CIRCULAR SUPPLY CHAINS: AN IOT APPLICATION FOR ROTTEN PRODUCT DETECTION IN AGGREGATE FOOD INDUSTRY

(DÖNGÜSEL TEDARİK ZİNCİRLERİ: TOPLAM GIDA ENDÜSTRİSİNDE BOZUNMUŞ ÜRÜN TESPİTİ İÇİN BİR IOT UYGULAMASI)

by

Candan ERGELDİ, B.S.

Thesis

Submitted in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE

in

INTELLIGENT SYSTEMS ENGINEERING

in the

GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

of

GALATASARAY UNIVERSITY

Supervisor: Prof. Dr. Orhan Feyzioğlu

Jun 2023

ACKNOWLEDGEMENTS

I would like to express my gratitude to my supervisor, Prof. Dr. Orhan Feyzioğlu, for his continuous support and feedback during this research. Throughout the thesis exertion, he provided a constant source of encouragement and was always eager to help in any possible way. He guided me through the core papers on the circular economy and sustainability practices, and raised questions that led me to go farther and experiment with other approaches specifically for the multi-class classification. Meetings and discussions we held were of great consequence for driving me to look at things from other viewpoints and build a full and objective critique.

Jun 2023
Candan ERGELDİ

TABLE OF CONTENTS

LIST OF SYMBOLS

3R : Reduce, Reuse, Recycle
A2A : Application to Application
A2P : Application to Person
Agri-food : Aggregate Food
Al_2O_3 : Aluminum Oxide
ANN : Artificial Neural Network
AWS : Amazon Web Services
BD : Big Data
BGR : Blue, Green, Red
CE : Circular Economy
CH_4 : Methane
C_2H_4 : Ethylene
C_6H_{14} : Hexane
CO : Carbon monoxide
CO_2 : Carbon dioxide
CNN : Convolutional Neural Network
CPU : Central Processing Unit
DL : Deep Learning
E-mail : Electronic Mail
FFNN : Feed-forward Neural Network
GHG : Green House Gases
GND : Ground
HTTPS : Secure Hypertext Transfer Protocol
I4.0 : Industry 4.0
IC : Integrated Circuit
IDE : Integrated Development Environment
IoT : Internet of Things
JSON : JavaScript Object Notation
LoRaWAN : Long-Range Wide Area Network
LPG : Liquefied Petroleum Gas
ML : Machine Learning
MOS : Metal Oxide Semiconductor
MQTT : Message Queue Telemetry Transport
$MTCO_2e$: Metric Tons Of Carbon Dioxide-Equivalent
Ni-Cr : Collector Collector Voltage
NTC : Negative Temperature Coefficient
O_2 : Oxygen
PPM : Parts-per-million
Pt : Platinum

RAM : Random Access Memory
ReLU : Rectified Linear Unit
RGB : Red, Green, Blue
RH : Relative Humidity
SC : Supply Chain
SDG : Seven Development Goals
SMS : Short Message Service
SnO_2 : Tin Dioxide
SNS : Simple Notification System
SSD : Solid State Disk
TCP : Transmission Control Protocol
USDA : United States Department of Agriculture
WSS : Web sockets Secure
V_{cc} : Collector Voltage
QoS : Quality of Service

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Today, majority of food created is wasted rather than consumed, which has a negative impact on worldwide hunger and the economy. Improvements to aggregate supply chains are at the forefront of the actions needed to meet these problems. One of such improvements noted in this research was aggregate food storage. ESP8266 Microcontroller, along with DHT11 temperature and humidity sensor and MQ3 alcohol sensor, is used to measure the storage conditions of fruit products. Data gathered is sent to AWS-Cloud-Computing-Service via MQTT protocol and is stored in databases using rules defined. As result, the sensor data in the database is examined using AWS IoT Analysis. As temperature rises, crops begin to decompose. Accordingly, a rule in cloud is defined to fire with out-of-range measurements, and AWS Simple Notification Service is triggered to send ambient readings to user via SMS and e-mail. Convolutional Neural Network model was developed to classify fruits based on their variety and freshness using binary and multiclass classification. Models was first taught using images of 1693 fresh apples, 1581 fresh bananas, 1466 fresh oranges, 2342 rotten apples, 2224 rotten bananas, and 1595 rotten oranges and tested with images of 395 fresh apples, 381 fresh bananas, 381 fresh oranges, and 388 rotten apples, 601 rotten bananas, and 530 rotten oranges over 50 epochs. Binary model had 99.15% training and 97.59% testing accuracy whereas multi class model had training accuracy of 99.59% and testing accuracy of 98.99%.

Keywords: Circular Economy, Sustainability, Industry 4.0, Agrifood Supply Chain, Internet of Things

ÖZET

Günümüzde üretilen gıdanın büyük bir kısmı tüketilmek yerine israf edilmekte ve bu durum dünya çapındaki açlığı ve ekonomiyi olumsuz etkilemektedir. Agregat tedarik zincirlerindeki iyileştirmeler, bu sorunları çözmek için gereken eylemlerin başında gelir. Bu araştırmanın sonucunda agregat gıda depolaması alanında bu tür bir gelişme kaydedilmiştir. ESP8266 Mikrodenetleyici, DHT11 sıcaklık ve nem sensörü ve MQ3 alkol sensörü ile birlikte meyve ürünlerinin saklama koşullarını ölçmek için kullanılmaktadır. Toplanan veriler, MQTT protokolü aracılığıyla AWS Bulut Servisi'ne gönderilir ve tanımlanan kurallar kullanılarak veritabanlarında depolanır. Sonuç olarak, veritabanındaki sensör verileri AWS IoT Analysis kullanılarak incelenir. Sıcaklık arttıkça mahsuller çürümeye başlar. Bu nedenle, parametreler belirlenen optimal aralığın dışına çıktığında kullanıcıya bildirmek üzere AWS IoT Uygulamasında kurallar tanımlanmıştır. AWS Simple Notification Service, bu kurallar etkisiyle tetiklenerek ortam sıcaklığı, nem ve metanol değerleri SMS ve e-posta ile kullanıcıya iletilmektedir. Evrişimli Sinir Ağı modeli, ikili ve çok sınıflı sınıflandırma kullanarak meyveleri çeşitliliklerine ve tazeliklerine göre sınıflandırmak için geliştirilmiştir. Modeller ilk olarak 1693 taze elma, 1581 taze muz, 1466 taze portakal, 2342 çürük elma, 2224 çürük muz ve 1595 çürük portakal görseli kullanılarak öğretilmiş ve 395 taze elma, 381 taze muz, 381 taze portakal ve 388 görselle test edilmiştir. 50 çağda çürük elma, 601 çürük muz ve 530 çürük portakal. İkili model %99,15 eğitim ve %97,59 test doğruluğuna sahipken, çok sınıflı model %99,59 eğitim doğruluğu ve %98,99 test doğruluğuna sahiptir.

Anahtar Kelimeler: Döngüsel Ekonomi, Sürdürülebilirlik, Endüstri 4.0, Tarımsal Gıda Tedarik Zinciri, Nesnelerin İnterneti

1. INTRODUCTION

Elimination of food waste is a major worldwide challenge and a vital prerequisite for economic development. Poor food management has a substantial impact on waste generation. Therefore, to reach the goal of zero food waste, sustainable practices should be monitored from farm to fork from an economic, social, and environmental standpoint. In the 2030 Agenda for Sustainable Development, it is stated that urgent measures should be taken on sustainable consumption and production, sustainable management of natural resources and climate change for the continuity of life on Earth.

The circular economy (CE) model is a production and consumption model that comprises sharing, leasing, reusing, repairing, refurbishing, and recycling existing goods for longer duration, and it is the most popular method in terms of food waste management (92%). The CE's primary focus in the supply chain (SC) is to ensure the optimal environmental results by enhancing the efficient use of resources. The optimized material flows in circular SCs, leads to higher economic rewards compared to the traditional take-make-dispose aspect of linear ones. As a result, applying the CE in aggregate food (agri-food) SCs has the potential to alleviate the negative operational, economic, and environmental effects of food waste. These mostly include energy and resource waste, idle time wasted on food production, raised emissions of greenhouse gases, and increased food expenses due to decreased food supply.

Food service organizations must improve efficiency at each phase of the supply chain, through procurement to logistics, production to marketing, and sales to after-sales support. Specifically, the agri-food sector additionally is volatile and complicated, necessitating the use of advanced techniques to maximize efficiency by optimizing resources, and reducing costs (Duncan et al., 2019.; Angkiriwang et al., 2014.). At this

stage, industry 4.0 (I4.0) is playing an essential role in the circularization of agri-food SCs by transforming supply networks into intelligent systems capable of attaining full traceability of goods (Zhang, 2020).

Internet of Things (IoT) and Big Data (BD) are major aspects in the I4.0 framework, proposing a linear-to-circular model transition to assist sustainability issues. IoT has an impact on agri-food supply chain through improved connectivity, reduced interference from humans, and energy conservation (Vaibhav et al., 2022). Utilization of IoT in food chains arouse with numerous linked devices that span portable agricultural equipment, tools, and hardware to residential devices and temperature-sensing devices (Rao & Clarke, 2019). Moreover, the utilization of big data has advanced IoT networks by precisely documenting SC variables based on sensor network data; improved information visualization across the food network; promoted higher transparency, efficiency, and data-driven decision-making (Ji et al., 2017; Li & Wang, 2017). As a result, more intelligent decisions are made, allowing food chains to run more efficiently, minimize expenditures, expedite the decision-making procedure, and reduce risks.

This thesis is structured as follows; Literature review section includes an evaluation of the works on I4.0 applications in the agri-food SC. The section then provides the findings of the previous studies in Convolutional Neural Network (CNN) based fruit image classification and Wireless Sensor Networks (WSN) for the rotten crop detection. Methodology section describes how the study is conducted in detail: CNN structure for image classification and WSN model. Results and discussion section presents the outputs of the implementation that are classification accuracy, sensor outputs and user information measures. Finally, Conclusion section restates the findings in terms of a sustainability and under the improvement purpose of circular SCs and presents relevant research for future investigations.

2. LITERATURE REVIEW

2.1 Agri-Food Supply Chains

Aggregate food (agri-food) products can be defined as agricultural goods, forest fleece, marine and freshwater species, and foodstuffs that are obtained from these raw materials and semi-products. Their production rate has grown drastically from the beginning of the century in order to meet the demands of emerging markets like China and India (Accorsi et al., 2019). Researchers predict that agri-food production volumes will be even more by 2050 due to the predicted doubling of global food demand (Koning and van Ittersum, 2009). Nevertheless, the anticipated expansion will also have unfavorable side effects. It is more likely that the agri-food supply chains (SCs) will require assistance in distributing food equitably and sustainably as they grow (Ahumada and Villalobos, 2009; World Bank, 2013). Furthermore, nutrient flows across the world are dramatically linearized both in the supply and in the sink sides as a result of industrialization, globalization, and agricultural specialization. This linearization widens gap between urban and rural populations and hence, causes an uneven movement of agricultural nutrients from rural areas to cities. This, along with the international agricultural commodity trading, tends to cause nutrient depletion. Yet, a sizable amount of the food is wasted during supply in a period of such tremendous demand for food. The need to put policies into place that support the shift to sustainable systems was stressed by the international community (The FAO, 2017). Therefore, the primary goal of international unions and organizations has been to eradicate food waste that is thrown out at the retailer and consumer tiers (The FAO, 2019). The World Bank declared that food waste accounts for 30% of the world's food output during the procurement process. Hence, studies on the food waste eradication solutions have become more crucial than ever. The 2030 Agenda for Sustainable Development, particularly asks for fundamental reforms in governments, society, and corporate sectors, to reflect this demand. The Commission's "Farm to Fork" strategy, which forms the backbone of the European Green Deal, desires to create equitable, healthy, and environmentally friendly agri-food systems by developing a uniform policy

for all Europe Member States that is capable of interfering in all stages of the SC, from production to consumption (European Commission, 2020; Romero et al, 2021).

2.2 Waste in Agri-food Supply Chains

To begin with, identification of the “food waste” and “food loss” has vital importance. Although used interchangeably, these two words have different meanings. Food "loss" occurs as a result of issues in the manufacturing, storage, processing, and distribution stages prior to consumption. In contrast, food waste happens when food fit for consumption is discarded during the retailing or consuming stages. According to food waste research, one-third of all food produced is wasted or lost, resulting in yearly cost savings of 1.3 billion tons (Mangla et. al, 2018). The SCs are accountable for the most of these wastes that occur. As an example, it is stated that 40% of the entire food is wasted in the food SC (USDA Economic Research Service, 2020). Therefore, urgent measures shall be taken on sustainable consumption and production, management of natural resources and climate change for the continuity of life on Earth (Agenda for Sustainable Development, 2020). Regarding this issue, most research highlights the importance of using industry 4.0 (I4.0) tools such as internet of things (IoT) and big data (BD) to promote sustainable food SC models that recognize agricultural waste. IoT-based waste management systems, for example, can differentiate municipal waste, recognize waste traits, and identify sustainable waste treatment methods (Fatimah et al., 2020).

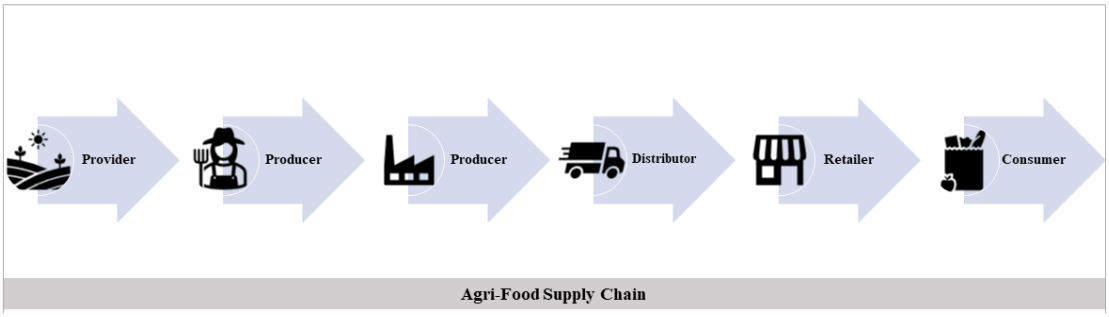


Figure 2.1: Figure Showing Agri-Food Supply Chain

2.3 Circular Economy Model for Sustainable Agri-food Supply Chains

A closed model of production and consumption is a circular economy (CE). The CE model prioritizes sharing, renting, reusing, repairing, renovating, and recycling of produced goods and materials. The CE emphasizes waste valorization, cost savings, and improved resource efficiency in contrast to the conventional linear economy paradigm with the "take-make-dispose" method. Hence, implementing CE in agri-food SCs could mitigate the negative operational, financial, and environmental effects of food waste including energy and waste of resources, time consumed producing food, an upsurge in greenhouse gas (GHG) emissions, and higher prices for food due to shortage of food. Therefore, CE model could effectively counteract mentioned undesirable situations and results in improved resource regeneration, stability in prices, financial resilience, conservation of resources, and GHG mitigation.

Modern retailers' corporate image is positively impacted by recovering food waste, however food that is unsellable because of its condition is often wasted. This is due, in part, to the lack of a recycling option for merchants or others, for instance a dedicated waste management facility, for unsalable foods. The best course of action for retailers in this situation, however, should be to reuse and redistribute food that would otherwise go to waste because of seasonal demand and a slow rate of sales. In particular, foods that are meant for waste could be used to make derivative items like salad or juice, give customers who buy such foods a discount, or redirect these food items for an alternate purpose through other channels as food banks. The grade and quality of the food produced, nevertheless, is a component that often has an impact on this approach. For instance, a case study revealed that retailers preferred to discard excess fruits and vegetables rather than selling them for less money or donating them due to the fact they were concerned about their brand's reputation (Pagotto & Halog, 2016). Academics further discussed the use of food outside of salable conditions. Their study recommended selling goods that do not meet requirements and are therefore unsuitable for modern stores in traditional markets or donating them to food banks. Additionally, certain goods can be utilized to make electricity, compost, or bio-fertilizer, or they can be sold as raw materials for animal feed (Pagotto & Halog, 2016). The fact that the food was reused is directly related to CE

principles. Cascading direction, waste reduction, maximizing efficiency, ecological awareness, optimizing the good's preserved value, and leakage mitigation are the six fundamental concepts of CE. As an example, maximization of the product's retained value pertains to the conversion of commodities into other derivatives, such as animal feed or substance to various types of food. Moreover, leakage mitigation ensures that no resources are lost during the biological cycle and end up as compost or as plant fertilizer.

However, although adopting the CE has many benefits as indicated, there are some restrictions to place the model in application. These include lack of knowledge regarding transition to the CE Model, high investment costs for technological setups, food SC system security concerns, lack of research-related digital technologies in the realm of agri-food sustainability along with the workers' technical competence requirements.

2.4 The Circular Economy and Industry 4.0 Relationship

I4.0 is transforming how businesses produce, enhance, and distribute their goods by combining information and communication technologies (ICTs) and production and manufacturing processes with its fundamental attributes as scalability, security, QoS (Quality of Service), reliability, modularity, and the networking capabilities of the system. In I4.0 perspective, the IoT, cloud computing analytics, artificial intelligence (AI), machine learning (ML), industrial automation (robotics), simulations, system integration, cybersecurity, and additive manufacturing are incorporated into production facilities and operations as a whole. Thanks to its inclusive structure with the indicated sub-areas, I4.0 offers various applications in different sectors such as food, textile, electronics, and automobile. In terms of the food sector, I4.0 improves traceability of material flow in food SC, provides accurate data accumulation and hence, ensures the reduction of adverse environmental and social effects by lowering waste, recycling, and minimizing natural resource utilization (Javaid et. al., 2022). Therefore, it could be said that the modernization and industrialization of food SCs have reached a significant turning point with I4.0 by incorporating cutting-edge technology innovations.

The I4.0 framework proposes a linear-to-circular model shift that supports sustainability considerations in agri-food SC, with IoT, a real-time data transfer system for products, services, processes, activities, or tasks, and BD, serving as the key contributors. IoT advantages the agri-food SC by improving connectivity, requiring less human interaction, and managing energy (Vaibhav et. al, 2022). Its use in food chains has risen with a variety of linked devices, from mobile agriculture instruments, and machines as well as residential appliances and temperature-sensing systems. (Rao & Clarke, 2019). Moreover, in order to manage dynamic storage, packaging, distribution, and selling based on sensor network data, BD enhanced IoT networks by correctly capturing time and temperature data and communicating it with exchange partners and improved information visualization across the entire food network and encourages great openness, efficiency, and informed choice-making (Li and Wang, 2017; Ji et al., 2017). As a result, now smarter decisions are made enhancing the efficiency, cost-savings, and risk-reduction of food chains.

2.5 Barriers to I4.0 Application in the Agri-Food SC

For resource maximization, cost minimization, and process optimization purposes, efficiency management is required in the agri-food SC (Angkiriwang et al., 2014). Food service businesses should therefore make sure that efficiency is improved at all points throughout the SC, from sourcing to logistics, production to marketing, and sales to after-sales support. To ensure and maximize this efficiency, the agri-food industry needs complex tools due to its diversity and dynamic nature (Duncan et al., 2019). However, although technologies appropriate for accomplishing nutrient circulation on an expansive basis are being investigated under the I4.0, their implementation and upscaling application are still unknown framework (Harder et al., 2019; Johannesdottir et al., 2020). Academics indicate that the main barriers for the implementation are absence of digitalization strategies coupled resource shortages, inefficient I4.0 integration methods with sustainability objectives, and a lack of finances for I4.0 initiatives (Raj et al., 2020). On the other hand, to retain sustainable systems, reliable methods for diagnosing and simulating the variables affecting the circularity of nutrient flows must be developed. In consideration of these matters, the study indicated in paper provides an efficient way to

overcome these barriers by being cost-effective in-terms of the selection of necessary hardware and software and by offering an end-to-end integrated and inclusive model with full surveillance of the agri-foods during storage.

2.6 Effect of Storage Conditions on Fruit Freshness

By having a high consumer demand, fruits are fundamental crops in global agriculture. The typical SC for fruit goods incorporates quality inspections to ensure freshness, flavor, and safety. Therefore, the level of ripening is an essential component in determining fruit quality. However, fruits are prone to rotten and have a short shelf life due to physiological and biochemical activities (Kader, 1986). Their degeneration starts from the moment when they are aparted from the parental stem (Kays, 1991). The tissue cells of cultivated plant commodities constantly respire, absorb oxygen (O_2) from their surroundings, and emit carbon dioxide (CO_2). Thus, respiration is a primary factor contributes to perishable post-harvest losses (Bhande et. al., 2007). The post-harvest respiratory activity of fresh produce is affected by storage air temperature, relative humidity (RH) and the O_2, CO_2, and ethylene (C_2H_4) (if the type of the fruit is sensitive to C_2H_4) composition. In general, enzymatic reactions are reduced and fruit post-harvest life is extended over its usual lifespan by giving relatively low temperatures, moderate O_2 level, and relatively higher CO_2 (Kader, 1986; Saltveit, 2004; Bhande et. al., 2007). As a result, the major elements controlled to keep the fruit in optimal condition over its usual season are the ambient temperature, RH and air composition of the storage area and handling process of the fruit before and during storage (Bhande et. al., 2007)

When assessing the quality of fruits, additional to their physiological responses to the storage conditions, their physical (weight loss, color, and firmness) and chemical properties (total soluble solids (TSS), pH, titratable acidity (TA), and vitamin C/ascorbic acid) are evaluated, as well. The required level of these factors varies according to the type and cultivar of the fruit. In this research, our dataset for rotten and fresh image classification dataset involves the pictures of apples, oranges, and bananas. Therefore, this section specifically focuses on the assessment of physical and chemical attributes of

these types of fruit and provides the necessary criteria to attain the optimal storage for them.

2.6.1 Cold Storage of Fruits

Most fruits require cold storage since they are notoriously perishable and consequently have a short shelf life due to their high-water content (80% - 90%) (Tripu & Farcuh, 2021). Their characteristics related to quality alter swiftly at ambient temperature, limiting storability. Permanent ambient storage conditions cause alterations to color, texture, flavor (soluble solids, pH, aroma), and nutritional content. (Tripu & Farcuh, 2021). Regarding this issue, cold storage is an effective method for preserving fruit quality, reducing losses, and increasing harvest amount. Integrating cold storage into operations extends the market potential by prolonging fruit durability. Thus, fruits in cold storage can be marketed all year despite their season of growth. The physical appearance, volatiles, and scents of the fruit have a major effect on purchasing decisions of the customers (Tripu & Farcuh, 2021). Therefore, preserving the look of fruits is highly important for the retailers, since it is considered to be an indicator of freshness, crispness, and sweetness by the customers (Tripu & Farcuh, 2021). In addition to sales, the use of cold storage facilitates export facilities as well. These benefits imply a higher profit margin for the agri-food sector.

2.6.2 The Impact of Storage Conditions on Apple Ripening

Apples are one of the most durable fruits. However, their storage still needs specific attention since it could cause chilling effect or over-ripeness of the product. Also, the storage of apples has high influence on their quality. As an example, ascorbic acid/Vitamin C and glucose levels peak at the climacteric and begins to lower as the fruit ripens. Regarding this issue, there are various studies in the literature to determine the optimal storage conditions for apples. Although these conditions slightly vary from one apple cultivar to another, parameters affecting storage (O_2 and CO_2 levels, C_2H_4 susceptibility, RH) fall within the same range, in general.

Apples ripen normally in temperatures that range from 15° to 30°C, with the fruits susceptible to fungal attack at degrees over 25°C. If the apples are only partially ripe, they can be stored cold for 2-3 months, whereas completely ripened apples can only be kept cold for less than a month. Mature apples can be stored at 0°C to ~3.89°C for 6 months based on the cultivar. Also, via careful inspection, this duration can be prolonged to 12 months. In addition, ripening for apples is accelerated through the CO_2 elimination and supplying O_2 to the storage area and diminishes otherwise.

On the other hand, apples are more susceptible to chilling injuries than ripeness. Certain chilling sensitive cultivars become damaged by prolonged exposure to low non-freezing temperatures. Therefore, temperatures below 15°C of storage areas cause chilling injuries to apples. Figure 2.2 shows the chilling injury occurred in a Honeycrisp apple.

Apples stored at low relative humidity matures rapidly compared to those stored at high relative humidity. Therefore, optimal humidity level for the apples is within the range of 85% – 95%.

Figure 2.2: Figure showing soft scald due to chilling sensitivity in Honeycrisp apple. (Tripu & Farcuh, 2021.)

Apples are mainly sensitive to C_2H_4 although some cultivars such as Annona Squamosa L. is not visibly affected from C_2H_4 (Broughton & Guat, 1979). Since this work focuses on the general assumption for the optimal storage conditions on the fruits,

apples are assumed as sensitive to C_2H_4. However, storage parameters defined in this thesis are general assumptions and exemplary for the use cases based on the literature review. Therefore, their values can be changed afterwards by the client user based on the cultivar.
durability for sea shipment, and mitigates chilling damage (Bishop, 1996; Chauhan et al., 2012; Morelli et al., 2003).

2.6.3 The Impact of Storage Conditions on Banana Ripening

One of the most vital foods traded globally is the banana commodities. Based on Food Codex for Bananas, they must be, sound, with no rotting or deterioration that renders it unfit for consumption; free of any foreign odor and/or taste; firm; without any harm due to low temperatures; and devoid of bruising, according to the unique requirements for each banana category within tolerances (Banana Codex, 1997). Nevertheless, as bananas are climate-dependent, they undergo significant physiological alterations following harvest (Stover & Simmonds, 1987). Also, the shelf life of bananas that ripen on the tree is only about 7-10 days which is criticized as a short shelf life to be feasible for mass market exportation. As a result, bananas are harvested when they are green and handled differently than other fruits during their storage to avoid ripening. They are artificially ripened when they arrive at their target destination.

The ripening involves the artificial yellowing process. In yellowing phase of the bananas, fruits undergo specific processes to reach their bright yellow color. The yellowing and ripening process takes place in specifically constructed chambers that are equipped with heat and gas insulation, ventilation systems, and temperature and humidity control. The following conditions are required for an efficient yellowing and ripening procedure:

1. Temperature (for yellowing: 20-28°C, for ripening: 14-20°C)
2. Humidity (85% - 95%)
3. Ethylene gas (3-10 ppm for yellowing, 100-1000 ppm for ripening)
4. Air movement (to circulate the volume of the room 25-30 times per hour)
5. Time (48-96 hours for painting and ripening)

Figure 2.3 shows the 7 stages of food ripening before decay. According to these stages, the most appropriate step for the marketing of bananas is 4, and the most appropriate step for consumption is 6.

Numerous investigations have demonstrated that optimal banana preservation conditions are between 12 and 16°C and 90% to 95% RH, that delays ripening, maintains the Bananas are normally stored at temperatures ranging from 13.33 to 14.44°C for a long-term transportation and storage. Research indicated that as the storage temperature increases, banana fruit ripens faster, and the shelf life decreases exponentially as the storage temperature rises. (Yoshika et al., 1978; George and Marriott, 1985; Saymour et al., 1987; Kim and Lee, 1988; Peacock, 1980).

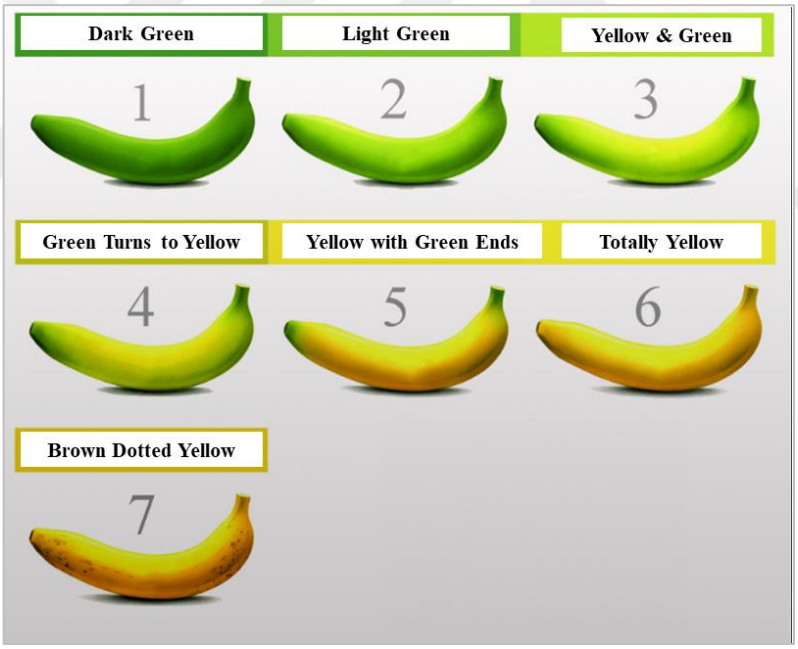


Figure 2.3: Yellowing Stages of the Banana

This fact might be due to fact that respiration rate is higher at high temperatures. On the other hand, bananas should not be exposed to temperatures below ~12.78°C since they are highly prone to chill damage. Storage life of banana increases by the increase in of the RH storage atmosphere. Optimum RH for bananas is about 90.00% - 95.00%. (Patil and Magar, 1976; Ramana et al., 1989). Susceptibility to C_2H_4 of bananas differ based on

their level of maturity. Green bananas are sensitive to C_2H_4 whereas ripen bananas are not. This could be due to fact that the ripening process causes a significant rise in the rate of C_2H_4 production. Accordingly, C_2H_4 is used for ripening bananas in commercial practices. C_2H_4 emission of bananas accelerated by the rise in temperature.

2.6.3 Storage Conditions and Ripening of the Oranges

Oranges require close surveillance of the handling and storage conditions after harvesting. Temperature and relative humidity are the most critical factors in preserving quality and increasing the storage and shelf life of the orange. Their softness and shelf life are diminished as a result of post-harvest water loss in fruits and vegetables (Smith et al., 2006). In addition to high water content, they have a high ascorbic acid concentration which makes the fruit more prone to substantial degradation during post-harvest handling and storage than other antioxidants (Wilhelmina, 2005). Fruit rotting has also been related to a decrease in beneficial substances such phenolics and ascorbic acid (Sanusi et al., 2008). On the other hand, different from the other types of fruits mentioned, based on the findings in literature review oranges are not sensitive to C_2H_4. Hence their ripening is not accelerated by the increase in C_2H_4 emission. The optimal storage temperature of oranges shall vary between 4.4 to 7.2°C and their optimal storage area must have a level of RH between 90.00% - 95.00%.

To sum up the findings from the literature review based on storage conditions of the fruits, the optimal storage conditions of the apples, bananas, and oranges, in terms of temperature, RH, and C_2H_4 are summarized in Table 2.1.

2.7 Industry 4.0 Applications for the Detection of Food Freshness

Regarding its importance, there are various studies in literature to determine the fruit maturity. ML and deep learning (DL) techniques drive the top-performing methods provided in these studies. Many researchers utilized different classification methods to accurately classify different fruits and vegetables based on their freshness. Moreover, researchers studied different IoT configurations to measure fruit storage parameters, as well. This section provides detailed information regarding these works.

Table 2.1: Table displaying the optimal storage conditions for the selected fruits.

Fruit	Temperature (°C)	RH (%)	Cold Chilling Sensitivity	C_2H_4 Sensitivity
Apple	-1.1 - 4.4	90 – 95	Yes	Yes
Green Banana	16.7 - 21.1	85 - 95	Yes	Yes
Ripen Banana	13.3 - 15.6	85 - 95	Yes	No
Orange	4.4 - 7.2	90 – 95	Yes	No

2.7.1 Fruit Image Classification

Creating methods to automate the fruit grading process is of great interest. Support vector machines, decision trees, and K-Nearest Neighbor algorithms are examples of ML techniques that have been effectively used to solve classification issues in the literature, notably for fruit categorization. Recently, artificial neural networks (ANNs) and convolutional neural networks (CNNs), have also been used to the fruit classification, with encouraging results (Rizzo et. Al., 2023).

In computer vision applications including image classification, and object identification, convolutional neural networks are often utilized due their advantages as: focusing on the different parts rather than the complete picture (Chakraborty et. Al., 2021), and attaining higher accuracy (Das et. Al, 2020).

As indicated in Table 2.2, numerous studies have been conducted that aim to monitor the freshness of the food image classification techniques. However, these studies constitute only a small part of fruit surveillance and/or freshness detection. In consideration of the aggregate food SC, it is undeniable that a much more comprehensive, holistic, and integrated system is needed. Therefore, to meet this demand, we built an end-to-end integrated wireless sensor network for the surveillance of storage area temperature,

humidity and alcohol composition using microcontroller coupled with CNN model that classifies the fruits based on their type and level of freshness then, transmits these data to cloud model and analyzes for the further decision-making process of the retailer. Regarding this issue, our study combines provides a profound system that integrates the image classification and sensor gathered data in one and uses cloud platform as a baseline. In addition, most studies regarding the agri-food SC improvement are concerned with the production-to-retailer phase; whereas our study provides a different aspect for the waste management since, it targets the retailer-to-consumer phase, and focuses on the waste mitigation at the retailer storage area.

Table 2.2: Literature Review on CNN Models for Fruit/Vegetable Classification.

References	Dataset	Model	Accuracy
Zhang et. al. (2015)	UEC-FOOD100	5-layer CNN	80.80%
Mureşan et. al. (2018)	Fruits-360	AlexNet, GoogLeNet, Own CNN	~99.00%
Lu, Y. (2019)	ImageNet	5-layer CNN	With augmentation 90.00%
Sakib et. al. (2019)	Fruits-360	Own CNN	99.79%

Table 2.3: Literature Review on Fruit Freshness Classification Using CNN.

References	Dataset	Model	Accuracy
Mudaliar et. al. (2021)	Tomato 500	2-layer CNN	98.74%
Nuanmeesri et. al. (2021)	Own Dataset of red apple, rose apple, and rambutan fruit with 500 images of each.	VGG16 CNN	91.33 %

2.7.2 Freshness Detection of Fruits via Sensors

In addition to fruit image classification, academics further progressed to develop models for the detection of fruit freshness via sensors and cameras. Mostly cameras were utilized to take pictures of the fruits initially, then classification methods were deployed to determine whether fruits are fresh or rotten.

2.7.3 Necessity of Cloud Architecture

IoT applications rely on sensor data to function, but processing and storing it may be quite difficult. Data aggregation from many and remote sources of information causes several privacy issues regarding information leaking (Guo & Wang, 2019). In addition, using many IoT devices to monitor and assess various parameters, adds to the intricacy due to the variety of sensors and data formats in usage. Therefore, a sound acquisition and monitoring system is needed for the data security of the aggregate food storage. One of the key applications for companies is maintaining the volume, speed, variety, and quality of data gathered by sensors in a cloud-based or hybrid IoT infrastructure.

There are several benefits to storing data on the cloud. One of the primary benefits is that it enables for more efficiency because users may access their files remotely. This removes the need to physically access a gadget. Another advantage is the ability to store a large amount of data. Cloud storage is additionally affordable and scalable. Also, the storage capacity could be expanded or reduced without purchasing extra hardware. Furthermore, cloud storage services give high levels of data security and protection. They safeguard your data from illegal access and cyber dangers by utilizing powerful encryption technology. Finally, cloud storage is remarkably dependable and offers simple data recovery solutions in the event of data loss or system failure. Moreover, compared to conventional infrastructure, cloud storage and computing can offer environmental benefits. This is due to the fact that cloud service providers optimized their data centers for energy efficiency by powering their data centers with renewable energy sources thus, lowering their carbon impact. They also, cut power consumption and enhanced resource usage by utilizing modern cooling systems, server virtualization, and other technologies. In addition, cloud providers can combine multiple virtual servers onto fewer physical

servers, decreasing total energy consumption and physical space requirements. Considering these issues, our study provides a cloud-based data acquisition system that ensures security of the data gathered via key and certificate assurance in addition to credentials for root (administrator) and I AM (client) user roles.

Table 2.4: Literature Review on Freshness Detection of Fruits via Sensor.

References	Configuration	Method	Accuracy
Jayasankar et. al. (2018)	Raspberry Pi, Proximity and Gas Sensors, Load Cell	Digital Image Processing (OpenCV Method)	80.00%
		CNN	95.00%
Shobana et. al. (2022)	Raspberry Pi, Virtual digicam, Ultrasonic sensor, Speaker	Visual Geometry Group	82.00%
		Random Forest Classification	80.00%
		Light Gradient Boosting Machine	74.00%
		Logistic Regression Classification	77.00%
Betal, S. (2019)	Arduino Mega 2560, TCS3200 Sensor, Conveyor belts, DC motors.	Image classification methods were not applied.	If sensor values are outside the acceptable range of freshness, prototype model alerts its users.
Jayasinghe et. al. (2022)	ESP8266 NodeMCU Microcontroller, O_2	Keras Sequential Model	98.90%

Ayan et. al. (2018)	Sensor (ME_2- O_2 -20), CO_2 Sensor, Humidity Sensor (DHT11) Photoelectric Proximity Sensor, Telemecanique ATV18U18M2 Inverter, Electrical Panel and PLC	. Image Processing	Avg. 92.00%
Sonwani et al. (2022)	Raspberry Pi, Arduino, Heat, Humidity & Gas Detection Sensor, Cooling Module	CNN	95.00%

3. MATERIALS and METHOD

The proposed model, illustrated in Figure 3.2, is initiated by taking input data from the shelf of retail stores or from their warehouses. It includes temperature, humidity, and gas sensors, and a camera for real-time detection of objects. The camera is used to take images of the products in retail stores. The sensors' humidity, excess gas, humidity reading outputs, and image info are fed to the Amazon Web Services (AWS) Cloud Platform. In AWS Cloud Service, taken images are processed using classification methodologies via convolutional neural network (CNN). Within the context of this study, fruits are classified using both a binary and a multiclass classification model. The aim of using binary model was to determine the freshness of the products solely whereas, for multiclass classification both the freshness and the respective type of the fruits were evaluated. By doing so, the project contributes to the advance in retailer-to-consumer phase of the aggregate food supply chain (agri- food SC) by enabling better surveillance for the storage conditions of the retailer.

3.1 Convolutional Neural Network

A machine learning method is often created by establishing and training an architecture to learn parameters. A typical neural network training approach consists of the set of the configurations to their default settings and the definition of the optimization algorithm. Then the following procedures are repeated until the optimal value of the cost function is gained:

1. Forward inputting.
2. Determination of the cost function.
3. Backpropagation is used to calculate the gradients of the cost with regard to parameters.

4. Utilizing gradients, each parameter is updated along with the optimization process.
5. Following that, the model can then be used to predict the class of a new data point given a new data point.

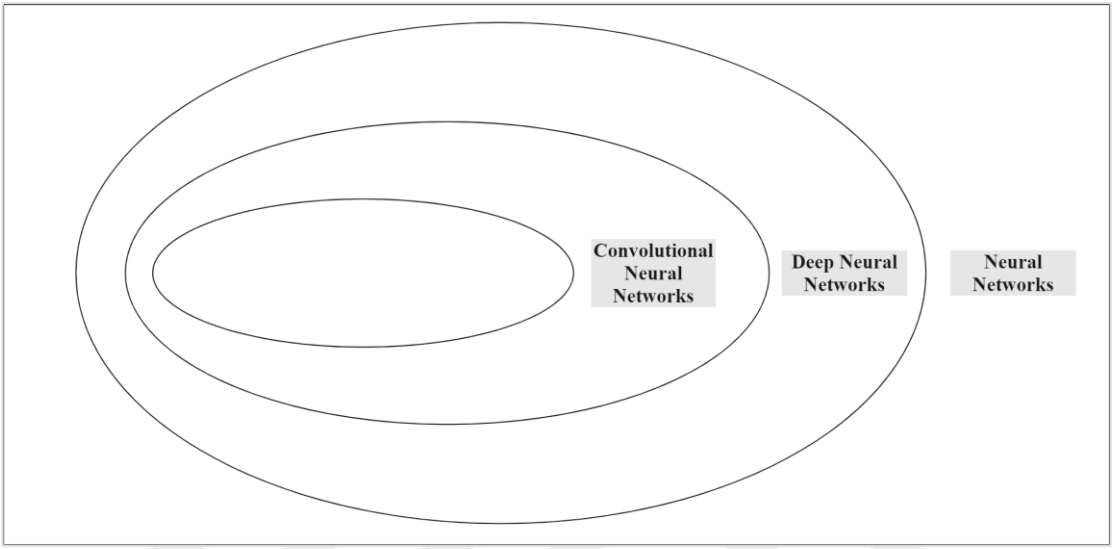


Figure 3.1: Figure Showing Relation Between Neural Networks.

CNN is a multilayer feed-forward neural network (FFNN) that can swiftly identify, categorize, and recognize features in images using perceptrons. CNNs are beneficial for image and natural language processing among other cognitive tasks. However, they are mostly utilized for image classification applications. Massive amount of data can be forwarded into the networks. Therefore, CNNs could be trained for recognition of certain characteristics accurately.

With the light of this knowledge and based on the literature review, “CNN” based approach was selected as the methodology to perform bothbinary and multi-class classification tasks for the determination of fruit type and freshness due to its advantages as:

1. Local receptive fields: Contrary, in fully connected networks, wherein the value of each unit is determined by the entire input, a unit in convolutional networks is

2. determined by a subset of the input. This region in the input is the unit's receptive field (Gao et. al., 2022).
3. Shared weights: By assigning identical weights to all input samples (pixels), weight sharing encourages a neural network to recognize prevalent 'local' patterns throughout an image (Took and Mandic, 2022.)
4. Pooling: The fundamental goal of pooling is to decrease the size of feature maps, a process that in turn speeds up computation by reducing the amount of training parameters.

In a CNN, the convolution process consists of the following steps:

1. The kernel matrix is placed over each pixel of the image and multiplied by each kernel value.
2. The multiplied values are then added, and the result is provided as the new value of the center pixel.
3. This procedure is repeated throughout the image.

The methodology used in this study for CNN for multiclass classification is as following:

1. The fruit images are selected and resized to 32 x 32.
2. Then, BGR (Blue, Green, Red) images are converted into RGB (Red, Green, Blue).
3. The dataset is normalized by division to 255.
4. Class labels are pre-processed, and the model architecture is designed as:
   4.1 Kernel initializer (*layer weight initializer*) was selected as “he uniform”.
   4.2 Activation functions for convolutional and 2D-convolutional layers are selected as 'ReLU' function.
5. In the Pooling Layer, “Max Pooling” is selected.
6. Activation function for the fully connected layer is selected as sigmoid.
7. Model is taught with training set over 50 epochs.
8. The model is compiled with Adam optimizer and categorical-cross entropy with learning rate = 0.001.
9. As the final step, model is tested with 50 epochs using testing set.

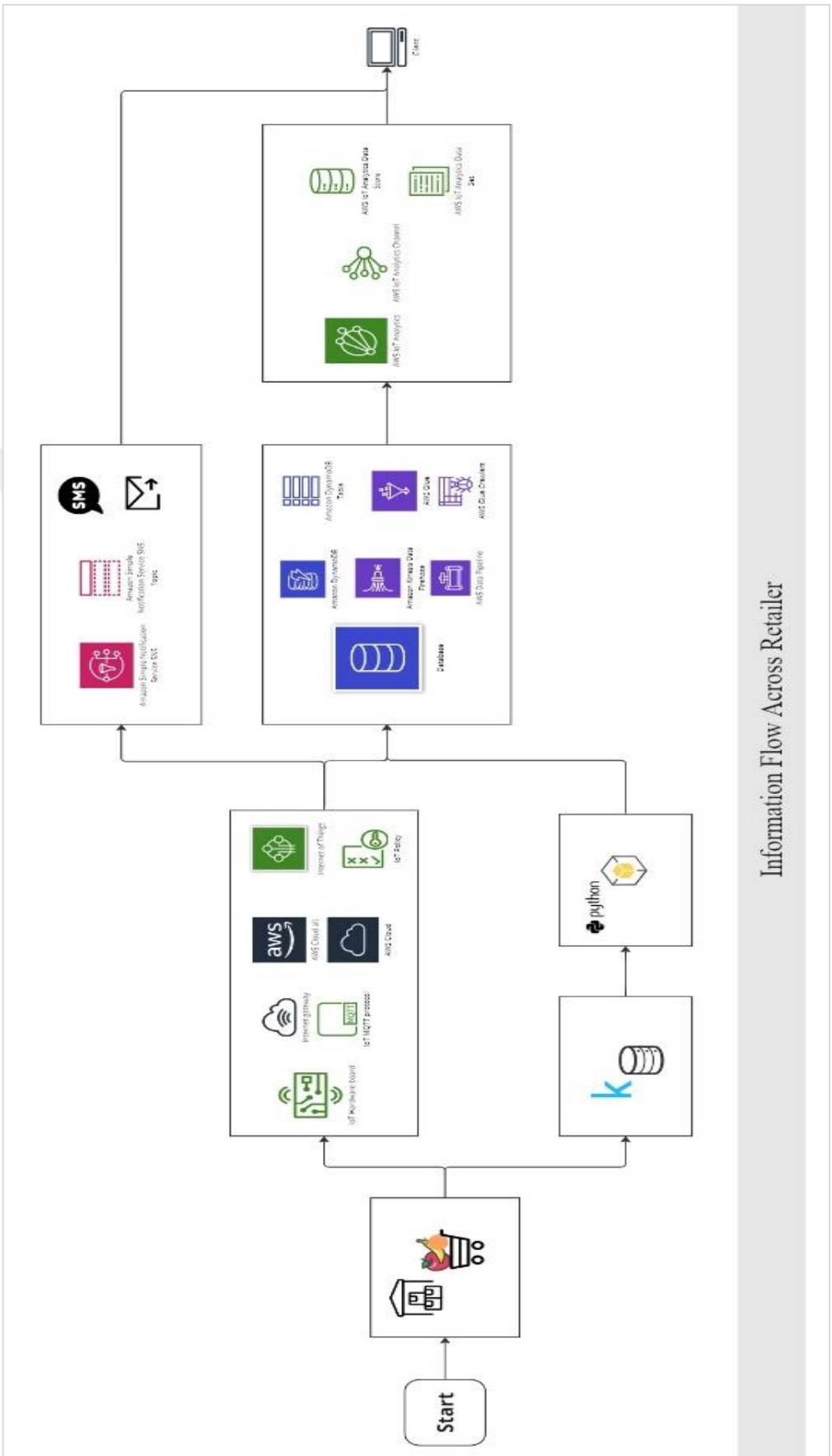


Figure 3.2: Figure Showing Proposed Model.

The methodology used in this study for CNN is as following:

1. The fruit images are selected and resized to 100 x 100.
2. Then, BGR (Blue, Green, Red) images are converted into RGB (Red, Green, Blue).
3. The dataset is normalized by division to 255.
4. Class labels are pre-processed, and the model architecture is designed as:
   4.1 Kernel initializer (*layer weight initializer*) was selected as “he uniform”.
   4.2 Activation functions for convolutional and 2D-convolutional layers are selected as 'ReLU' function.
5. In the Pooling Layer, “Max Pooling” is selected.
6. Activation function for the fully connected layer is selected as sigmoid.
7. Model is taught with training set over 50 epochs.
8. The model is compiled with Adam optimizer and binary-cross entropy with learning rate = 0.001.
9. As the final step, model is tested with 50 epochs using testing set.

The details relating to the CNN-model, the size, and outputs of the indicated layers of this study is provided in Table 3.1. Moreover, Figure 3.3 and 3.5 shows the three-dimensional view of the CNN-model and the model flow diagram respectively.

3.1.1 Data Pre-processing for CNN

An RGB image is the result of three distinct computer matrices. It determines the color displayed by each pixel which is performed by the definition of the red component in the initial matrix. Followed by the green component in the second, and the blue component in the last one respectively. Thus, for a 3x3 pixel image, there are three independent 3x3 matrices. Every pixel in these matrices that are sent into the network must be evaluated. In this study, the data in our dataset was in initially BGR format. Therefore, the data was converted to RGB format then, normalized (converted to grayscale) by the division by 255 to have a value between (0-1).

Table 3.1: Details of the Binary CNN Model Architecture.

Layer Type	Size	Output Shape
Input	(100,100,3)	-
Convolution 2D + ReLU	32 × (3, 3) filter	(100, 100, 32)
Separable Convolution 2D + ReLU	32 × (3, 3) filter	(100, 100, 32)
Max Pooling + Dropout	(2, 2) filter	(50, 50, 32)
Separable Convolution 2D + ReLU	64 × (3, 3) filter	(50, 50, 64)
Separable Convolution 2D + ReLU	64 × (3, 3) filter	(50, 50, 64)
Max Pooling + Dropout	(2, 2) filter	(25, 25, 64)
Convolution 2D	128 × (3, 3) filter	(25, 25, 128)
Convolution 2D	128 × (3, 3) filter	(25, 25, 128)
Max Pooling 2D + Dropout	512 × (3, 3) filter	(12, 12, 128)
Hidden (ReLU + Dropout)	9437696 Hidden Units	512
Hidden (ReLU + Dropout)	65664 Hidden Units	128
Sigmoid	1 ways	1

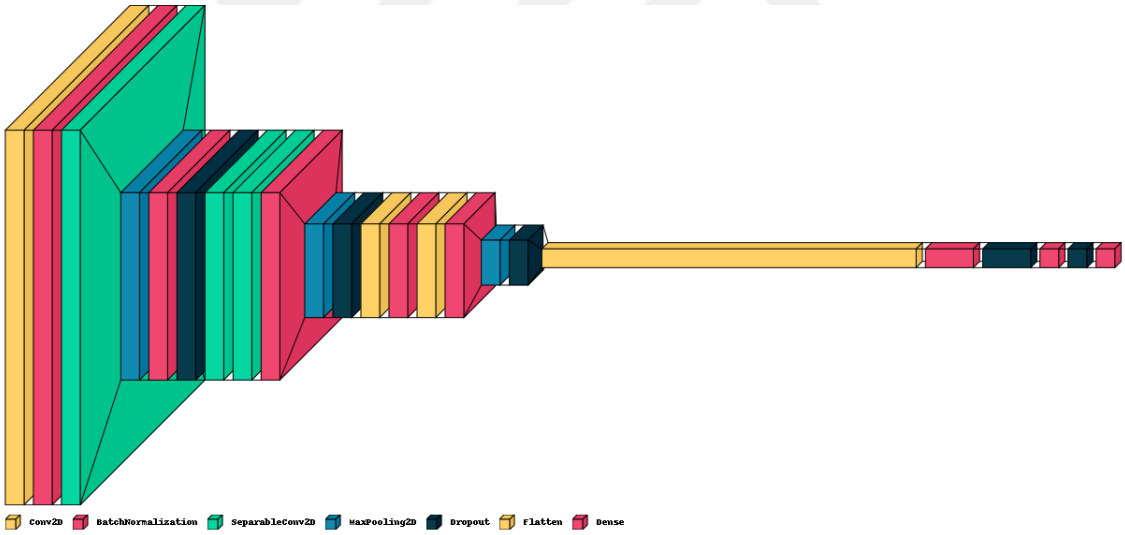


Figure 3.3: Binary CNN Layer Structure of the Model.

Table 3.2: Details of the Multiclass Classification CNN Model Architecture.

Layer Type	Size	Output Shape
Input	(32,32,3)	-
Convolution 2D + ReLU	32 × (3, 3) filter	(32, 32, 32)
Separable Convolution 2D + ReLU	32 × (3, 3) filter	(32, 32, 32)
Max Pooling + Dropout	(2, 2) filter	(16, 16, 32)
Separable Convolution 2D + ReLU	64 × (3, 3) filter	(16, 16, 64)
Separable Convolution 2D + ReLU	64 × (3, 3) filter	(16, 16, 64)
Max Pooling + Dropout	(2, 2) filter	(8, 8, 64)
Convolution 2D	128 × (3, 3) filter	(8, 8, 128)
Convolution 2D	128 × (3, 3) filter	(8, 8, 128)
Max Pooling 2D + Dropout	(2, 2) filter	(4, 4, 128)
Hidden (ReLU + Dropout)	1049088 Hidden Units	512
Hidden (ReLU + Dropout)	65664 Hidden Units	128
Sigmoid	6 ways	-

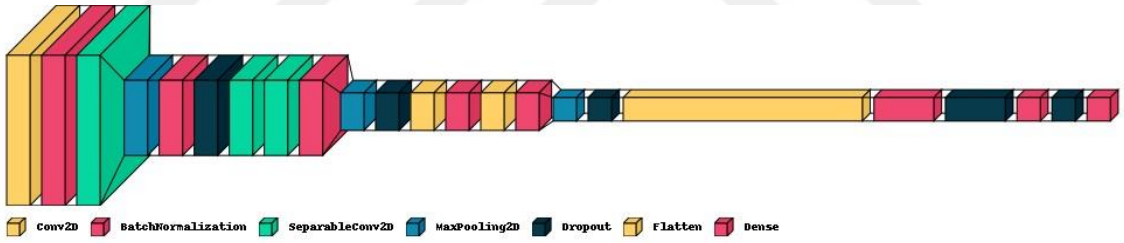


Figure 3.4: Multiclass CNN Layer Structure of the Model.

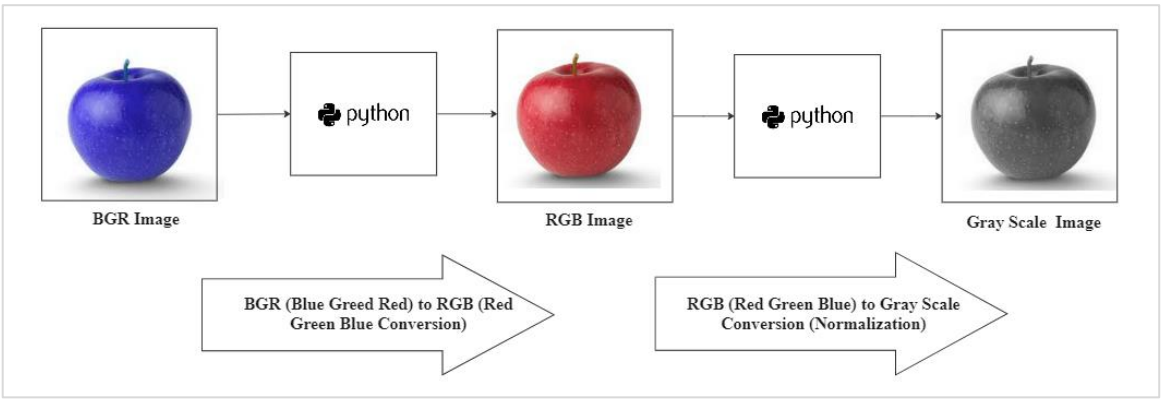


Figure 3.5: BGR to Gray Scale Image Conversion

3.1.2 Layer weight initializer: He Uniform Class

Weights are critical to initialize because they reflect the strength of connections among neurons in the network and characterize the impact of all neurons on the final result. Therefore, selection of the correct method to aasign weights is fundamental, since weight initializers govern the way that the CNN layers' starting random weights are allocated. Regarding this manner, "he" initialization is currently the preferred approach for initializing the weights of neural network layers and nodes using the rectified linear (ReLU) activation function, and used in this study, since it converges faster with higher level accuracy, there occurs a constant increase in accuracy with each epochs and it is very stable. The he initialization technique is computed as a random number using a Gaussian probability distribution (G) and a standard deviation of square root of (2/n), where n is the number of inputs to the node. He Uniform Class obtains samples from a uniform distribution within [-limit, limit], where:

$$Limit = \sqrt{\frac{6}{n_{input\ units\ in\ the\ weight\ tensor}}} \quad (1)$$

3.1.3 Activation Functions

The activation function works as a mathematical gate between an input of a certain neuron and its output into the subsequent layer, indicating if the neuron should be activated.

- **Rectified Linear Unit (ReLU) Activation Function**

The ReLU function returns the original value when the input number is zero or greater. If the input is less than zero, the ReLU function outputs zero. The ReLU function is a nonlinear function with one linear component having a constant derivative and the other having a zero derivative.

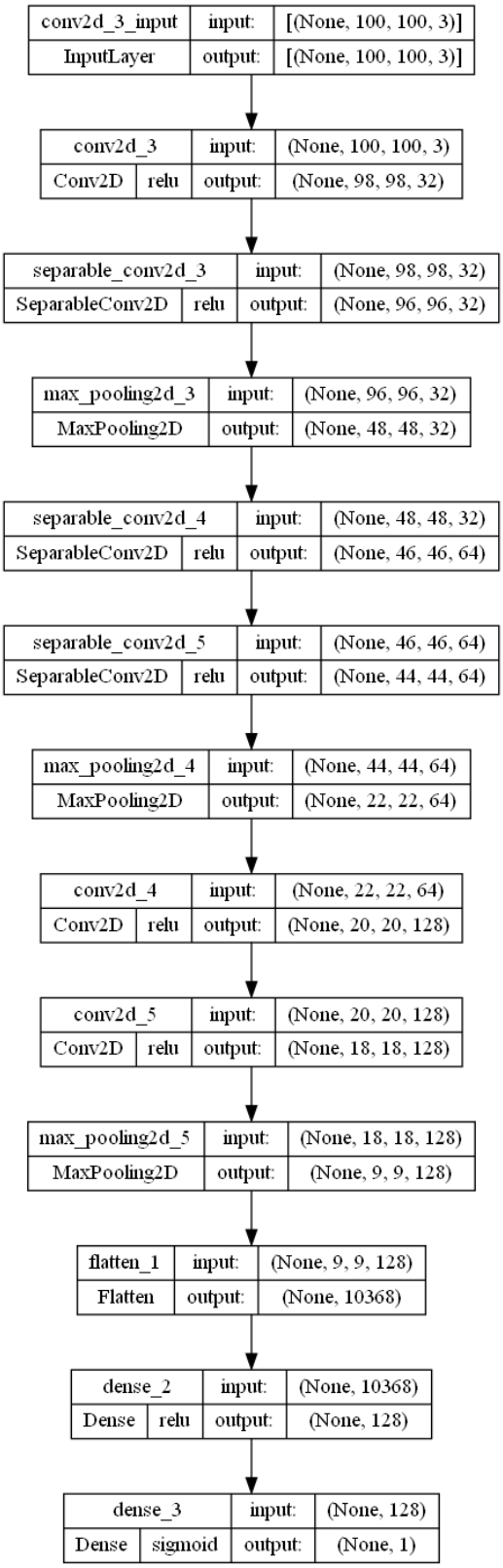


Figure 3.6: Figure Showing CNN Layer Details of the Model

As a consequence, it converges rapidly than the sigmoid and tanh functions, resulting in a 6-fold faster learning process and greater processing efficiency (Chou et al., 2017). Therefore, in current CNN models, the ReLU function, shown in Figure 3.7, is the default activation function for hidden layers. If the input value is zero or greater, the ReLU function outputs the original value. The ReLU function returns zero if the input is less than zero. It is a piecewise linear function since it is composed of two linear components. The ReLU function is, in fact, a non-linear function with a fixed derivative for one linear component and a zero derivative for the other. Therefore, it converges faster than sigmoid and tanh functions, resulting in a faster learning process and computationally efficiency.

$$ReLU\ Function: \qquad \sigma(z)_i = \max(0, \mathrm{x}) \qquad (2)$$

- **Sigmoid Activation Function**

A sigmoid function, formulated in equation 3 and plotted in Figure 3.8, is a bounded, differentiable, real function that is defined for all real input values and has a non-negative derivative at each point. The sigmoid function converts a continuous real number to an interval of (0,1), allowing the subsequent layer's input value to be between the specified range and the weight to be more robust (Chen et al., 2023). In CNNs, sigmoid function is generally used for multi-class multi class classification tasks.

$$Sigmoid\ Function: \qquad \sigma(z)_i = \frac{1}{1 + e^{-x}} \qquad (3)$$

- **SoftMax Activation Function**

The Softmax function computes the probability value of an event (class) over K different events. The softmax function turns a vector of K real values that equal to one. The function always returns a value between 0 and 1, which can be utilized as a the probability score. The input can be either positive or negative, but the output can only be between 0 and 1 (Dwiwedi et al., 2023). Since the sum of all probability is one, all events are

mutually exclusive. The SoftMax function is formulated in equation 4 and plotted in Figure 3.9.

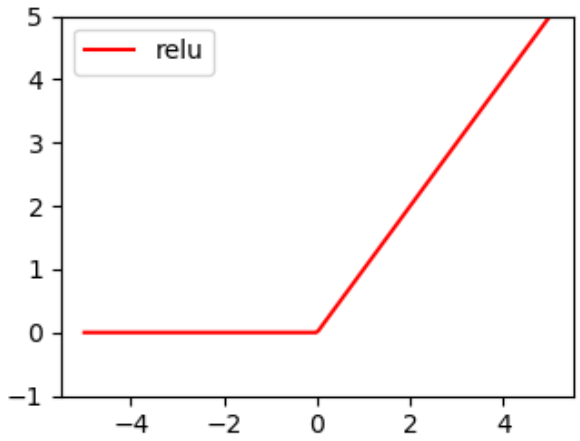


Figure 3.7: Graph Showing ReLU Function.

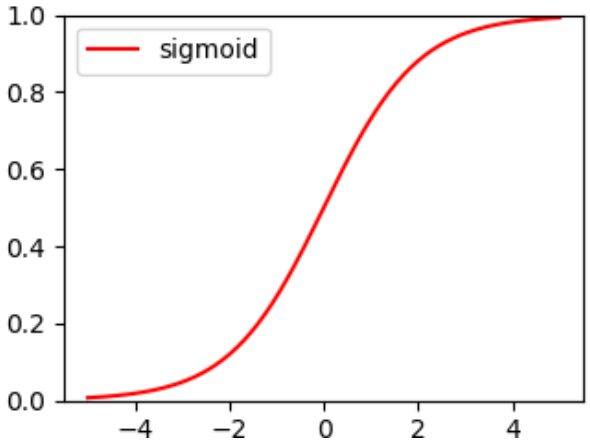


Figure 3.8: Graph Showing Sigmoid Function.

$$Softmax\ Function: \quad \sigma(z)_i = \frac{e^{z_i}}{\sum_{j=i}^{K} e^{z_j}} \qquad (4)$$

Task	Last layer activation function
Binary classification	Sigmoid
Multiclass single-class classification	Softmax
Multiclass multiclass classification	Sigmoid
Regression to continuous values	Identity

Figure 3.9: Commonly applied last layer activation functions for various tasks (Yamashita et. al., 2018)

- **Adam Optimization Function**

During training, optimizers are employed to reduce the loss function. Adam is an optimization method based on AdaGrad and RMSProp that produces the highest precise outcomes on the training set (Tieleman & Hinton, 2012; Graves, 2013). Adam optimization is a stochastic gradient descent approach regarding adaptive estimate of 1st and 2nd order moments. Compared to other optimization algorithms, this technique is cost effective, requires low memory, robust to diagonal rescaling of gradients, and appropriate for cases involving massive data or high number of parameters (Kingma et al., 2014).

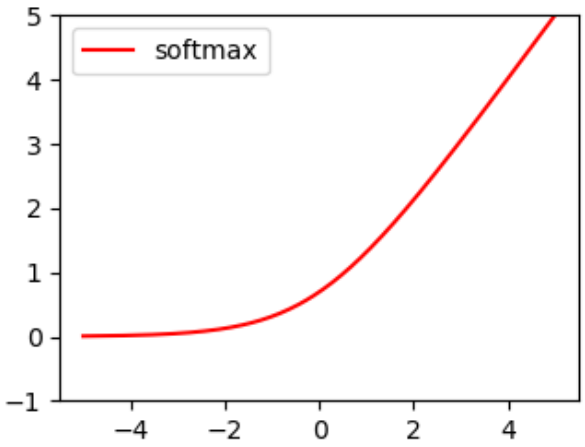


Figure 3.10: Graph Showing SoftMax Function.

Figure 3.9 shows the decrease in the value of cost functions of Adam Optimizer on training set. As it could be observed from the figure, compared to other optimizers Adam Method provides the lowest cost on the same dataset. The pseudocode for the Adam optimizer is as following:

while *w(t) not converged* **do:**

$$(1)\ t = t + 1$$

$$(2)\ m_t = \beta_1 \times m_t + (1 - \beta_1) \times \frac{\delta L}{\delta w_t}$$

$$(3)\ v_t = \beta_1 \times v_t + (1 - \beta_2) \times \left(\frac{\delta L}{\delta w_t}\right)^2$$

$$(4)\ \widehat{m_t} = m_t \div (1 - \beta_1^t)$$

$$(5)\ \widehat{v_t} = v_t \div (1 - \beta_2^t)$$

$$(6)\ w_t = w(t-1) - \alpha * \left(\frac{m_t}{\sqrt{v_t}} + e\right)$$

end

return *w(t)*

- **Learning rate**

During optimization, the algorithm must descend the error in a sequence of steps. Each steps have two properties: size and direction. The gradient (derivative) determines the direction of the step. To determine the direction, compute the derivative of the error regarding the weight values. The learning rate determines the step size that controls how quickly or slowly the optimizer descends the error curve. Common learning rates are 0.1, 0.01, and 0.001. These numbers are "default" recommendations for an effective learning rate (Hinz et. al., 2018). By using these default values as their starting point researchers proven that for all resolutions, the optimal learning rate is to be a value between 0.01 and 0.1 (Hinz et al., 2018). Therefore, when using stochastic gradient decent, a lower learning rate for the convolution layers is frequently employed in practice (Kingma and Ba, 2015). Regarding this, in this thesis, 0.1 is selected as the initial value of the learning rate and lowered via multiplying with 0.1 in every 5 epochs unless the training accuracy is increased, and the training loss is decreased.

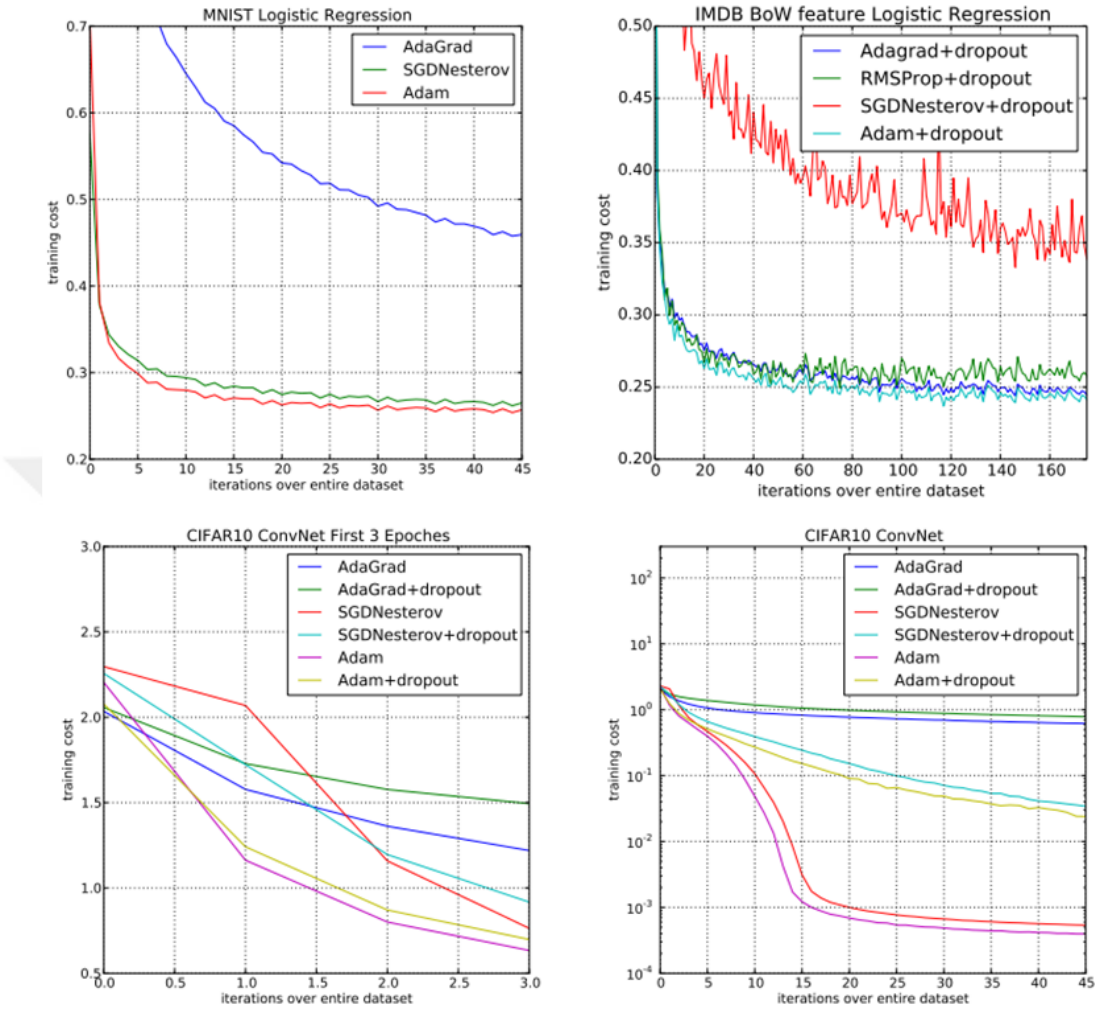


Figure 3.11: Figure Showing Efficiency of the Adam Optimizer (Kingma and Ba, 2015).

- **Number of Layers**

Number of layers affect the fitting status of the dataset. If there are many layers in the model then, the model is likely to overfit. On the other hand, when there are less number of layers the model do not learn on the trainset and underfit the dataset. Therefore, the number of hidden layers must be kept constant across all input sizes (Hinz et al., 2018). Moreover, the total number of convolutional layers should be smaller for an input with smaller pixel inputs and bigger for an input with bigger pixel size.

For our model, three convolutional, two separable convolutional and two hidden layers along with an input and output layer to yield the best level of fitting to the dataset.

- **Batch Normalization**

This method is required to add regularization effects while also accelerating the learning process. Batch normalization is accomplished by completing normalization procedures in each iteration, ensuring that the learning process takes into account the means and variances of each layer's inputs. Batch normalization restricts the extent to which changing parameters in previous layers can influence the distribution of data seen by the current layer. This is because the algorithm uses the mean and variance rather than the actual numbers (Rahman et al., 2020). Within our study, both after the convolutional and pooling layer batch normalization is applied.

- **Regularization**

Regularization is a strategy for reducing model error through preventing overfitting while also training the model to function accurately. Common regularization methods include determination of dropout rates, lambda 1 and lambda 2 regularization. For this thesis dropout is selected as the regularization method to prevent dataset from the overfitting.

- **Dropout**

Dropout regularization is accomplished by eliminating nodes from the network at random during training (Srivastava et al., 2014). Random weights are selected randomly to be discarded in each training iteration in this method (Hinton et al., 2012b; Fathi & Shoja, 2018). It is suggested to use 0.5 as the dropout rate of final hidden layers and lower rates for the preceding layers. Regarding this fact, dropout ranges selected for the model vary between the range of (0.3-0.4-0.5).

3.1.4 The Input Layer

For image processing tasks, input layer most commonly represents the pixel matrix having values varying between 0-255 of the image in neural networks. Therefore, in this project, for both of the classification models, the input layer is composed of the pixel matrices of the images of fresh and rotten apple, orange, and banana fruits.

3.1.5 The Convolutional Layer

The convolutional layer is the initial layer used for extracting different characteristics of the inputted images. In this layer, the image is divided into smaller sub-images. Convolution operation is performed in this layer, among the input image and a filter of size "M x M*"*. The dot product of the filter and the sections of the input image regarding the filter size is calculated by sliding the filter across the input image and feature map is outputted. The feature map has the information regarding the specific features of the image. It is then, forwarded to next layers to derive other characteristics from the inputted image. Therefore, the filter of the convolutional layer sets the dimension and spacing among the sub-images. As a result, the resolution of the image is diminished.

- **Conv2d Class for the Convolutional Layer**

This layer generates a convolution kernel, that is convolved with the layer input to for the generation of a tensor of outputs. Parameters specific to CNNs as padding, and stride, along with other common network features as filter, and input sizes all influence the output sizes of images in a CNN.

- **Number of Filters**

Number of layers affect the fitting property of data to our model. With fewer hidden layers, network configuration is simpler that the model could underfit the training dataset. As a result, complicated patterns are not learned. In contrary, when model contains many hidden layers, the network becomes larger and thus, overfits the training

data. In this case, instead of learning patterns, network attempts to recall the training data. As a result, it fails to apply well to previously unseen (test) data. Therefore, the total number of filters per convolutional layer should be kept small at the very beginning of the research and could range between 10, 20, 30, 40, and 50 (Hinz et al., 2018). Regarding these, we selected 32 as the initial number of the filters since model detects only the sharp points and edges in the initial layers and continued increasing the number by multiplying with two for the detection of the deeper patterns within the image.

- **Batch Size**

The batch size specifies the total quantity of samples that will be processed and sent across the network. Breuel (2015) discovered that smaller batches tend to perform better and as a result of his study, he suggested starting with relatively small batch sizes (i.e. 10, 20, 30, 40, 50) Moreover, it is suggested that with a rise in input size, the batch size should be reduced (i.e. from 40-60 to 20-30). In our thesis, since our input image sizes are selected as 32x32 and 100x100 which are considered as relatively smaller sizes, we experimented with batch sizes 16 and 32 to determine the one provides the optimal results.

- **Epoch Number**

Epochs are used to train neural networks. An epoch is a complete iteration of the training dataset. Each epoch comprises of one forward along with a backpropagation passing over all of the training data provided. In literature, most CNN models are trained via maximum of 50 epochs (Hinz et. al., 2018). Regarding this our model is tested over 50 epoches as well.

- **Kernel Size**

In literature, the filter sizes are generally set to be the odd values between three and seven due to the fact that feature matrix should be symmetric, and have the results of the summation resulted from the kernel and input matrix multiplication in its center.

Therefore, the identical quadratic filter dimension (i.e. 3x3, 5x5, or 7x7) is used to initialize all filters on a specific convolutional layer. These values have been stated in the literature often (such as Cox and Pinto, 2011; and Khorrami et al., 2015). Larger filter sizes, typically greater than 200x200 pixels, should be assigned to larger input images. For all inputs, the filter size of 3x3 generated the best results (Hinz et al., 2018). In the models used in this study (3x3) is selected as the filter size, since the input images are in the size of 32x32 for the multiclass classification and 100x100 for the binary classification respectively.

- **Padding & Stride**

Padding is a technique that adds additional pixels to the boundaries of input images to retain the output picture size identical to the input image. The most common way of padding is to add zeros, which is also named as the “zero padding”. Another technique used in CNNs is the “stride”. Stride is the step size for each iteration of moving an image with a convolutional window. In terms of this work, zero padding is utilized to match the size of the filters with the inputs and the stride is selected 1 which indicates that each value of the pixel matrix is evaluated.

3.1.6 SeparableConv2D Layer

Separable convolutions are formed by first performing a depth wise spatial convolution that functions independently on each input channel followed by a pointwise convolution that combines the resulting output channels. The depth multiplier argument in the depth wise step controls the number of output channels created for each input channel. Separable convolutions are simply considered as an approach to partition a convolution kernel into a pair of smaller kernels.

3.1.7 The Pooling Layer

Between the convolutional and the fully connected layer, there exists the pooling layer. The pooling layer is mainly used to reduce the size of the convolved feature map for reducing cost. This is obtained by diminishing the links among layers and working on

every feature map. The processes are identical as those in the convolution layer, with the exception that the mean or maximum value of the output is taken dependent on the application. This method is useful for preserving features in pixels that are necessary for completing the task at hand. The pooling layer also removes image noise and features within the image which are not relevant to the categorization process.

- **Max Pooling**

The most common techniques for pooling are the “Max” and “Average” pooling techniques. The appropriate technique between them is chosen based on the use case of the classification task. Max pooling is a pooling procedure which computes the maximum value for patches in a feature map and utilizes it to produce a pooled feature map. In this study, max pooling technique was preferred since it performs faster, yields a lower cost and extracts only the most salient features of the data compared to average pooling. For the application, pooling filter size is selected as (2x2) for both models of our study.

3.1.8 The Fully Connected Layer

The fully connected layer brings together all the sub-images to identify the linkages and execute the categorization process. It generates a neuron for each sub-image entry and connects it to all neurons in the following layer which lowers the number of dimensions and hence fastens the training procedure. Lastly, the layer learns the bits of the image are needed to do the categorization task.

3.1.9 The Output Layer

In this layer, flattening task is performed to change the output into the number of classes requested by the network for the classification. Exemplary model showing the determination of the final class in the output layer is provided in Figure 3.10. For the binary classification model, the number of output nodes were defined as one to determine whether the fruit is fresh or rotten. For the multiclass classification model, the number output nodes were defined as six since the model includes six different classes.

3.2 Hardware Configuration

Hardware configuration of the prototype includes the connection and wiring process of DHT11 temperature and humidity, and MQ3 gas sensors to ESP8266 NodeMCU microcontroller. Output pin of the DHT11 temperature and humidity sensor is connected to D2 (GPIO4) analog node of the microcontroller, and the output pin of the MQ3 sensor is connected to A0 analog pin of the microcontroller. Both sensors require their V_{cc} to be connected to 3.3V and are grounded via the GND pins of the ESP8266 NodeMCU Board. The corresponding configuration is provided in Figure 3.12.

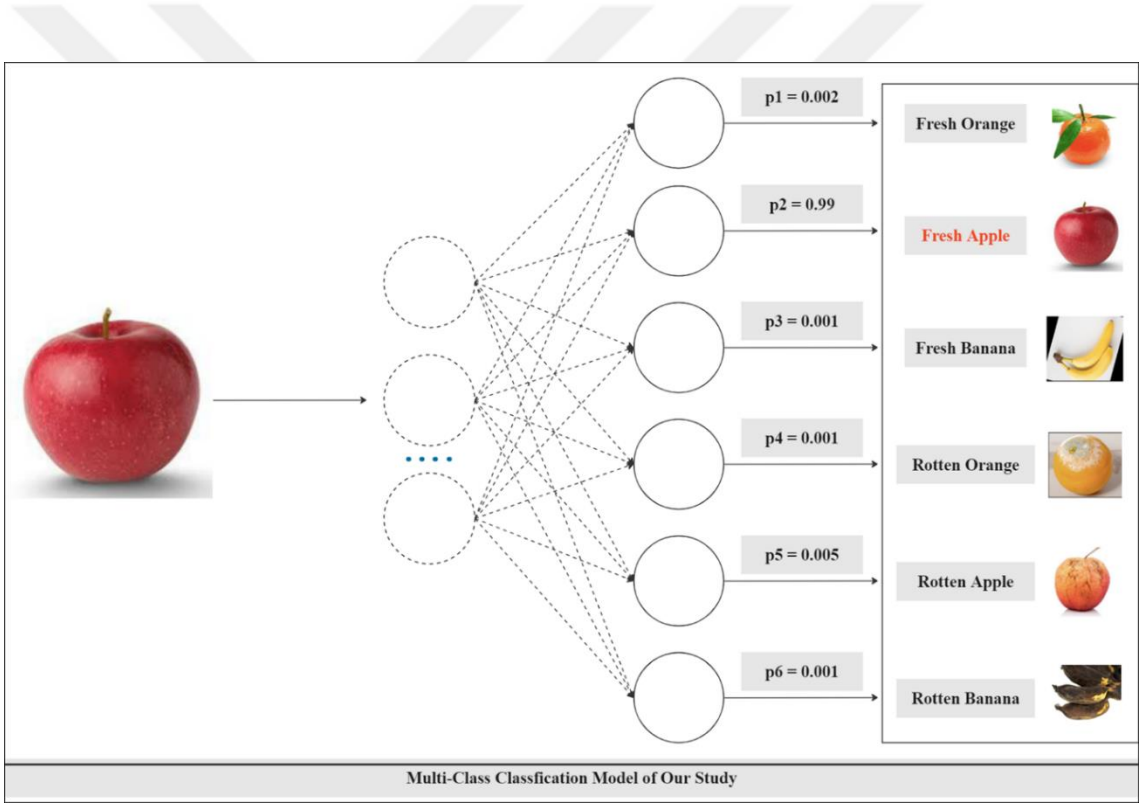


Figure 3.12 Figure Showing Output Layer of Our Model.

The methodology used in this study for hardware configuration is as following:

1. Analog output of the DHT11 temperature and humidity sensor is connected to D2 of ESP8266.
2. V_{cc} of the DHT11 temperature and humidity sensor is connected to 3.3V and the sensor is grounded via the GND pin of the ESP8266.

3. Analog output of the MQ3 alcohol sensor is connected to A0 of ESP8266.
4. V_{CC} of the MQ3 alcohol sensor is connected to 3.3V and the sensor is grounded via the GND pin of the ESP8266.

3.2.1 ESP8266 NodeMCU Microcontroller

The microcontroller utilized in this research is ESP8266 NodeMcu due to its ease of use, programming availability with Arduino IDE and for its lower cost compared to other commercial products in the market as ESP-32 or Raspberry Pi. It is a prominent and frequently used development board thanks to the ESP-12E WiFi (Wireless Fidelity) Module which integrates programming using the Arduino IDE, a C++ based platform, with WiFi capabilities.

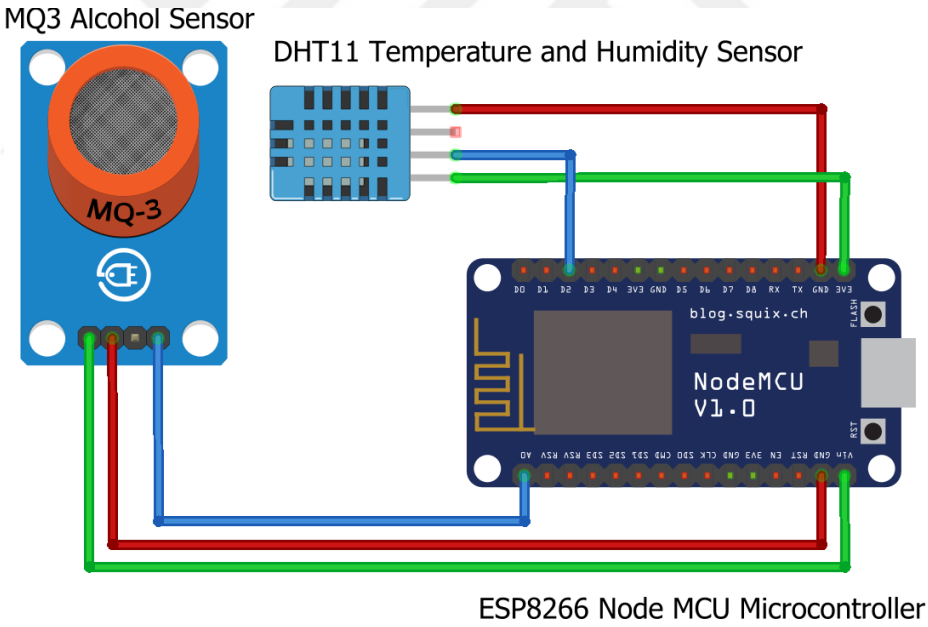


Figure 3.13: Hardware Configuration of the Study.

3.2.2 DHT11 Temperature and Humidity Sensor

The DHT 11 sensor delivers calibrated digital output signals. It is extensively used in applications primarily because of its advantages of being reliable and stable throughout long periods of operation. It has an 8-bit microcontroller inside and responds promptly and clearly. It measures the ambience temperature between 0 and 50°C with a ±2°C error margin.

Figure 3.14: ESP8266 NodeMCU – CP2102 - 12E

For the humidity parameter, the sensor measures ambiance RH between 20% - 90% with a ±5% error margin. The sensor configuration contains three pins indicated in Figure 3.12 are: "GND", "VCC", and "S" for grounding the inner circuit, voltage input, and digital signal output, respectively.

Table 3.3 DHT11 Measurement Specifications.

Parameter	Measurement Range	Error Margin	Unit
Temperature	0 - 50	±2	℃
Humidity	20 - 90	±5	%RH

As its working principle, DHT 11 sensor utilizes a capacitive humidity sensor and a thermistor for measuring the ambient air and outputs a digital signal on the data pin without the requirement of an additional analog input pins. The thermistor inside the sensor is designed so that its resistance changes dramatically with temperature. In addition, DHT11 sensor has a sampling rate of 1Hz, indicating that the sensor provides new data once every second (1 data package/sec). Inside the DHT11 sensor, there is an NTC (Negative Temperature Coefficient) thermistor and a humidity sensor.

The humidity sensing part has of two electrodes aparted by a moisture-holding substrate (typically made up of a type of salt or conductive plastic polymer). As humidity increases, the substrate absorbs water vapor, causing the discharge of ions and the drop in resistance between the two electrodes. The variation in resistance is proportional to the humidity and can be measured for estimating relative humidity. The 8-bit integrated circuit (IC) in the sensor measures and analyzes the analog signal utilizing preset validation parameters, transforms the analog signal to digital, and generates a digital signal with the temperature and humidity.

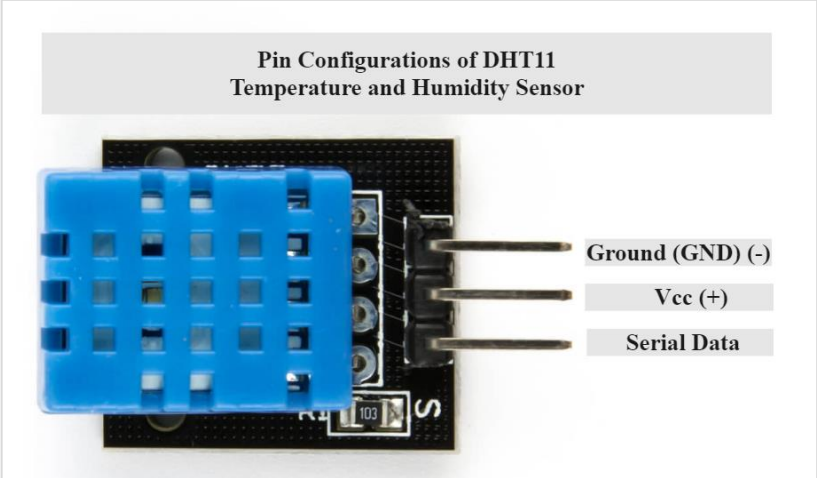


Figure 3.15: DHT11 – Temperature & Humidity Sensor

3.2.3 MQ3 Gas Sensor

MQ3 gas sensor is a widely used MOS (Metal Oxide Semiconductor) sensor for the alcohol detection that also known as chemiresistors since they detect the variation in resistance of the material under exposure to chemical as alcohol.

The MQ3 is a heater-driven sensor. Therefore, it is covered by two layers of fine stainless steel to protect the device from the explosion. The sensor is also protected from suspended particles by filtering them out letting only gaseous constituents to pass through the chamber. Thus, it is appropriate for the detection of Alcohol, Benzine, Methane (CH_4), Ethylene (C_2H_4), Hexane (C_6H_{14}), LPG (Liquefied Petroleum Gas), CO (Carbon Monoxide). Moreover, since the sensor is highly sensitive and has a fast response time, it is appropriate for gas leakage detection for commercial uses.

The sensor works by measuring the concentration of Alcohol, Benzine, C_2H_4, CH_4, Hexane, LPG, CO composition of the ambient and outputs this reading as an analog voltage. Inside of it there is a sensing element and six connecting legs as indicated in Figure 3.16. Two (H) of the six leads detect heating and are linked via a nickel-chromium (Ni-Cr) alloy coil. Platinum (Pt) wires link the rest of the four signal-carrying leads (A and B). These wires are attached to transmit minor fluctuations in the current flowing across the sensing element.

Figure 3.16: MQ3 Gas Sensor

The tubular sensor element is comprised of an aluminum oxide (Al_2O_3) ceramic with a tin dioxide (SnO_2) covering. Because it is sensitive to alcohol, SnO_2 is the most significant substance. In contrast, the ceramic substrate enhances heating efficiency and ensures that the sensor area is heated to the operating temperature. In conclusion, the heating system consists of a Ni-Cr coil and an Al_2O_3 based ceramic, whilst the sensing system consists of Pt wires with a SnO_2 covering. The working procedure of the sensor is as following: O_2 is absorbed on the surface of a SnO_2 semiconductor layer when it is subjected to an elevated temperature. Oxygen molecules attract electrons from the SnO_2 conduction band when the air is pure. In this case, SnO_2 particles form an electron depletion layer near the surface. As a result, the SnO_2 layer becomes highly resistive and unable to conduct current. The surface density of O_2 absorption drops in the presence of alcohol when it reacts with the alcohol, diminishing the potential barrier. By releasing electrons into the SnO_2, the sensor permits current to flow freely.

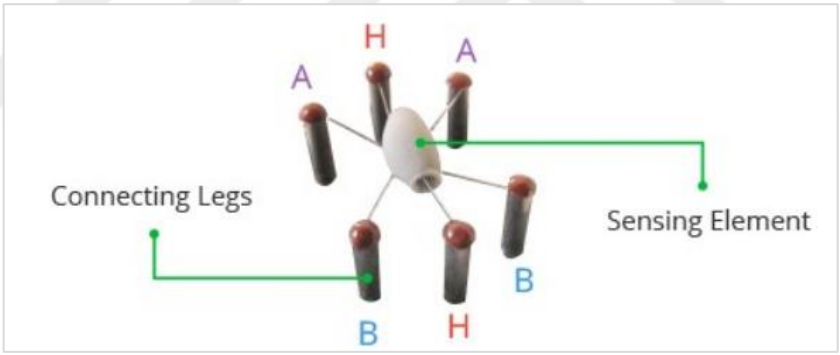


Figure 3.17: Inner Configuration of the MQ3 Gas Sensor

3.3 Software Configuration

3.3.1 Arduino IDE

To program Arduino boards and upload written programs, the Arduino Software (IDE) is used. It is one of the most popular software for IoT applications. Although all Arduino boards are compatible with the Arduino IDE, the microcontroller used in this study is ESP8266 NodeMCU as indicated previously. The main reason for the selection of the

Arduino IDE for the development is that the the controller is compatible with pre-defined libraries such for MQTT communication, AWS Cloud Platform subscription/publishing (PubSubClient Library) and the sensor libraries (DHT11 and MQ3 Sensor Libraries). In addition, Arduino IDE provides fast and secure connection with cloud platforms and can transmit JSON packages (ArduinoJSON Library). In this study, the project code consists of a main file ("main.ino") and a header file ("credentials.h"). The header file includes private information to be secured by the industry as Wi-Fi password, device name, MQTT host (AWS IoT endpoint), device certificates (root CA1, certificate) and the private key. On the other hand, main file includes the programs to perform sensor operations, to convert readings to meaningful information and then, transmit the results to cloud system. Appendix A contains the complete Arduino code of the study.

3.3.2 Cloud Architecture

The cloud platform is a form of digital infrastructure that allows remote computers and processing engines to be accessed via the Internet using a web browser. Large clouds often feature functions distributed across numerous locations, each of which constitutes a data center. Today cloud platforms are widely used by different businesses for data storage and processing. Therefore, there are various commercial cloud providers including AWS Cloud, Microsoft Azure Cloud, Oracle Cloud, and Google Cloud. For this study, the implemented cloud architecture is based on the AWS Cloud due its robust applications to be deployed and in terms of cost aspect.

The methodology used in this study for the cloud configuration is as following:

1. AWS region is selected.
2. A Thing named ESP8266 is created on the AWS IoT Core Platform.
3. Device certificate for the thing named ESP8266 was created.
4. MQTT policy containing connection, subscription, receive, and publish to thing topic was created and attached to the device certificate indicated in Step 3.
5. Rules regarding data storage S3 and DynamoDB databases were created.
6. Rules regarding e-mail and SMS updates of the users were defined.
7. Rules regarding sending data to IoT Analytics was defined.
8. MQTT test module was subscribed to ESP8266 NodeMCU microcontroller topic.

Via the ESP8266 Node MCU microcontroller, the data obtained from the DHT11 temperature and humidity sensor and the MQ3 alcohol sensor is sent to the AWS IoT Core Application, the first step of the AWS Cloud Architecture with the MQTT communication protocol. The microcontroller code is based on Arduino IDE 2023. A virtual microcontroller equivalent to the original microcontroller is defined on the cloud (Thing Definition) to display the data instantly with the IoT Core Application as in Figure 3.16. For data security, determination of the parties that can access the data and have the authority to modify it, a policy that provides subscription, receive, and publish authorizations to the cloud platform by creating a device certificate is prepared and placed in the certificate as in Figure 3.17. In our model, sensors' data is stored in S3 storage buckets via Kinesis Firehose Streamline for the SQL users and in the DynamoDB Database for the NoSQL users for the effect of acquaintance on the ease of analysis and both having different advantages.

The data accumulated in databases are then further analyzed in IoT Analytics platform for the purpose of enhanced monitoring and the visualization of the conditions in the retail store. In addition, to improve the user interaction with the system and to inform the users in case of rottenness, sensors' data are sent to SNS Push Notification Channel from IoT Core Application via the defined rules for message routing. By this way, system users are notified with an SMS and an e-mail including the values of the temperature, humidity, and gas level variables when the storage conditions, are out of the defined ranges.

3.3.3 AWS IoT Core Service

AWS IoT Core is a hosted cloud solution enabling connected devices to communicate with cloud apps and other devices in a simple and safe manner. Through its use, various devices and messages can be accommodated. The communications can be processed and routed reliably and securely to other devices and to AWS destinations. Moreover, it offers device software for integrating IoT devices into AWS IoT-based applications.

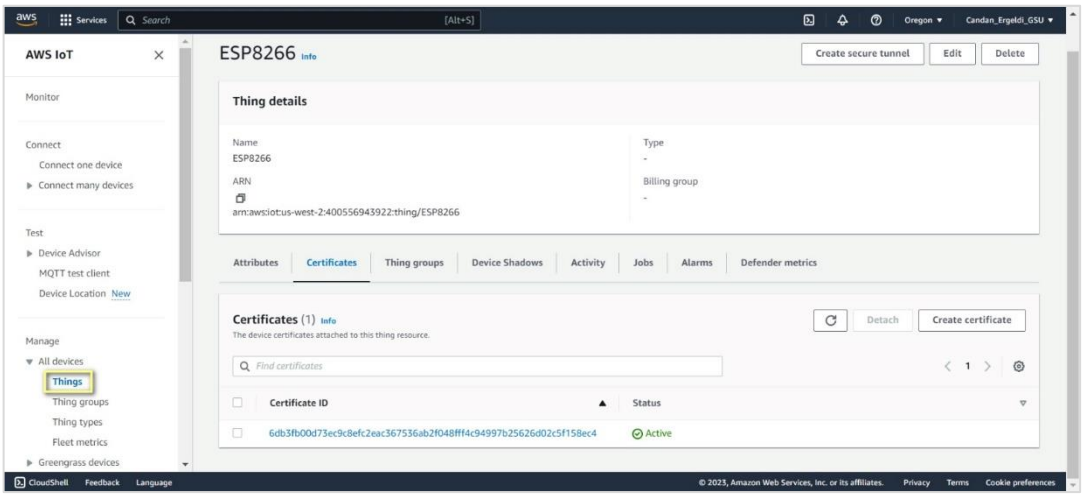


Figure 3.18: ESP8266 Thing on AWS IoT Core.

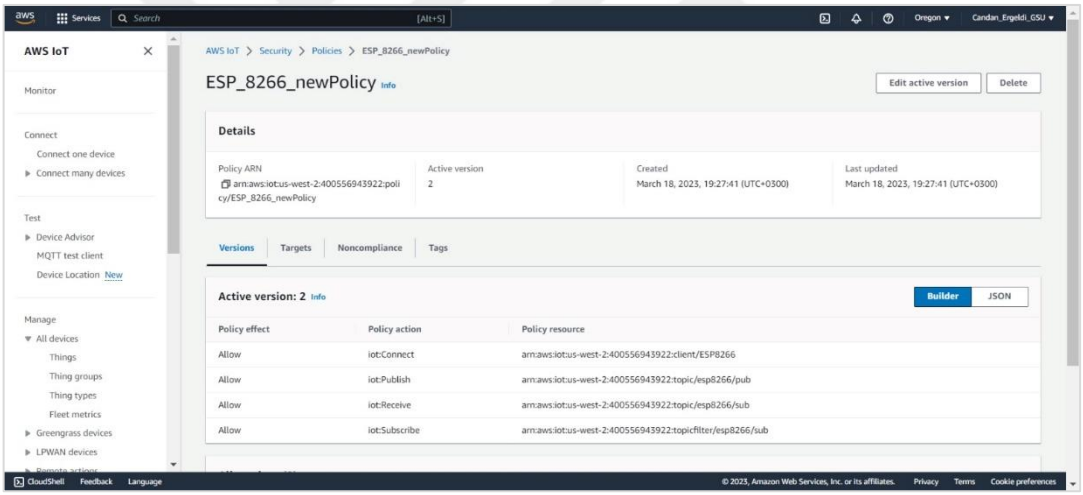


Figure 3.19: Policy Configuration.

The AWS IoT Core supports MQTT, MQTT over WSS (Web sockets Secure), HTTPS (Secure Hypertext Transfer Protocol) and Long-Range Wide Area Network (LoRaWAN) protocols. In this study, MQTT is utilized for the data transmission to AWS IoT Core Service.

The Core Service is comprised of manage, act, connect, secure and test layers. The manage layer facilitates the categorization of devices into logical groupings such as thing

groups and the act layer allows creation of rules for routing data from devices to cloud services. The connect layer assists in linking standalone gadgets and creates templates for connecting several devices. Finally, the secure layer manages the device authentication and authorization actions finally the test includes tools for testing MQTT communication and client devices. For each AWS IoT Core Application a “Thing” associated with the publisher device must be created.

When creating a “Thing” on the Cloud system, below application steps are followed:

I. Specification of thing properties
II. Configuration of the device certificate
III. Policy attachment to certificate

For this study, “ESP8266” thing was created on the AWS IoT Core Platform using the Manage - Things path as provided in Figure 3.16.

Then, device (ESP8266 NodeMCU) certificate containing the security credentials of the thing is attached to model. The private key and root certificate is saved for the further use in Arduino IDE publishing topic. The certificate attached includes a policy to define the actions between the device and the IoT Core. “ESP8266_Policy” attached in the device certificate (Figure 3.17), allows device to connect and to publish data to IoT Core. Along with the policy IoT Core is enabled to subscribe to MQTT topic and receive data from the microcontroller. Rule creation is one of the most important steps in the creation of the IoT Cloud Architecture of AWS. Rules allow the further processing of acquired data from the publisher. In this study, rules declared as indicated in Figure 3.18. The rules (Kinesis Firehose and DynamoDB) are for the sensor data storage, SNS (Simple Notification Service) push notifications via e-mail and SMS (Short Message Service) and IoT Analytics for transmitting data to analytics dashboard.

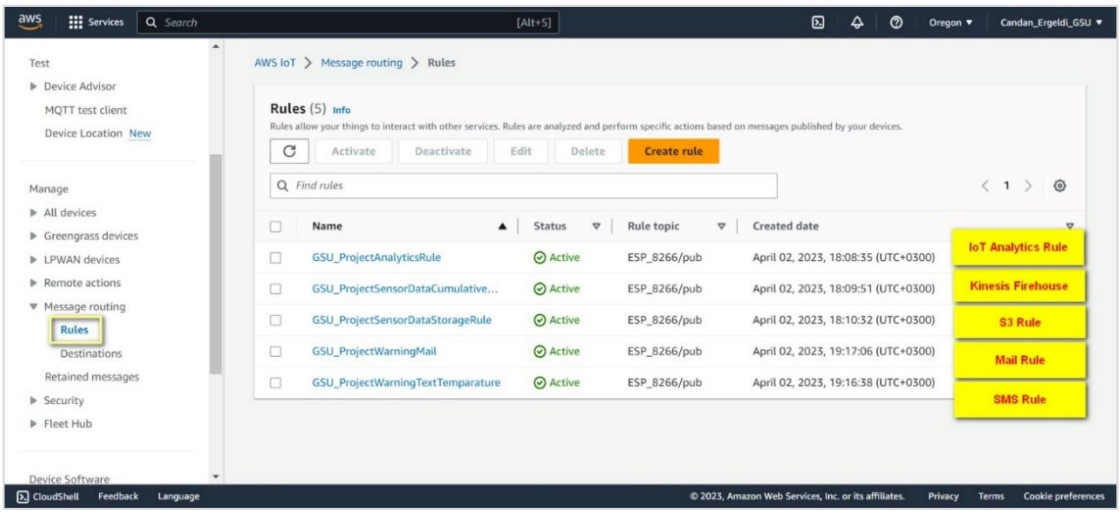
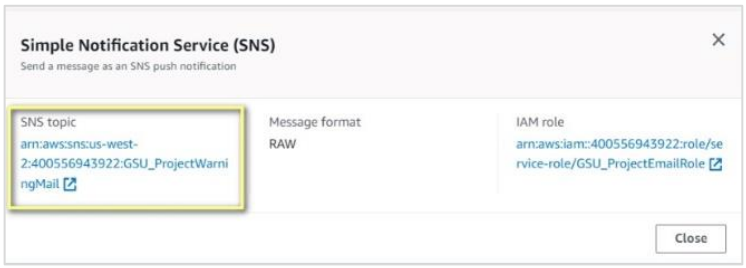


Figure 3.20: Rules defined in the AWS IoT Core.

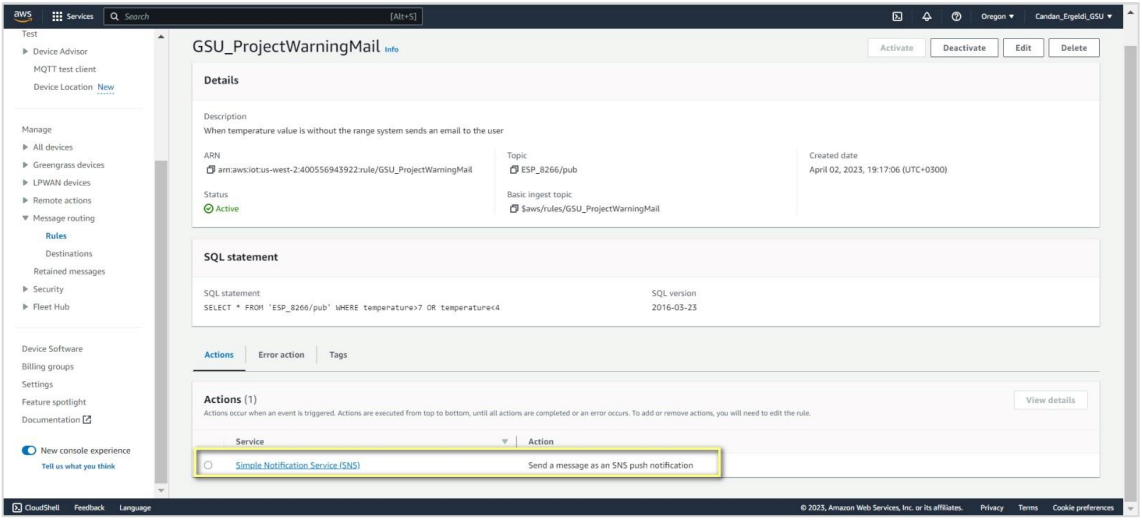
3.3.4 AWS SNS Push Notification Service

A2A (Application to Application) and A2P (Application to Person) notifications are sent through Amazon Simple Notification Service (SNS). A2A is a high-throughput, push-based, messaging protocol that connects dispersed systems, microservices, and event-based serverless apps. Amazon Simple Queue Service (SQS), Amazon Kinesis Data Firehose, AWS Lambda, and other HTTPS endpoints are among these applications.

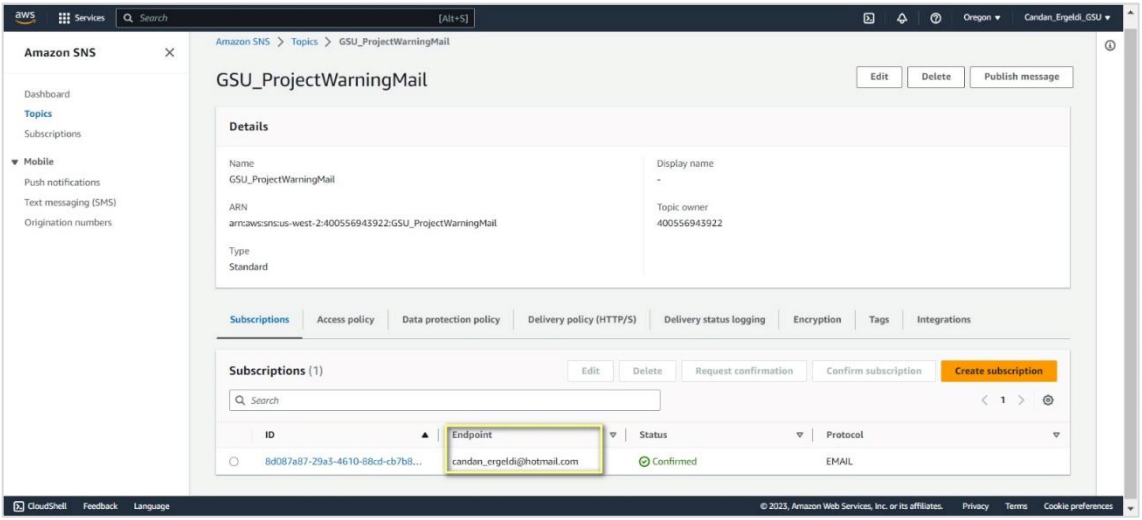
Clients can use A2P for sending messages via SMS texts, push notifications, and electronic mail (e-mail). In this study, A2P is deployed to send e-mail and text message notifications to client in case temperature, humidity or alcohol levels are out of their defined range as in Figure 3.19 and Figure 3.20. By this way, clients of the system are alerted when the parameter values are out of the pre-defined ranges for the fruit freshness indicating the potential decay of the fruit.



(a)



(b)



(c)

Figure 3.21: Figure Showing SNS Configuration.

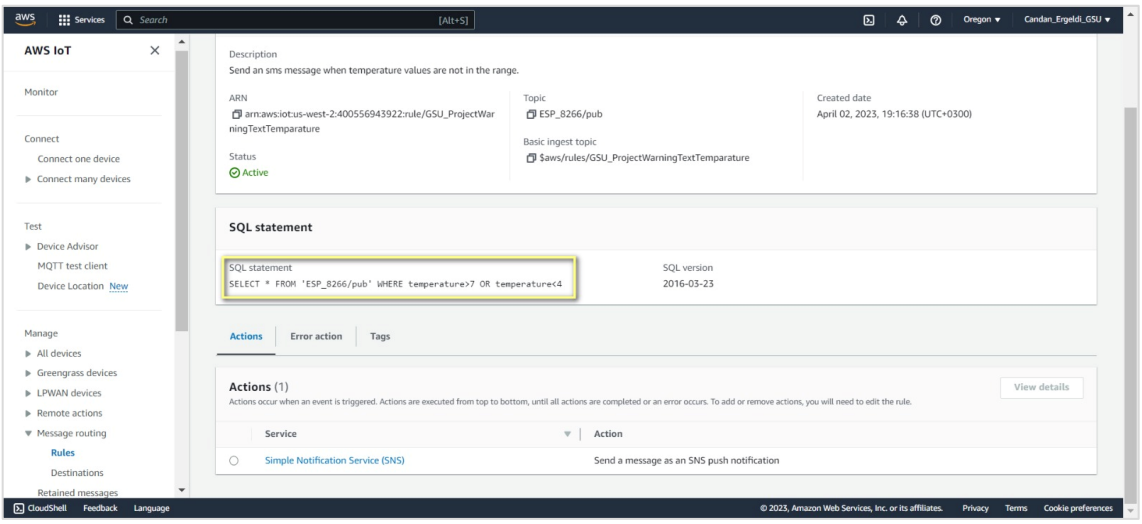


Figure 3.22: Figure Showing Rule Configuration.

3.3.5 MQTT Protocol

IoT communication at the application layer is generally deployed via MQTT, which uses transmission control protocol (TCP) as a base transfer layer protocol. Its TCP-based interface provides a more robust protocol, and its small MQTT header makes it compatible with IoT systems (Patel & Doshi, (2020). An MQTT Protocol involves three fundamental components that are: publisher, subscriber, and broker. Publisher receives the data from different sources as deployed sensors and publish them on a particular topic. The subscriber on the other hand, subscribe to the topic on which the publisher is publishing data. Lastly, broker obtains data from the publishers and transmit it to subscribers. Generally, brokers store data in cloud services, and the user accesses it via this service. In this study, publisher corresponds to the ESP8266 NodeMCU and the data sources are DHT11 and MQ3 sensors. The output data acquired from these sensors are published by the publisher agent ESP8266 NodeMCU to the MQTT broker topic subscribed by the AWS IoT Core subscriber. The MQTT Network of this study is provided in Figure 3.21. Quality of service (QoS) in MQTT facilitates communication among publisher and subscriber regarding data delivery approval. There are three forms of QoS for communication. The first one is the QoS-0. In QoS-0, data is communicated for once, at most. In the second type, QoS - 1, data is transmitted at least once and in the QoS - 2: just one data transmission occurs. The QoS provided in this study is QoS - 0.

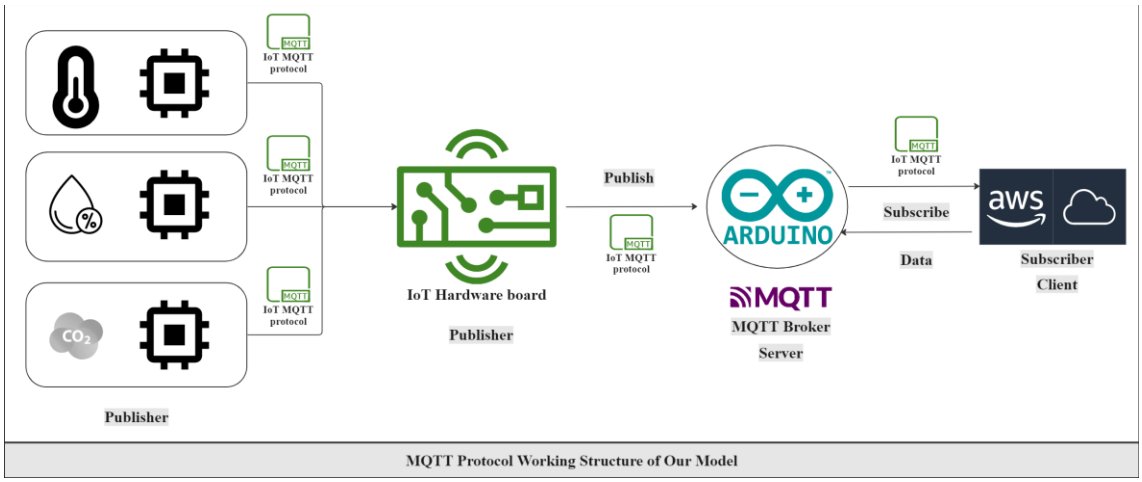
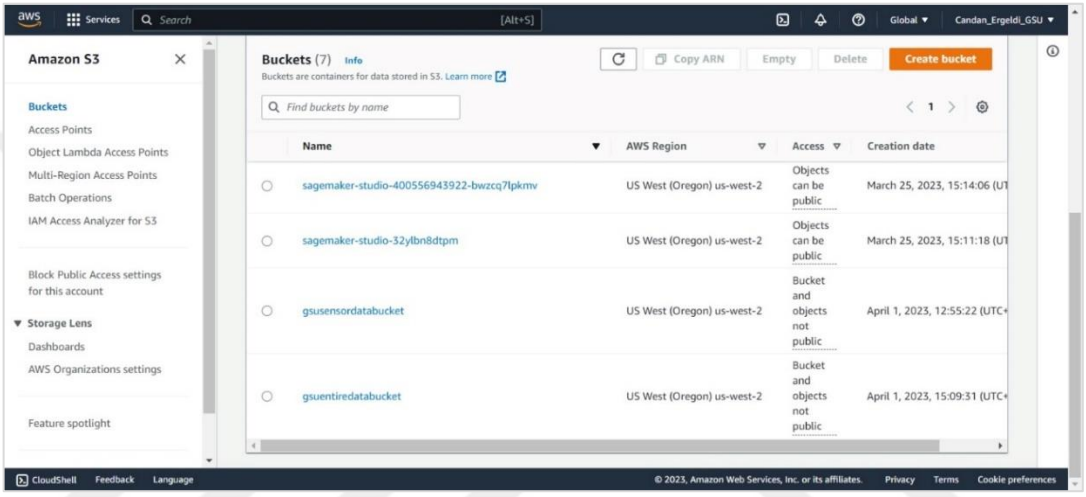


Figure 3.23: Figure Showing MQTT Structure of Our Model.

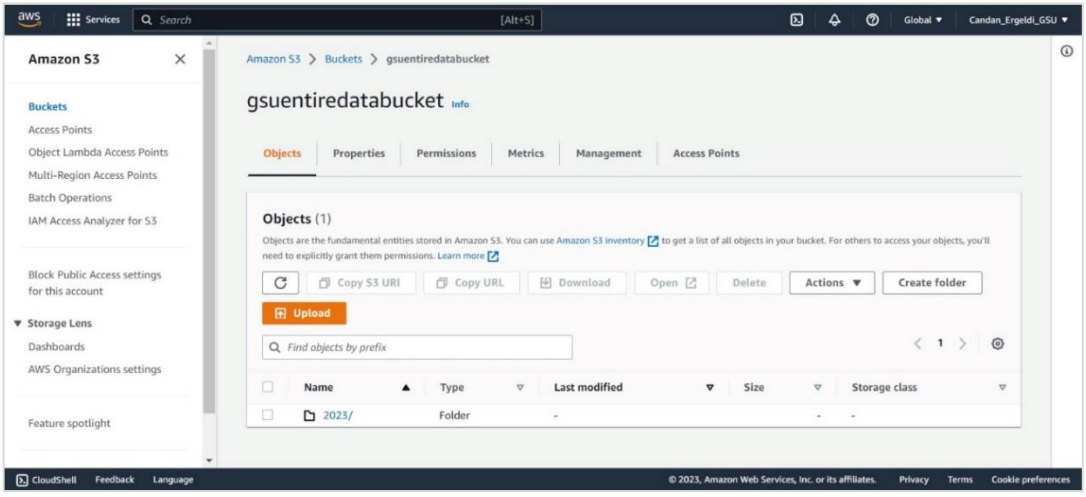
3.3.6 Model Databases

Cloud storage is one of the most prevalent and rapidly rising types of data storage as businesses seek methods to avoid the capital costs associated with having their own storage systems. It provides encrypted backup of the data for the business owners and eradicates the need of external hard drives. The ability to shift storage from on-premises systems to the cloud brings several possibilities for enterprises to simplify their storage. Since they could be updated and synced remotely it provides smart storage for a lifetime. Cloud computing solutions offer enterprises an efficient way to store and backup data which could also be used later for the BD analytics. This study proposes and employes two services for data storage, that are S3 Buckets and DynamoDB, namely. Amazon S3 is an SQL based object storage service that provides data security and DynamoDB is a NoSQL database server with seamless scaling and quick and predictable performance. The reason of deploying both SQL and NoSQL databases is regarding their different advantages. NoSQL databases employ many data models, such as graph, key value, document, and in-memory search. NoSQL databases have dynamic structure, whereas SQL databases have preset structure. In a SQL database, data is stored in tables compromised of rows, however with NoSQL, there are no standardized architecture requirements that data must follow.

In this study, DynamoDB is used to generate a database table that can store and retrieve temperature, humidity and alcohol levels and handle any degree of request traffic. The DynamoDB service dynamically distributes the table's data and traffic among an appropriate number of servers to handle the client's request capacity and the volume of data stored. In addition, a data stream in Kinesis Firehose is created to transmit sensor data to S3 Bucket via pipeline in real time.



(a)



(b)

Figure 3.24: Figure Showing Model Databases.

3.3.7 IoT Data Analytics

IoT Analytics ensures enhanced data analysis for IoT devices. AWS IoT Analytics is a time-series data storage that allows you to gather, manipulate, and store IoT data. The data can then be analyzed by performing queries. The IoT Data analytics is included in the proposed model since it would provide a better understanding of the sensor readings for the client user with its interface donated with different graphics.

4. RESULTS and DISCUSSION

The outcomes of this research have provided insight into the detection of the rotten crops. The temperature, humidity, and poisonous gas levels were measured by using our wireless sensor network (WSN) model. The measured levels were transmitted simultaneously to the cloud system via message queue telemetry transport (MQTT) protocol. The sensor readings were compared to the predefined values of the parameters based on the literature review to determine whether they fall within an acceptable range. For the measurements that do not lie within this confidence range, the client user was warned using SMS (Short Message Service) and electronic mail (e-mail) notification channels defined on the SNS Push Service of the Amazon Web Service's (AWS) Cloud Platform. In addition to the WSN deployment, a convolutional neural network (CNN) based image classification technique was utilized to improve the determination of fruit freshness process. Our study, in which both WSN and CNN combined, provided a robust and cost-efficient model for the fruit freshness determination that could be effectively used for the advancement of cold-storage conditions by agri-food supply chain (SC) shareholders. Regarding this issue, this section provides and discusses the results of the work done, that aims to contribute to the circularity of the agri-food SC with the improve in monitoring of cold-storage conditions.

4.1 Image Classification Dataset

Based on the research on perishable product datasets, we preferred to use the dataset from Kaggle[1] website named as "fruits: fresh and rotten for classification" in this study. This dataset includes images of high demand and commonly traded perishable fruits. It contains the following categories: fresh apples, fresh bananas, fresh oranges, rotten

[1] https://www.kaggle.com/datasets

apples, rotten bananas, and rotten oranges and has different folders to train and test images. The images present in the dataset were illustrated in Figure 4.1.

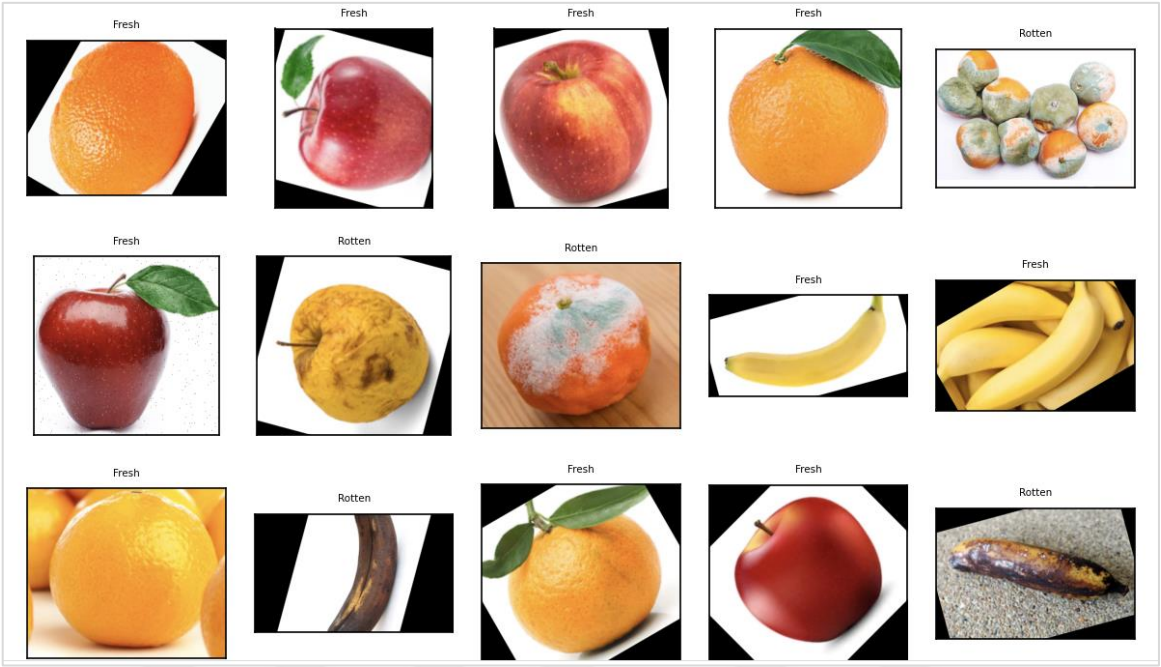


Figure 4.1: The Dataset Showing Fresh and Rotten Fruits.

Before performing data analysis, exploration of the dataset was performed. The dataset was visualized via plotting the class labels and the number of instances in these classes. As details are provided in Table 4.1 and Figure 4.2, the training dataset consists of 10,901 image instances with 4,740 fresh and 6,161 rotten fruits (apples, bananas and oranges). Meanwhile, the test dataset contains 1,157 fresh and 1519 rotten fruit images. Accordingly, both training and test datasets include around 43% fresh and 57% rotten fruits. The details about the test dataset are indicated in Table 4.2 and Figure 4.3.

4.2 Classification Accuracy

The trials were run on Anaconda Software Jupyter Lab Notebook via a computer with an Intel® 11th Gen i7- 11800H CPU processor running at 2.30GHz, a 1 TB SSD, and 8 GB of RAM to enable parallel processing and boost the classification task's computing

capacity for both the binary and multiclass classification models. The outputs of the convolutional (both before and after the fresh/rotten images are inputted to activation function) and pooling layers are plotted for the better visualization of the models as indicated in Figure 4.4 and Figure 4.5. As result, binary CNN model had a training accuracy of 99.15% and training loss of 3.22% and a testing accuracy of 97.59% and testing loss of 5.85% over 50 epochs both. Resulting accuracy and loss graphs are plotted in Figure 4.4 and 4.5 respectively.

Table 4.1: Table Showing Number of Instances in Training Classes

Class Name	Number of Instances
rottenbanana	2224
freshoranges	1466
rottenoranges	1595
freshbanana	1581
rottenapples	2342
freshapples	1693

Table 4.2: Table Showing Number of Instances in Testing Classes

Class Name	Number of Instances
rottenbanana	601
freshoranges	381
rottenoranges	530
freshbanana	381
rottenapples	388
freshapples	395

Figure 4.2: Figure Showing Training Class Distributions.

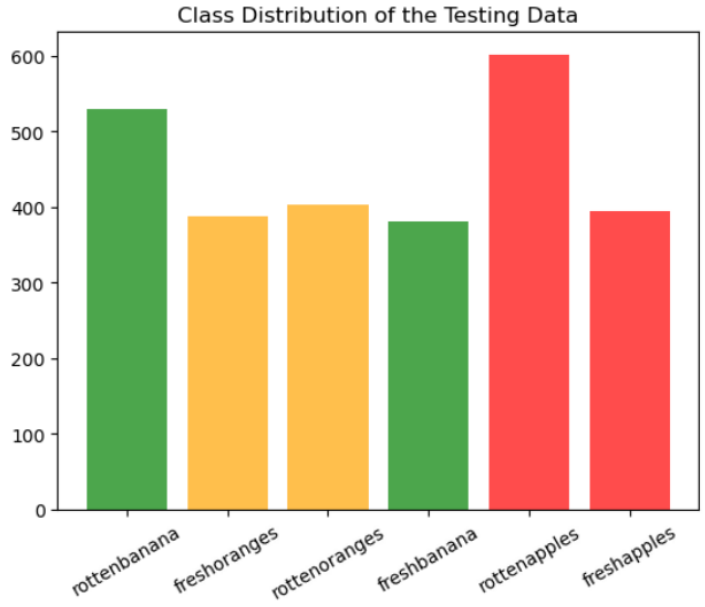


Figure 4.3: Figure Showing Testing Class Distribution.

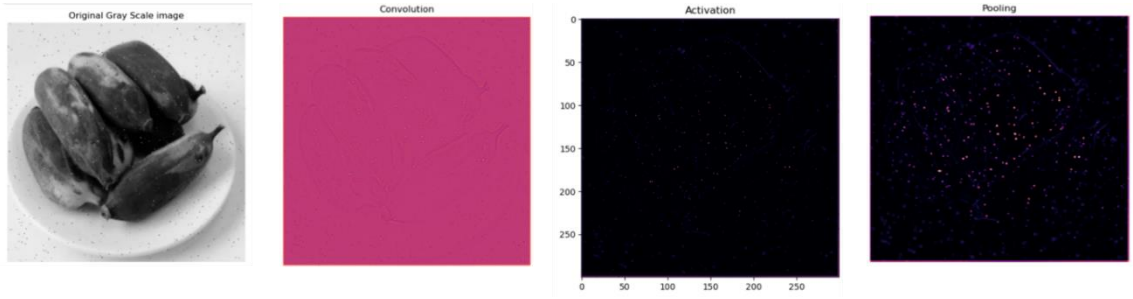


Figure 4.4: Respective Layer Outputs of the Inputted Rotten Image

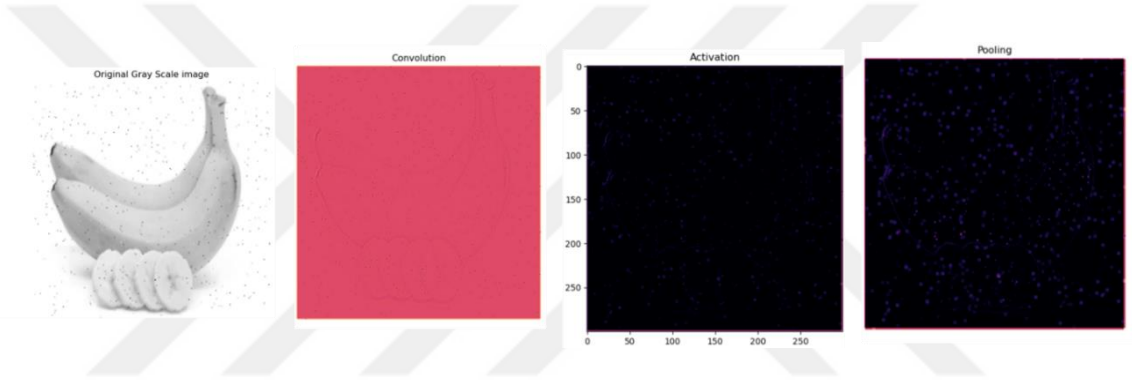


Figure 4.5: Respective Layer Outputs of the Inputted Fresh Image

For the multiclass classification model, our model had a training accuracy of 99.59% and training loss of 1.27% and a testing accuracy of 98.99% and testing loss of 2.32% over 50 epochs both. In addition, multiclass classification model was tested using an image from the test data to determine the classification ability of the model and hence, as it is indicated in Figure, it classified the inputted rotten banana correctly by turning its respective class instance in the array by 1; where 0 : freshapples, 1 : freshbanana 2 : freshoranges, 3 : rottenapples, 4 : rottenbanana, 5 : rottenoranges.

4.3 Sensor Readings

The prototype's hardware configuration involves the connection and wiring of DHT11 temperature and humidity sensors, as well as MQ3 gas sensors, to the ESP8266 NodeMCU microcontroller. For the sensor measurement, samples of fresh and rotten apple, banana and orange crops are used. The results should be interpreted with caution

due to the limitations of the current research. This study aims to offer a concept of the required hardware and software for the detection of freshness in which our target audience is the retailers; therefore, the study shall be tested in a retailer cold storage unit for better results.

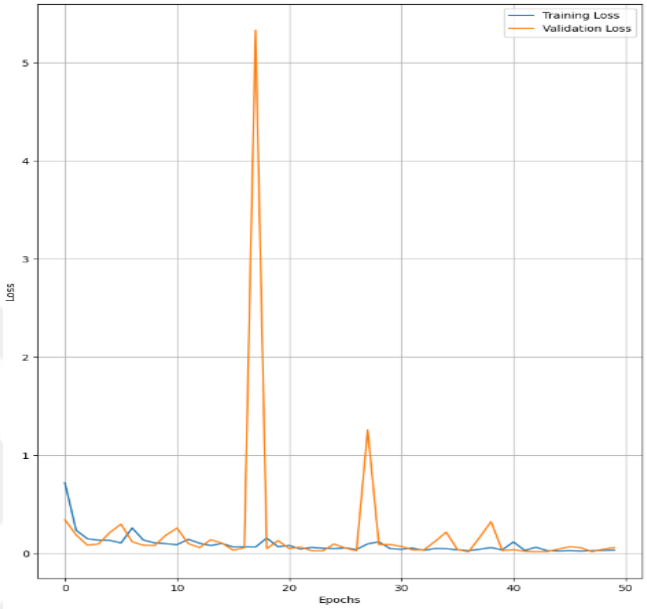


Figure 4.6: Training and Testing Loss of the Binary CNN Model.

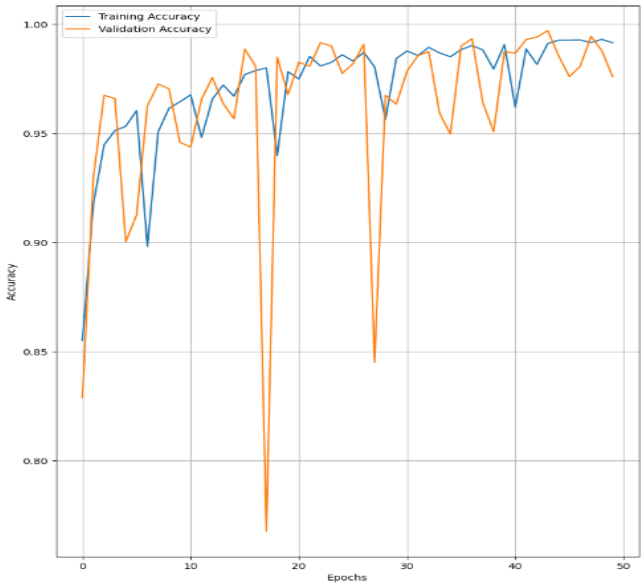


Figure 4.7: Training and Testing Accuracy of the Binary CNN Model.

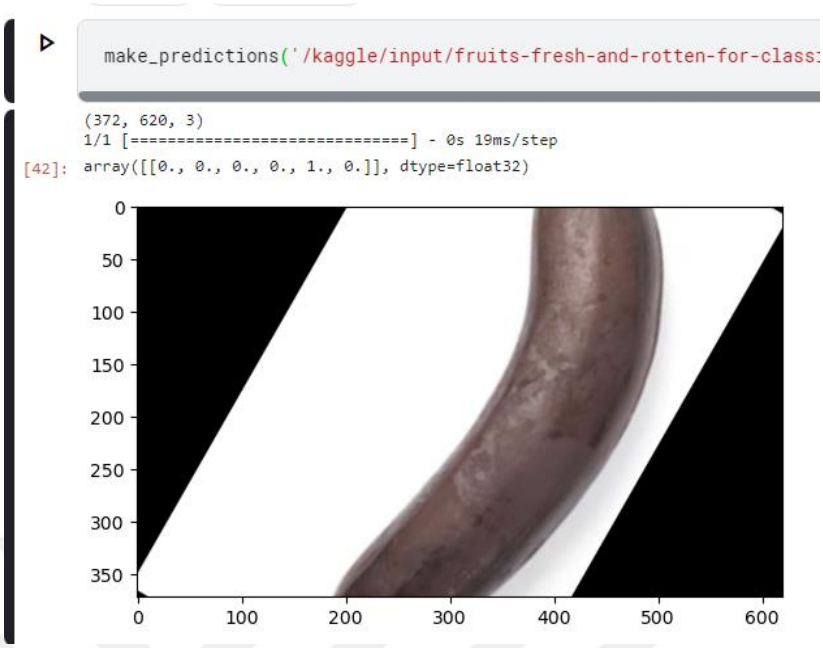


Figure 4.8: Multiclass Classification Testing Sample.

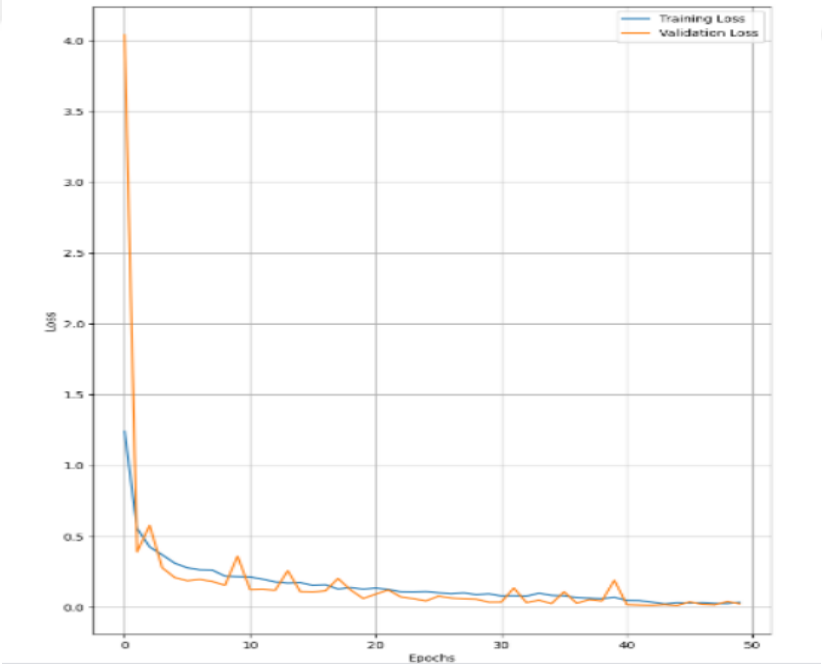


Figure 4.9: Training and Testing Loss of the Multiclass Classification Model

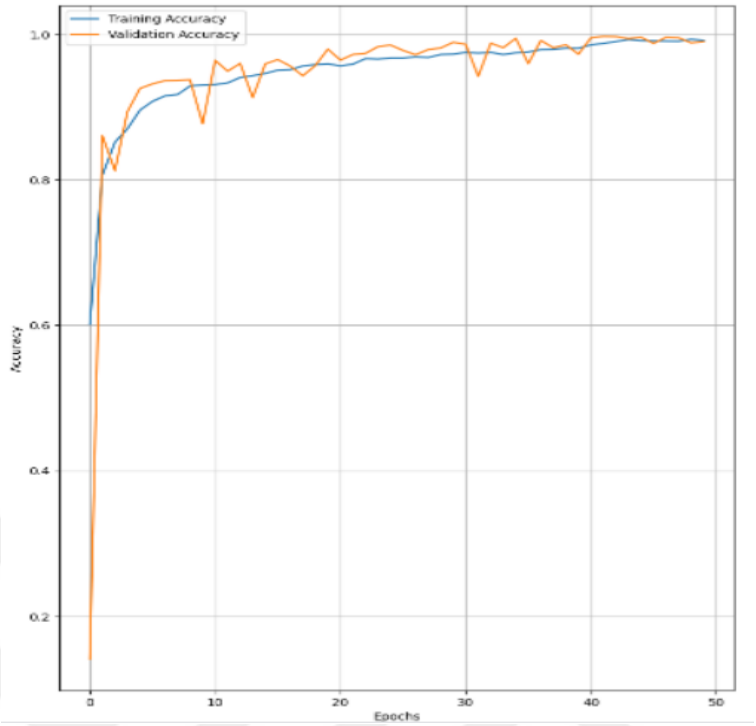


Figure 4.10: Training and Testing Accuracy of the Multiclass Classification Model

In this phase, the humidity, temperature, and alcohol composition levels of the fruit measured for 10 times are tabulated in Table 4.3. While considering alcohol levels, it should be noted that for the MQ3 sensor, reading ≤ 120 indicates the absence of alcohol composition whereas reading ≥ 400 indicates the high level of alcohol composition and hence, urgent measures shall be taken. An outtake of the sensor readings on the Arduino Serial Monitor is provided in Figure 4.6.

4.4 MQTT Communication & Cloud Architecture

DHT11 temperature & humidity, and MQ3 alcohol sensor data are transmitted to the AWS IoT Core Application, the initial stage of the AWS Cloud Architecture, using the ESP8266 Node MCU microcontroller via MQTT communication protocol. Therefore, to test our WSN, initially our hardware configuration was connected to the computer and Arduino IDE serial monitor was started with baud rate of 115200 bits per second. Then, MQTT test client was initialized on the AWS IoT Core Platform. On the MQTT test client screen, our microcontroller named ESP8266 was subscribed to view the published

messages and sensor readings on the cloud communication test screen. Based on the ESP8266 topic subscription, readings measured via microcontroller was sent to the cloud server immediately and was shown on the publishing screen. An example of such publishing is shown in Figure 4.7.

Table 4.3: Table Showing Sensor Outputs for the Apple Crop.

Fruit	Time Stamp	Temperature (°C)	RH (%)	Alcohol Level	Result
Apple	33128	23.79	163	654	Out of range
Apple	39460	23.89	59	586	Out of range
Apple	45784	23.89	59	546	Out of range
Apple	52384	23.89	59	497	Out of range
Apple	58771	24.00	59	475	Out of range
Apple	65415	24.00	59	450	Out of range
Apple	71793	24.00	58	428	Out of range
Apple	78142	24.00	58	411	Out of range
Apple	84803	24.00	58	393	Out of range
Apple	91189	24.00	58	379	Out of range

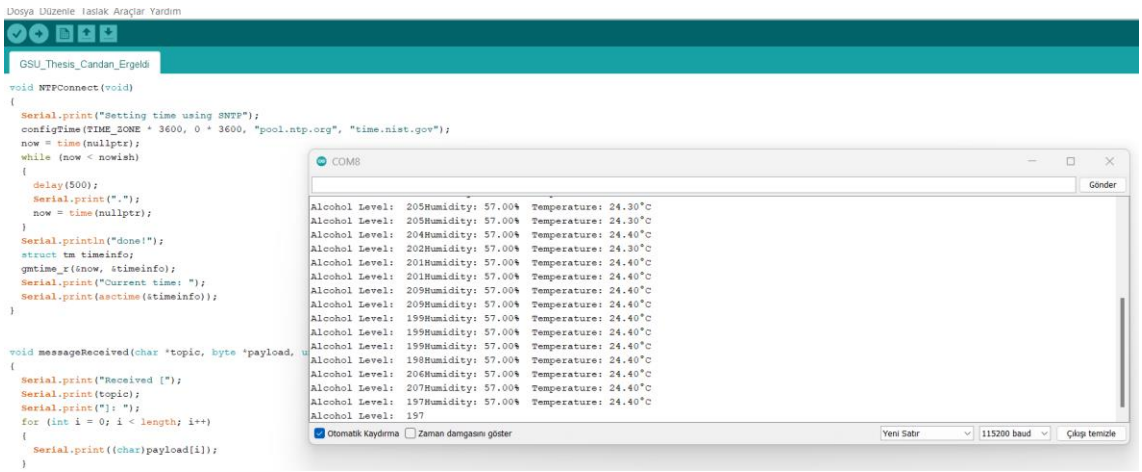


Figure 4.11: Figure Showing Sensor Readings on Arduino Serial Monitor.

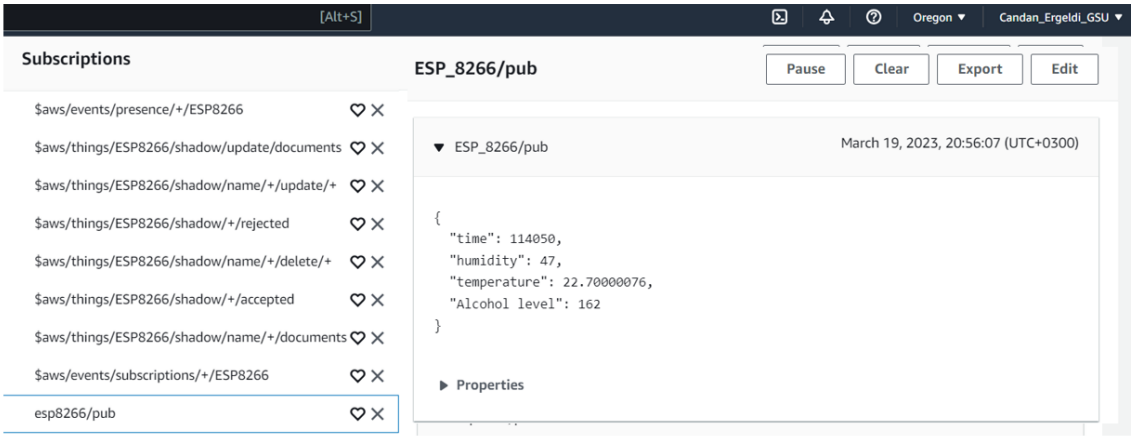
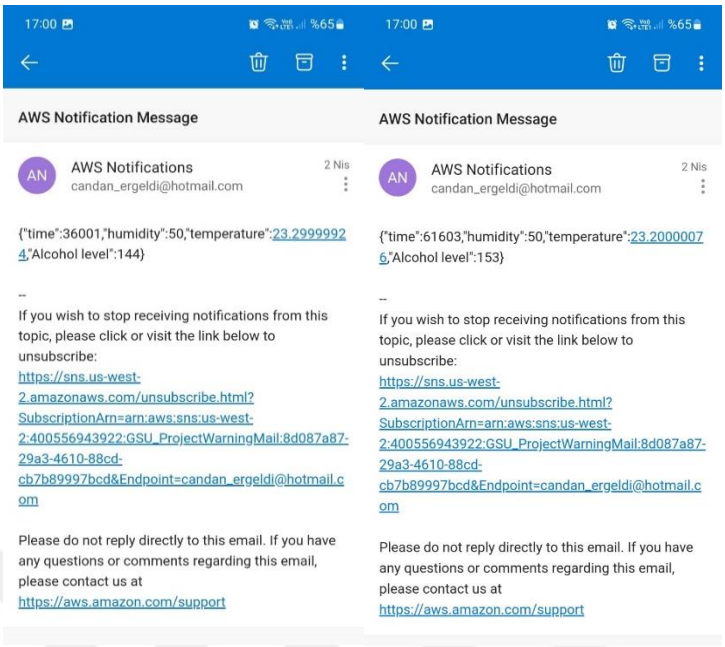


Figure 4.12: Sensor Data Readings on AWS Cloud IoT Core

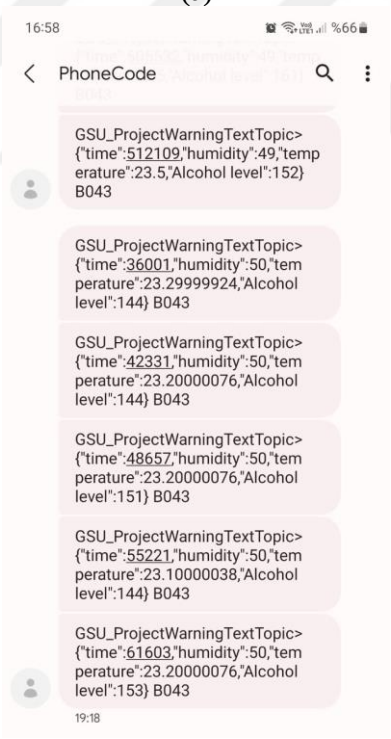
4.5 Client User Notification & Output Data Sustainability

After the cloud connection, based on the rules defined in IoT Core Service, the readings outside of the predefined range initiates the notification architecture in SNS Push Notification Service. Therefore, via message routing in SNS Push Notification Channel, user - system involvement is enhanced, and the client users are immediately alerted by receiving SMS and email notifications with the values of the temperature, humidity, and gas level parameters involved as shown in Figure 4.8 in the case of rottenness. With this way, the freshness of the crops is targeted to be kept in the desired level and the agri-food SC shareholders, our client users, responsible from the cold-storage of the fruits were warned in primary stages of the over ripeness that they could take the necessary actions for the rehabilitation beforehand. The frequency of the notifications is adjustable and could be changed based on the preference of the client users.

The sensor readings are stored in AWS database platforms DynamoDB, NoSQL based, and S3 Buckets shown in Figure 4.9, SQL based, for the accumulation and better tracking of historic data for the retailers. The data accumulated in the databases could be downloaded in JSON format for local storage and to be analyzed on other tools as shown in Figure 4.10. In addition, the readings are tabulated in AWS IoT Data Analytics for the better visualization and monitoring of the cold storage parameters.



(a)



(b)

Figure 4.13: SNS Push Notifications (E-mail (a) and the SMS (b) Outtakes)

Moreover, using AWS IoT Data Analytics, the sensor readings are sent to Jupyter Lab for the data analysis as provided in Figure 4.17 for the data driven decision making of the retailers.

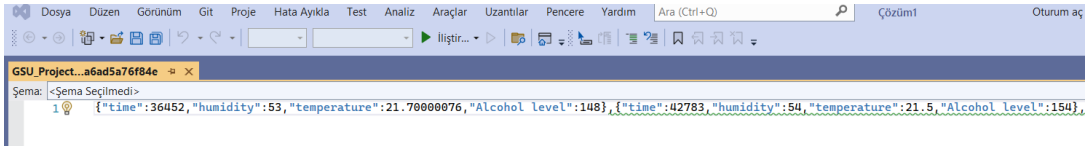


Figure 4.14: Figure Showing S3 Bucket Export.

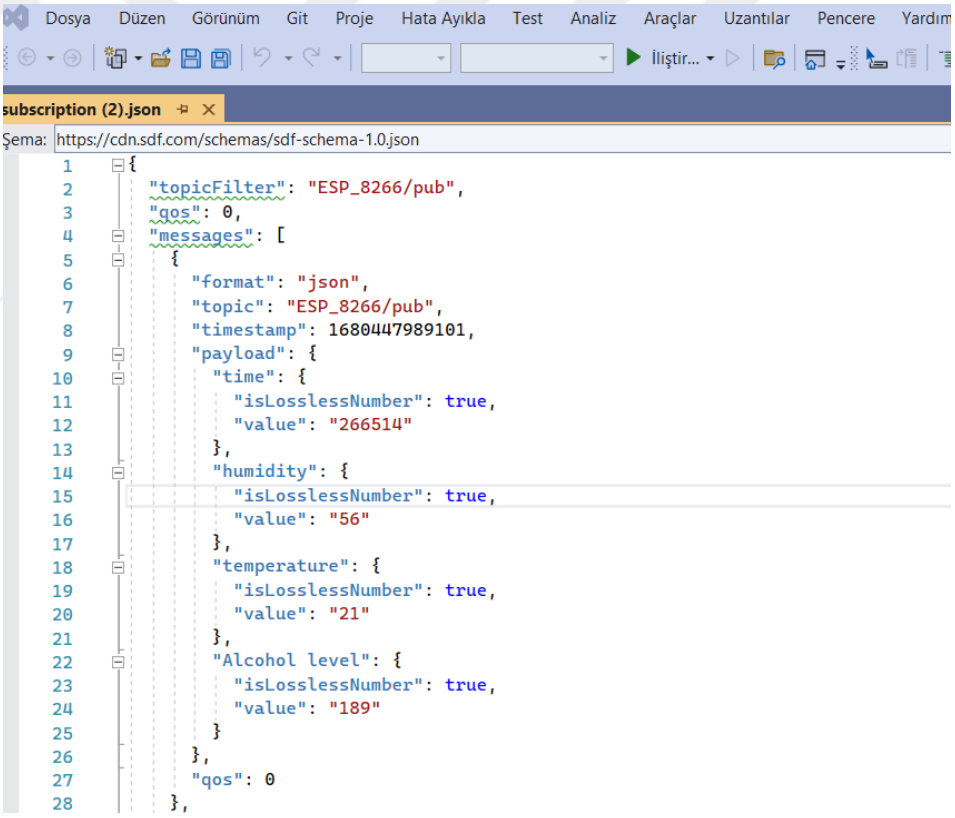


Figure 4.15: Figure Showing Data Exported in JSON Format.

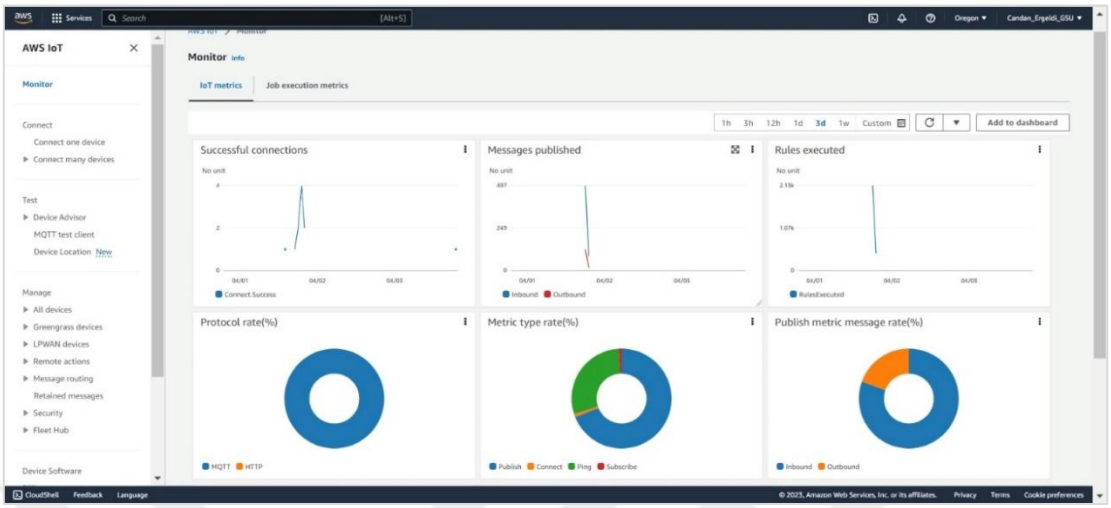


Figure 4.16: Figure Showing IoT Dashboard.

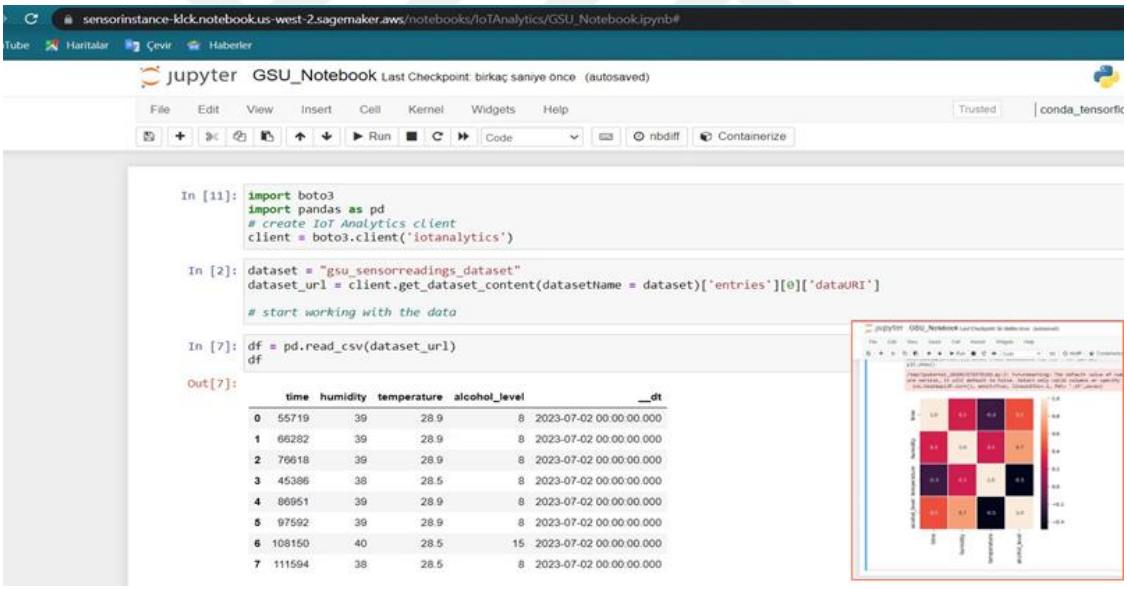


Figure 4.17: Jupyter Lab Connection of AWS Cloud IoT Analytics.

4.6 Effect of Cloud Utilization on Carbon Emission

The CO_2 emission report of the work was evaluated using the Cloud Platform. Based on the Cloud Platform's evaluation report, our study has not caused any metric tons of carbon dioxide-equivalent ($MTCO_2e$) (based on the report that we could receive results from the

3 months before) which is an industry-standard measure for carbon emissions as indicated in Figure 4.18. This measurement considers a variety of greenhouse gases, such as carbon dioxide, methane, and nitrous oxide in which all greenhouse gas emissions are translated to the amount of carbon dioxide required to cause the same level of warming. These carbon emissions statistics are based on the Greenhouse Gas Protocol and ISO Regulations.

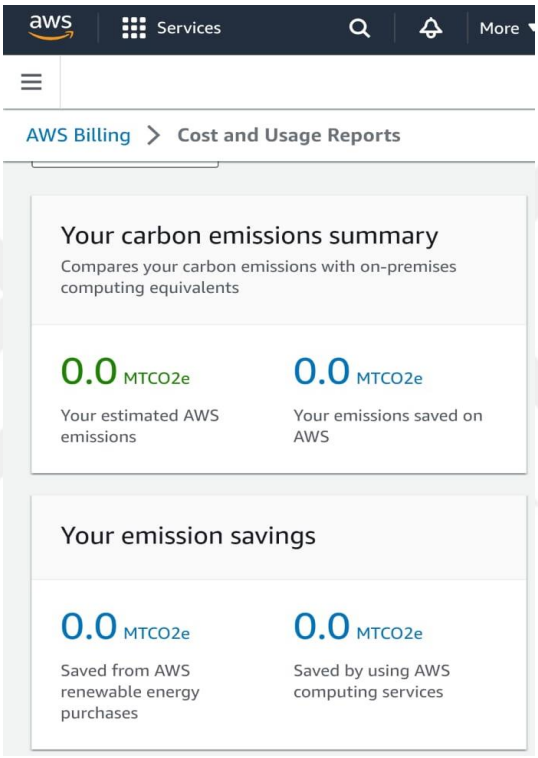


Figure 4.18: Cloud Carbon Emission of the Study.

5. CONCLUSION

The circular economy has the potential to increase food security and attain price stability. Regarding this issue, there have been industry 4.0 practices in agri-food supply chains, such as to improve sustainability for automating the fruit grading process. However, there is a gap in the making of a profound model that combines the sensor data and convolutional neural network (CNN) model together onto a cloud platform. To cope with this issue, we provided in this study a wireless sensor network model that was coupled with a CNN to determine the freshness of the food. The suggested application recognizes the crops and categorizes them according to their freshness using a trained CNN model. Additionally, to improve the accuracy of the ripeness level, our ensemble model interfaces with the ESP8266 Node MCU microcontroller to relay the temperature, humidity, and alcohol level to the Amazon Web Services cloud system. As result, our binary CNN model showed training accuracy of 99.15% and a testing accuracy of 97.59% over 50 epochs for each; and our multiclass classification CNN model had a training accuracy of 99.59% and training loss of 1.27% and a testing accuracy of 98.99% and testing loss of 2.32% over 50 epochs both.

The approach used in this thesis can precisely identify images of the inputted fruit types and classify them based on their ripeness. In addition, our model rapidly transmits data to the AWS Cloud. Hence, the model is secure for industrial purposes since the data accumulated in cloud and could be further analyzed. Our study provides a robust and a cost-efficient model that could easily be established in agri-food retail sector with various benefits as high accuracy in freshness detection, 7/24 tracking of sensor outputs on cloud server via any internet connected device, immediate notification of the parameter values based on pre-defined conditions.

For the future work, maturity levels of the fruits could be classified and indicated on a (0-100% level). By using this way, based on the freshness/ripeness level of the fruits they could be categorized based on their relevant further use. The suggested further use areas include subsidiary use of the crops such as for juice making, animal feed production or for other use cases such as their donation to a food bank or for composting. With the indicated adjustment, up to 100% circularity of the fruits would be ensured. Another improvement to be done include the dynamic price modelling of the fruits again based on their freshness percentage. Such improvement would attract most agri-food retailers since it would be easier for them to determine their pricing policies under a rational approach. In addition, they could have improved sales with least amount in loss of crops. Moreover, in current situation the model only covers the highly perishable products as fruits. However, the model could be applied and tested on other cold-storage food such as meat after research of required parameters of storage and training via relevant dataset, as well.

REFERENCES

Accorsi, R., Ferrari, E., & Manzini, R. (2019). Modeling inclusive food supply chains toward sustainable ecosystem planning. 10.1016/B978-0-12-813411-5.00001-6.

Agrawal, R., Wankhede, A.W., Kumar, A., Luthra, S., Huisingh D. (2021). Progress and trends in integrating Industry 4.0 within Circular Economy: A comprehensive literature review and future research propositions. *Business Strategy and the Environment.*

Ahumada, O. & Villalobos, J.R. (2009) Application of Planning Models in the Agri-Food Supply Chain: A Review. *J. European Journal of Operational Research*, 195, 1-20.

Atif, S., Ahmed, S., Wasim, M., Zeb, B., Pervez, Z., & Quinn, L. (2021). Towards a Conceptual Development of Industry 4.0, Servitisation, and Circular Economy: A Systematic Literature Review. *J. Sustainability. Vol 13. 6501.*

Angkiriwang, R., Pujawan, I.N., Santosa, B. (2014). Managing uncertainty through supply chain fexibility: reactive vs. proactive approaches. *Prod Manuf Res* 2:50–70.

Anitha, G. & Shivashankar, P. T. (2023). Fruits Fresh and Rotten Detection Using CNN & Transfer Learning. *Department of CSE, Faculty of Engineering and Technology Jain Deemed-to-be-University.*

Ayan, O., Demirez, Z., Kızılöz, H. K., İnci, G., İşleyen, S., Ergin, S. (2018). The Detection of Spoiled Fruits on a Conveyor Belt Using Image Processing Techniques and OPC Server Software. *J. IJCESEN Vol. 4, No.1 pp. 11-15*

Betal, S. (2019). Automatic Rotten and Fresh Apple Sorting Machine using Arduino and TCS3200. *J IJSRD Vol. 7, No. 2, 2321-0613*

Bhande, S.D., Ravindra, M.R., Goswami, T.K. (2008). Respiration rate of banana fruit under aerobic conditions at different storage temperatures. J *Journal of Food Engineering. 87. 116-123.* 10.1016/j.jfoodeng.2007.11.019.

Bishop, D. (1996). *Controlled atmosphere storage*, Edited by: Dellino, C.J.V. Blackie, London: Cold and chilled storage technology. (ed.).

Birkel, H, Müller, J. M. (2021). Potentials of industry 4.0 for supply chain management within the triple bottom line of sustainability e A systematic literature review. *J. Journal of Cleaner Production Vol. 289 No. 125612*

Breuel, T.M. (2015). The Effects of Hyperparameters on SGD Training of Neural Networks. arXiv:1508.02788.

Broughton, W.J. & Guat T. (1979). Storage conditions and ripening of the custard apple Annona squamosa L. *J Scientia Horticulturae, Vol 10, Issue 1, pp. 73-82, ISSN 0304-4238*

Chakraborty, S., Shamrat, F. M., Billah, Md., Jubair, Md., Alauddin, Mdi., Ranjan, R. (2021). Implementation of Deep Learning Methods to Identify Rotten Fruits. 10.1109/ICOEI51242.2021.9453004.

Chauhan, B., Singh, R., Mahajan, G. (2012). Ecology and management of weeds under conservation agriculture: A review. *Crop Protection. 38.* 57–65. 10.1016/j.cropro.2012.03.010.

Chen, Y., Li, L., Li, W., Guo, Q., Du, Z. & Xu ,Z. (2024) AI Computing Systems. Chapter 2 - Fundamentals of neural networks. *Morgan Kaufmann*. 17-51. ISBN 9780323953993.

Chou, C., Shie, C. K., Chang, F. C., Chang, J. & Chang, E. (2017). Representation Learning on Large and Small Data.

Codex Standard for Bananas. (1997). (205, AMD. 1-2005). URL: https://unece.org/fileadmin/DAM/trade/agr/meetings/ge.01/document/Codex%20bananas%20E.pdf

Coleman-Jensen, A., Rabbitt M. P., Gregory, C. A. & Singh A. (2021). Household Food Security in the United States in 2020, *ERR-298*, U.S. Department of Agriculture, Economic Research Service.

Cox, D. & Pinto, N. (2011). Beyond Simple Features: A Large-Scale Feature Search Approach to Unconstrained Face Recognition. In IEEE International Conference on Automatic Face Gesture Recognition and Workshops, 8–15.

Das, P., Yadav, J.K.P.S., Yadav, A.K. (2021). An automated tomato maturity grading system using transfer learning based AlexNet. *Ingénierie des Systèmes d'Information, Vol. 26, No. 2, pp. 191-200.*

Duncan, S.E., Reinhard, R., Williams, R.C., Ramsey, F., Thomason, W., Lee, K., Dudek, N., Mostaghimi, S., Colbert, E. & Murch R. (2019). Cyberbiosecurity: a new perspective on protecting U.S. Food and Agricultural System. *Front Bioeng Biotechnol 7.*

Dwivedi, D., Ganguly, A. & Haragopal, V.V. (2023). Statistical Modeling in Machine Learning. 6 - Contrast between simple and complex classification algorithms, *Academic Press. 93-110.* ISBN 9780323917766.

European Commission, Directorate-General for Climate Action, Directorate-General for Energy, Directorate-General for Mobility and Transport, De Vita, A., Capros, P., Paroussos, L.et al. . (2021). EU reference scenario 2020 – Energy, transport and GHG emissions : trends to 2050. *Publications Office.*

Faasema, J., Abu, J.O. & Alakali, J.S. (2011). Effect of packaging and storage condition on the quality of sweet orange (Citrus cinesis). J *Journal of Agricultural Technology Vol. 7(3): 797-804* ISSN 1686-9141.

Fakhrou, A., Kunhoth, J. & Maadeed, S. (2021). Smartphone-based food recognition system using multiple deep CNN models. *Multimedia Tools and Applications.*

FAO. 2017. The future of food and agriculture. (2017). Trends and challenges. Rome.

FAO. 2019. The State of Food and Agriculture. (2019). Moving forward on food loss and waste reduction. Rome. Licence: CC BY-NC-SA 3.0 IGO.

Fathi, E. & Shoja, B. (2018). Deep Neural Networks for Natural Language Processing. 10.1016/bs.host.2018.07.006.

Fatimah, Y., Govindan, K., Murniningsih, R. & Setiawan, A., (2020). Industry 4.0 based sustainable circular economy approach for smart waste management system to achieve sustainable development goals: A case study of Indonesia. *J Journal of Cleaner Production*. 269. 122263. 10.1016/j.jclepro.2020.122263.

Gao, S., Li, Z. Y., Han, Q., Cheng, M. M., & Wang, L. (2022). RF-Next: Efficient receptive field search for convolutional neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *45*(3), 2984-3002.

Gebhardt, M., Kopyto, M., Birkel, H. & Hartmann, E. (2021). Industry 4.0 technologies as enablers of collaboration in circular supply chains: a systematic literature review. *J. International Journal of Production Research.*

George, J.B., Marriott, J. (1985). The effect of some storage conditions on the storage life

of plantains. *Acta Horticulturae,* 158: 439-447.

Graves, Alex. (2013). Generating Sequences With Recurrent Neural Networks.

Guo, T, Wang, Y. (2019). Big data application issues in the agricultural modernization of China. *Ekoloji* 28:36773688.

Harder, R., Giampietro, M. & Sean, Smukler. (2021). Towards a circular nutrient economy. A novel way to analyze the circularity of nutrient flows in food systems *J Resources, Conservation and Recycling Vol 172*, 105693, ISSN 0921-3449.

Hinz, T., Navarro-Guerrero, N., Magg, S. & Wermter, S. (2018). Speeding up the Hyperparameter Optimization of Deep Convolutional Neural Networks. *J. International Journal of Computational Intelligence and Applications.* 17. 1850008. 10.1142/S1469026818500086.

Hussain, I., He, Q. & Chen Z. (2018). Automatic Fruit Recognition Based on DCNN for Commercial Source Trace System. *J. IJCSA Vol.8, No.2/3.*

Jana, S., Ranjan, P. & Sarkar, B. (2021). Detection of Rotten Fruits and Vegetables Using Deep Learning. *B. Computer Vision and Machine Learning in Agriculture*. 31-49.

Javaid, M., Haleem, A., Singh, R., Suman, R., Santibanez & Gonzalez, E. (2022). Understanding the adoption of Industry 4.0 technologies in improving environmental sustainability. *J. Sustainable Operations and Computers*.

Jayasankar, K., Karthika, B., Jeyashree, T., Deepalakshmi, R. & Karthika G. (2018). Fruit Freshness Detection Using Raspberry Pi. *J. International Journal of Pure and Applied Mathematics Vol. 119 No. 15*, 1685-1691.

Jayasinghe, P. K. S. C. & Sammani, S. (2022). Detection of Freshness of the Fruits using Machine Learning Techniques. *J. SLJoT:* 8-17.

Ji, G., Hu, L., Tan, K.H. (2017). A study on decision-making of food supply chain based on big data. *J Syst Sci Syst Eng* 26:183–198.

Johannesdottir, S., Macura, B., Mcconville, J., Lorick, D., Haddaway, N., Karczmarczyk, A., Ek, F., Piniewski, M., Ksiezniak-Machane, M. & Osuch, P. (2020). What evidence exists on ecotechnologies for recycling carbon and nutrients from domestic wastewater? A systematic map Environmental Evidence. *J Environmental Evidence. 9. 24.* 10.1186/s13750-020-00207-7.

Kader, A. A. (1986) Biochemical and Physiological Basis for Effects of Controlled and Modified Atmospheres on Fruits and Vegetables. J *Food Technology*, 40, 99-104.

Kalluri, S. R. (2018) URL : https://www.kaggle.com/datasets/sriramr/fruits-fresh-and-rotten-for-classification, Kaggle.

Kalsoom, T., Shehzad, A., Rafi-ul-Shan, P. M., Azmat, M., Akhtar, P., Pervez, Z., Imran, M. A. & Ur-Rehman, M. (2021). Impact of IoT on Manufacturing Industry 4.0: A New Triangular Systematic Review. *J Sustainability Vol. 13, No. 12506.*

Kays, S. (1991). Postharvest Physiology of Perishable Plant Products. Springer, 978-1 4684-8257-7.

Kim, M.H. & Lee, S.K. (1988). Effect of storage temperatures on mature green bananas, J. *Korean Soc. Hort. Sci., 29*: pp. 64-70.

Kingma, D.P. & Ba, J.L. (2014). Adam: A Method for Stochastic Optimization.

Khorrami, P., Paine, T. & Huang, T. (2015). Do Deep Neural Networks Learn Facial Action Units When Doing Expression Recognition? In IEEE International Conference on Computer Vision, 19–27.

Koning, N. & Ittersum, M. K. Will the world have enough to eat ? (2009). *J: Current Opinion in Environmental Sustainability. Vol 1. Issue 1: 77-82, ISSN 1877-3435.*

Li, D. & Wang, X. (2017). Dynamic supply chain decisions based on networked sensor data: an application in the chilled food retail chain. *Int J Prod Res 55:5127–5141.*

Liu, C., Wang, X., Ni, J., Cao, Y., Liu, B. (2019). An Edge Computing Visual System for Vegetable Categorization.

Mangla, S., Luthra, S., Mishra, N., Singh, A., Rana, N., Dora, M. & Dwivedi, Y. (2018). Barriers to effective circular supply chain management in a developing country context. *J Production Planning & Control.*

McGuire, S. (2010). U.S. Department of Agriculture and U.S. Department of Health and Human Services, Dietary Guidelines for Americans. 7th Edition, Washington, DC U.S. Government Printing Office, January 2011. *Adv Nutr.* 2011 May;2(3):293-4.

Morrelli, K. L., Hess-Pierce, B. M. & Kader, A. A. (2003). Genotypicvariation in chilling sensitivity of mature -green bananas and plantains. J. *Hort. Technology, 13, 328 – 332.*

Mudaliar, G. & Priyadarshini, R. B. K. (2021). A Machine Learning Approach for Predicting Fruit Freshness Classification, *J. IRJET Vol. 8 No. 5.*

Mureșan, H. & Oltean, M. (2018). Fruit recognition from images using deep learning. Acta Universitatis Sapientiae, *J. Informatica*. 10. 26-42. 10.2478/ausi-2018-0002.

Naranjo-Torres, J., Marco, M., Ruber, H. G, Barrientos, R. J, Claudio, F. & Valenzuela, (2020). A Review of Convolutional Neural Network Applied to Fruit Image

Processing. J. Applied Sciences Vol. 10. No. 3443.

Naseem, M. H & Yang J. (2021). Role of Industry 4.0 in Supply Chains Sustainability: A Systematic Literature Review. *J Sustainability Vol. 13, No. 9544.*

Nuanmeesri, S., Poomhiran, L., Ploydanai, K. (2021). Improving the Prediction of Rotten Fruit Using Convolutional Neural Network. *J. IJETT Vol.69 No. 7, pp. 51-55*

Pagotto, M. & Halog, A. (2015). Towards a Circular Economy in Australian Agri-food Industry: An Application of Input – Output Oriented Approaches for Analyzing Resource Efficiency, and Competitiveness Potential. J: Journal of Industrial Ecology pp. 20.

Patel, C. & Doshi, N. (2020). A Novel MQTT Security Framework in Generic IoT Model. J. *Procedia Computer Science, Vol 171. pp. 1399-1408.* ISSN 1877-0509.

Patil, D.L. & Magar, N.G . (1976). Physicochemical changes in banana fruits during ripening. J. *Maharastra Agrie. Univ., 1:* pp. 95-99.

Peacock, B.C. (1980). Banana ripening effect of temperature on fruit quality. Oueensl. J. *Agrie. Anim. Sci.* 37: pp. 39-45.

Rahman, T, Chowdhury, M. E. H., Khandakar, A., Islam, K. R., Islam, K. F., Mahbub, Z. B., Kadir, M. A. & Kashem, S. (2020). Transfer Learning with Deep Convolutional Neural Network (CNN) for Pneumonia Detection Using Chest X - ray. *Applied Sciences.* 10(9):3233.

Raj, A., Dwivedi, G., Sharma, A., Lopes de Sousa Jabbour, A. B. & Rajak S. (2020). Barriers to the adoption of industry 4.0 technologies in the manufacturing sector: An Inter - country comparative perspective. *J: International Journal of Production Economics, Vol 224,* 107546, ISSN 0925-5273.

Ramana, S. V., Kumar, B. L. M. & J ayaraman, K. S. (1989). Effect of postharvest treatments and modified atmosphere on the storage life of fresh banana and guava under ambient temperature. J *Indian Food Packer,* 43: pp. 29-35.

Rao, AR, Clarke D. (2019). Perspectives on emerging directions in using IoT devices in blockchain applications. *Internet Things 100079.*

Ravindhar, N. V. & Sasikumar, S., An effective monitoring, storage and analyze on industrial process on cloud big data by data publishing in industrial wireless sensor network, *J Measurement: Sensors, Volume 24,* 2022, 100525, ISSN 2665-9174

Rizzo, M., Marcuzzo, M., Zangari A., Gasparetto, A. & Albarelli, A. (2023). Fruit ripeness classification: A survey, *Artificial Intelligence in Agriculture, Vol. 7,* 44-57.

Romero, C. A. T, Castro, D. F, Ortiz, J. H, Khalaf, O. I. & Vargas, M. A, (2021). Synergy between Circular Economy and Industry 4.0: A Literature Review. J. Sustainability Vol. 13, No. 4331.

Roy, K, Chaudhuri, S. S & Pramanik, S. (2021). Deep learning based real-time Industrial framework for rotten, and fresh fruit detection using semantic segmentation. *J: Microsystem Technologies.*

Sahu A, Agrawal S, Kumar G. (2022). Integrating Industry 4.0 and circular economy: a review. *J: Journal of Enterprise Information Management Vol. 35 No. 3.*

Sakib, S., Ashrafi, Z., Siddique, M. A. (2019). Implementation of Fruits Recognition Classifier using Convolutional Neural Network (CNN) Algorithm for Observation of Accuracies for Various Hidden Layers.

Saltveit, M. (2002). Repiratory metabolism. The commercial storage of fruits, vegetables, and florist, and nursery stock. Fundamentals of Temperate Zone Tree Fruit Production, *311-313.*

Sanusi, R., Nwozo, S. & Ogunro, Y. (2008). Effect of Storage Time on Ascorbic Acid Content of Some Selected "Made in Nigeria" Fruit Preserves. J: Pakistan Journal of Nutrition. 7. 10.3923/pjn.2008.730.732.

Seymour, G.B., Thompson, A.K., John P. (1987). Inhibition of degreening in the peel of bananas ripened at tropical temperatures. I. Effect of high temperature on changes in the pulp and peel during ripening. *Ann. Applied Biology*, 110: pp. 145-151.

Shobana, G., Reethu, Sudheksha S, Vinothini, K. (2022). Fruit Freshness Detecting System Using Deep Learning, and Raspberry PI. International Conference on ICSES Innovative Computing, Intelligent Communication, and Smart Electrical Systems.

Sonwani, E., Bansal, U., Alroobaea, R., Baqasah, A. & Hedabou M. (2022). An Artificial Intelligence Approach Toward Food Spoilage Detection and Analysis. J: *Frontiers in Public Health.* 9. 10.3389/fpubh.2021.816226.

Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. J: Journal of Machine Learning Research, 15(1):1929–1958.

Stover, R. H. & Simmonds N. W. (1987) Classification of Banana Cultivars. In: Stover, *Eds., Bananas, Wiley*, New York, 97-103.

Tamasiga, P., Miri, T., Onyeaka, H. & Hart, A. (2022). Food Waste, and Circular Economy: Challenges and Opportunities. *J. Sustainability Vol 14. No. 9894.*

Tavares - Lehmann, A. & Varum, C. (2021). Industry 4.0, and Sustainability: A Bibliometric Literature Review. *J. Sustainability Vol 13. No. 3493.*

Tieleman, T. & Hinton, G. (2012). Lecture 6.5 – rms prop: Divide the Gradient by a Running Average of Its Recent Magnitude. 26-31.

Took, C. C. & Mandic, D. (2022). Weight sharing for LMS algorithms: Convolutional neural networks inspired multichannel adaptive filtering, *J. Digital Signal Processing Vol 127.* ISSN 1051-2004.

Tripu, D. & Farcuh, M. (2021). Keeping it Cool: Cold Storage Recommendations for Apples and Peaches. University of Maryland.
URL: https://extension.umd.edu/resource/keeping-it-cool-cold-storage-recommendations-apples-and-peaches.

Umezuruike, L. O., Rashid, A., Nafla, A., Fahad, A. S., Majeed, A. A., Annamalai, M., Adel, A. M. (2013). Postharvest Responses of 'Malindi' Cavendish Banana to Various Storage Conditions. J *International Journal of Fruit Science*, 13:4, 373- 388.

United Nations General Assembly. (2015). Transforming our world: the 2030 Agenda for Sustainable Development.
URL: https://www.refworld.org/docid/57b6e3e44.

United States Department of Agriculture (USDA). (2010). Food Security Assessment, 2010 - 2020. Report GFA-21, *Economic Research Service*, Washington DC.

Vaibhav, S., Narwane, A., Gunasekaran, & Gardas, B. (2022). Unlocking adoption challenges of IoT in Indian Agricultural and Food Supply Chain, *Smart Agricultural Technology, Volume 2*, 100035, ISSN 2772-3755.

Wilhelmina, K. (2005). Effects of production, and Processing Factors on Major Fruit and Vegetable Antioxidants. J *Journal of Food Science Vol. 70, Nr, 1. pp. R11-R19.*

Wojciuk, M., Swiderska - Chadaj, Z., Siwek, K., Gertych, A. (2022). The Role of Hyperparameter Optimization in Fine-Tuning of Cnn Models.

Wu, J., Chen, X. - Y., Zhang, H., Xiong, L. - D, Lei H. & Deng, S. - H. (2019). Hyperparameter optimization for machine learning models based on bayesian optimization. *J. Journal of Electronic Science and Technology. Vol. 17, issue 1, 26 – 40.*

Yamashita, R., Nishio, M., Do, R. K. G. *et al.* (2018). Convolutional neural networks: an overview and application in radiology. *Insights Imaging* 9, 611–629

Yoshioka, H., Ueda, Y. & Ogata, K. (1978). Effect of elevated temperatures on ripening

of banana fruits. J. *Japanese Soc. Food Sci. Techn., 25*: pp. 607 – 611.

Zhang, C., Chen, Y., Chen, H. & Chong, D. (2021). Industry 4.0 and its Implementation: a Review. *J. Information Systems Frontiers.*

Zhang, X., Chen X., Yao L., Ge C., Dong, M. (2019). Deep neural network Hyper parameter optimization with orthogonal array tuning. C: International Conference on neural information processing, 287 – 295.

APPENDICES

Appendix A. Arduino IDE Code

Main File

```
#include <ESP8266WiFi.h>
#include <WiFiClientSecure.h>
#include <PubSubClient.h>
#include <ArduinoJson.h>
#include <time.h>
#include "credentials.h"
#include "DHT.h"

#define DHTPIN 4
#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);

float h ;
float t;
unsigned long finalMillisecond = 0;
unsigned long previousFromFinalMillisecond = 0;
const long interval = 5000;

#define AWS_IOT_PUBLISH_TOPIC   "ESP_8266/pub"
#define AWS_IOT_SUBSCRIBE_TOPIC "ESP_8266/sub"

WiFiClientSecure net;

BearSSL::X509List cert(caCertificate);
BearSSL::X509List client_crt(clientCertificate);
BearSSL::PrivateKey key(privateKey);

PubSubClient client(net);

time_t now;
time_t nowish = 1510592825;
```

```
int mq3 = A0;
int data = 0;

void NTPConnect(void)
{
  Serial.print("Time is setting.");
  configTime(TIME_ZONE * 3600, 0 * 3600, "pool.ntp.org", "time.nist.gov");
  now = time(nullptr);
  while (now < nowish)
  {
    delay(500);
    Serial.print(".");
    now = time(nullptr);
  }
  Serial.println("Time is set.");
struct tm timeinfo;
gmtime_r(&now, &timeinfo);
Serial.print("Current time: ");
Serial.print(asctime(&timeinfo));
}


void messageReceived(char *topic, byte *payload, unsigned int length)
{
 Serial.print("Received [");
 Serial.print(topic);
 Serial.print("]: ");
 for (int i = 0; i < length; i++)
 {
  Serial.print((char)payload[i]);
 }
 Serial.println();
}

 struct tm timeinfo;
 gmtime_r(&now, &timeinfo);
 Serial.print("Current time: ");
 Serial.print(asctime(&timeinfo));
}

void messageReceived(char *topic, byte *payload, unsigned int length)
{
 Serial.print("Received [");
 Serial.print(topic);
 Serial.print("]: ");
 for (int i = 0; i < length; i++)
 {
  Serial.print((char)payload[i]);
```

```
 }
 Serial.println();
}

void connectAWS()
{
 delay(3000);
 WiFi.mode(WIFI_STA);
 WiFi.begin(WIFI_SSID, WIFI_PASSWORD);

 Serial.println(String("Attempting to connect to SSID: ") + String(WIFI_SSID));

 while (WiFi.status() != WL_CONNECTED)
 {
  Serial.print(".");
  delay(1000);
 }

 NTPConnect();

 net.setTrustAnchors(&cert);
 net.setClientRSACert(&client_crt, &key);

client.setServer(MQTT_HOST, 8883);
client.setCallback(messageReceived);


Serial.println("Device is connecting to AWS IoT Core Platform.");

while (!client.connect(THINGNAME))
{
 Serial.print(".");
 delay(1000);

  }

  if (!client.connected()) {
   Serial.println("Connection has been failed.");
   return;
  }
  client.subscribe(AWS_IOT_SUBSCRIBE_TOPIC);

  Serial.println("Device is connected to the AWS IoT Core Platform.");
 }
```

```
void publishMessage()
{
  StaticJsonDocument<200> doc;
  doc["time"] = millis();
  doc["humidity"] = h;


  doc["temperature"] = t;
  doc["Alcohol level"] = data;
  char jsonBuffer[512];
  serializeJson(doc, jsonBuffer); // Publishing data to client

  client.publish(AWS_IOT_PUBLISH_TOPIC, jsonBuffer);
}
void setup()
{
  Serial.begin(115200);
  connectAWS();
  dht.begin();
}
void loop()
{
 h = dht.readHumidity();
 t = dht.readTemperature();
 data = analogRead(mq2);

 if (isnan(h) || isnan(t) )
 {
Serial.println(F("DHT11 sensor reading has been failed, please try again."));
return;}

if (isnan(data) )
 {
Serial.println(F("Sensor reading has been failed, please try again"));
return;}

Serial.print(F("Humidity: "));
Serial.print(h);
Serial.print(F("%  Temperature: "));
Serial.print(t);
Serial.println(F("°C "));
Serial.print(F("Alcohol Level:  "));
Serial.print(data);
```

```
  delay(2000);

  now = time(nullptr);

  if (!client.connected())
  {
   connectAWS();
  }
  else
  {
   client.loop();
   if (millis() - finalMillisecond > 5000)
   {
    finalMillisecond = millis();
    publishMessage();
   }
  }
}
```

Appendix B. Python Code for Binary Classification

```
import numpy as np
import pandas as pd
import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
import cv2
import matplotlib.pyplot as plt
from tqdm import tqdm
from random import shuffle
from keras.utils  import to_categorical
import pickle
import tensorflow as tf
import keras
```

```
from keras.layers import Dense,Dropout, Conv2D,MaxPooling2D , Activation, Flatten,
BatchNormalization, SeparableConv2D
from keras.models import Sequential

def freshness():
    quality=['fresh', 'rotten']
    X,Y=[],[]
    z=[]
    for cata in tqdm(os.listdir('/kaggle/input/fruits-fresh-and-rotten-for-
classification/dataset/train')):
        if quality[0] in cata:
            datasetPath = os.path.join('/kaggle/input/fruits-fresh-and-rotten-for-
classification/dataset/train',cata)
            for imageName in os.listdir(datasetPath):
                image=cv2.imread(os.path.join(datasetPath,imageName))
                image=cv2.resize(image,(100,100))
                image=cv2.cvtColor(image,cv2.COLOR_BGR2RGB)
                z.append([image,0])
        else:
            datasetPath = os.path.join('/kaggle/input/fruits-fresh-and-rotten-for-
classification/dataset/train',cata)
            for imageName in os.listdir(datasetPath):
                image=cv2.imread(os.path.join(datasetPath,imageName))
                image=cv2.resize(image,(100,100))
                image=cv2.cvtColor(image,cv2.COLOR_BGR2RGB)
                z.append([image,1])

    print('Data is shuffling...')
    shuffle(z)
```

```
    for images, labels in tqdm(z):
        X.append(images);Y.append(labels)
    return X,Y

X,Y=freshness()
X=np.array(X)
Y=np.array(Y)
ySeries=pd.Series(Y)
ySeries.value_counts()
def freshnessTestData():
  quality=['fresh', 'rotten']
x,y=[],[]
  z=[]
  for cata in tqdm(os.listdir('/kaggle/input/fruits-fresh-and-rotten-for-classification/dataset/test')):
      if quality[0] in cata:
          datasetPath = os.path.join('/kaggle/input/fruits-fresh-and-rotten-for-classification/dataset/test',cata)
          for imageName in os.listdir(datasetPath):
              image=cv2.imread(os.path.join(datasetPath,imageName))
              image=cv2.resize(image,(100,100))
              image=cv2.cvtColor(image,cv2.COLOR_BGR2RGB)
              z.append([image,0])
      else:
          datasetPath = os.path.join('/kaggle/input/fruits-fresh-and-rotten-for-classification/dataset/test',cata)
          for imageName in os.listdir(datasetPath):
              image=cv2.imread(os.path.join(datasetPath,imageName))
              image=cv2.resize(image,(100,100))
              image=cv2.cvtColor(image,cv2.COLOR_BGR2RGB)
              z.append([image,1])
```

```
    print('Data is shuffling...')
    shuffle(z)

    for images, labels in tqdm(z):
        x.append(images);y.append(labels)
    return x,y

xTest,yTest=freshnessTestData()

yTest=np.array(yTest)
xTest=np.array(xTest)
ySeries=pd.Series(yTest)
ySeries.value_counts()

model = Sequential()

model.add(Conv2D(32, (3, 3), kernel_initializer='he_uniform', padding='same',
activation='relu', input_shape=(100,100,3)))
model.add(BatchNormalization())
model.add(SeparableConv2D(32, (3, 3), kernel_initializer='he_uniform',
padding='same', activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(BatchNormalization())
model.add(Dropout(0.3))

model.add(SeparableConv2D(64, (3, 3), kernel_initializer='he_uniform',
padding='same', activation='relu'))
model.add(SeparableConv2D(64, (3, 3), kernel_initializer='he_uniform',
padding='same', activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D((2, 2)))
model.add(Dropout(0.4))
```

```
model.add(BatchNormalization())
model.add(Dropout(0.3))

model.add(SeparableConv2D(64, (3, 3), kernel_initializer='he_uniform',
padding='same', activation='relu'))
model.add(SeparableConv2D(64, (3, 3), kernel_initializer='he_uniform',
padding='same', activation='relu'))

model.add(BatchNormalization())
model.add(MaxPooling2D((2, 2)))
model.add(Dropout(0.4))

model.add(Conv2D(128, (3, 3), kernel_initializer='he_uniform', padding='same',
activation='relu'))
model.add(BatchNormalization())
model.add(Conv2D(128, (3, 3), kernel_initializer='he_uniform', padding='same',
activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D((2, 2)))
model.add(Dropout(0.5))

model.add(Flatten())

model.add(Dense(512, activation='relu', kernel_initializer='he_uniform'))
model.add(Dropout(0.5))
model.add(Dense(128, activation='relu', kernel_initializer='he_uniform'))
model.add(Dropout(0.3))

model.add(Dense(1, activation='sigmoid'))

model.summary()
learningRate=keras.callbacks.ReduceLROnPlateau( monitor='val_loss', factor=0.5,
patience=6, verbose=1, mode='max',min_lr=0.00002, cooldown=2)
```

```
check_point=tf.keras.callbacks.ModelCheckpoint(
filepath='/kaggle/working/fruitFreshness.h5', monitor='val_loss', verbose=1,
save_best_only=True, save_weights_only=False, mode='min')

model.compile(loss=keras.losses.binary_crossentropy, optimizer =
keras.optimizers.Adam(learning_rate = 0.001), metrics=['accuracy'])
X = X/255.0
xTest = xTest/255.0

history = model.fit(X,Y,batch_size=20,validation_data=(xTest,yTest),epochs=
50,callbacks=[check_point])


plt.figure(1, figsize = (20, 12))
plt.subplot(1,2,1)
plt.xlabel("Epochs")

plt.figure(1, figsize = (20, 12))
plt.subplot(1,2,1)
plt.xlabel("Epochs")
plt.ylabel("Loss")
plt.plot( history.history["loss"], label = "Training Loss")
plt.plot( history.history["val_loss"], label = "Validation Loss")
plt.grid(True)
plt.legend()

plt.subplot(1,2,2)
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.plot( history.history["accuracy"], label = "Training Accuracy")
plt.plot( history.history["val_accuracy"], label = "Validation Accuracy")
plt.grid(True)
plt.legend()
```

Appendix C. Python Code for Multiclass Classification

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import cv2
os.environ["TF_CPP_MIN_LOG_LEVEL"] = "2"
import warnings
warnings.filterwarnings('ignore')
from sklearn.metrics import confusion_matrix, classification_report
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Activation, BatchNormalization, Conv2D, Dense,
Dropout, Flatten, MaxPooling2D
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.losses import CategoricalCrossentropy
from tensorflow.keras.callbacks import ReduceLROnPlateau, EarlyStopping


train_dataset_path = '/kaggle/input/fruits-fresh-and-rotten-for-classification/dataset/train'
validation_dataset_path = '/kaggle/input/fruits-fresh-and-rotten-for-classification/dataset/test'
```

```
IMG_WIDTH = 32
IMG_HEIGHT = 32
BATCH_SIZE = 32
train_datagen = ImageDataGenerator(rescale=1.0/255,fill_mode='nearest')

train_generator =
train_datagen.flow_from_directory(train_dataset_path,target_size=(IMG_WIDTH,
IMG_HEIGHT),batch_size=BATCH_SIZE,class_mode='categorical',shuffle=True)
validation_datagen = ImageDataGenerator(rescale=1.0/255)
validation_generator = validation_datagen.flow_from_directory(validation_dataset_path,
                                        target_size=(IMG_WIDTH, IMG_HEIGHT),
                                        batch_size=BATCH_SIZE,
class_mode='categorical',
shuffle=True)
for i in range(2):
    for j in range(5):
        label = labels[np.argmax(train_generator[0][1][idx])]
        ax[i, j].set_title(f"{label}")
        ax[i, j].imshow(train_generator[0][0][idx][:, :, :])
        ax[i, j].axis("off")
        idx += 1
plt.tight_layout()
plt.suptitle("Sample Training Images", fontsize=21)
plt.show()
def create_model():
    model = Sequential([
        Conv2D(filters=32,input_shape=(32, 32, 3),kernel_size=(3, 3),
```

```
        kernel_initializer='he_uniform',activation = 'relu', padding='same'),
        BatchNormalization(),
        SeparableConv2D(32, (3, 3), kernel_initializer='he_uniform', padding='same',
activation='relu'),
        MaxPooling2D(pool_size=(2, 2)),
        BatchNormalization(),
        Dropout(0.3),
        SeparableConv2D(64, (3, 3), kernel_initializer='he_uniform', padding='same',
activation='relu'),
        SeparableConv2D(64, (3, 3), kernel_initializer='he_uniform', padding='same',
activation='relu'),
        BatchNormalization(),
        MaxPooling2D((2, 2)),
        Dropout(0.4),

Conv2D(128, (3, 3), kernel_initializer='he_uniform', padding='same', activation='relu'),
BatchNormalization(),
Conv2D(128, (3, 3), kernel_initializer='he_uniform', padding='same', activation='relu'),
BatchNormalization(),
MaxPooling2D((2, 2)),
Dropout(0.5),
    Flatten(),
    Dense(512, activation='relu', kernel_initializer='he_uniform'),
    Dropout(0.5),
    Dense(128, activation='relu', kernel_initializer='he_uniform'),
    Dropout(0.3),
    Dense(6, activation='sigmoid')
])
```

```
    return model
cnn_model = create_model()
print(cnn_model.summary())
reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.5, patience=5)
optimizer = Adam(learning_rate=0.001)
cnn_model.compile(optimizer=optimizer, loss=CategoricalCrossentropy(),
metrics=['accuracy'])
history = cnn_model.fit(train_generator, epochs=50, validation_data=validation_generator,
                    verbose=2,
                    callbacks=[reduce_lr])
plt.figure(1, figsize = (20, 12))
plt.subplot(1,2,1)
plt.xlabel("Epochs")
plt.ylabel("Loss")

plt.plot( history.history["loss"], label = "Training Loss")
plt.plot( history.history["val_loss"], label = "Validation Loss")
plt.grid(True)
plt.legend()
plt.subplot(1,2,2)
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.plot( history.history["accuracy"], label = "Training Accuracy")
plt.plot( history.history["val_accuracy"], label = "Validation Accuracy")
plt.grid(True)
plt.legend()
cnn_model.evaluate(train_generator)
cnn_model.save('/kaggle/working/multiclass.h5')
from keras.models import Sequential,Model, load_model
trainedModel=load_model('/kaggle/working/multiclass.h5')
trainedModel.evaluate(validation_generator)
```

```
import matplotlib.image as mpimg
from tensorflow.keras.preprocessing import image as image_utils
from tensorflow.keras.applications.imagenet_utils import preprocess_input
def show_image(image_path):
    image = mpimg.imread(image_path)
    print(image.shape)
    plt.imshow(image)
def make_predictions(image_path):
    show_image(image_path)
    image = image_utils.load_img(image_path, target_size=(32, 32))
    image = image_utils.img_to_array(image)
    image = image.reshape(1,32,32,3)
    image = preprocess_input(image)
    input_image = preprocess_input(image)

    input_image = preprocess_input(image)
    predictions = cnn_model.predict(np.array([input_image]))
    predicted  class = np.argmax(predictions)
   predicted_probability = predictions[0][predicted_class]
   print("Predicted Class:", predicted_class)
   print("Probability:", predicted_probability)
   return preds
def predict_image_class(image, model, class_labels):
   input_image = preprocess_input(image)
   predictions = cnn_model.predict(np.array([input_image]))
   predicted_class_index = np.argmax(predictions)
   predicted_class_label = class_labels[predicted_class_index]
   print("Predicted Class:", predicted_class_label)
```

BIOGRAPHICAL SKETCH

Candan Ergeldi is a senior intelligent systems engineering master's degree student at the Galatasaray University. She received a bachelor’s degree in electrical and electronics engineering and industrial engineering from İstanbul Aydın University. Her research interests include artificial intelligence, image processing, cloud computing, and data science.

PUBLICATIONS

Ergeldi, C. & Feyzioğlu, O. (2023). Circular Supply Chains: An Internet of Things Application for Rotten Product Detection in Aggregate Food Industry. *The Eurasia Proceedings of Science, Technology, Engineering & Mathematics (EPSTEM)*.