100%**	**100%**	**100%**	**92%**	**87%**	**100%**	**99%**	**92%**	**93%**
LSTM	87.01%	98%	79%	42%	0%	88%	81%	**100%**	0%	93%	80%	59%	0%
RNN	83.07%	99%	67%	0%	0%	92%	54%	0%	0%	95%	60%	0%	0%
ANN	71.26%	**100%**	0%	0%	0%	71%	0%	0%	0%	83%	0%	0%	0%

This table shows the results (accuracy, recall, and precision) of our model compared with other models: LSTM, RNN, and ANN. Our model achieved accuracy of 98.82% and demonstrates high recall and precision for class 0, class 1, and class 3. It also has a good recall for class 2 (92%) and relatively high precision for most classes. The F1 scores indicate a good balance between precision and recall for all classes.

The LSTM model achieved an accuracy of 87.01%. It exhibits high recall and precision for class 0 but has lower values for class 1, class 2, and class 3. The F1 scores reflect this discrepancy, with significantly lower scores for class 2 and class 3.

The RNN model achieved an accuracy of 83.07%. It demonstrates high recall and precision for class 0, but recall and precision are lower or absent for class 1, class 2, and class 3. Consequently, the F1 scores for these classes are 0%.

The ANN model achieved the lowest accuracy of 71.26%. It shows perfect recall for class 0 but fails to recall any instances for class 1, class 2, and class 3. The precision values are also quite low. Consequently, the F1 scores for all classes are 0%.

In summary, our model performs well compared to the LSTM, RNN, and ANN models in terms of accuracy, recall, precision, and F1 scores. It achieves high accuracy and balanced recall and precision across most classes, providing a more reliable and accurate classification.

We can see that our models are better than other models in pest prediction. In the next part, we will install and program the device.

4.3.2. Implementation of second section (prototype)

We utilize a LilyGO T-SIMA7670sa development board, which integrates the SIMA7670s module for GSM communication. The board also features an external antenna for better signal reception. Along with the GSM capabilities, we incorporate a DHT22 temperature and humidity sensor. The sensor provides accurate readings of temperature and humidity levels. Additionally, a solar panel is utilized to harness solar energy, which can be used to power the system. To store energy, a lithium-ion battery is integrated, ensuring continuous operation even in the absence of direct sunlight. the device can transmit data over the GSM network, and power the system using solar energy and a rechargeable battery.

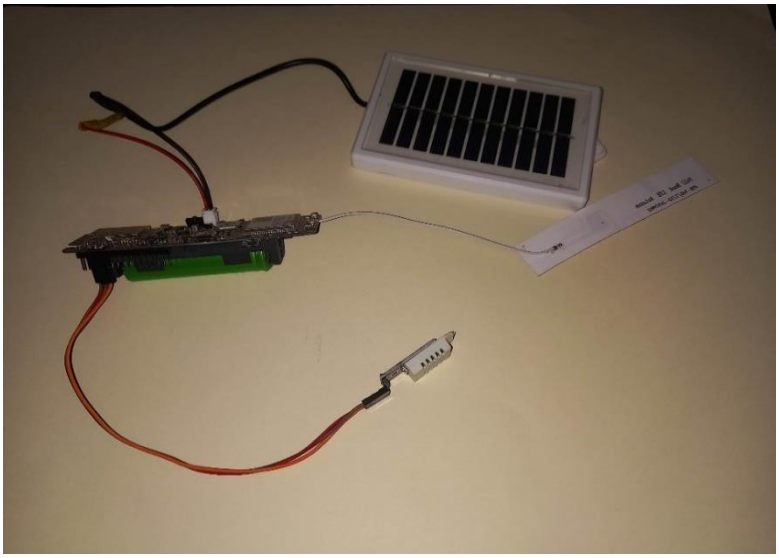


Figure 4.14: Implementation of device.

We installed a solar panel in a specially designated location, along with a SIM card, and proceeded to connect a DHT22 sensor to the relevant components. The sensor was connected using the VCC 3v3 (power supply), GND (ground), and GPIO 22 (general-purpose input/output) pins.

Chapter 4 Implementation and Results

Figure 4.15: Implementation of device.

We programmed LilyGO T-SIM A7670e using Arduino ide we send data to ThingSpeak, ThingSpeak is an Internet of Things (IoT) platform and data analytics service that allows users to collect, and visualize sensor data in real-time. Users can send data from their IoT devices to ThingSpeak using HTTP or MQTT protocols. It supports both numerical and textual data. We use HTTP protocol to send data. The image below shows an example of the data sent.

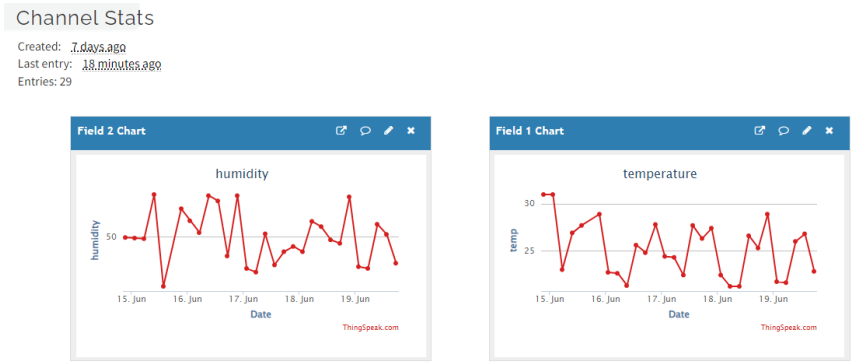


Figure 4.16: ThingSpeak GUI.

The Raspberry Pi acts as the center for data processing. It retrieves data from the cloud platform ThingSpeak, which serves as a repository for various sensor readings or other relevant data. Once the data is fetched, the Raspberry Pi analyzes the number of data available.

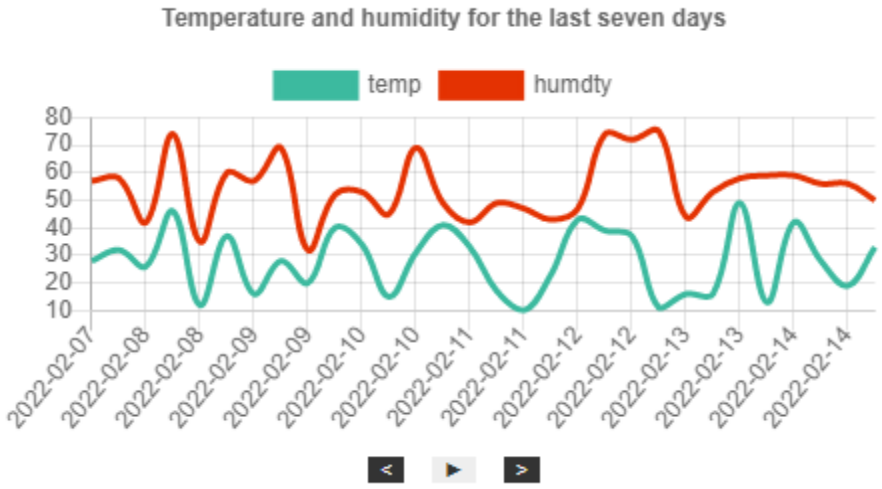


Figure 4.17: Data GUI

Based on the number of data, the Raspberry Pi employs one of three prediction models. These models are specifically designed to handle different ranges of data. For instance, if the number of data points is small (e.g., less than 60), Model 30 is selected. If the number of data points falls between 60 and 89, Model 60 is chosen. Finally, for a larger number of data points (over 90), Model 90 is utilized.

Once the appropriate model is selected, the Raspberry Pi makes predictions based on the provided data. The predictions are then displayed in a graphical user interface (GUI) developed using a Python GUI framework Eel.

Figure 4.18: Prediction GUI

In addition to the previous steps mentioned, we also considered energy consumption during the implementation. To optimize power usage, we incorporated a deep sleep mode in the LilyGO T-SIM A7670e module.

Deep sleep mode allows the LilyGO module to enter a low-power state when idle or during periods of inactivity. By utilizing this feature, we can significantly reduce energy consumption and prolong the device's battery life.

The LilyGO module, specifically designed for low-power applications, efficiently manages power usage during this mode.

To resume normal operation, lilygo/t-a7670sa we programmed to wake up periodically, and send new data to ThingSpeak.

By incorporating deep sleep mode in the LilyGO T-SIM A7670e module, we achieve an energy-efficient solution that balances data processing capabilities with minimal power consumption, making it suitable for remote or battery-powered applications where energy efficiency is a critical factor.

In addition to the aforementioned steps, we also focused on the physical design and protection of the LilyGO T-SIM module. To ensure its safety and durability, we designed a custom cover specifically tailored to fit the LilyGO T-SIM module.

we utilized 3D printing technology. Using a 3D printer, we transformed our digital model into a physical object. We selected a suitable filament material, considering factors such as

strength, durability, and temperature resistance to ensure the cover's reliability in various operating conditions.

The 3D printing process allowed us to produce a high-quality, customized cover with precise dimensions and a perfect fit for the LilyGO T-SIM module. The cover offers protection against external elements, such as dust, moisture, and accidental impact, safeguarding the module from potential damage.

Figure 4.19: cover

Figure 4.20: cover

4.4. Conclusion

In conclusion, this chapter presented the implementation of an early prediction system using LSTM and GRU models. The hardware/software environment was described, with Google Colab chosen as the development platform due to its features and access to processing resources like TPUs. The Python programming language, along with libraries such as Keras.

The hardware components used included the Lilygo T-SIM module for communication and connectivity purposes, the Raspberry Pi 4 Model B with 8GB RAM as a single-board

computer, and the DHT22 sensor for temperature and humidity measurements. Additionally, a rechargeable lithium-ion battery and a solar panel were incorporated for power supply.

The implementation of the models involved data preparation, restructuring, and normalization. The data set was obtained from the Scientific and Technical Research Center on Arid Regions, containing temperature and humidity records from three locations.

The models' architectures were built from scratch, and three models were trained using different timeframes (30 days, 60 days, and 90 days). The training process showed promising results, with high accuracy and low loss values.

General conclusion

General conclusion:

Boufaroua disease poses a severe threat to date palm productivity, with infection rates increasing under conditions of insufficient irrigation and neglect of maintenance. During dry and warm years, the damage caused by Boufaroua disease can exceed 80% of the crop, making it essential to address this issue promptly. However, Boufaroua disease is difficult to detect in its early stages visually, and controlling its spread becomes increasingly challenging as it progresses rapidly. Farmers often struggle to determine the appropriate timing for implementing chemical treatments or other preventive to combat this pest effectively.

we have successfully implemented our early prediction system for palm disease using sensors. We have also explained the data set used for training our models and the data preparation steps involved.

We implemented three different models based on the time frames of 30 days, 60 days, and 90 days using LSTM and GRU architectures. These models were trained and validated using the collected data, and we have presented the network architectures and the detailed code for each model.

After completing the training process, we evaluated the performance of our models. The results showed promising accuracy and predictive capabilities for the detection of palm disease. The confusion matrices provided insights into the classification accuracy for each class.

Overall, our implemented system has shown great potential in effectively monitoring palm trees, detecting disease at an early stage, and providing timely intervention for farmers. By combining IoT technology and deep learning, we have developed a solution that can significantly reduce the reliance on manual inspections and expensive chemical treatments. The use of sensors and automated predictions can help farmers make informed decisions and take proactive measures to mitigate the impact of palm disease on crop yields.

Considering the potential future work, there are several perspectives:

General conclusion

- Develop modular or adaptable equipment that can be easily adjusted or modified to handle different categories of pests or plant types. This approach would allow for flexibility and scalability, enabling the equipment to be tailored to specific needs.
- add a method to get rid of the insect automatically: drone or robot.

References

[1] https://www.aljazeera.net/ebusiness/ Accessed May 2023.

[2] https://www.statista.com/statistics/1182206/agricultural-area-in-algeria-by-crop-type/ Accessed May 2023.

[3] Lakhwani, Kamlesh, et al. "*Internet of Things (IoT): Principles, paradigms and applications of IoT*". Bpb Publications, 2020.

[4] Ali, Zainab H., Hesham A. Ali, and Mahmoud M. Badawy. "Internet of Things (IoT): definitions, challenges and recent research directions." *International Journal of Computer Applications* 128.1 (2015): 37-47.

[5] Patel, Keyur K., Sunil M. Patel, and P. Scholar. "Internet of things-IOT: definition, characteristics, architecture, enabling technologies, application & future challenges." *International journal of engineering science and computing* 6.5 (2016).

[6] Abdelhaq, Maha. "Internet of Things Fundamentals, Architectures, Challenges and Solutions: A Survey." *IJCSNS* 22.1 (2022): 189.

[7] Kurose, James F., and Keith W. Ross. "Computer Networking: A Top-Down Approach. 7th ed. ", Pearson, 2017.

[8] Arasteh, Hamidreza, et al. "Iot-based smart cities: A survey." *2016 IEEE 16th international conference on environment and electrical engineering (EEEIC)*. IEEE, 2016.

[9] Moysiadis, Vasileios, et al. "Smart farming in Europe." *Computer science review* 39 (2021): 100345.

[10] https://eos.com/blog/smart-farming/ Accessed May 2023.

[11] El-Shafie, Hamadttu AF. "The Old World date palm mite Oligonychus afrasiaticus (McGregor 1939)(Acari: Tetranychidae), a major fruit pest: biology, ecology, and management." *CABI Reviews* 2022 (2022).

[12] Simpson, Michael G. *Plant systematics*. Academic press, 2010

[13] Al-Mssallem, Ibrahim S., et al. "Genome sequence of the date palm Phoenix dactylifera L." *Nature communications* 4.1 (2013): 2274.

References

[14] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." *nature* 521.7553 (2015): 436-444.

[15] developer.ibm.com/developer/default/articles/an-introduction-to-deep-learning/images/AI-ML-DL.png Accessed May 2023.

[16] Deng, Li, and Dong Yu. "Deep learning: methods and applications." *Foundations and trends® in signal processing* 7.3–4 (2014): 197-387.

[17] Zhang, W. J., et al. "On definition of deep learning." *2018 World automation congress (WAC)*. IEEE, 2018.

[18] Khan, Murad, Bilal Jan, and Haleem Farman. "*Deep learning: convergence to big data analytics*. " Singapore: Springer, 2019.

[19] microsoft. (n.d.). "Deep Learning and Neural Networks. "

[20] Chapelle, Olivier, et al. "*Semi-Supervised Learning*."

[21] Gustineli, Murilo. "A survey on recently proposed activation functions for Deep Learning." *arXiv preprint arXiv:2204.02921* (2022).

[22] Karpathy, Andrej, Justin Johnson, and Li Fei-Fei. "Visualizing and understanding recurrent networks." *arXiv preprint arXiv:1506.02078* (2015).

[23] Soori, Mohsen, Behrooz Arezoo, and Roza Dastres. "Artificial intelligence, machine learning and deep learning in advanced robotics, A review." Cognitive Robotics (2023).

[24] Agarwal, Kartikeya, and Tismeet Singh. "Classification of skin cancer images using convolutional neural networks." *arXiv preprint arXiv:2202.00678* (2022).

[25] Tsimenidis, S. "Limitations of deep neural networks: A discussion of G." *Marcus' critical appraisal of deep learning. ArXiv* (2020).

[26] Abu-zanona, Marwan, et al. "Classification of Palm Trees Diseases using Convolution Neural Network." International Journal of Advanced Computer Science and Applications 13.6 (2022).

References

[27] Al Shidi, Rashid H., et al. "Relationship of date palm tree density to Dubas bug Ommatissus lybicus infestation in Omani orchards." *Agriculture* 8.5 (2018): 64.

[28] Mohammed, Maged, Hamadttu El-Shafie, and Muhammad Munir. "Development and Validation of Innovative Machine Learning Models for Predicting Date Palm Mite Infestation on Fruits." *Agronomy* 13.2 (2023): 494.

[29] crstra.dz Accessed May 2023

[30] Ndung'u, Rachel N. "Data Preparation for Machine Learning Modelling." (2022).

[31] colah.github.io/posts/2015-08-Understanding-LSTMs/img/LSTM3-chain.png Accessed May 2023

[32] 149695847.v2.pressablecdn.com/wp-content/uploads/2021/08/GRU.png Accessed May 2023

[33] Princeton.edu, mcgrawect.princeton.edu/guides. Accessed May 2023.

[34] Pedregosa, Fabian, et al. "Scikit-learn: Machine learning in Python." *the Journal of machine Learning research* 12 (2011): 2825-2830.

[35] Team, Keras. "Keras Documentation: About Keras." Keras.io, keras.io/about/

[36] Kramer, Oliver, and Oliver Kramer. "Scikit-learn." *Machine learning for evolution strategies* (2016): 45-53.

[37] NumPy. "Overview — NumPy V1.19 Manual." Numpy.org, 2022, numpy.org/doc/stable/.

[38] Pandas. "Python Data Analysis Library — Pandas: Python Data Analysis Library." Pydata.org, 2018, pandas.pydata.org/.