Optimization of AI models as the Main Component in Prospective Edge Intelligence Applications

A dissertation submitted to the
College of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in the Department of Electrical and Computer Engineering
University of Saskatchewan
Saskatoon

By
Julio Wladimir Torres Tello

Abstract

Artificial Intelligence (AI) is a successful paradigm with application in many fields; however, there can be some challenging scenarios like the deployment of AI models in remote locations or with limited connectivity, possibly needing to perform inference closer to where data is collected. A potential solution is the study of ways to optimize AI models, for deployment of intelligent algorithms closer to the edge.

This thesis focuses on applications of AI that need to have characteristics that make them suitable for implementation on portable devices (e.g., aeroponics container, drone, mobile robot); thus, the development and implementation of custom models, and their optimization (i.e., reduction in size and inference time) is of upmost importance and the main goal of this dissertation. For this task, a number of options have been explored, including developing techniques to select relevant features from the samples that the model analyzes, and pruning and quantization. Therefore, this thesis proposes a scheme for the development, implementation, and optimization of custom AI models used mainly in agriculture, so that they have the desired characteristics that are needed for their deployment in edge devices. This main goal is fulfilled by implementing a number of sequential steps that include the validation of the hypothesis that there is at least an AI model capable of generating useful predictions for the applications being studied, the exploration and implementation of an approach for their optimization, and their final implementation in hardware of limited resources.

The main contributions of this thesis are on the general workflow for optimization of custom models, as well as in the proposed scheme for feature selection based on model interpretability approaches. This carries most of the novelty of the thesis, since we have not found previous implementations of these ideas, at least in the field under study. This optimization is mainly based on a feature selection approach, but finally complemented with pruning and quantization. The implementation of some of these models on an edge-like device, demonstrates the feasibility of this approach.

Finally, although this thesis tries to be a self-contained work, encompassing all the aspects required to go from AI model design to implementation on an edge device, there are some aspects that could be further studied, analyzed, and improved. Furthermore, it is almost impossible to keep the pace with all the new developments in the fields of AI, edge computing, hardware and software tools, etc. which opens the field for new discussions and proposals. This work tries to fill some gaps and to propose some ideas that hopefully will be useful for future researchers in the development of new technologies and solutions.

Acknowledgements

First of all, I would like to thank Prof. Seok-Bum Ko for giving me the opportunity to develop this project, for all his support, advise, patience, and dedication to this work; because without his guidance and knowledge I could have never gone through it.

Also, I would like to thank the University of Saskatchewan and its faculty and staff for making it possible for me to complete this degree. I am grateful for the knowledge and experience that I have gained throughout these years. And I would like to acknowledge the importance of the collaboration with Farmboys Design and the Plant Phenotyping and Imaging Research Centre.

Furthermore, I would like to mention that this work was supported by the scholarship CZ02-000745-2018 from Secretaría de Educación Superior, Ciencia, Tecnología e Innovación SENESCYT, and thanks to the leave conferred by Universidad de las Fuerzas Armadas ESPE, so that I can dedicate my time to the completion of my doctorate degree.

Additionally, I would like to thank my examining committee constituted by professors Ebrahim Bedeer Mohamed, Khan Wahid, Francis Bui, Ian Stavness, and Colleen Josephson, for their comments and feedback that have improved the final thesis. Also, especial thanks to my friends Andrés Erazo, Jaime Cárdenas, Adriana Noguera, Meghan Sired, and Dr. Juan Yépez, for their comments and revisions to this document.

Finally, I would like to thank everyone who contributed in some way or another to this achievement. Especial thanks to my family and friends for all their emotional support.

To my beloved parents.

Contents

List of Tables

List of Figures

List of Abbreviations

AI	Artificial Intelligence
ANN	Artificial Neural Network
AUC	Area Under the Curve
BFWA	Binary Firework Algorithm
BPTT	Backpropagation Through Time
CEC	Constant Error Carousel
CNN	Convolutional Neural Network
CPMI	Canola-Pod-Maturity Index
DAS	Days After Sowing/Seeding
DL	Deep Learning
DNN	Dense Neural Network
FPGA	Field-Programmable Gate Array
GC-MS	Gas Chromatography - Mass Spectrometry
GRU	Gated Recurrent Unit
HSI	Hyperspectral Imaging
IMS	Ion-Mobility Spectrometry
IoT	Internet of Things
kNN	k Nearest Neighbors
LAI	Leaf Area Index
LDA	Linear Discriminant Analysis
LSTM	Long Short Time Memory
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
ML	Machine Learning
MOX	Metal OXide
MSE	Mean Squared Error
MSI	Multispectral Imaging
NDVI	Normalized Difference Vegetation Index
NIR	Near-Infrared
NN	Neural Network
PCA	Principal Component Analysis
PLSR	Partial Least Squares Regression
PSNDb	Pigment Specific Normalized Difference b

P2IRC	Plant Phenotyping and Imaging Research Centre
RBF	Radial Basis Function
RCBD	Randomized Complete Block Design
ReLU	Rectified Linear Unit
RF	Random Forest
RGB	Red, Green, Blue
RMSE	Root Mean Square Error
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic
RS	Remote Sensing
SBS	Sequential Backward Selection
SHAP	SHapley Additive exPlanations
SIANN	Shift Invariant ANN
SVM	Support Vector Machine
SVR	Support Vector Regression
TDS	Total Dissolved Solids
UAV	Unmanned Aerial Vehicle
VGG	Visual Geometry Group
VOC	Volatile Organic Compounds
VPD	Vapour Pressure Deficit
WSN	Wireless Sensor Network

Part I

Preface

1 Introduction

Artificial Intelligence (AI) has achieved great success in recent years. Although commercially profitable applications such as image and speech recognition currently compete with humans in terms of accuracy, there are many other areas that could benefit from these technologies, but some obstacles have to be surpassed first. One of the current issues to adopt and adapt AI to help solve new problems is the difficulty of porting AI models into applications that are further away from compute servers and with limited connectivity, which leads to the possible need to perform inference closer to where the data is collected. A potential solution is the study of ways to optimize AI models that allow for the deployment of intelligent algorithms closer to the edge, where data is being generated.

This thesis evaluates a number of approaches to generate AI models that use different kinds of data, such as tabular, sequential, and multi- and hyper-spectral images (MSI and HSI, respectively); while trying to understand which features are more relevant for the predictions generated by these DL models. This understanding is relevant not only from a purely scientific curiosity point of view, but also as a tool for implementing smaller and faster models, that could run on hardware of restricted characteristics, allowing them to finally be deployed under edge intelligence frameworks.

A general scheme of the steps followed to reach these goals is presented in Fig. 1.1, and serves as a basic outline of the works described throughout this thesis. The feedback arrows represent how the products of each step are or could be used for the improvement of the outcomes of the proposed workflow. In the feedback, the continuous line represents the main flow of information applied in this thesis, while the dashed lines represent mostly what is left for future works.

1.1 The success of Machine and Deep Learning

AI is a concept as old as computers themselves, and it saw its firsts real advances in the mid twentieth century [1, 7], where some scientists tried to mathematically and electronically replicate the basic understanding that was available at the time of how neurons and the human brain work. However, in its beginnings, most successful AI approaches consisted on what is called symbolic AI, or the idea that intelligence could be achieved by programming a sufficiently large set of rules. This approach lacked flexibility and has now almost entirely been replaced by Machine Learning (ML), a subfield of AI (Fig. 1.2), that uses learning algorithms to extract information from data in order

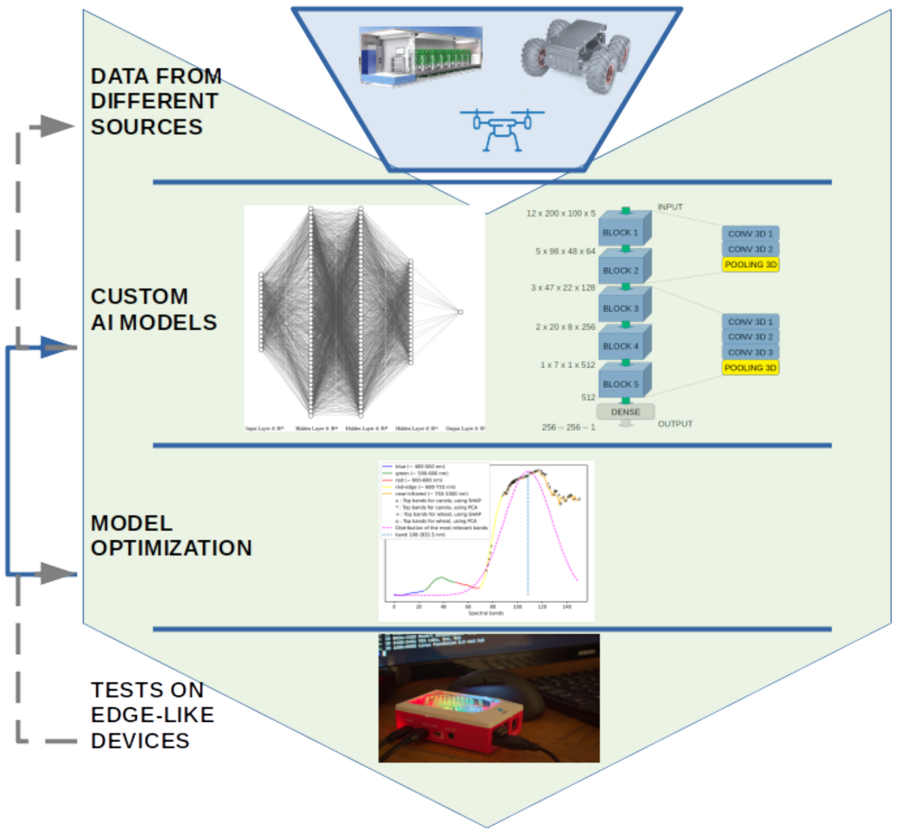


Figure 1.1: Graphical representation of the general workflow of this thesis.

to make predictions, instead of humans trying to derive complicated rules [7].

There are many definitions of ML in literature; however, all of them agree on the basic idea of machines being able to learn from data[1, 3, 7, 8]. Some also point out that it is mainly an engineering problem [3], due to its statistical foundations but high reliance on computers and relaxed confidence intervals [8]. That highly application-oriented nature makes it a science, but also an art [3, 7]. All of this has some implications; the most important being a paradigm shift in programming, which is better visualized in Fig. 1.3. In classical programming, a human has to write a set of rules that operate over data to provide some answers, while with ML, a computer uses the data and known answers in order to derive a set of rules that can then operate over new data to generate original answers [1]. This process, of course, still has the need for humans to write a program. The main change is that programmers now do not need to code the rules, but instead they need to specify a limited model space within which the computer needs to find an optimal model that fits the data, with the help of a feedback signal [1]. This process is called training.

Going deeper into Fig. 1.2, we have that DL is a particular form of ML [8], a mathematical framework that is not based on how the brain works [1]. This approach has gained attention in this century due to its recent advances [3], mainly powered by developments in hardware, data collec-

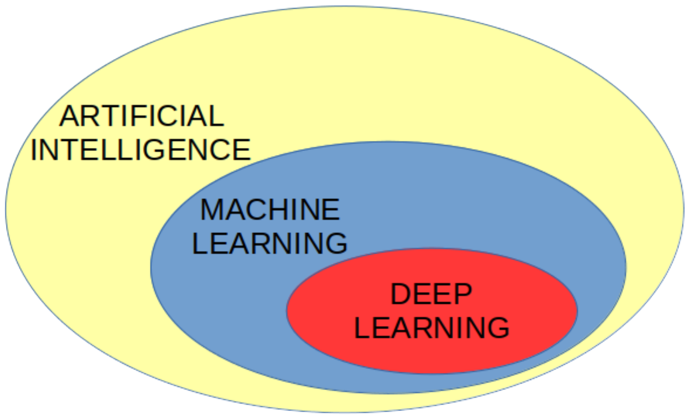


Figure 1.2: Artificial intelligence, machine learning, and deep learning [1].

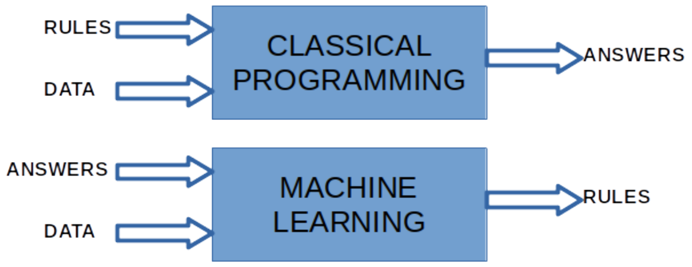


Figure 1.3: Machine learning: a new programming paradigm [1].

tion, new algorithms, and the amount of new applications that it finds (new ones being constantly realized)[1]. The main characteristic of DL is the added layers of increasingly meaningful representations [1]. In such networks, each layer generates a feature map that extracts essential and unique features within the samples that will eventually be used by the network for its characterization [9].

1.2 Motivation

1.2.1 Importance of the Applications of AI in this Thesis

ML and DL have achieved great success in many applications in recent years, such as medicine, robotics, and climatology. However, there are many other applications that can benefit from these technologies. In the case of this thesis, the main application for the developed ML and DL models is agriculture. Agriculture in general is a broad field encompassing different areas of knowledge, but the main challenges addressed in this work correspond to the development and implementation of models that can predict the yield [5, 10, 11, 12] (and moisture content in one particular case [13]) of a crop, based on some sensor data. The sensors utilized to collect the data, range from in-situ [5, 11] (e.g., temperature, light, etc.) to remote [10, 12, 13] (e.g., MSI and HSI images).

Importantly, regardless of the data collection method and the information contained, AI models can be adapted to deal with other applications. Provided that the used methods are alike, similar models and approaches can be implemented. To demonstrate this, one work [14] in this thesis, focuses on a security application based on a chemical sensing system. The common ground between this application and the agricultural ones is the similar data collection process, which becomes virtually transparent for the phase of AI model development and following steps.

Applications in Agriculture

Agriculture is one of the oldest and most important economical activities of humanity, which has undergone numerous changes throughout history. It began as a primitive process that eventually became what we call modern agriculture. Traditional methods for plant phenotyping (identification of external characteristics) are time consuming and prone to error [15]; therefore, optimization of this process is required. Furthermore, plant phenotyping, and specific tasks such as yield prediction or moisture content determination, could be the basis of fully automated control systems in which the final yield could be maximized by setting the variables under control (e.g., irrigation, light conditions, nutrients, temperature) [16]. Considering that the Government of Canada forecasts a production of 33.6 million tonnes of wheat for the 2020-2021 season [17], as an example, the improvement of the previously mentioned tasks can have a considerable impact in agricultural-based economies such as in the case of Saskatchewan.

In recent years, there have been some attempts to use ML for agricultural applications, as described in a review [25] about remote sensing and ML for crop water stress determination in various crops. There are also studies utilizing DL models and high-resolution image technology were able to develop more precise predictions of yield. A 2019 study in corn yield prediction found that a DNN model had a superior prediction accuracy on average yield [56]. [57] used CNNs that outperformed the yield prediction when compared with the traditional RS based model. In literature there are also some attempts to use SHAP values to interpret a CNN trained to predict soil organic carbon content [35].

Finding important features for yield and moisture content prediction is also relevant for the development of small, low-cost sensors that focus on capturing only the information that is needed for a given task. It is also important for the implementation of data collection strategies that could even become automated. All of this could open new research possibilities in the development of Internet of Things (IoT) based solutions for plant phenotyping. Thus, there are economical, scientific, and technical reasons that make agriculture an interesting application field for AI.

Chemical Sensing

Chemical sensing is a type of sensing where chemical substances are under analysis. As a proof of generalization of the main concepts, this thesis presents one work that deals with one specific chemical sensing topic: the identification of explosive compounds by means of an electronic nose, which tries to emulate the complex biological olfactory system [14]. This application is very relevant for the analysis of volatile organic compounds (VOC) that can be generated by different chemical processes. At first, this situation might seem very dissimilar from the agricultural applications; however, once data is collected, it can be processed and analyzed in a similar fashion to data obtained from in-situ sensing for agricultural applications. Furthermore, the time-series nature of the data and the related target for a portable, fast, and accurate application, makes it a good candidate for sharing architectural and training constraints when developing a DL model. Finally, it is important to mention that electronic noses are an active field or research in applications related to agriculture, such as quality assurance and identification of ingredients/components [18, 19, 20, 21, 22].

1.2.2 The Need for Custom Architectures

ML and DL are finding application in new fields of study all the time; however, those applications usually mean the use of pre-trained networks and in some cases a fine tuning process to fit the needs of such application. This happens mainly because of the lack of computational power for training new architectures, the long heuristic process of finding the optimal hyperparameters of a network, and the relative uniformity of the datasets that are being used. In the case of this thesis, the absence of the latter has been the driving force for having to implement and test customized architectures. The datasets used throughout this work, have been collected for different projects and with different goals in mind; thus, they are not conventional images, videos, or sequences that can easily be adapted to public pre-trained networks. Therefore, the development of architectures tailored to the different datasets is the first and one of the major components of this thesis. For that, it was necessary to use relatively large computational resources and the exhaustive search for optimal hyperparameters, as will be clearly described in each chapter.

1.2.3 Models Optimized for Edge Computing

Given that the applications of AI analyzed here have as a final goal to be implemented on portable devices, the optimization of the models is of upmost importance and it is the main focus of this thesis. For this task, a number of options have been explored, like techniques to select relevant features from the samples that the model analyzes, and pruning and quantization.

1.2.4 AI, a Growing Business

AI is a technology that is expected to grow in the near future, as shown in Fig. 1.4. Other data that support this trend mention that the worldwide enterprise spending on AI hardware, software, and services, are expected to reach $204 billion in 2025; or that by 2024, 60% of G2000[a] companies will expand the use of AI across all business-critical functions [2]. However, it is not only the adoption of AI, but also its efficient implementation, which implies the optimization of AI models for their use in edge devices, keeping in mind that this is also a promising field. Enterprise spending on edge hardware, software, and services will reach an estimated $250 billion by 2024 [23]. Therefore, innovations in relevant technologies and the inclusion of new applications are critical.

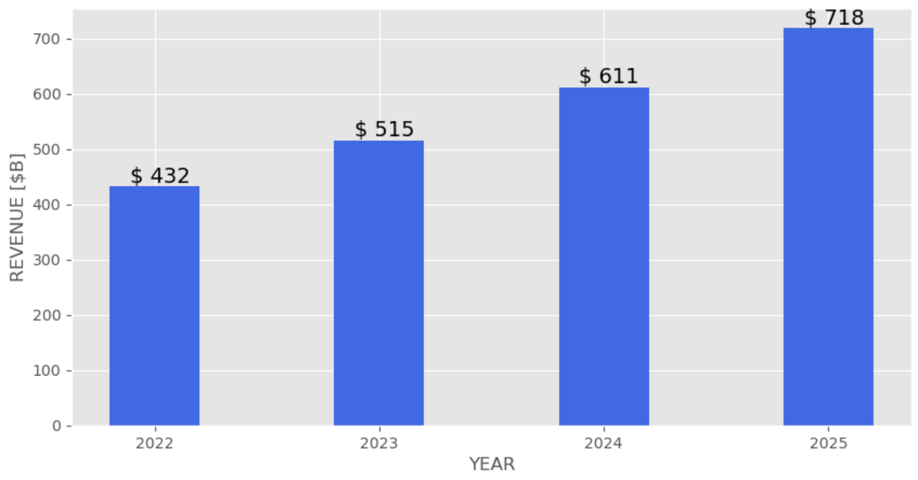


Figure 1.4: Worldwide AI Hardware, Software, and Services Revenue ($B), 2022–2025 [2].

1.3 Objectives

The main objective of this thesis is to propose a scheme (which general concept is described in Fig. 1.1) to develop, implement, and optimize custom AI models used mainly in agriculture, so that they have the characteristics needed to be deployed in edge devices. This main goal is built on the following specific objectives, which have to be accomplished as sequential steps:

1. The first objective, in which all the others are supported, is to validate the hypothesis that there is at least a ML/DL model capable of generating useful predictions for the applications being studied: yield and moisture content prediction from remote sensing sources, yield prediction from an aeroponic farm, and explosive detection by means of an e-nose.

2. Provided that the first objective is fulfilled, the second one is to explore, propose, and implement an approach for the optimization of such models, using feature selection as the main component.

[a]The Forbes Global 2000 (G2000) is an annual ranking of the top 2000 public companies in the world by Forbes magazine.

3. The final objective is to demonstrate that such models, which original format might contain an elevated number of parameters, can be optimized and implemented in hardware of limited resources, facilitating the implementation of edge intelligence. This should be done on a general purpose computer, in order to facilitate the adoption and deployment of the proposed solution.

1.4 Overview of Research Works

In this thesis, based on the concepts discussed throughout this chapter, some solutions are provided in order to first implement customized models for the particular tasks being addressed, and then for the optimization of some of those models. This optimization is mainly based on a feature selection approach, but finally complemented with pruning and quantization. It has also been an important task to implement some of these models on an embedded device, in order to demonstrate the feasibility of this approach. In the end, the main contributions of this thesis are on the general workflow for model optimization, as well as in the proposed scheme for feature selection based on model interpretability approaches. This carries most of the novelty of the thesis, since we have not found previous implementations of these ideas, at least in the field under study. The whole thesis is composed of five parts with nine chapters shown as follows:

- **Part I Preface** includes:
  - Chapter 1 Introduction: presents the importance of deep learning and the motivations, the objectives, and the contributions of the research works.
  - Chapter 2 Background: introduces the technical information required to present the research works.
- **Part II Custom Machine Learning Models** includes:
  - Chapter 3 Ensemble Learning for Improving Generalization in Aeroponics Yield Prediction: presents an introduction to the use of ML techniques, particularly for analyzing data collected from unconventional sources (an aeroponics growing module). This chapter attempts to verify if ML could be a viable approach for processing data obtained from various kinds of electronic sensors.
- **Part III Feature Selection in Deep Learning** includes:
  - Chapter 4 Identifying Useful Features in Multispectral Images with Deep Learning for Optimizing Wheat Yield Prediction: presents the use of DL for the analysis of MSI taken at different times and locations, with yet another customized model. Moving one step forward into this thesis main workflow, this chapter also explores the possibility of

identifying the most relevant features for the predictions of the DL model. A simple algorithm is used in this initial stage.

  - Chapter 5 A Novel Approach to Identify the Spectral Bands that Predict Moisture Content in Canola and Wheat: presents a game-theory-based method for feature selection, based on a feature attribution algorithm designed for model explainability/interpretability (SHAP). This approach is used to analyze which features generate certain predictions. One advantage of the proposed technique is that this can preserve the physical meaning of the original features, providing more meaningful results than feature extraction algorithms like PCA. Another benefit of this line of action is that it can generate explanations of the trained model, focused on a target (e.g., moisture content), a feature (e.g., spectral or spatial), or a subset of the samples. If a comprehensive explanation is needed, the individual contributions can be aggregated into a single SHAP value per feature, due to its additive property. This proposed approach is the main novelty of this work.

- **Part IV Model Optimization and Edge Intelligence** includes:

  - Chapter 6 Interpretability of Artificial Intelligence Models that use Data Fusion to Predict Yield in Aeroponics: presents concepts that elaborate on previous chapters; first, it extends the use of DL models for sensor-data, also applying data fusion, and second, it introduces the use of SHAP-based feature selection for model optimization, by using only the most relevant features in the dataset. This is the first step towards the implementation of these DL models in embedded devices. This chapter is also the only one that explores the ideas of model interpretability, by focusing on the ways that some features affect the final predictions, their meaning, and their relationship to the current scientific knowledge of the particular phenomena.

  - Chapter 7 Improving the Detection of Explosives in a MOX Chemical Sensors Array with LSTM Networks: presents an example of the whole proposed process, starting with the implementation of customized AI models, going through model optimization, and finally ending in the implementation on an edge-like device. In this particular case, it implements a simple approach for feature selection, by taking only the initial time-steps of a sequence in order to make a prediction.

  - Chapter 8 Optimizing a Multispectral-Images-Based DL Model, Through Feature Selection, Pruning and Quantization: presents the use of pruning and quantization, which are model optimization techniques that complement the feature selection approach, and further improve the task of creating models for embedded applications. This chapter demonstrates that after applying some optimization techniques, a DL model can be implemented on an edge device, which could in turn be mounted on or near a data gathering devices (such as a drone, an exploration robot, or an aeroponics growing chamber).

- **Part V Conclusion** includes:
  - Chapter 9 Conclusions and Future Research: presents the conclusions of the thesis and proposes ideas for future work.

1.5 Summary of Contributions

In this thesis, some new ideas are proposed particularly in the feature selection field, and its usage as an optimization tool for DL models. The papers that support this thesis present examples of the implementation and application of these ideas. Another important component is the implementation of customized AI models that serve as an initial step towards that end. Therefore, all the papers here described contribute in one way or another to the achievement of the aforementioned goals. Below is the list of publications, arranged according to the order of appearance in this thesis:

- **Chapter 3: J. Torres-Tello**, S. Venkatachalam, L. Moreno, and S.-B. Ko, "Ensemble Learning for Improving Generalization in Aeroponics Yield Prediction," in 2020 IEEE International Symposium on Circuits and Systems (ISCAS), Spain, Oct. 2020, pp. 1–5. doi: 10.1109/ISCAS45731.2020.9181283.
- **Chapter 4: J. Torres-Tello** and S.-B. Ko, "Identifying Useful Features in Multispectral Images with Deep Learning for Optimizing Wheat Yield Prediction," in 2021 IEEE International Symposium on Circuits and Systems (ISCAS), May 2021, pp. 1–5. doi: 10.1109/ISCAS51556.2021.9401360.
- **Chapter 5: J. W. Torres-Tello**, S. Ko, K. D. Singh, and S. J. Shirtliffe, "A novel approach to identify the spectral bands that predict moisture content in canola and wheat," Biosystems Engineering, vol. 210, pp. 91–103, Oct. 2021, doi: 10.1016/j.biosystemseng.2021.08.004.
- **Chapter 6: J. Torres-Tello** and S.-B. Ko, "Interpretability of artificial intelligence models that use data fusion to predict yield in aeroponics," J Ambient Intell Human Comput, Sep. 2021, doi: 10.1007/s12652-021-03470-9.
- **Chapter 7: J. Torres-Tello**, A. V. Guamán, and S.-B. Ko, "Improving the Detection of Explosives in a MOX Chemical Sensors Array With LSTM Networks," IEEE Sensors Journal, vol. 20, no. 23, pp. 14302–14309, 2020, doi: 10.1109/JSEN.2020.3007431.
- **Chapter 8: J. Torres-Tello** and S.-B. Ko, "Optimizing a Multispectral-Images-Based DL Model, through Feature Selection, Pruning and Quantization," in 2022 IEEE International Symposium on Circuits and Systems (ISCAS), In press.

- **Other publications that are not included in this thesis:** P. Pérez, **J. Torres-Tello**, and S. Ko, "Low-Cost 2-D Map Generation System for a Mobile Robot," in 2019 IEEE International Symposium on Circuits and Systems (ISCAS), May 2019, pp. 1–5. doi: 10.1109/ISCAS.2019.8702764.

2 Background

This chapter is a review of the main concepts discussed in this thesis. The main goal is to introduce the reader to the general concepts before they appear in the following chapters, where these concepts and ideas are presented again and/or further discussed.

This chapter begins with the description of basic concepts on sensors and data types that will be used to train the AI models. Then, it continues with a description of ML/DL and the particular techniques applied in this thesis. Later, it introduces the concepts that support the idea of model optimization, mainly focusing to the identification of salient features in the datasets. Finally, some characteristics of the edge-like devices and their model implementation are mentioned.

2.1 Sensors and Types of Data

A transducer is a device that transforms some kind of energy into another one, typically electric. On top of it, a sensor is a kind of transducer in which the output signal is a function of the measured variable. In a broad sense, this measurement also implies some kind of data processing and/or the display of results, being either in-situ or remote; functions that are usually integrated in the same device. This has been possible mainly due to three technological advances: computer miniaturization, improvements in semiconductor fabrication, and distributed control schemes [24].

There are many different kinds of sensors, but the ones used in this thesis can be classified in two categories regarding the type of output: time-series and image sensors.

2.1.1 Time-series

This is the most basic kind of data collection scheme, in which a sensor samples a signal at a given frequency. Thus, the sample contains a number of datapoints at fixed time intervals. Different sensors can be measuring the same or different variables at the same time, to construct a unique sample, in univariate (one sensor at a time) and multivariate (multiple sensors at the same time) situations. Fig. 2.1 shows an example of time-series data used in this thesis, which include magnitudes like relative humidity, temperature, gas concentrations, etc. These sensors are usually cheap, small, and energy efficient, and their measurements are valuable inputs to ML algorithms, although they usually need some level of signal pre-processing.

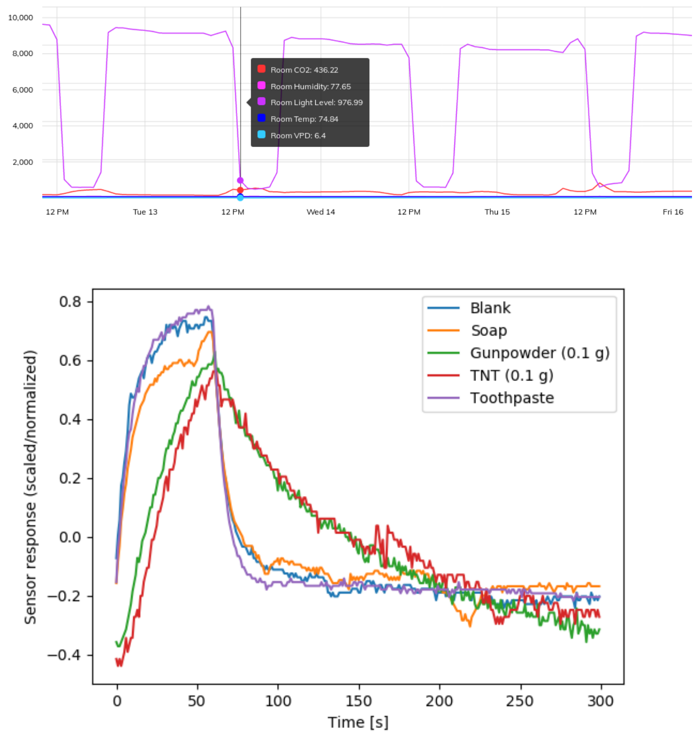


Figure 2.1: Examples of time-series data used in this thesis

2.1.2 Image Sensors

A more complex kind of sensor is one that has to construct a two-dimensional spatial representation of an object. This is the case of image sensors, like the ones included in photographic cameras, as well as in MSI/HSI sensors. Images are usually used for remote sensing, in which there is no physical contact between the sensor and the measured object or magnitude. These sensors in particular respond to the electromagnetic spectrum reflected by an object. In that sense, they constitute passive sensors, meaning that they do not inject any energy into the system. For this thesis, MSI and HSI are used.

Multi- and Hyper-spectral Images

Traditional photography or imaging is based on the idea that color can be represented as a combination of three channels: red, green, and blue (RGB). This is usually enough to represent the colors that the human eye can distinguish; however, the electromagnetic spectrum contains more information that can be collected by other sensors. An HSI camera is a passive sensor that captures the spectral reflectance that corresponds to the wavelength of the electromagnetic field collected from objects on the ground. This information is then associated with biochemical and biophysical properties of the surface, and the number of spectral bands recorded can range from a few to a few hundred (150 in this thesis) [25].

In that sense, MSI is a subfield of HSI, in which the number of spectral bands is very limited. In the case of the dataset used in this thesis MSI samples contain only five bands (RGB plus red-edge and near infrared channels). This difference also means that HSI with more bands usually have a better spatial resolution (each band is narrower).

Thus, HSI is a combination of digital imaging and spectroscopy, best matched to applications in which spectral information is more reliable or measurable than shape or morphology [26], as it is the case of the studies described in following chapters. Considering that most current approaches that involve HSI and plant phenotyping tend to use spectral (or vegetation) indexes, which are mathematical combinations of two or more spectral bands [25, 27], ML seems a viable option to find similar and probably more complex relationships that could generate better results, despite the problems associated to it, including high dimensionality of the data, redundant information, and insufficient samples [28].

As an example, Fig. 2.2 shows RGB representations of a canola field where HSI have been collected, as well as the spectral curve corresponding to a canola plot (mean values for all pixels in the plot). Each pixel in the HSI data contains spectral curves similar to the one presented in this figure, and this is the information fed to our DL models. Also, this information is later analyzed with a feature attribution/selection method.

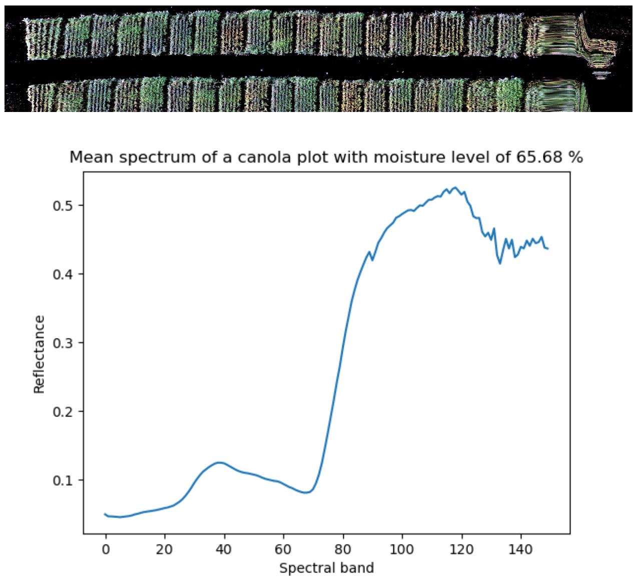


Figure 2.2: HSI samples of a canola field. Top: RGB representation. Bottom: Spectral curve of a sample.

In the end, regardless of the sensor used to collect the data, samples will be organized into multi-dimensional arrays that can be operated upon by the ML and DL algorithms implemented for each particular case.

2.2 Machine and Deep Learning

As it was mentioned in the previous chapter, multiple ML and DL models have been implemented throughout this thesis. This section contains a brief description of those algorithms. It is important to remember that the main goal of these algorithms is to learn the relationship between a set of samples X and their target values Y, in what is called a supervised training. Once that relationship has bean learned, the algorithm should be able to predict a value $\hat{y}$ based on a new sample x with minimum error. The most difficult part of this process is to find a trade off between the capacity of the model (how much it can learn) and its generalization capability (how good it performs when dealing with unseen samples).

2.2.1 Machine Learning

There are several ML techniques that have been developed in past years, all of them with different limitations and target applications. Some chapters in this thesis implement ML models, in most cases with the goal of evaluating that this programming paradigm could be used for the problem under analysis. These implementations are also useful to establish a baseline in order to evaluate the performance of more complex (DL) models. The ML models implemented in this thesis are:

k Nearest Neighbors

k Nearest Neighbors (kNN) is a very simple algorithm that assigns the class of a sample, among the k nearest neighbors (samples), by majority vote. One downside of this method is that the model needs to memorize the whole training set in order to be able to make a decision, which makes it poorly scalable.

Support Vector Machine

This algorithm is widely used due to its good performance in terms of training speed and unique solution. The main idea behind a Support Vector Machine (SVM) is the definition of a hyperplane that separates the samples in their respective classes; however, not all the samples are used for this task. Only the samples closest to the hyperplane (i.e. the hardest to classify) are used, and they are called support vectors; hence the name. A parameter ϵ defines the epsilon-tube inside of which no penalty is associated in the training loss function. There is also a regularization parameter C that controls the misclassification penalty. To deal with non-linear data, SVM implements the so-called kernel trick that transforms the data into a higher dimensional feature space via a mapping function. The kernel trick has a γ parameter associated to it, that indicates how soft the decision boundary would be. Although SVM is initially defined for classification, it can be applied to regression problems as well. In this case, the acceptable error is defined and the model finds a hyperplane to fit the data.

Random Forest

A Random Forest (RF) is an ensemble or combination of multiple decision trees that generates a final prediction by majority vote. This approach attempts to generate a strong learner, based on the weaker decision tree; which is a simple concept where the model learns a series of questions that define cut-off values for the different features in the dataset. It separates the samples based on how the features relate to those cut-off values, starting at the root and sequentially splitting the data on the feature that generates the largest information gain, in an iterative process that finishes when the leaves are pure or at a predefined depth [7].

Artificial Neural Network

An Artificial Neural Network (ANN) is a concept roughly based upon the understanding of how the human brain works, and it was first introduced in the 1940's [7]. Its basic element is a processing unit called neuron that connects to other neurons and can be arranged in layers in order to fit complex functions. The connections between neurons have weights associated to them, which are the learnable parameters. These weights are updated simultaneously on every pass of the training set, or epoch, and this is performed by an optimization algorithm that calculates the gradient of a loss function. The final goal is to minimize the loss function. Fig. 2.3 shows a simple ANN known as a perceptron, where the neuron computes the weighted sum of the inputs and then applies a step function (or a threshold - so this is also known as a Threshold Logic Unit) in order to generate the output [3]. Modern ANN's replace the step with other activation functions.

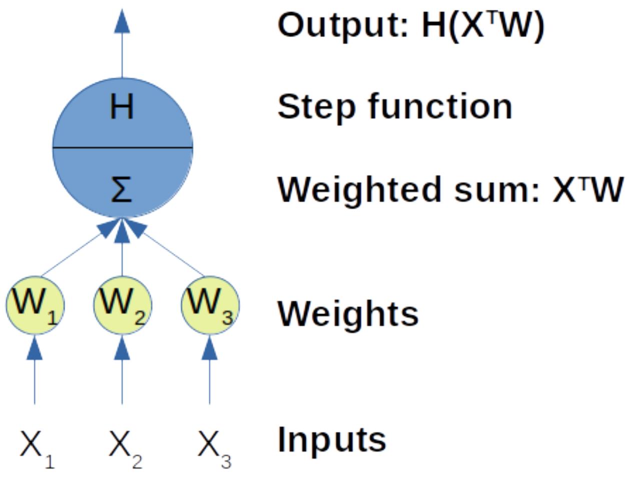


Figure 2.3: Scheme of a simple perceptron [3].

2.2.2 Deep Learning

The main idea behind DL is stacking multiple layers of ANN's in order to capture more complex features with each layer and the term deep comes from the relatively high depth of the network caused by the multiple layers. DL has evolved into two main branches: Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN), both of which are used in this thesis. The former is usually applied to image analysis, while the latter focuses on sequential data.

Convolutional Neural Networks

CNN's are very popular especially in computer vision, given the good results they provide. The main idea behind a CNN is the implementation of multiple layers of feature detectors, that automatically identify patterns within the data. In order to learn those patterns, convolutional layers use a sliding filter (kernel) that covers the whole sample (input feature map), as shown in Fig. 2.4, and generates an output feature map that captures the important characteristics that are finally used by a classifier or regressor layer(s). Many architectures have been proposed and used in the last years, and the first successful implementation was the LeNet-5 [29], that was used to classify handwritten digits. LeNet-5 consists of a stack of convolutional-pooling blocks and has been used as the basis for following architectures that stack more layers and add different types of connections between those layers, such as the case of VGG-16, used in some of the chapters of this thesis.

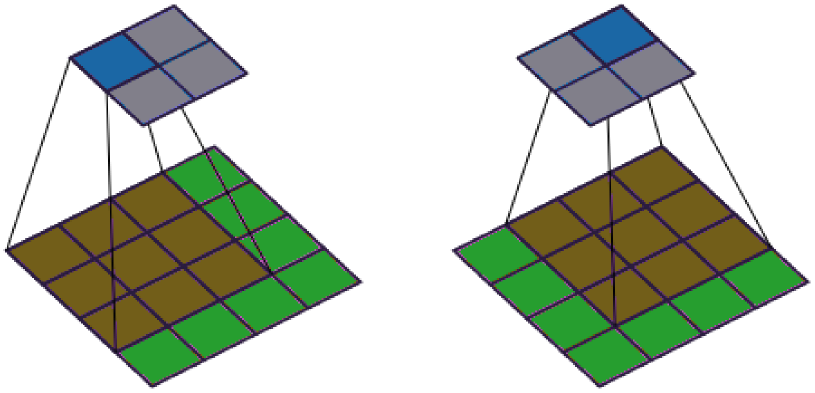


Figure 2.4: Sliding kernels on a CNN [4].

CNN's have been designed for processing visual data [30], which requires two dimensional kernels; however, they can handle any number of dimensions [4]. The works described in this thesis use 1 to 3 dimensional kernels depending on the application. The use of 1D convolutions means that the network uses the same filter for each of the individual n channels at its input. For 2 dimensions, the kernel slides over height and width, and so on.

CNN's are a good choice due to their relatively reduced number of weights (they are shared) and translation invariance (CNN is also known as shift invariant ANN, or SIANN) [31]. The use of the pooling operation contributes to achieve invariance to small local distortions and reduce dimensionality of the feature space [31].

Recurrent Neural Networks

RNN's are a kind of network for processing sequential data, mainly because they are able to scale to much longer sequences than other networks [8]. They are able to achieve this task by keeping information from the past. The original RNN design suffers from vanishing and exploding gradient problems, as the backpropagation through time (BPTT) procedure depends exponentially on weights for each time step, failing to learn information that goes over a few time steps. Thus, new designs such as Gated Recurrent Unit (GRU) or Long Short Term Memory (LSTM) have been proposed. Among these RNN candidates, despite various efforts, no real improvements have been achieved over LSTM. For example, [31] compares six different methods for time series prediction and finds out that including CNN, LSTM and autoencoders, LSTM is one of the best performers on all the evaluated datasets. The most important concept about RNNs is that they connect neurons outputs to their own inputs. This closed loop gives RNNs the ability to memorize information regarding trends.

Long Short Term Memory LSTM has emerged as an effective and scalable model for several learning problems related to sequential data. Previous methods for addressing these problems have either been specifically designed towards a particular case or did not scale well [32]. One downside of this network is that it involves more operations than other options and therefore demands more computational resources. LSTM is a special implementation of the general RNN, that deals with the vanishing gradient problem by means of three gates: forget, input, and output; which are described by equations (2.1), (2.2), and (2.3) [8].

The forget gate controls the self-loop weights, and a unit $f_i^{(t)}$ for the time step t and cell i, is defined by (2.1).

$$f_i^{(t)} = \sigma\left(b_i^f + \sum_j U_{i,j}^f x_j^{(t)} + \sum_j W_{i,j}^f h_j^{(t-1)}\right), \tag{2.1}$$

where b^f, U^f, and W^f are the biases, input weights, and recurrent weights for those forget gates, in that order; and x^t and h^t are the current input and hidden layer vectors, respectively. Note that the final value of the gate is set to be between 0 and 1, due to the sigmoid activation function. Accordingly, the input $g_i^{(t)}$ and output $q_i^{(t)}$ gates are defined in a similar fashion, but with their own parameters.

$$g_i^{(t)} = \sigma \left(b_i^g + \sum_j U_{i,j}^g x_j^{(t)} + \sum_j W_{i,j}^g h_j^{(t-1)} \right), \tag{2.2}$$

$$q_i^{(t)} = \sigma \left(b_i^0 + \sum_j U_{i,j}^0 x_j^{(t)} + \sum_j W_{i,j}^0 h_j^{(t-1)} \right). \tag{2.3}$$

Other Networks for Sequential Data As it was mentioned before, CNN's can be implemented using 1D kernels, which makes them appropriate for sequential data processing, with the advantage of using less computational resources than RNN's; however, this comes with a reduced performance in most cases.

Finally, it is worth mentioning that most current approaches for sequential data involve the use of transformers, which are a networks based on the concept of the attention mechanisms that have revolutionized the natural language processing world, because they overcome some short memory downsides of LSTM networks, making them suitable for long sequences. Transformers also involve the idea of an encoder-decoder scheme, which is very useful for sequence to sequence tasks, such as language translation.

Improving predictions

Some of the custom models that were first tried in this thesis did not perform as expected. Thus, in some cases, some additional techniques have been used to improve the training process or the final predictions. Among these, we have the use of data augmentation, data fusion, pre-trained networks, and ensemble of models.

Data augmentation is a common approach to fight overfitting and it simply means generating artificial samples in order to increase the size of the dataset. The most common and understandable example comes from computer vision and it implies rotating and/or flipping images so that the dataset contains a few times more samples with new images that were not initially there. This thesis uses data augmentation by rotating images (Chapters 4 and 8), using a sliding window to take pieces of the sample (Chapter 5), or using a moving average (Chapter 7).

Another useful technique to improve training is the use of combined data sources to generate a single prediction. This is implemented in Chapter 6 where tabular and sensor-acquired data are combined by a single DL model to generate predictions about the yield. Thus, the model should mix the information to learn the patterns that relate those different inputs to the given output.

As it was already indicated, the use of pre-trained networks also facilitates the DL workflow whenever possible, because we have a model that was trained using millions of samples and we can just fine-tune it to our particular needs. The resulting model can be then combined with other models or used as it is. A good approach for using a pre-trained network is to use the initial layers until they are able to extract the features relevant to a given problem, and then to train a customized dense network on top of it [1]. In Chapter 5 a pre-trained model is used as one of the components.

Finally, the ensemble of models is also implemented in this work. This is used to increase the generalization performance of models by combining them into a meta-model. The error probability of an ensemble is always better than the error of any of its components, provided that their individual error is better than a random guess [7]. This is implemented in Chapters 3 and 5.

2.3 Model Optimization

As it was mentioned in Chapter 1, most of the novelty of this thesis is in the feature selection for model optimization. The most relevant approach used for this purpose is the use of SHAP values, a feature attribution method for model interpretability.

2.3.1 Feature Selection

In ML there is a well know issue called *the curse of dimensionality.* Intuitively, we can think that the more features (dimensions) a dataset has, the more information it carries, and thus the easier it will be to find a model that can fit the dataset. While this might be true up to a certain point, eventually the dataset will likely contain too many features (compared to the number of samples) that it is represented in so many dimensions that the samples are too sparse; in such a way that it becomes impossible to find a model which: (i) can fit the dataset, and (ii) can generalize well to previously unseen samples. At this point, it is important to find ways to alleviate this issue, using different approaches like adding more samples whenever possible (which makes the samples to be closer to each other), projecting the samples to a lower-dimensional space (dimensionality reduction or feature extraction), or by selecting the most important features in the dataset, called feature selection.

Feature selection can be implemented in a few different ways. The most obvious one is having an expert that can choose them by hand; which is useful mostly in tabular datasets that have well known characteristics (a typical example is the one of real estate, where an expert can assess market prices based on their experience). However, when the datasets are more complex and with less understood features, we have to rely on other tools. Two main approaches have been used in this thesis: (i) Sequential Backward Selection (SBS), and (ii) Feature Attribution. Both of these approaches imply the selection of features after an initial model has been trained and analyzed.

It is important to mention that the feature selection approach, besides palliating the curse of dimensionality, allows for the implementation of smaller and faster models, which is one of the goals of this work.

Sequential Backward Selection

When dealing with the selection of a subset of the original features, SBS is a classic algorithm that sequentially removes them from the full feature set until the new feature subspace contains the desired number of features [7], or assures a certain performance level. In this thesis, the process of sequentially removing features is applied in two ways: (i) Chapters 4 and 8 implement a scheme in which different features (temporal or spectral) are removed in different combinations, and (ii) in Chapter 7 only the temporal component is affected, by progressively reducing its size. Both cases look forward to the implementation of DL models of a reduced size by means of selecting a subset of the original features.

Feature Attribution

Feature attribution consists in assigning some level of relevance to the features in a dataset. After attributing this relevance, it is possible to pick only a number of those feature for further analysis or to train new models. This process is relatively straightforward for ML algorithms like decision trees or random forests [7], but it becomes more cumbersome for DL algorithms. Thus, there is the need to find some feature attribution mechanisms that works well for the DL case. In this thesis, the use of a model agnostic interpretability approach is proposed.

The issue of explainability and interpretability arises from the fact that AI models are so complex and constituted by a very high number of parameters (it is usual to have thousands or millions) that they are called black boxes. The name comes from the fact that data enters on one side of the model, and the predictions are obtained on the other, without really knowing how these were obtained, or which features in the sample or the dataset were the ones that influenced those predictions [33]. The most typical example is the algorithm that decides if a person qualifies or not for a bank loan, which might be efficient and resource saving for the bank, but it might not be fair for the person being analyzed. Questions like the kind of biases or possible discrimination of race, gender, religion, among others might appear. These questions have fostered the implementation of model explainability/interpretability methods.

In the particular case of this work, it is proposed that these explanations could be used as a tool to asses the features (mainly either spectral bands or the kind of sensor used to collect the data) in the datasets that influence the predictions of the implemented AI models. The benefit of this approach is twofold, since besides gaining knowledge about the model and data [3], the results are used as an optimization tool because the models could be retrained on the dataset containing only the selected features, which in turn generates smaller models which are a first step towards the implementation of light-weight prediction models that run on computers with very limited resources. It is not always trivial to determine which variables will be good predictors to the output variable, but a relationship between an input variable and the output gives an idea of which features are likely to be important for a statistical model to make good predictions [34].

While in some cases, the distinction between interpretability and explainability might not be clear [35], interpretability refers to the degree of human comprehensibility of a given model or decision [33], and explainability is a previous condition to reach that level of understanding. Thus, most approaches involve the two steps: (i) explainability in the sense of at least feature attribution, and (ii) offering some tools for the analysis of those results, tending to facilitate the understanding of the predictions, the model, and the data. The general workflow of this dissertation focuses on the former, as a tool for feature selection and model optimization; however, Chapter 6 explores the concept and some implications of interpretability.

Explainability methods can be applied to a global (all possible datapoints) or local level (only one or some samples). The advantage of the latter is that it can explain single predictions made by a model, but its results can also be combined to explain the whole or a subset of the dataset. In any case, these explanations are not full scientific explanations (general relationships or scientific laws), but rather causal relationships between the set of variables in a given model. Thus, local approximations can be useful to explore slices of a model for prototyping or debugging [33].

A Brief Description of SHapley Additive exPlanations (SHAP)[1]

SHAP is a game theory technique originally introduced by Lundberg and Lee in 2017 [36] that treats each feature as a player in a game, where that player (feature) contributes to increase or decrease the predicted value. SHAP unifies six previous additive feature attribution methods (Local interpretable model-agnostic explanations (LIME), DeepLIFT, Layer-wise relevance propagation, Classic Shapley value estimation, Shapley sampling values, and Quantitative input influence) that use the same linear explanation model.

SHAP is a local method, which means that it studies the effects of each feature on individual predictions. To explain a prediction $f(x)$ based on a single input x, a function $x = h_x(x')$ maps the original input to a simplified x'. SHAP views any explanation of a model as a simple linear model (Eq. 2.4) that approximates the decision of the original model in the neighborhood of the point of interest.

$$g(z') = \phi_0 + \sum_{i=1}^{M} \phi_i z'_i, \tag{2.4}$$

Where g is the explanation model (approximation of the original model f), M is the number

[1]The content of this subsection corresponds to the Supplementary Information of a paper originally published in Biosystems Engineering [11]. It has been reformatted for inclusion in this thesis.

of features (maximum coalition size in cooperative game theory), $\phi_i \in \mathbb{R}$ (Shapley values in game theory) represents the feature attribution values of feature i, and $z'_i \in \{0,1\}$ is the observed feature (coalition vector in game theory). Eq. 2.4 shows that each binary variable z'_i is given an effect ϕ_i, and these are summed M times to approximate the original prediction $f(x)$. Binary variables ($z'_i \in \{0,1\}$) activate or deactivate a particular feature to produce simplified inputs x'. Local methods like SHAP try to ensure that $g(z') \approx f(h_x(z'))$ for $z' \approx x'$.

SHAP uses the expected value $E[f(z)]$ over the analyzed dataset as the base value (predicted value if no feature is known), and the effects of ϕ_i are then subsequently added for each feature. The sum of these effects adds up to the predicted value.

2.3.2 Reducing the size of the model

As previously mentioned, selecting a subset of features from the original dataset allows for the implementation of smaller models with fewer parameters and thus fewer computational operations. This in turn produces models that can run on computers with limited resources. However, there is more that can be done to optimize these models. Two techniques that are popular for this task are pruning and quantization.

The main idea behind pruning is that the already trained weights (parameters of the model) below a threshold are removed from the network [37]. On the other hand, quantization reduces the precision of the numbers and works well on pruned networks [37]. For their applicability, they are usually implemented together. The goal is to have in the end a model that can be implemented on embedded devices for inference purposes [22, 38, 39].

2.4 Edge Computing and Edge Intelligence

The goal of optimizing AI models is that they could be deployed in devices with limited computational capabilities, that are closer to where data is being collected. This scheme of performing computations on devices that are closer to the edge (where data is generated and collected) is referred to as edge computing, and when that computing involves the use of AI models, it is known as edge intelligence. This is one of the fastest growing trends in enterprise computing, because reducing the distance between where data is collected and where it is processed allows users to react quickly to real-time insights [23].

Fig. 2.5 shows a general scheme of this collaboration architecture, in which edge computing and AI implement real-time small data processing at the edge, and transmit results back to the cloud for long-term and big data processing [2]. This working model is important for this thesis in order to implement inference on the data collected from the sensors [23], shortening the long round-trip time between the sensor and cloud or the datacenter that in many cases prevents AI from further playing its role [2], as is usually the case of the applications analyzed in this thesis.

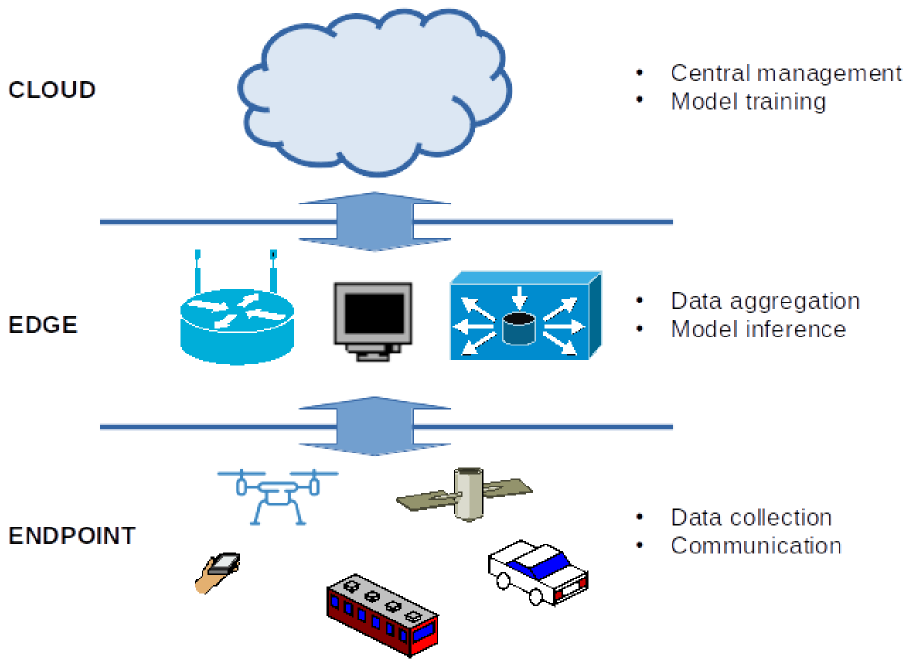


Figure 2.5: Cloud-Edge-Endpoint Collaboration Architecture [2].

In order to validate the hypothesis that the optimized models were small enough to be deployed into portable devices, these were implemented and tested on Raspberry Pi boards. From there, they could be used as on-site processing elements directly connected to the acquisition devices, which could contribute to the edge intelligence component. Models 3 and 4 of the Raspberry Pi computer were used for Chapters 7 and 8, respectively. The main challenge in these cases was running newer versions (and not yet officially supported) of Tensorflow. Tests and results will be explained in the aforementioned chapters.

Part II

Custom Machine Learning Models

3 Ensemble Learning for Improving Generalization in Aeroponics Yield Prediction [2]

This chapter presents an introduction to the use of ML techniques, particularly for analyzing data collected from sensors, in-situ. The goal of this chapter in the context of this thesis, is to verify if ML could be a viable approach to processing data obtained from various kinds of electronic sensors. As it has been previously described, the customization of AI models is important when data collected is unconventional, as it is the case here. Thus the design and implementation of tailored architectures, and the procedure to do so is relevant throughout this thesis. In later chapters we will explore optimization options that will add more value to these experiments.

Agriculture plays a crucial role in economy of several countries and yield prediction is essential for production management and operation planning. ML is a growing trend in determining yield as a complex function of multiple input variables. Aeroponics is one of the efficient sustainable farming methods and allows all season farming despite hostile outdoors growing environment. In this chapter, yield prediction in aeroponics is studied using ML. We have compared and analyzed three popular supervised ML methods - DNN, RF based on decision trees and Support Vector Regression (SVR). Air quality and water quality measurements including temperature, humidity, CO_2, pH and Total Dissolved Solids (TDS) are used for yield prediction. Other static inputs such as number of days before and after transplant are also used. Six crops are studied (garlic chives, basil, red chard, rainbow chard, arugula, and mint). DNN performs particularly well with the prediction. The root MSE, MAE and coefficient of determination (R^2) are calculated to estimate the efficiency of the method. Mean square error and R^2 score of DNN are 0.10 and 0.67, RF follows DNN correctness with MSE and R^2 of 0.12 and 0.62, and SVR achieves 0.18 and 0.45 respectively, all of these values over the validation dataset. In addition to individual models, the two top performing models are combined as an ensemble model to improve overall performance, which shows an average

[2]The content of this chapter is originally published in ISCAS 2020 [5]. The manuscript has been reformatted for inclusion in this thesis.
Julio Torres-Tello: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data Curation, Writing - Original Draft. Suganthi Venkatachalam: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation. Lyman Moreno: Resources, Data Curation, Writing - Review & Editing, Funding acquisition. Seok-Bum Ko: Conceptualization, Writing - Review & Editing, Supervision, Project administration, Funding acquisition.

R^2 score over the whole dataset divided by crop of 0.81.

3.1 Introduction

There is an increasing trend in demand for healthy fresh foods. On a global scale, there is a need to meet food demands and predicting yield with better accuracy is important to plan import and export in case of deficit or excess [40]. Methods like green house, hydroponics, and aeroponics allow for year around harvest, protection from harsh cold or warm weather, portability, cultivation of diverse crops and disease free cultivation. Among these alternatives, aeroponics has been evolving as a promising and efficient modern day plant growing method in multiple countries [41]. Studies in [42] show that, compared to traditional farming, aeroponic farming shows an increase of yield ranging from 7% to 65% based on the type of crop. Along with faster crop cycle, they also have reduced water, pesticide and fertilizer usage [43].

Aeroponics systems are soil-less growing methods where plants get their nutrition from a mix of water and nutrient tonics. Aeroponics are more commonly closed or semi-closed systems, with automated controlled variables that influence the growth of the plants. Unlike traditional field agriculture, air quality variables such as temperature, humidity, CO_2 and light can be maintained within specific ranges by automated systems. Water quality variables including pH and TDS can be controlled as well, and sprayed to roots to deliver nutrients. They also allow for more dense growing environment since plants can be grown in a stacked tower structure.

Yield prediction is a complex task based on various parameters including crop type, environment and water quality. Human based yield prediction is time consuming and prone to error. The outcome of the harvest is hard to predict in advance, but yield knowledge can help growers model and plan price, supplies, and future techniques. Recently, ML is becoming an important tool for predictions in medicine, robotics, economic sciences, climatology and yield prediction is no exception.

Furthermore, yield prediction could be the basis of a fully automated control system in which yield could be maximized by setting the variables under control (light, nutrients, etc.) in the best possible way. Such a system would increase revenues by using the minimum possible resources to produce the maximum yields. This is extremely valuable in aeroponics, where the farmer has much more control over the environmental conditions when compared to traditional farming.

In traditional agriculture, yield predicted from remote sensing data is extensively used. NDVI is used as an main indicator of yield projection [44, 45]. However, prediction of crop yield with better accuracy demands other inputs like water nutrients, fertilizer and pesticides information. In controlled aeroponics farming, where sensors and controllers are an integral part, this information is used for yield prediction and typically without the need for remote sensing.

In our chapter, data is collected from a sustainable and organic aeroponic farming device called AeroPod of Farm Boys Design, a corporation based in Saskatchewan, Canada. A typical Aeropod system and a single stacked tower in AeroPod are given in Fig. 3.1.

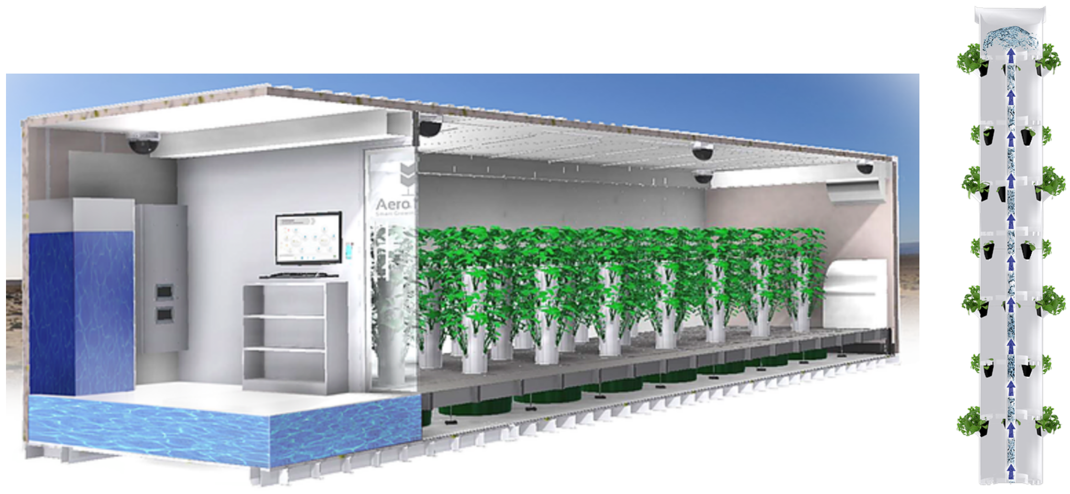


Figure 3.1: Aeroponics container (a) and one of the towers in AeroPod (b).

Three ML techniques - DNN, RF and SVR are studied and compared. DNN is a neural network structure with multiple layers which has dramatic breakthroughs in multimedia recognition, object detection and classification [46, 47]; RF, proposed in [48] is used to improve prediction accuracy by taking in account large number of decision tree models. SVR [49] is an effective regression method with an advantage of one global optimum compared to neural network approach. Of the three methods, DNN is found to have better yield prediction, followed by RF. An ensemble regressor that puts together these two models is analyzed as well.

This chapter is organized as follows: Section 3.2 discusses what data is used as inputs to the models and three ML models used in our work are explained in detail. Section 3.3 discusses mean square, mean absolute error and R2 score of all our models. Section 3.4 concludes the chapter.

3.2 Methodology

3.2.1 Growing Methods and Data Collection Materials

The crops taken for studying yield prediction are garlic chives, basil, red chard, rainbow chard, arugula and mint. Seeds are sown in rockwool grow cubes medium in a tray, until they germinate and attain early stages of growth. Then, they are transplanted to stacked towers in AeroPod. Rockwool absorbs nutrients and retains oxygen for fast growth. Each layer in a tower consists of few spots which can accommodate a rockwool cube. In the AeroPod, nutrient mixture comes in

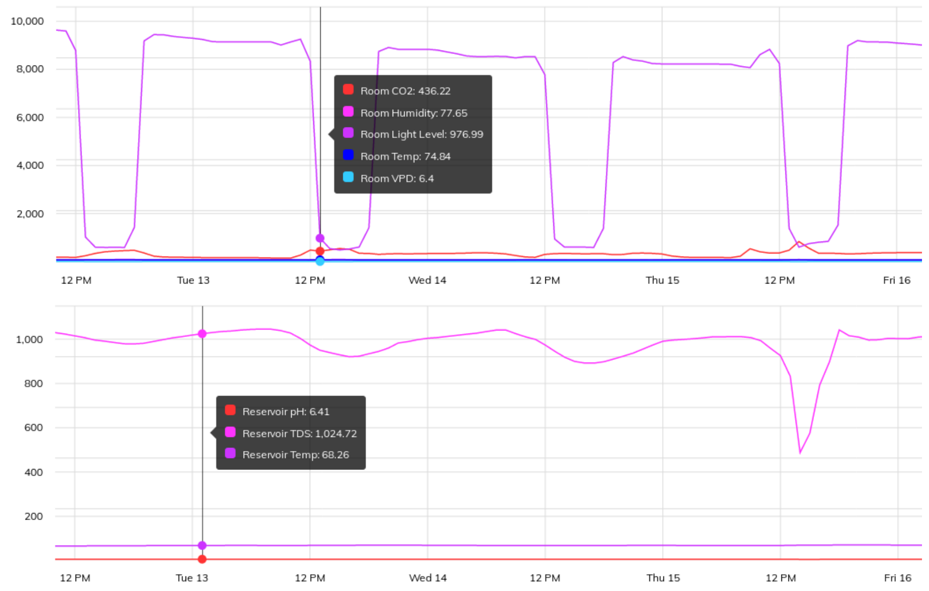


Figure 3.2: Screenshot of the data acquisition platform.

contact with rockwool cubes at constant intervals, thereby nourishing the roots. When the plants reach the expected harvest growth, they are harvested and yield is measured as weight per spot.

Air quality and water quality sensors are implemented in AeroPod container (an example of the data collection system is shown in Fig. 3.2), whose measurements are valuable inputs to ML algorithms. Environment and nutrient input variables include average (calculated for the time frame in which each plant has been in the AeroPod) of hourly values of room Carbon-di-oxide levels, room relative humidity, room light level, room temperature, room vapour pressure deficit, water pH, water TDS and reservoir temperature. Number of days in tray, number of days in tower, harvest number (how many times the plant has been harvested since first transplanted to AeroPod) and grow number (how many times a plant has been transplanted into that spot) are given as inputs to the model. The label of the ML model is yield measure of weight per spot (oz/spot). A block diagram of our model is given in Fig. 3.3. For this study, 200 samples have been collected between November 2018 and August 2019.

3.2.2 ML models and training parameters

The training process of all the algorithms has involved k-fold cross validation (k=10). This resampling technique without replacement is applied in order to obtain an error rate as independent as possible of the train-validation split of the dataset.

The only pre-processing technique applied to this dataset is the standardization of the features

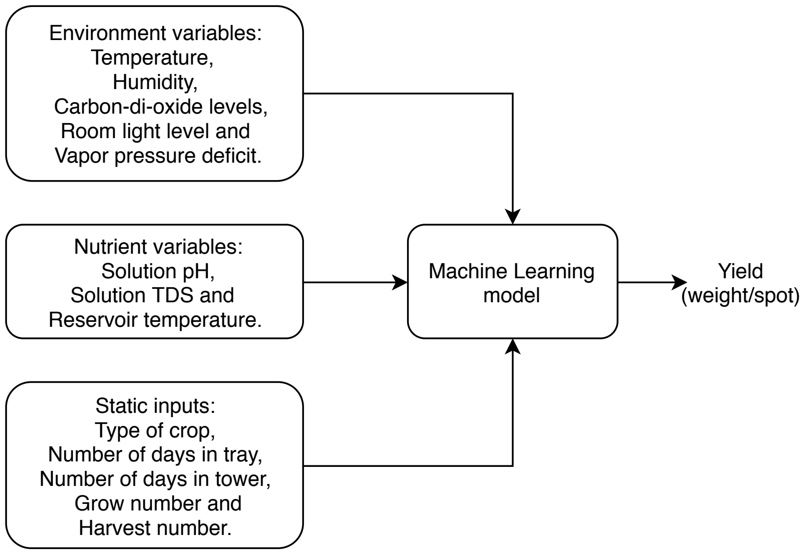


Figure 3.3: Block diagram of the machine learning approach used in our analysis.

by subtracting the mean and scaling to unit variance, according to (3.1):

$$y = \frac{x - \mu}{\sigma}, \tag{3.1}$$

where x is the original feature, y the resulting one, μ is the mean of the samples, and σ is their standard deviation.

3.2.3 Support Vector Regression model description

SVR is used in yield prediction to provide an estimate of output as a non-linear function of inputs. The kernel function plays a crucial role in transforming inputs into higher dimensional space. Here, we use a non-linear kernel called Radial Basis Function (RBF). In our model, the influence of single training example gamma is taken as 0.1, factor C is 100 which decides the increase or decrease of margin, and a margin of tolerance epsilon = 0.1.

3.2.4 Random Forest model description

RF is an effective classification and regression method. It involves ensemble learning of multiple decision trees. We implemented an RF algorithm and tuned the hyperparameters in order to obtain the best possible results. The results of the tuning process indicated that the ideal number of trees for the ensemble is 100 and their maximum depth must be 20. One important difference between RF and the previous ML model is that RF does not require data normalization.

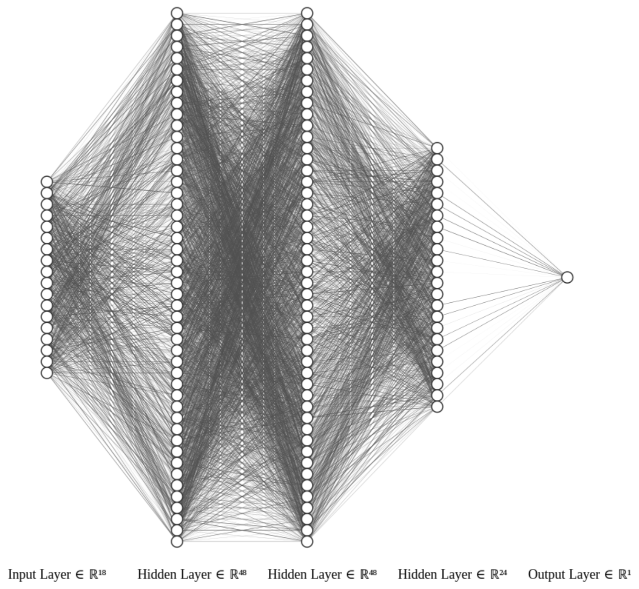


Figure 3.4: Architecture of the DNN implemented for this work.

3.2.5 Deep Neural Network model description

The DNN model used in our work is given in Fig. 3.4. The model uses rectified linear unit (ReLU) as activation function for all layers, including the output. Rectified linear function can be given as:

$$R(z) = max(0, z), \tag{3.2}$$

which is basically a linear function that cancels out any negative value. Usually, regression problems use linear activation for the output neuron; however, in our case we had the issue that in some cases the DNN would predict negative yields (mathematically possible due to the linear response of the function, but impossible from the physical interpretation of the yield) so we decided to make those predictions zero (i.e. to use a ReLU activation function). This modification improved the results of the predictions of the DNN, and not only for the values previously predicted as negative, but for all of them.

Our DNN has three hidden layers with 48, 48 and 24 neurons respectively. After trying different combinations of number and size of layers and regularization techniques (we tested dropout and L2 with different coefficients), the architecture shown in Fig. 3.4 was our optimal solution, without the need of using any other method to fight over-fitting than the reduced size of the network itself, which in the end has 4,465 weights. Some of the important training parameters are the optimizer, loss function and metrics (adam, MSE and MAE, respectively), and that the model was trained for between 101 and 327 epochs per fold (early stopping was implemented), using a batch size of 4. The average of training and validation losses of this algorithm are shown in Fig. 3.5.

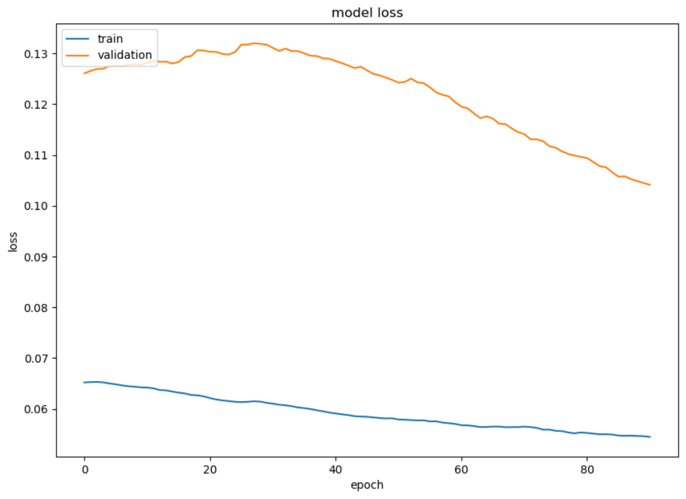


Figure 3.5: Training and validation losses (10-fold average) of the DNN model.

3.2.6 Ensemble of DNN and RF model description

A common technique used in the field of ML in order to increase the generalization performance of models is to combine them into a meta-model, in which the error probability of an ensemble is always better than the error of any of its components, provided that their individual error is better than a random guess [7]. As we will see in the following section, the best performing algorithms are RF and DNN, and they are used to create the ensemble learner finally used in this work.

3.3 Results and Discussion

Our ML models were implemented in Python 3, using Scikit Learn for SVR and RF, and Keras with Tensorflow backend for the DNN. All of them were trained on an machine with 12 Intel(R) Core(TM) i7-8750H CPU's at 2.20GHz, 16 GB of RAM, and a Nvidia GTX 1060 GPU used only for the DNN.

The main results of the regression process over the above described dataset are summarized in Table 3.1, where we can see that the best performing model is DNN, when measuring both the error (either MAE or MSE) and the coefficient of determination. However, RF shows similar although not as good results as DNN. Fig. 3.6 is a plot of the predictions generated by the (a) DNN and (b) RF models for the training and validation sets, after the 10-fold training process.

Both, Table 3.1 and Fig. 3.6 show that these models have a good performance when dealing with the training set, but their error increase when dealing with the validation set (especially SVR shows a poor R^2 for validation), which means that they do not have a good generalization power. In our implementations we have tried to tackle overfitting in many ways, but we believe that in our case the only remaining options are the implementation of a meta-model by means of an ensemble

Table 3.1: Mean Squared Error (MSE), Mean Average Error (MAE) and coefficient of determination (R^2), for the training and validation sets.

	MSE		**MAE**		**R^2**	
	train	**val**	**train**	**val**	**train**	**val**
SVR	0.08	0.18	0.15	0.26	0.86	0.45
RF	0.05	0.12	0.13	0.21	0.90	0.62
DNN	0.05	0.10	0.11	0.18	0.91	0.67

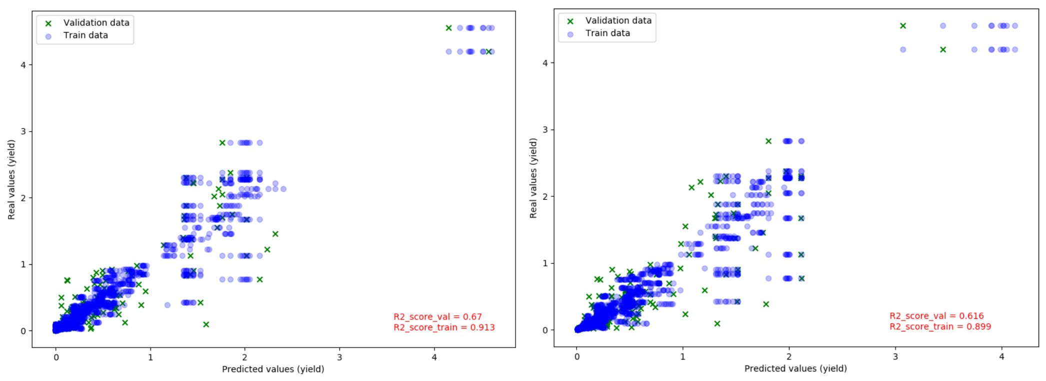


Figure 3.6: Predictions over the training and validation sets with (a) the DNN model and (b) the RF model.

of regressors, and as the main future work, to increase the size of our dataset by collecting new samples.

Additionally, it was important to evaluate how the models perform on the different crops that conform our dataset. Table 3.2 presents the coefficient of determination calculated with the two best performing models (RF and DNN) over each of the six different crops. It is interesting to note that the predictions are acceptable for most of them, except for the rainbow chard which will need further analysis of the phenotype of this plant.

Finally, we built an ensemble of these two models in order to improve the prediction and generalization capabilities of our ML models. This ensemble gave the best result with weights of 48% to DNN and 52% to RF, and obtained the results presented in Table 3.2, which are better that the individual-model predictions, and it is quite obvious that this increases the coefficient of determination especially for the case of garlic chives. As a matter of example, Fig. 3.7 shows the results of the ensemble model for the best (Garlic Chives) and worst (Rainbow Chard) predicted crops.

These results show that with an adequate tuning of weights in the ensemble model, we could tackle future generalization issues when we increase our dataset, as a future work.

Table 3.2: Coefficient of determination (R^2) calculated over the complete dataset divided by type of crop, for the two best performing ML algorithms and the ensemble of the two.

	RF	**DNN**	**Ensemble**
Garlic Chives	0.80	0.71	0.89
Basil	0.85	0.89	0.88
Red Chard	0.83	0.86	0.87
Rainbow Chard	0.70	0.61	0.66
Arugula	0.74	0.73	0.74
Mint	0.80	0.83	0.82
Average	0.79	0.77	0.81

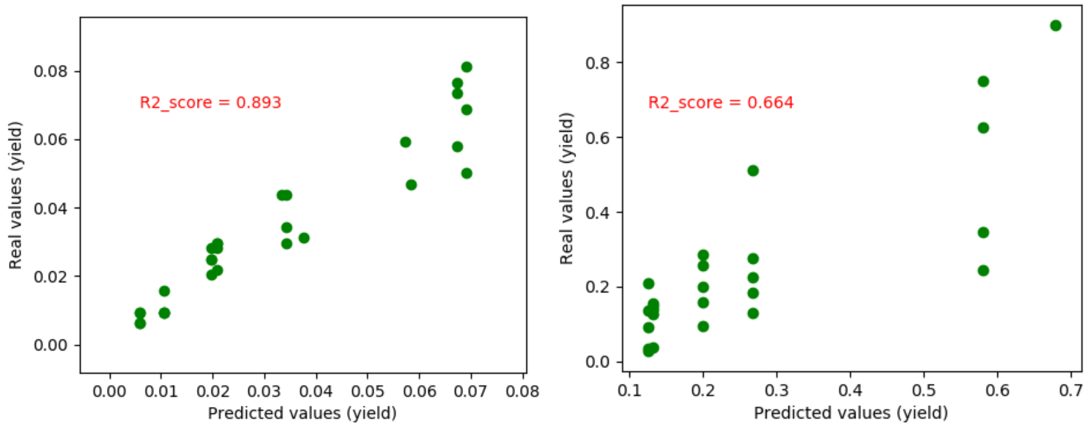


Figure 3.7: Predictions with the ensemble of models, over the best -Garlic Chives- (a) and worst -Rainbow Chard- (b) performing crops.

3.4 Conclusion

This work presents a yield prediction model for aeroponic crops in a controlled environment, based on the environmental variables that can be controlled and/or measured in the production system. For this purpose we have used 200 samples covering 6 different crops, and we have reached an average coefficient of determination value $R^2 = 0.81$ when testing an ensemble of the two best models (DNN and RF) over the whole dataset separated by type of crop. This and the other results presented in this chapter show the potential for implementing a yield prediction model that could become the first step towards the full automation of a crop production system based on aeroponics, such as the one shown here.

This work draws the main path to be followed in order to have a robust yield prediction (and automated production) tool that would eventually require less human intervention with a bigger profit margin. The next step towards that goal is the collection of more data, organized in such a way that it does not present the issues that have had to be addressed for this work. We believe that it would allow us to generate more accurate and self-explanatory results, that could be easily implemented in a final control system.

Part III

Feature Selection in Deep Learning

4 Identifying Useful Features in Multispectral Images with Deep Learning for Optimizing Wheat Yield Prediction[3]

This chapter explores the use of DL for the analysis of multispectral images taken at different times and locations. The main challenge comes from the additional dimension (time) and two extra spectral bands (red-edge and near infrared), when compared to traditional image processing, which means that there was the need to train a model from scratch (another customized model). Given the aforementioned constraints, it was decided to also explore the possibility of identifying the features (either in time or spectrum) that are more relevant for the predictions of the DL model, with the final goal of model optimization. The results of this research could be a tool for the development of more efficient sensors and strategies for data collection in plant phenotyping that would acquire images with only the useful spectral bands, or at the best stages of the crops.

Since unmanned aerial vehicles have been utilized in plant phenotyping, they have revolutionarily improved its accuracy. In this chapter, we introduce a deep learning based approach for optimizing the yield prediction process of spring wheat (*triticum aestivum*), using multispectral images. We assessed both the temporal features to find the most valuable time to take images, as well as the contribution of spectral bands. We processed full stage multispectral images from four site-years (two sites during two years) of a wheat breeding project, and determined the prediction accuracy of the image-based predicted yields and compared them to the harvested yields taken in the field. The results compared the wheat images throughout the season and validated the most crucial flying times for acquiring images were at late-heading, late-flowering, dough-development, and harvesting stages. The two most useful colour-bands for yield prediction were red and red-edge. We found that removing these bands significantly decreased the prediction correctness. The results of this research could be a tool for the development of more efficient sensors and strategies for data

[3]The content of this chapter is originally published in ISCAS 2021 [10]. The manuscript has been reformatted for inclusion in this thesis.

Julio Torres-Tello: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data Curation, Writing - Original Draft. Seok-Bum Ko: Conceptualization, Methodology, Writing - Review & Editing, Supervision, Project administration.

We would like to thank the Plant Phenotyping and Imaging Research Centre (P2IRC) for providing the dataset.

collection in plant phenotyping.

4.1 Introduction

The Government of Canada forecasts a production of 33.6 million tonnes of wheat for the 2020-2021 season [17]. Traditional methods for the selection of high yielding wheat varieties are time consuming and prone to error [15]; therefore, optimization of yield prediction is required.

Phenotype is the observable expression of an organism and the interaction effects of genotype and environment [50]. Unmanned Aerial Vehicles (UAV) are a relatively novel phenotyping platform in Remote Sensing (RS) that allow a fast, high-throughput, and high-quality image of a large field-base experiment at different crop stages. High-throughput plant growth observations aim to improve the prediction accuracy of trait determination and selection speed [51]. Plant breeders are especially interested in phenomics because of the potential to quickly image and select desirable genetic lines.

Finding important features for yield prediction is also relevant for the development of small, low-cost sensors that focus on capturing only the information that is needed for a given task. It is also important for the implementation of data collection strategies that could even become automated. All of this could open new research possibilities in the development of IoT based solutions for plant phenotyping.

Neural Networks (NN) have emerged as one of the most important tools for data analysis, and within them there is an area called DL in which the NNs have multiple layers ranging from just a few to hundreds. DL is one of the main trends in artificial vision due to its good results when visual characteristics are hard to identify. This is the case of the images available for this work. A popular form of DL, especially when dealing with images, is the CNN, which is composed of multiple convolutional layers. In such networks, each layer generates a feature map that extracts essential and unique features within the image that will eventually be used by the network for the characterization of a given sample [9].

Most studies found in literature related to RS and plant phenotyping, have focused on generating predictions based on vegetation indices, such as chlorophyll content and NDVI [27]. Similar works on yield prediction with multispectral images include, among others, [52] that found that certain colour channels can be more useful than others for yield estimation, such as narrow wavelengths of the red-edge channel, which could be used as an alternative measure for leaf area index (LAI), or [53] that found that red and near-infrared (NIR) portions of the spectrum can be used to develop NDVI, which indicates a high correlation to the yield potential in potato, cotton [54], and rice [55]. Thus, multispectral images are important not only because they provide spatial information just like any other image, but also because they give additional information regarding the reflectance spectrum of an object. In the case of our dataset, we have information on the red-edge and NIR channels that would not be available in traditional RGB images.

In recent years, studies utilizing DL models and high-resolution image technology were able to develop more precise predictions of yield. A 2019 study in corn yield prediction found that a DNN model had a superior prediction accuracy on average yield [56]. [57] used CNNs that outperformed the yield prediction when compared with the traditional RS based model.

In this chapter, we introduce the development of a DL model for optimizing yield prediction of spring wheat using multispectral images. We assessed both the temporal features to try to find the most valuable UAV flight times, as well as the contribution of the spectral bands. We utilized several multispectral images from a wheat breeding project and correlated the predicted yield values to the actual yield measurements taken at harvest.

The first goal of this study is to implement a DL model able to predict the yield from each multispectral image. The second goal is to determine which days and spectral bands contribute the most to yield prediction. This research could eventually lead to the speed up in the variety selection process, and with more training data, become available as a tool for plant breeders. Also, our results could be an aid for the development of more efficient sensors and strategies for data collection in plant phenotyping.

This chapter is organized as follows: Section II describes the materials and methods used in our work, including the data collection process, the implemented DL model and the process followed to find out the best features (dates and channels) for yield prediction. Section III presents the results of this work, and Section IV concludes the chapter.

4.2 Materials and Methods

Images were initially pre-processed (stitched together) in order to create an orthomosaic and then annotated with the yield information, which is used by our model. An example is shown in Fig. 4.1.

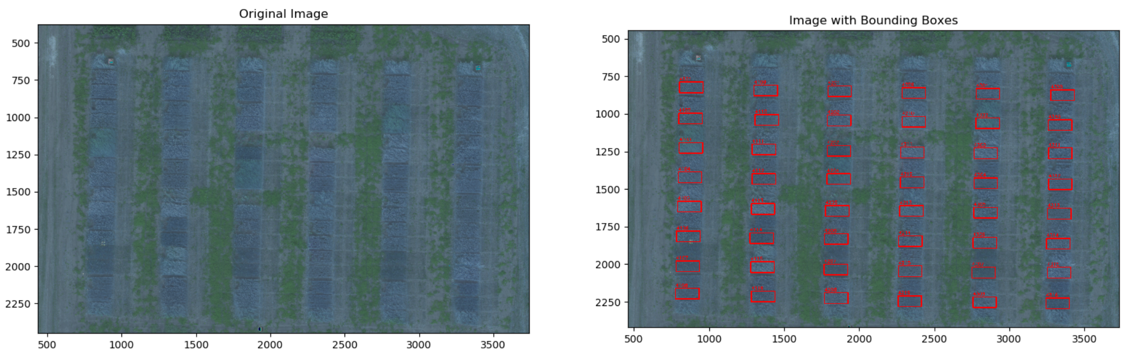


Figure 4.1: Example of an orthomosaic image. Left: Plotted using only RGB channels. Right: Bounding boxes (annotations) over the wheat plots.

4.2.1 Site Description and Image Collection

The wheat breeding sites are located east of Saskatoon, Canada, at the Kernan Research Farm. Trials were set up using a Randomized Complete Block Design (RCBD) including 16 wheat varieties and 3 replications (or experimental blocks). Two fields were seeded near the middle of May, during the 2017 and 2018 growing seasons, for a total of four trials.

In total, we had 48 orthomosaic images to train and test the network, from 12 different dates across four site-locations (two locations in two different years). From the images, we extracted 48 plots per location for a total of 192 samples, each containing information of 12 dates and 5 colour bands. Out of these 192 samples, in order to get predictions for all the wheat varieties in the four locations, the dataset was split in 144 images (2 replicates) for training and 48 (1 replicate) for testing (75% - 25%). This took into account that the same wheat varieties are present in each repetition and in all locations, so we trained the model with the same varieties that we tested (which was not guaranteed with a random split of the dataset).

In order to increase the number of samples, we used a common data augmentation technique, which consists of rotating the images. Given the non-square dimensions of the crops, the images were only rotated 180 degrees, which gave us a training dataset with 288 samples. Finally, we split the training set into training and validation, randomly dividing the dataset in 80% and 20% respectively.

To summarize, the dataset was split by crop replication to extract the test set (e.g., Replication 1 in four locations). On the remaining two replications (for the four locations), we applied data augmentation and then we split randomly into training and validation. And this process was performed three times (one per replication) in a 3-fold cross validation fashion, in order to guarantee a small variance of the error in the predictions with respect to the split [7] and to obtain predictions for all the samples in our dataset.

4.2.2 The Deep Learning model

The DL model chosen for this work was based on CNN. In the field of artificial vision, there are some well known models with good performance. For our work, we have chosen VGG-16 as a reference architecture, given its proven good results, and easy implementation and modification (as it will be important to reshape the network). There are new architectures that might offer greater performance, but they are usually much deeper and complex, especially when trying to adapt them to achieve our goals.

Fig. 4.2 shows the architecture of the CNN model implemented for this study. In this case, there are five convolutional-pooling blocks that are used to extract the useful features from the images, and finally we have a block with two fully connected layers that make the final prediction.

However, as it is shown in Fig. 4.2, our model is not exactly the same as the original VGG-16 architecture. The main difference is that we have implemented convolutional layers with three dimensional kernels. Despite 2D convolution being usually implemented for image analysis, it can be generalized to N-D convolutions (3D in our case), given that we have a dataset that contains a temporal dimension as well (images of the same location taken at different dates). The main difference in the latter case is that the kernel would be a cuboid and would slide across the height, width and depth (time) of the different input feature maps [4].

This model considers the use of all dates and all spectral bands. In the tests, these 2 dimensions are variable according to the different experiments.

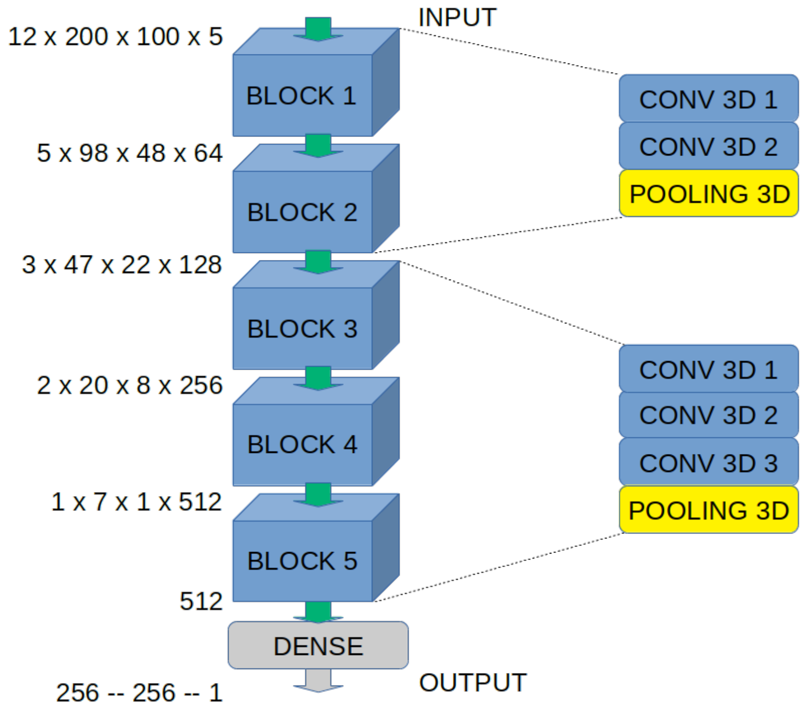


Figure 4.2: Block diagram of the implemented DL model (based on VGG-16).

The CNN described above was implemented in Keras with Tensorflow backend. Initial tests were done on a GeForce GTX1060 Nvidia GPU, and in order to speed up the analysis all the final and reported tests were performed on a Tesla V100 Nvidia GPU. The training process took place for around 500 to 1000 epochs, depending on each case (early stop implemented) and it used a batch size of 8, Adadelta optimizer, and dropout after the convolutional blocks (40%) and dense layers (30%) in order to avoid or at least minimize overfitting. Mean Absolute Percentage Error (MAPE) was used as a measure of the performance of the CNN in each case.

4.2.3 Experimental set up to decide which traits contribute to the yield prediction

Once we had a model that was able to predict yield from the sequence of multispectral images, we tested which dates and channels were more important for this prediction. For this process, we used reduced versions of the model already introduced in Fig. 4.2. In the case of the temporal analysis (best dates to fly the UAV), we used a 2D version of the model which was trained and tested for predicting yield based on one date at a time. For the spectral analysis (best channels), we only discarded one channel at a time. This later approach was used considering that there is a spectral correlation in the images, which is in fact the basis of the popular use of vegetation indexes in plant phenotyping.

To determine which temporal features to select for the final computer model, we ran the results for each individual flight day alone. From there, we selected which days had the greatest coefficient of determination (R^2) value. This was considered a relevant feature (date) only when the (R^2) value was higher than the baseline, which for this process is the result of the 3D CNN model because it was tested with all 12 dates at the same time.

When determining which colour bands were most useful in optimizing yield prediction, we first ran the control test which included all 5 spectral bands of the multispectral images, with the 3D CNN model. Following this, we took out one selected colour band at a time to see how the results would reflect the loss of that spectral band, leaving a four-channel image at a time in different combinations. In this case, a relevant feature (colour) was considered when the results had a much lower R^2 value than the baseline because in this case, that would be a measure of how much the absence of such colour would affect the yield prediction.

4.3 Results

The final trained model, for all cases, had around 10 million parameters. Fig. 4.3 shows the relationship between real and predicted yields for the results generated with the 3D CNN network. These predictions are the baseline to compare the results of the other experiments and decide which dates and spectral bands contribute the most to the predictions.

4.3.1 Effect of different imaging stage for yield prediction

These results are shown in Table 4.1, where we have highlighted the flight dates that achieve an R^2 value higher than the baseline (0.676, using all dates). We see in the table that the best dates to flight the UAV are 5, 7, 11, and 12. These dates correspond to 62 (late-heading), 72 (late-flowering), 91 (dough-development), and 95 (harvest) days after seeding (DAS), respectively, in the

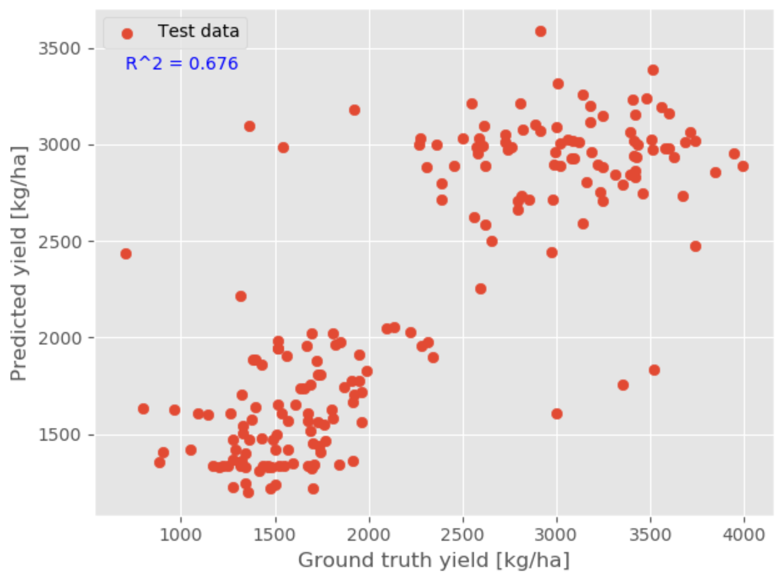


Figure 4.3: Real and predicted values for the whole test set, using all features.

development stages of the crop.

4.3.2 Effect of colour channel contribution for yield prediction

These results are presented in Table 4.2. There we have highlighted the lowest values of the R^2 coefficient. We must remember that in this case we have removed one band at a time, so these low values mean that when the multispectral image does not have those channels, the predictions are bad, or in other words, those low values mean that red and red-edge channels are the most important for yield prediction. We can see that by removing any of the other three channels, the coefficient of determination is similar or even higher than the baseline value (not removing any channel). Of the other three channels, only removing the green one affects the predictions. Although red and red-edge channels are expected to be important features, since they are used for vegetation analysis [26, 52, 53, 54, 55], it is somehow unexpected that NIR did not seem to contribute that much to yield prediction. It could be that when this channel is absent, red and red-edge compensate the missing information.

4.3.3 Representations learned by the CNN

Some additional and interesting results of this work are the graphical representation of the patterns that the filters in the network look for in an image because that is what DL for image processing is about: finding interesting patterns. Fig. 4.4 shows as an example, the patterns that the filters of the last conv-pooling block look like. This corresponds to the original 3D CNN network, trained with all dates and all colours. It is interesting to see that the filters look for rugous patterns in the

Table 4.1: R^2 values, for independent dates

Dates	***Coefficient of Determination***
all	0.676
1	0.631
2	0.305
3	0.622
4	0.619
5	**0.766**
6	0.575
7	**0.698**
8	0.170
9	0.504
10	0.659
11	**0.728**
12	**0.747**

Table 4.2: R^2 values, for independent channels

Channel not used	***Coefficient of Determination***
none	0.676
Blue	0.710
Green	0.638
Red	**0.240**
Red-edge	**0.244**
Near-infrared	0.716

centre of the images, which in the spatial context of the crop, probably would relate to how the wheat plants are packed and organized within the plot.

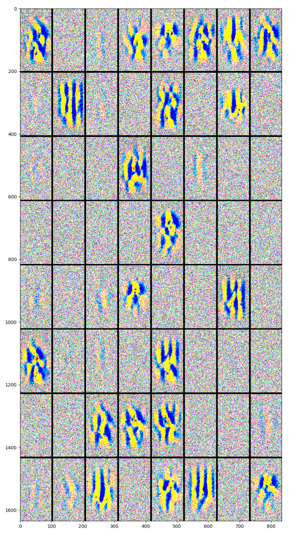


Figure 4.4: Filter patterns that the 3D CNN model looks for in the samples, after the last conv-pooling block.

As previously mentioned, the main objective of this work was to find the useful features that allow yield prediction, which would lead to the implementation of a more robust yield predicting model. In order to achieve this, it is also important to collect and use bigger datasets. For this work, we have only used images from two locations corresponding to two growing years (four site-years in total), which could make our model weak against any environmental variations that may occur within a growing season. Due to differences in precipitation between 2017 and 2018, yield varies considerably in our dataset and it gives us some robustness. These differences are clearly seen as two clusters of yield values in Fig. 4.3.

4.4 Conclusion

The results presented in this chapter show that a DL model is capable of finding useful features within a multispectral image dataset for yield prediction purposes. The comparison between single-date image and whole season images found the best timing for phenotyping wheat yield prediction were late-heading, late-flowering, dough-development, and harvesting stages. Furthermore, even a single flight at one of these stages would be better than using an entire season of images. Exploiting the optimal features would be a methodology to improve yield prediction.

Analyzing the predicting efficiency among the five bands in the multispectral images allowed us to determine that the model that lacked red and red-edge bands decreased the prediction accuracy. We hope this provides a direction to utilize and develop vegetation indices for wheat yield prediction.

This research found that using less spectral bands would also allow us to use software tools designed to handle only three channels; for example, a pre-trained VGG network using green, red and red-edge. That could be matter of a future work.

Finally, this work is of great value since it gives us guidance relative to data collection, storage and processing. As well, this work lets us know which characteristics we should take into account in the implementation, training and validation of our final yield prediction DL model for spring wheat. Furthermore, these results could be used for implementing more efficient data collection strategies and in the design of sensors tailored for a specific application that would acquire images with only the useful spectral bands.

5 A Novel Approach to Identify the Spectral Bands that Predict Moisture Content in Canola and Wheat[4]

This chapter continues to explore the relevance of different features for the predictions made by DL models. Instead of using a traditional approach, as in the previous chapter, we now propose the use of a game-theory-based technique designed for model explainability/interpretability (SHAP). This method tries to figure out the reasons why a model predicted a certain value, and this could be used to analyze which features generate certain predictions. One advantage of the proposed technique is that this can preserve the physical meaning of the original spectral bands, providing more meaningful results than feature extraction algorithms like PCA. Another benefit of the proposed approach is that we can generate explanations of the trained model, focused on a target (moisture content), a feature (spectral or spatial), or a subset of the HSI. If a comprehensive explanation is needed, the per-pixel contributions can be aggregated into a single SHAP value per covariate (feature), due to its additive property. As it has already been mentioned, this approach could eventually become a resource for the elimination of redundant bands, which is helpful for saving computational resources and improving model performance. Moreover, wavelength selection is essential for the implementation of real-time HSI applications.

Due to the relevance of agriculture in economy and human development, the inclusion of technology in this activity is of utmost importance, and moisture content prediction is relevant for assessing the degree of maturity of a crop, which relates to efficient harvesting and quality control. This chapter presents an accurate deep learning model for the prediction of the moisture content of canola and wheat crops, based on hyperspectral images taken by several drone flights. This model serves as the starting point for a supervised band selection process that involves a novel

[4]The content of this chapter is originally published in Biosystems Engineering [13]. The manuscript has been reformatted for inclusion in this thesis.

Julio Torres-Tello: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data Curation, Writing - Original Draft, Visualization. Keshav Singh: Data Curation, Formal Analysis, Resources, Investigation. Steve Shirtliffe: Data Curation, Formal Analysis, Resources, Investigation, Funding acquisition. Seok-Bum Ko: Conceptualization, Methodology, Writing - Review & Editing, Supervision, Project administration, Funding acquisition.

approach based on a game-theory model-interpretability analysis. The deep learning model for moisture content prediction included a final ensemble of two branches for analysis of spatial and spectral features, and it reached a coefficient of determination of 0.916 and 0.818 for the canola and wheat test datasets, respectively. SHapley Additive exPlanations analysis allowed us to study the individual predictions of the models, which is the most important contribution of this chapter because this approach could eventually lead to the design and implementation of more tailored software and hardware for the analysis of spectral information. The obtained results validate the idea that using this approach actually obtains the spectral bands that are important for this task, since they are similar to PCA results, and they fall on the NIR part of the spectrum, which is widely used in moisture measurement of agricultural products and vegetation analysis.

5.1 Introduction

Agriculture is one of the oldest and most important economical activities of humanity, which has undergone numerous changes throughout history. It began as a primitive process that eventually became what we call modern agriculture, where we see the inclusion of new technologies [58]. In the case of our study, these technologies involve the use of RS, UAV, HSI, and AI.

Conventional data collection techniques in agriculture, either manual or sensor based, are usually costly, time-consuming, labour-intensive, and invasive; RS on the other hand, being a contactless method to obtain and analyse data, could alleviate those issues [25]. RS data collection could be achieved by different means; however, the most common approaches are the use of satellites and more recently UAVs. Modern agriculture uses UAV technology to reduce farm labour and increase productivity. This flexible and cost-effective solution for non-intrusive, precise, and rapid phenotyping is commercially available as UAV platforms, which could include several sensors (e.g., HSI cameras) and software [59].

HSI is a combination of digital imaging and spectroscopy, relevant for the study of plant phenotyping. An HSI camera is a passive sensor that captures the spectral reflectance that corresponds to the wavelength of the electromagnetic field collected from objects on the ground. The spectral reflectance is associated with biochemical and biophysical properties of plants [25]. Plant breeders, researchers, and industry professionals have to test a large number of crop varieties, and HSI technology has the potential to speed up the process with direct use in the fields, in a non-destructive way [60, 61].

Current approaches that involve HSI tend to use spectral (or vegetation) indexes, which are mathematical combinations of two or more spectral bands [25], to extract the relevant information. However, more recently, ML has started to contribute to the RS field with HSI applications because it constitutes an effective approach for creating regression models of non-linear and multivariate systems [25]. Specifically DL techniques have been widely employed in RS in recent years in many fields, including agriculture [62].

Statistical models like Partial Least Squares Regression (PLSR) and classical ML approaches like RF and SVM are now common in RS [25]. However, despite their popularity and constant improvement, they are limited in the amount of spectral information they can extract [63]. New DL based models have been successfully applied in many fields such as medicine [64, 65], transportation [66], and security [14], mainly because they can effectively discover complex structures in large datasets [31]. The use of ML (and DL) and RS has already reshaped precision agriculture, and the future of farming depends largely on the adoption of such new technologies [25].

DL models such as LSTM and CNN are hot topics in RS applications [31]. CNN models are well known for their success in artificial vision, and they are currently relevant due to the growing amount of UAV and satellite images being collected [59]. On the other hand, RNN and LSTM, by extension, are natural candidates for HSI analysis due to their ability to process sequential data [67]. Furthermore, feature fusion is a common method for improving results in RS and DL tasks, and it can involve different approaches such as features being extracted by different branches of a single model [63, 68], or the use of ensembles of models [69, 70]. Ensembles combine multiple learners with the advantage of easy parallelization [71].

The DL approach is useful to find hidden relationships within the data [71], but despite being able to make accurate predictions, DL models lack the interpretability that is desired in many applications [31]. The study of this interpretability and explainability, or how a model arrived at a prediction [33], has led to interesting methods such as SHAP [36]. These methods try to figure out the reasons why a model predicted a certain class or value, and this could be used to analyse which features generate certain predictions.

A DL model can be inspected using SHAP [35], and that idea is used for example in the analysis of soil mapping in the spatial context [35]. In the case of our study, SHAP was used as a tool for supervised band selection. Band selection can preserve the physical meaning of the original spectral bands, providing more meaningful results than feature extraction algorithms like Principal Component Analysis (PCA), with the additional difference that supervised band selection methods use label information [28]. This approach could eventually become a resource for the elimination of redundant bands, which is helpful for saving computational resources and improving model performance [72]. Moreover, wavelength selection is essential for the implementation of real-time HSI applications [73].

Similar models found in literature that have provided ideas for this chapter include works like [59], that used UAVs, AI, and multispectral images for counting, locating and evaluating trees, and [68], that implemented a dual branch model that extracts spectral features plus serialised information about space with a LSTM network. The later approach uses a 3D CNN to extract information about the correlation of spectral and spatial features, which are cascaded before being fed to a classifier. [74] also integrated spatial and spectral features, although in a very different manner; this is a sequence of blocks that classify easier samples first. Moreover, [10] tried to evaluate the best time and spectral bands to predict wheat yield using multispectral images and 2D/3D CNNs. Similarly, [74] used 2D/3D dense networks for HSI classification, with a focus on mixed pixels. Finally, regarding model interpretability and feature selection, in the literature we have found other interesting proposals like the binary firework algorithm (BFWA) which selects bands before using PLSR [60]. We also found [75], that visualizes spatio-temporal attention weights as means of interpretability of the model predictions.

However, to the best of our knowledge, current works focus on the implementation of accurate prediction models which in most cases do not look for the features that generate those predictions, and even if they look for optimal wavelengths [60], they do not use HSI or moisture content information related to crops in development stages. In other cases, they use traditional statistical approaches [76]. Thus, we try to fill this gap in the literature by implementing a DL model that is able to predict moisture content of crops based on HS images of real-life scenarios, and moreover we look for the spectral bands that have an impact on those predictions. The prediction of moisture content is important to determine the degree of maturity of the crops [61, 76] and to implement quality control schemes [60, 77], which might have an impact on the revenue of farmers.

To summarise, this chapter presents two main contributions: (i) an accurate DL model for the prediction of the moisture content of canola and wheat crops as an indirect estimate of maturity, based on HSI taken by several UAV flights, using spatial and spectral information; and (ii) a novel approach for assessing the importance of spectral bands in those predictions, using a game-theory model-interpretability analysis. The chapter is organized as follows: Section 5.2 presents the AI models, their architecture, hyperparameters and training process, and introduces the SHAP analysis; Section 5.3 shows the way in which we found the optimal hyperparameters for the DL models; Section 5.4 presents the results of our study and compares them to other well-known methods; and Section 5.5 concludes the chapter.

5.2 Materials and Methods

5.2.1 Acquisition of Hyperspectral Images

Plant breeders traditionally need to manually assess each plot (or some of them) based on the plant characteristics they are seeking. Assessing the traits of a crop is usually performed by point-sampling techniques, which are costly, laborious, and spatially unrepresentative [78]. Crop leaves and canopies usually show changes in the plants with variations in their pigment contents, such as xantophylls, chlorophylls, and carotenoids [78]. These changes are exploited by the analysis of HSI, which provides discriminative spectral signatures despite the problems associated with it including high dimensionality of the data, redundant information, and insufficient samples [28].

HSI data samples were collected throughout the growing season of the two crops under analysis, at different time points, using a multi-rotor drone (DJI M600) UAV unit with a Corning microHSI SHARK push-broom scanner of 150 spectral bands ranging from 400 to 1000 nm. A gimbal always kept the camera in horizontal position. The UAV was flown at an altitude of 30 m. All data were acquired before noon under ideal weather conditions (mostly during cloudless days). The radiometric correction was conducted using a non-uniformity correction (NUC) procedure which converts raw data to spectral radiance.

Noteworthy, the HSI samples and moisture measurements were taken by other collaborators of P2IRC at the University of Saskatchewan, following a procedure similar to what is described by [76]. This information was made available to the authors of this study and it has been used here.

In this chapter, we study the effectiveness of UAV-based HSI sensors to estimate the moisture content of canola and wheat crops.

5.2.2 Description of the datasets

The province of Saskatchewan, Canada, is home to more than 40% of cultivated farmland in the country and is the largest exporter of peas, lentils, durum wheat, mustard seed, canola, flaxseed, and oats in the world [79], making it important to study these crops. P2IRC as part of its projects, collects different types of data, including HSI and manual measurements of several crops. Using the available information, in the case of our work, two datasets corresponding to canola and wheat were assembled.

For our experiments we used these data because of the importance of these two crops, the availability of the data, and in order to evaluate how well the DL models perform on different crops, which could relate to the generalization capabilities of the resulting model and data processing pipeline.

Canola dataset

The canola trial was conducted during the 2019 growing season in a field located at Nasser near Kernen Crop Research Farm, University of Saskatchewan, in Saskatoon ($52°9'42.3786''N, 106°31'0.249''W$), and it contains HSI of 8 stages, including manual measurements of the pod moisture, among others. It was an experiment with 2 varieties of canola and 8 different nitrogen treatments, on 4 replicates. As per the aim of this study, we are interested to relate HSI to the degree of maturity of the pod, measured by its moisture content.

The HSI images and moisture content measurements took place at 77, 81, 88, 91, 95, 101, 108, and 117 Days After Sowing (DAS), during the months of August and September. Fig. 5.1 shows

Table 5.1: Samples per dataset

Crop	Plots in a Rep	# of Reps	# of dates	Total Plots
Canola	16	4	8	**512**
Wheat	15	3	2	**90**
Total samples				**602**

an example of an image corresponding to half of the field.

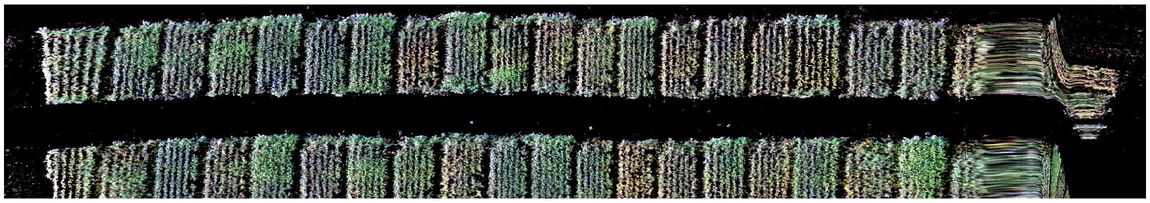


Figure 5.1: Two replicates within the canola field, RGB representation

Wheat dataset

This trial was conducted during the wheat growing season in the spring-summer of 2019 in the aforementioned Nasser field. The experiments were designed with 15 varieties and three replicates.

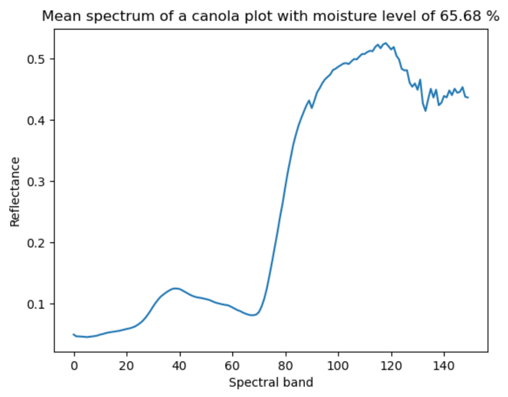
For this case, moisture content was measured only twice, at flowering and ripening stages; therefore, only these two batches of images have been used.

Table 5.1 summarises the number of samples and particularities of the canola and wheat datasets.

5.2.3 Reflectance Data Extraction

Narrow wavebands used in hyperspectral RS increase our ability to accurately extract phenological attributes in plants. Each pixel in an HSI has a spectral signature that contains the reflections of that spatial position in all the available wavelengths, which constitutes a feature vector [80].

Fig. 5.2 shows the spectral curves corresponding to two plots (mean values for all pixels in the plot). We can see that the spectra are similar for canola and wheat, with higher reflectance values in bands greater than 70, which correspond to red-edge and near-infrared zones. Each pixel in the HSI data contains spectral curves similar to the presented in this figure, and this is the information fed to our DL models and later on analysed with SHAP. We use all the spectral and spatial information contained in the areas of interest. Most current approaches use the data after dimensionality reduction which diminishes their performance, but DL allows an end-to-end approach [81].



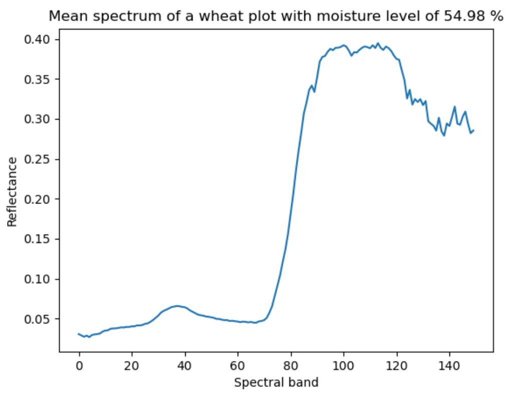


Figure 5.2: Spectral curves of two randomly selected samples. Left: canola. Right: wheat

Finally, we labelled the plots separately and tried to avoid the edges due to some overlaps of the plots and to avoid edge-effects. This would make measurements more robust, standardized, and repeatable.

5.2.4 Deep Learning Models

Non-linear regression problems require advanced solutions. Typical approaches include SVM and Naive Bayes; however, these might not be the best options when dealing with data in which features are dependent, since those models assume they are independent [71]. These methods also have the disadvantage that they use the spectral information of the pixels without including spatial details [68, 80, 82]. Spatial context is an important complement to the spectral feature [67] instead of using a single-pixel information [35].

In order to tackle those issues, DL approaches are being used in RS and HSI analysis. The most obvious option would be the use of 3D CNN in order to capture spatial and spectral features at the same time [10]. A 3D CNN could create a hierarchical representation of spectral-spatial data. However, the number of parameters grows exponentially when adding new dimensions, which makes the network prone to overfitting and highly increases the required computational load [63, 67, 74, 80, 81], besides decreasing their performance with depth [80].

Thus, in our proposed architecture, we have implemented two separate models for extracting spatial and spectral features in HSI. As depicted in Fig. 5.3, this information is fed into the models which generate predictions that are combined in an ensemble approach. Final predictions are then generated using both spatial and spectral features. These models and predictions are finally used in our analysis of the spectral bands that explain such predicted values, as well as the spatial contribution[a].

[a]Note added for the thesis: Here it is worth noting that the SHAP analysis is applied to the results of the spectral and spatial models, separately. Although the final predictions are obtained with the ensemble.

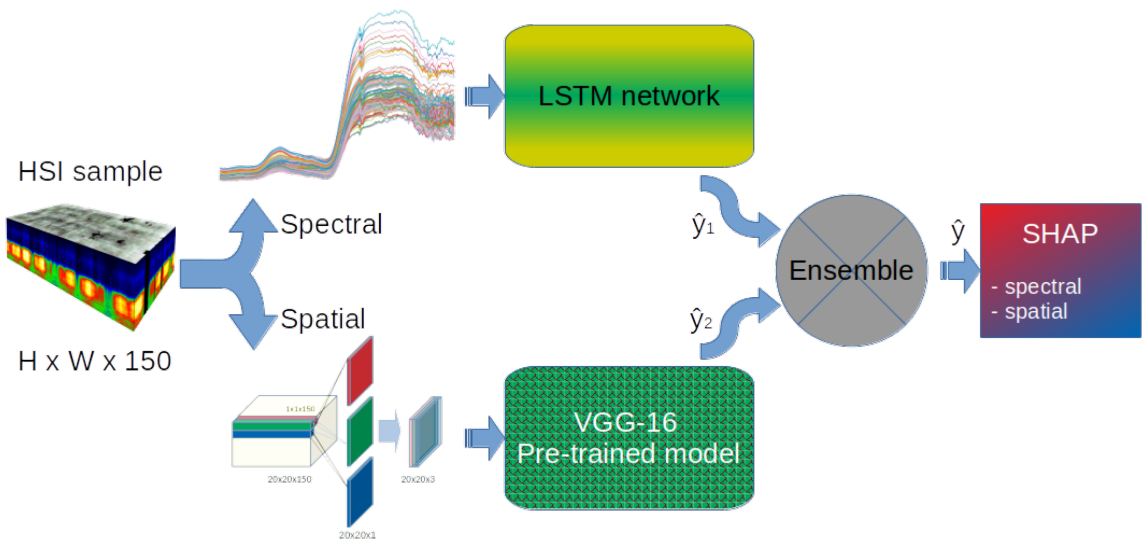


Figure 5.3: Block diagram that represents the whole information extraction scheme and analysis of predictions

Spatial Model

The model implemented for extracting spatial information (Fig. 5.4) relies on two main concepts: (i) the use of a pointwise convolution to convert the 150 bands into 3 that would correspond to the RGB channels used in everyday images[b] and (ii) the use of a pre-trained network to extract the spatial features.

A pointwise convolution is a regular convolutional operation in which the kernel is of size 1×1, and it is used to project the channels (bands) of an image onto a new channel space, capturing the cross-channel correlations [83]. The use of this operation is the first step to extract spatial information, and this replaces methods such as PCA which is commonly used as a preprocessing technique [82].

In the second step, we use a pre-trained VGG-16 network [71] fed with the information extracted with the pointwise convolution. In RS, the use of transfer learning from popular image classification models such as ImageNet is not uncommon [63, 71], mainly due to the small number of training samples (labelled HSI) [62].

Each of the layers (and blocks of layers) in a CNN is implemented to extract more complex features as we go deeper into the network. A good approach for using a pre-trained network is to use the initial blocks until they are able to extract the features relevant to a given problem (we tested different options for this), and then to train a customized ANN on top of it [1]. The regressor (a dense ANN) is based on the popular LeNet-5 architecture [29].

[b]J. Torres-Tello, K. Singh, S.-B. Ko, S. Shirtliffe, Transfer Learning from RGB to Hyperspectral Images by means of Pointwise Convolutions, 5th Annual P2IRC Symposium, Unpublished results (Oct. 2020). doi:10.13140/RG.2.2.31919.56480

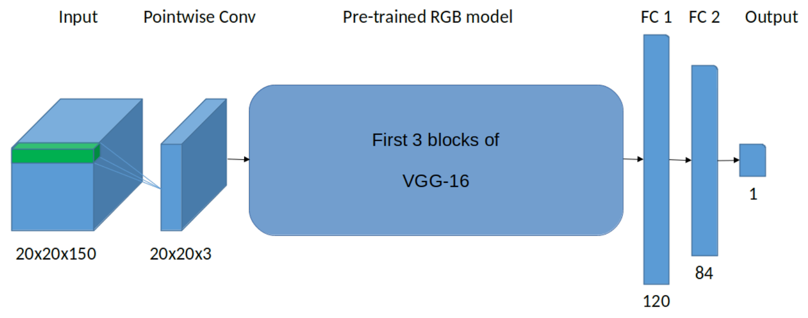


Figure 5.4: Block diagram of the DL model trained with spatial information

Spectral Model

RNNs are especially good at time-series tasks [31]. In the case of the spectral information contained in an HSI, we can treat it as a time-series feature vector in which each spectral band would correspond to a time step [68]. The original RNN design suffers from vanishing and exploding gradient problems, as the BPTT procedure depends exponentially on weights for each time step, failing to learn information that goes over 10 time steps. LSTM mitigates the vanishing gradient problem by introducing a linear unit (cell) called a Constant Error Carousel (CEC), which error flow control is conducted using gates [31].

Fig. 5.5 shows the architecture of the LSTM network that we have implemented. It has three LSTM blocks for extracting the spectral features, that are then fed to the same regressor implemented for the spatial branch. We used bidirectional (Bi-LSTM) layers because the spectral information has no semantic order and this approach is useful to exploit the forward and backward relationships of sequential data [68].

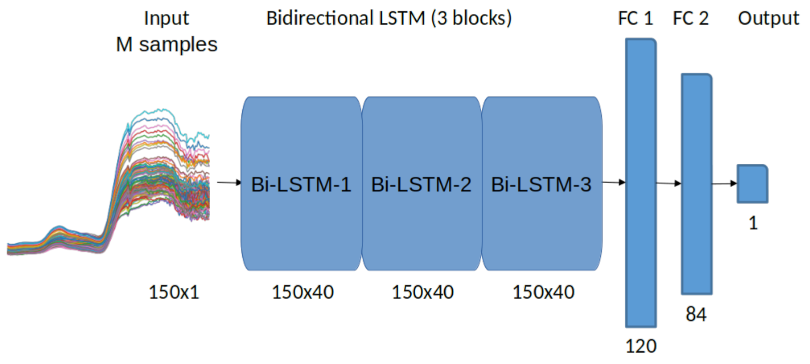


Figure 5.5: Block diagram of the DL model trained with spectral information

5.2.5 Ensemble Learning

Spectral signatures of HSI pixels that describe the same information may vary due to differences in imaging conditions, interference, or other factors, so that spatial information becomes an important complement that provides context and improves the final predictions, by combining them [68, 80, 82]. In spectral-spatial feature fusion, these features can be analysed individually or simultaneously [80]. In the case of our study, as seen in Fig. 5.3, we treated them separately and then combined their predictions in an ensemble approach, which is common technique used in ML/DL in order to increase the generalization performance of models [5].

The feature fusion strategy makes the process of generating predictions smoother (less noisy results), by using spatial information while maintaining the ability to extract spectral information [63]. An appropriate fusion rule is important; simple linear combinations are popular, weighted or not. In the case of a weighted combination, those weights should be proportional to performance of the models [80]. Thus, here we implemented a weighted linear ensemble method.

5.2.6 The training process

To train the DL models we had to separate the samples in the datasets previously described into different groups. One of the three or four replicates shown in Table 5.1 has been reserved for testing. The data for testing the model should include all kinds of characteristics in the image that might influence the prediction [62], and they should come from the same distribution of the data used for training. Reserving one replicate for testing assures that all varieties and treatments are included in the training and testing sets. The test set is only used for reporting the results of our test, in an attempt to make them as independent as possible of the training and validation (tuning) processes. In the case of the train-validation split, it was done in a random fashion with 80% of the samples used for train and 20% for validation.

Considering that in our case all the pixels in an HSI plot correspond to the same target and if all the pixels from an object are members of the same class, the scene can be represented by a single feature set [80], we adopted a split and average strategy for our samples. Splitting is used to generate a larger amount of training data, given the limited number of HSI images. This is a useful concept to fit the requirements of AI models, usually by a moving window strategy [62], which we implemented with non-overlapping regions. Those sub-samples were used to train the spatial model while a spectrum obtained by averaging the pixel values on that sub-sample was used to train the spatial model. This is useful to remove the noisy pixels from a segment [80] under the assumption that neighbouring pixels belong to the same class [69]. CNN and LSTM are sometimes avoided in RS applications because they require a large number of training samples [71], and splitting helps to overcome that issue.

In order to find the optimal hyperparameters of the models, we use the approach described in Section 5.3. The spatial model was trained for 2000 epochs and the spectral for 5000. Due to the difference between canola and wheat, both in phenotype and amount of samples, the models and their hyperparameters were tuned for canola, and then retrained (applying also some regularization in the form of dropout) for wheat samples. This would also help us evaluate the generalization capabilities of our model. In all cases, the loss function was MAPE, and the evaluated metrics were MAE and MSE.

All the models described here were implemented on Python 3.6.9, using ScikitLearn 0.24 and Tensorflow 2.1, and they were trained using the Microsoft Azure Machine Learning platform on a standard NC6 server with 6 Cores, 56 GB RAM, 380 GB Disk and a NVIDIA Tesla K80 GPU. The average time for training one of our models ranged from 10 to 173 hours.

5.2.7 Model Interpretability for Feature Selection

Feature selection is an important process in the analysis of HSI, mainly due to the high dimensionality of the raw data. The so-called *curse of dimensionality* refers to the difficulty of dealing with high dimensional data and the need for more data samples to tackle this problem [26].

There are useful tools to reduce the dimensionality of data, for example, PCA that is a linear transformation that generates components, where a few of them capture most of the data variability [26] becoming one of the most popular pre-processing techniques in HSI analysis [26, 72, 78, 82]. However useful and widely used, the PCA approach has two drawbacks: (i) it is an unsupervised method, meaning that it does not use the information contained in the labels of the HSI images and (ii) it transforms the data, so that the original features are not preserved. To tackle those issues, in this study we propose the use of a model interpretability approach that can point out the spectral bands (and areas of the plot) that contribute to generate a prediction, by explaining such predictions of DL models like the ones we have trained.

To this end, we have used SHAP which is a game theory technique that treats each feature as a player in a game, where that player (feature) contributes to increase or decrease the predicted value [35, 36]. The SHAP approach is also interesting because it generally shows a stronger agreement with human explanations [35, 36]. Nevertheless, it is noteworthy that black box models (such as DL) might be extremely complex and have an internal state composed of millions of interdependent values, and therefore the decision-making process might be impossible to completely understand [33].

SHAP is a local method, which means that it focuses on the effects of each feature on individual predictions, and this is the interesting point in our study, since we can analyse each spectral band as an individual feature. SHAP uses the expected value over the analysed dataset as the base value (predicted value if no feature is known), and the effects of each feature are then subsequently added. The sum of these effects adds up to the predicted value. SHAP generates a simple linear model (Equation 5.1) as the explanation of the trained model, which approximates the decision of the original model in the neighbourhood of the point of interest.

$$g(z') = \phi_0 + \sum_{i=1}^{M} \phi_i z'_i, \tag{5.1}$$

where g is the approximation of the original model, M is the number of features, $\phi_i \in \mathbb{R}$ represents the feature attribution values of feature i, and $z'_i \in \{0, 1\}$ is the studied feature.

5.2.8 PLSR as a reference model

PLSR is a popular statistical method used in the analysis of hyperspectral images [60, 61, 76], since it is a versatile tool that can be used with large datasets for which standard regression methods are not useful [84].

PLS methods in general relate two groups of data that describe the same set of observations, deriving optimal linear combinations of the variables of a group of data, and when the goal is to predict one group from the other, the technique is called a regression (PLSR) [84].

For this chapter, we have implemented PLSR as a reference to compare the results of the proposed DL models. In order to optimize the hyperparameters of the PLSR model, we used a grid search with a 5-fold cross-validation. The tuned hyperparameters were the maximum number of iterations (100, 500, 1000), the number of components to keep (2, 5, 10, 15), and the tolerance used as convergence criteria (1e-5, 1e-6, 1e-7). We used the best hyperparameters (that were tuned for canola and wheat separately) to train a new model and we used this retrained model in order to obtain the final predictions. As in the case of all other experiments, the results were averaged per plot, and they are presented in Table 5.4.

5.3 Calculation

The models described in Section 5.2 have some hyperparameters that can be tuned in order to minimize the error of the predictions. Tables 5.2 and 5.3 show the values that were tested for each of these parameters. The ones that produced the lowest MSE for the validation test were used for each case. In bold are the final hyperparameters used to train the model that was then used to generate the predictions shown in Section 5.4.

With the aforementioned hyperparameters, we have trained each of the models over each of the datasets described in Table 5.1. The initial predictions over the validation set were used to adjust an ensemble of models with a weighted addition of the predictions for each sample, as shown in Equation (5.2), where $\hat{y}$ corresponds to the predictions and w to the weights. These models and the final ensembles were used afterwards for generating predictions on the test set.

$$\hat{y}_{ensemble} = w_{spatial} \times \hat{y}_{spatial} + w_{spectral} \times \hat{y}_{spectral} \tag{5.2}$$

Table 5.2: Hyperparameters tested for the spectral model. The ones producing the best metrics are in bold

Hyperparameter	**Tested values**			
Neurons in layer	10	20	**40**	-
Number of layers	1	2	**3**	4
Optimizer	RMSprop	**Adam**	Adadelta	-
Pixels per sample	50	100	**200**	-

Table 5.3: Hyperparameters tested for the spatial model. The ones producing the best metrics are in bold

Hyperparameter	**Tested values**			
Optimizer	RMSprop	Adam	**Adadelta**	-
Box size	10 x 10	**20 x 20**	30 x 30	-
Output layer	block1_pool	block2_pool	**block3_pool**	block4_pool

where the optimal values for $w_{spatial}$ and $w_{spectral}$ are 0.1 and 0.9 for the canola dataset, and 0.2 and 0.8 for the wheat dataset, respectively. These weights were found through an exhaustive search that minimized the MSE of the weighted average of the predictions of the two models, on steps of 5% for each case.

Since our proposed model is an ensemble, it is useful to compare its ablation variants [70]. For this, we have considered our three possible combinations: (i) spatial model, (ii) spectral model, and (iii) ensemble of the two. Results are shown in Table 5.4. MAE and MSE were the metrics used to train and tune the models, and we also used the coefficient of determination R^2, which is often used to judge the quality of the prediction of a regression model and it should not be less than 0.5 for a valid model [75].

Table 5.4: Evaluation metrics for the test datasets, where * indicates predictions averaged per HSI plot

Crop	**Canola**			**Wheat**		
Metric / Model	Spatial	Spectral	Ensemble	Spatial	Spectral	Ensemble
MAE	5.823	3.094	3.272	2.453	2.062	1.928
MSE	62.772	16.893	20.750	9.405	7.019	5.954
R^2	0.559	0.826	0.854	0.623	0.698	0.761
MAE *	4.438	3.446	**2.709**	1.801	2.100	**1.704**
MSE *	38.352	21.767	**13.904**	4.635	7.203	**4.450**
R^2 *	0.767	0.868	**0.916**	0.810	0.705	**0.818**

5.4 Results and Discussion

5.4.1 Results of moisture content prediction

The results show that both averaging the predictions per plot and using the ensemble of models improve the moisture content predictions for canola and wheat datasets. We averaged the predictions per plot since the target value (moisture) was collected per plot. Thus, even though different pixels could generate different results in the regression process, the value per plot should be unique.

Results shown in Table 5.4 are difficult to compare to other works, but as a matter of reference, they were generated with a slightly higher number of samples than [35] that uses 485 samples and uses also SHAP for soil mapping, or [72] that used 72 samples with cross-validation. We have better results than [60], one of the most similar papers in literature, that uses PLSR for the analysis of HS soybean data during drying, achieving a Root Mean Square Error (RMSE) of 5.105 %, which squared would be 26.061, higher than the values obtained with our models (13.904 and 4.450 for canola and wheat respectively).

Predictions compared to a vegetation index

Since the 1970s, scientists have used RS-based spectral measurements of vegetation to characterise the environment. Vegetation indexes are the traditional tool for this analysis, and one of the most common indexes used is the NDVI, defined in Equation (5.3) [26].

$$NDVI = \frac{NIR - VIS}{NIR + VIS}, \tag{5.3}$$

where NIR and VIS are reflectances in near-infrared and visible bands respectively. The most commonly used bands to calculate NDVI are NIR=800 nm and VIS=670 or 675 nm [25]. We have used 800 and 670 nm, respectively.

RS methods based on spectral vegetation indexes are very popular [25]. For example [59] used NDVI as a base method for image segmentation and [78] used various indexes finding that NDVI produces a higher correlation with water content (> 0.87), similar to PSNDb (Pigment Specific Normalized Difference b).

In this Section, we evaluate the possibility of using the NDVI vegetation index to predict moisture content. To compare our prediction results to this common baseline method, Fig. 5.6 shows the correlation between NDVI and moisture content for our analysed datasets. The plots show that there is a relatively high correlation (R = 0.720 and 0.610 for canola and wheat respectively), but this relationship is not linear, as we can see when the R^2 values are also calculated and negative results are obtained. Sometimes, the NDVI is used as a logarithmic relationship which in our case does not improve the general correlation (R = 0.827 and 0.587 respectively).

These results imply that we need to establish a more complex relationship than the vegetation index in order to find a moisture prediction tool from the spectral information, otherwise, we would have ambiguous correspondences between moisture content and NDVI values. Fig. 5.6 (a) is a clear example of this ambiguity.

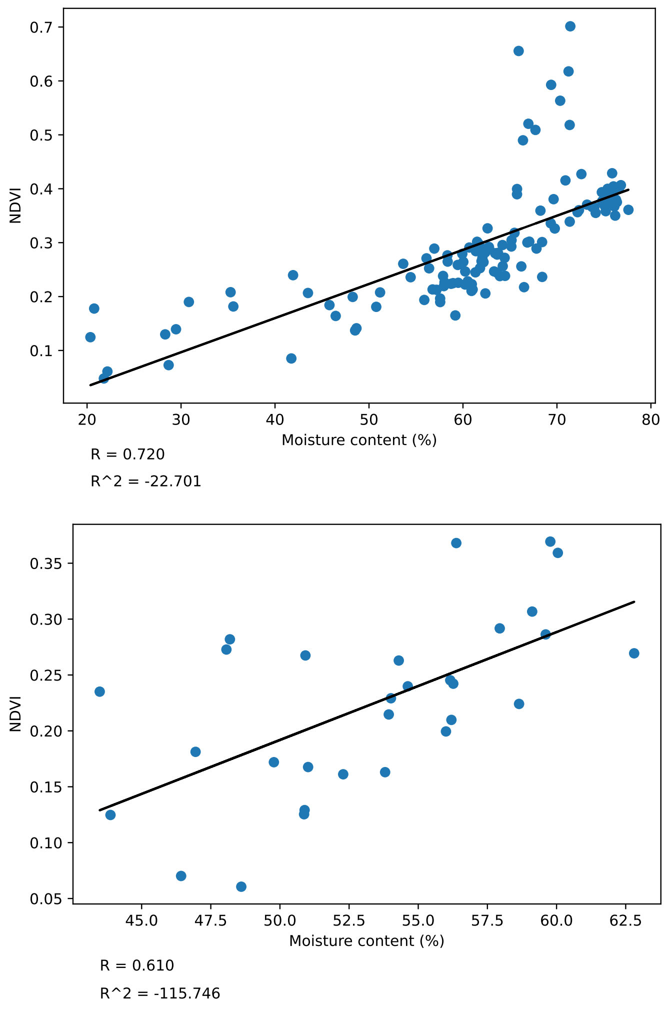


Figure 5.6: Representation of the correlation between NDVI and the moisture content. Top: canola (a). Bottom: wheat (b). Linear relationships

In literature, there are also customized indexes for particular tasks, such as the Canola-Pod-Maturity Index (CPMI) proposed by [76], that uses three spectral bands to predict the degree of maturity of the canola pods by using the moisture content. This index is given by Equation 5.4, where authors define R_{466}, R_{721}, and R_{813} as the reflectance values at 466.10 nm, 720.99 nm, and 812.59 nm, and the amplification factor A_f = 2.5, c1 = 2 (constant that adjusts the red-edge influence), and c2 = 1 (constant that minimizes the background signals on canopy spectra).

$$CPMI = A_f \times \left[\frac{R_{813} - R_{721}}{R_{813} + (c1 \times R_{721}) - R_{466} + c2} \right] \tag{5.4}$$

We have used the CPMI index as a reference value to compare the performance of our model for predicting moisture content in canola, which results are included in Table 5.4.

In order to summarise the results and to compare the performance of our proposed model to other methods or works, we put together the most relevant metrics in Table 5.5, for which we have also calculated the regression coefficient (R) of the predictions using our proposed model. Although comparison is complicated in these studies, we can see that our model performs better than the alternatives.

Table 5.5: Results of our model compared to other methods or works (best values are in bold)

		Metric		
Crop / Method		R	R^2	MSE
Soybean	[60]	-	-	26.061
Canola	CPMI [76]	0.771	-	-
Canola	NDVI -linear	0.720	-	-
	NDVI -log	0.827	-	-
	PLSR	0.937	0.854	24.099
	Proposed DL model	**0.963**	**0.916**	**13.904**
Wheat	NDVI -linear	0.610	-	-
	NDVI -log	0.587	-	-
	PLSR	0.915	0.816	4.479
	Proposed DL model	**0.936**	**0.818**	**4.450**

5.4.2 Feature selection with SHAP

SHAP analysis allows the study of individual predictions of a model. Unlike other approaches like PCA in which the information is extracted with disregard of the application or specific need, using the proposed approach, we can generate explanations of the trained model, focused on a target (moisture content), a feature (spectral or spatial), or a subset of the HSI. If we need a comprehensive explanation, the per-pixel contributions can be aggregated into a single SHAP value per covariate, due to its additive property [35]. This approach is the most important contribution of this chapter because it could lead to the design and implementation of more tailored software and hardware for the analysis of spectral information.

Spatial and spectral representations are shown in Fig. 5.7 and 5.8, respectively. In the first case, we can see two 20×20 windows (subplots) in grey (left) as a monochromatic representation of the HSI image in a given part of the field. On the right side, we have the representation of the pixels that spatially contribute to explain the final predicted value. Red and blue pixels represent the values that contribute to increase or decrease the final predicted value for that sample (ϕ values in SHAP analysis), or the most important areas of the plot in order to generate a prediction. This result could be used to analyse parts of the plot on which a farmer or breeder should focus on.

On the other hand, Fig. 5.8 is showing on the top part, all the spectral signatures collected on a single plot (each of the spectral representations corresponds to one sample or subplot in the 150 bands). In the lower part, we have the spectral bands that contribute the most to a predicted value. For each of the samples on the top part, we have the correspondent SHAP values on the bottom (red and blue for the bands that increase or decrease the predicted value, respectively). Each part of the plot has its own spectral signature and also its own explanation, but we can see that there is a trend in the most important bands for this prediction. For representation and comparison purposes, here we have chosen to aggregate the spectral features in this case.

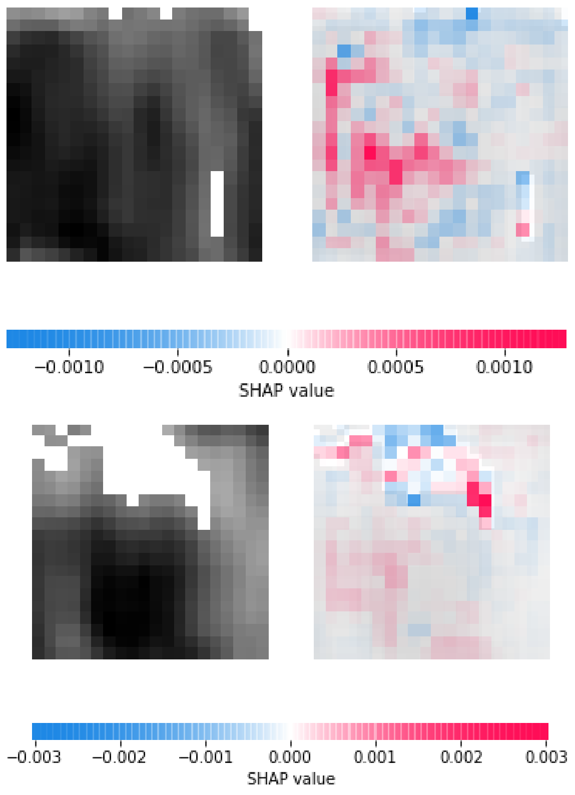


Figure 5.7: Examples of the spatial contribution to moisture predictions in canola crops (Left: greyscale portions of the plot. Right: SHAP values for those portions of the plot)

From the spectral and spatial analysis, the spectral part is the most relevant for our study. To

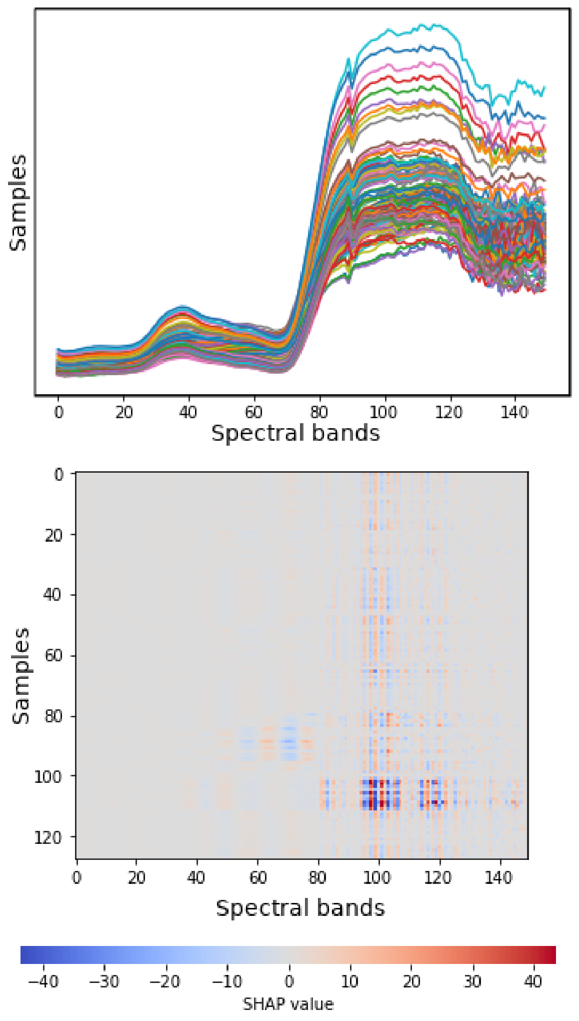


Figure 5.8: SHAP values per plot for the canola test samples. Top: Spectral signatures of the subplots. Bottom: SHAP values with respect to the spectral signatures above

validate our results, we have chosen 10 % of the top contributing bands in the aggregate (i.e. 15, which is similar to what [60] and [76] presented, where 12 and 16 bands were selected, respectively), which are shown in Fig. 5.9, where we can see that all of them are located in the NIR part of the spectrum, both for canola and wheat. The figure also includes the 15 bands that contribute the most to the first components after performing a PCA analysis, used as a sanity check, because analysing the whole spectra with PCA can also be helpful for band selection [72]. In this case, we see that only a few of the bands fall in the red-edge portion of the spectrum while all the others are located in the NIR.

These results validate the idea that using SHAP actually obtains the spectral bands that are important for this task, since they are similar to PCA, and they fall on the NIR part of the spectrum. It makes sense because water has several strong absorption wavelengths in the NIR, so NIR spectroscopy is widely used in moisture measurement of agricultural products [60]. It is also known that the range from 700 to 1100 nm is useful for vegetation analysis [26]. Moreover, it has been found that the NIR part of the spectrum contributes to plant phenotyping more than the RGB part [85]. Finally, one of the characteristics of SHAP is the consistency with human intuition [33, 36]. Fig. 5.9 shows that the mean value of those 15 bands is 832.5 nm and they are distributed on the NIR and red-edge spectrum.

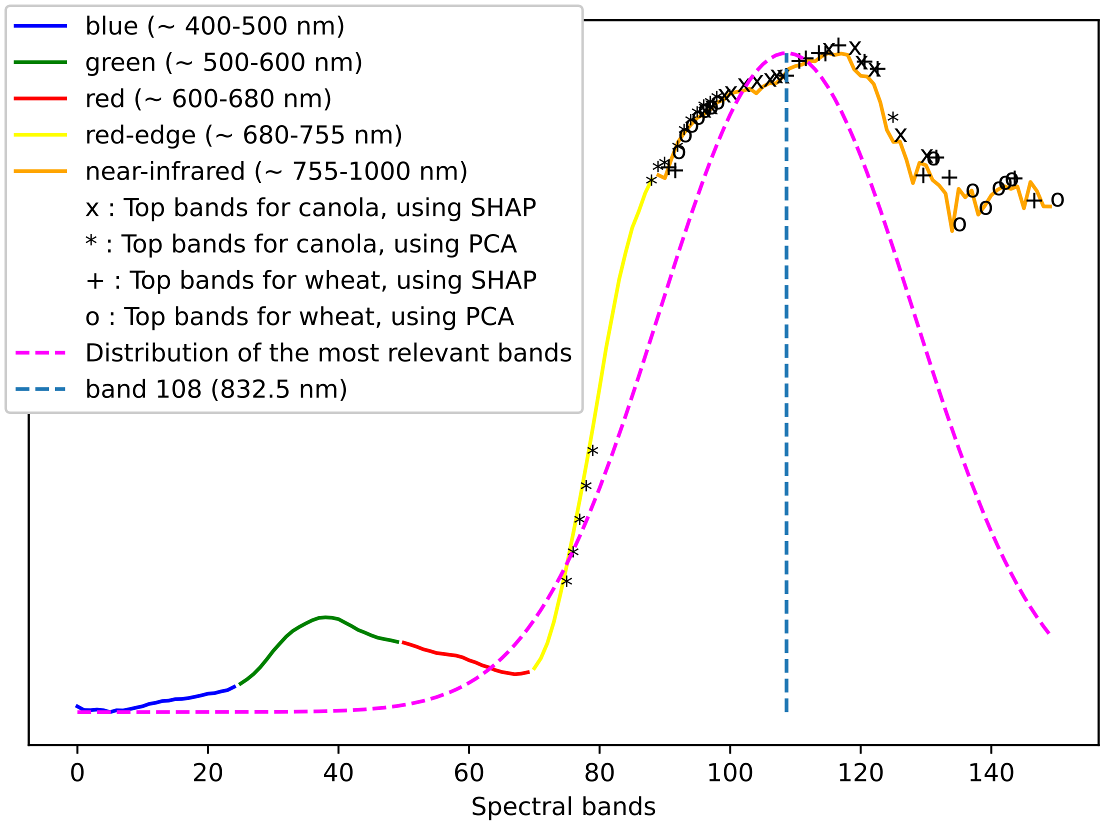


Figure 5.9: Position of the 15 most relevant bands obtained with SHAP values and PCA, plotted over a sample spectrum

5.5 Conclusion

As for the first goal of our study, we have demonstrated that the final ensemble models generated here obtained good results, even better than what we have found in literature. We mentioned that our models with an MSE of 13.904 (canola) and 4.450 (wheat) are better than results obtained by [60], with an MSE of 26.061 in a similar task. The use of our DL model is also better than reference methods such as PLSR and the NDVI vegetation index.

However, the relevance of the DL models is not only in the numerical results, but also in the architectural design of the network. We have implemented a spatial-feature model that uses a pointwise convolution to convert the 150 bands of the HSI into 3, that would correspond to the RGB channels used in everyday images; feeding this information into a pre-trained VGG-16 network that extracts the spatial features. Data fusion and DL methods for joint spectral-spatial generation of features are two main trends in recent literature [80].

As for the assessment of spectral bands, in this chapter we proposed the use of a model interpretability approach that by explaining the predictions of an AI model (such as the ones we have trained), it can point out the spectral bands (and areas of the plot) that contribute to generate such prediction. SHAP analysis allows us to study the individual predictions of a model, and this is one of the most important contributions of this study because this approach could lead to the design and implementation of more tailored software and hardware for the analysis of spectral information, and maybe the development of new vegetation indexes [76]. To the best of our knowledge, this game theory approach has not been used for supervised band selection in a regression model that uses HSI and DL.

The results of our analysis, when compared to PCA, validate the idea that using SHAP obtains the spectral bands that are important for this task. In both cases, the most relevant features are located on the NIR part of the spectrum, which is widely used in moisture measurement of agricultural products and vegetation analysis. Moreover, unlike other approaches such as PCA in which the information is extracted with disregard of the application or specific need, using the proposed approach, we can generate explanations of the trained model, focused on a target (moisture content), a feature (spectral or spatial), or a subset of the HSI.

Part IV

Model Optimization and Edge Intelligence

6 Interpretability of Artificial Intelligence Models that use Data Fusion to Predict Yield in Aeroponics[5]

This chapter elaborates on the previous ones; first, it extends the use of DL models for sensor-data, also applying data fusion, and second, it introduces the use of salient features selected by SHAP for model optimization. In the particular case of this chapter, optimization refers to generating smaller models by using only the most relevant features in the dataset, while keeping a performance similar to the predictions by the models that use all the features. This is the first step towards the implementation of these DL models in embedded devices. Assuming that different features contribute equally to the predictions of a model might be problematic in many applications because they have different discriminative capabilities. Thus, trying to understand the predictions of those models is important to gain information about the contribution of individual features, instead of assuming or using pure empirical knowledge. Therefore, this chapters is the only one that explores the ideas of model interpretability.

This chapter also explores the ideas of model interpretability, by not only finding that there are features that impact the predictions more that others (mainly reservoir temperature and room CO_2), but also by finding insight on how they do it. For example, results indicate that there is positive correlation between days in tower and yield and that reservoir TDS has a positive impact in the final yield. Moreover, higher reservoir temperatures have little or no effect on the model outputs and lower temperatures do; or for room CO_2, the relationship is the opposite. Importantly, these results are concordant with current literature in the sense that for example, the effect of CO_2 in plant growth is known and also the impact of the temperature of the roots has been studied before. Nonetheless, it would be important to further study these results, possibly designing new experiments for this particular case.

There is an increasing demand for healthy and fresh foods, and predicting yield effectively is important to improve production, especially in methods like aeroponics. This chapter has two main

[5]The content of this chapter is originally published in the Journal of Ambient Intelligence and Humanized Computing [11]. The manuscript has been reformatted for inclusion in this thesis.
Julio Torres-Tello: Conceptualization, Methodology, Software, Validation, Data Curation, Writing - Original Draft, Visualization. Seok-Bum Ko: Resources, Writing - Review & Editing, Supervision, Project administration.

goals: (i) use data fusion to improve yield prediction in aeroponics, and (ii) find which features are more relevant for yield prediction of six different crops. To reach these goals, a number of artificial intelligence models and an interpretability analysis based on SHAP have been implemented. The models were trained using 200 samples that were collected in a nine-month period, including information from different air and water quality sensors in addition to manually recorded data, reaching in the end a coefficient of determination value $R^2 = 0.752$ for the validation dataset in the best case (CNN-based model). As a result, two main features were identified in the dataset: Room CO_2 and Reservoir Temperature, along with other useful insights of how these features influence predictions. SHAP values also provided important information for feature selection. These results could be the first steps towards the full automation of an aeroponics crop production system.

6.1 Introduction

There is an increasing global demand for healthy and fresh foods. Methods like hydroponics and aeroponics allow for year-round harvest, protection from weather, portability, cultivation of diverse crops and disease-free farming. Among these alternatives, aeroponics is a promising and efficient technique. [42] showed that compared to traditional farming, aeroponics increases yield from 7 to 65% based on the type of crop. Along with faster crop cycles, aeroponics also improves water, pesticide and fertilizer usage. Nevertheless, improving yields in plant factories (fully controlled environments) is a major research problem [86].

Aeroponic systems are soilless growing methods using a mix of water and nutrient compounds to nourish plants. Unlike traditional field agriculture, air and water quality can be measured and controlled, while allowing for more dense farming since plants grow in a stacked tower structure. Data was collected from a sustainable and organic aeroponic farming device called AeroPod (Fig. 6.1) from Farm Boys Design, a corporation based in Saskatchewan, Canada.

The outcome of a harvest is hard to predict, time consuming, and prone to error. Recently, AI has become an important tool for predictions in different fields such as medicine [87, 88, 89, 90], robotics, and climatology, and the use of AI for yield prediction in agriculture should be no exception. Furthermore, yield prediction could be the basis of fully automated control systems in which yield could be maximized by setting the variables under control (e.g., light conditions, nutrients, temperature) [16].

Current literature includes some works that try to automate hydroponic or aeroponic production, some of which use AI for this task. For example, [91] developed an aeroponic system based on IoT, [92] implemented a fuzzy-logic based system for the control action in aeroponics, and [93] went one step further by using Deep Neural Networks for the control system in hydroponics, achieving an accuracy of around 88%. Other works used ML for generating recommendations or managing

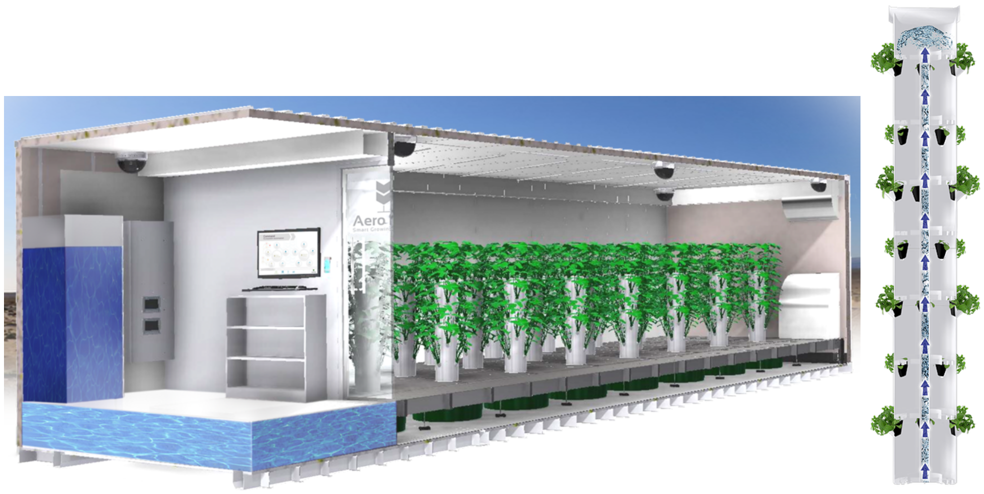


Figure 6.1: Left: Aeroponics container (a). Right: One of the towers in AeroPod (b) [5]

what they consider the most important variables in the system, such as the paper presented by [94] that focused on irrigation or [86] that considered temperature as the most important.

This study elaborates on a conference paper [5] (presented in Chapter 3) that uses three traditional ML techniques to create a yield prediction model. For this extended version, two new DL models based on LSTM and CNN have been included. The reason for using these DL models is that unlike the previous approaches, LSTM and CNN networks would be able to use all the data being gathered by the sensors in the AeroPod. The first goal of the chapter is to find AI models that could use the data as taken by the sensors to improve yield prediction in the aeroponics system. Eventually, this could be used for automating the production process.

Despite being able to make accurate predictions, ML and DL techniques generate complex models that lack the interpretability that is desired in many applications. This issue is particularly evident in disciplines like medicine [95] and finance [96], where the understanding of the features that cause a certain prediction is essential for the adoption of AI models. Likewise, with aeroponics, it is relevant to understand the features that could influence the final yield of a crop.

Assuming that different features contribute equally to the predictions of a model is problematic in many applications because they have different discriminative capabilities [97]. Thus, trying to understand the predictions of those models is important to gain information about the contribution of individual features to yield prediction. This would allow producers and researchers to learn, using statistical values, what variables they should act upon if they want to maximize yield, instead of assuming or using pure empirical knowledge [86, 94]. Therefore, the second goal of this chapter is to find which features are more relevant for yield prediction of six different crops. For this purpose, a technique known as SHAP [36] has been used.

This chapter is organized as follows: Section 6.2, Materials and Methods, describes data collec-

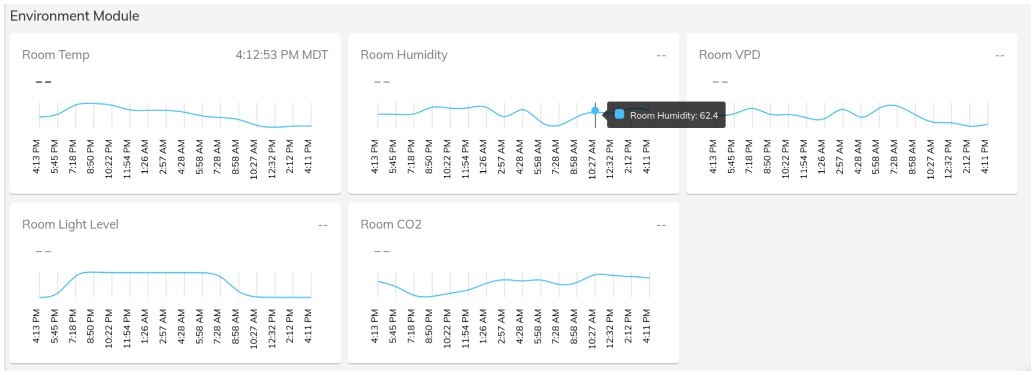


Figure 6.2: Screenshot of the data acquisition platform

tion methods, implementation and training process of the AI models, along with interpretability concepts. Section 6.3, Results and Discussion, presents the results achieved by our models and comments on their meaning. Finally, Section 6.4 presents the conclusions of the chapter.

6.2 Materials and Methods

6.2.1 Growing Method and Data Collection

Our experiments, conducted during production stages, included six different crops: garlic chives, basil, red chard, rainbow chard, arugula and mint. In all cases, seeds were sown in mineral wool in a tray, until they germinated and attained early stages of growth. Then, they were transplanted to stacked towers in the AeroPod. Each layer in a tower (Fig. 6.1b) consists of a few spots that can accommodate a mineral wool cube. In the AeroPod, nutrient mixture comes in contact with mineral wool cubes at constant intervals, thereby nourishing the roots. Plants were harvested and yield was measured as weight per spot (oz/spot then converted to g/spot for this study).

The data acquisition platform is implemented as part of the AeroPod and is used in the daily operations of this system. However, the data presented here, which is available online [98], was assembled for this study. Part of this data was used in Chapter 3 but the complete dataset has only been used for this study.

As shown in Fig. 6.2, air and water quality sensors implemented in AeroPod record hourly values of room carbon-di-oxide (CO_2) levels [ppm], room relative humidity [%], room light level [lux], room temperature [$^\circ F$], room VPD [mbar], reservoir water pH, water TDS [ppm] and reservoir temperature [$^\circ F$]. Manually collected data include number of days in tray, number of days in tower, harvest number (how many times the plant has been harvested since first transplanted to AeroPod) and grow number (how many times a plant has been transplanted into that spot). The target value that the AI models use is the crop yield (g/spot). For this study, 200 samples have been collected between November 2018 and August 2019.

Regarding the time series data, the samples were taken once per hour, as an average value during the last hour, for all the registered variables. We obtained the log files (raw information) for all these sensor readings during the months the experiments took place. From there, and using the transplant and harvest dates for each of the 200 spots, we constructed a dataset that included the readings from the sensors for the time of interest in each case. In the end, we had between 209 and 4431 data points per feature per sample. This difference in the length of the vectors happens mainly because not all the plants stay in the tower for the same amount of time (11 to 191 days) and also because there was some downtime in the data collection system, which values are not included.

6.2.2 Data preprocessing

The data was used to construct two different datasets. The first is a tabular dataset in which the values read by the sensors were averaged, similar to what is presented by [94]. The averages were calculated for the period in which each crop has been in the AeroPod, as described previously (see Sect. 6.2.1), so that each sensor contributed with one scalar feature per data point. This dataset was used to train the traditional ML models in a previous study [5], as well as in this chapter.

The second dataset is a mixed dataset in which the tabular data corresponding to manually acquired values (number of days in tray, number of days in tower, harvest number, and grow number) were kept unchanged, and in which each sensor contributed with a vector containing the time series response per data point. This dataset corresponds to the actual information gathered by the sensors (Fig. 6.2), combined with the manual readings, and it was used to train the DL models.

The only preprocessing on both datasets was performed on the tabular features. We standardized the features of the tabular dataset and normalized the tabular features of the mixed dataset. We used different approaches on these datasets considering that they do not have the same features. The tabular dataset contains average values of the sensors readings. The time series data was fed to the models with no preprocessing. In this case, we only used zero-padding to make the vectors fed to the DL models have the same length.

6.2.3 Regression models

A regression model f is the one that estimates the relationship between independent variables (features) x and a dependent variable (target) y. The goal of the AI models is to find an $f(x, w)$ such that:

$$\hat{y} = f(x, w) \approx y \tag{6.1}$$

Where $\hat{y}$ is an approximate value of the original dependent variable y, and w are the so-called weights that the model learns to fulfill Eq. 6.1.

6.2.4 Performance metrics

The validity of a regression model is given by how similar y and $\hat{y}$ are. To accomplish this task, there are some performance metrics that can be used. This chapter implements three of them: MAE, MSE, and coefficient of determination (R^2). These metrics are given by Eq. 6.2, 6.3, and 6.4, respectively, where n is the number of samples in the dataset.

$$MAE(y, \hat{y}) = \frac{1}{n} \sum_{i=0}^{n-1} |y_i - \hat{y}_i| \tag{6.2}$$

$$MSE(y, \hat{y}) = \frac{1}{n} \sum_{i=0}^{n-1} (y_i - \hat{y}_i)^2 \tag{6.3}$$

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=0}^{n-1} (y_i - \hat{y}_i)^2}{\sum_{i=0}^{n-1} (y_i - \bar{y}_i)^2} \tag{6.4}$$

As seen in the previous equations, MAE is just an average estimate of the error between real and predicted values, and MSE is useful in the sense that it is more sensitive to large errors (due to the square function). R^2 represents the proportion of variance that has been explained by a model, and is, therefore, a measure of how well the predicted values $\hat{y}$ fit the real ones y. In the case of MAE and MSE, values closer to 0 are better, and in the case of R^2, the best possible value would be 1.

6.2.5 Implementing and Training the AI Models

Traditional ML Models - Averaged Data

These models were already presented in Chapter 3 and they have been re-trained for this Chapter. Original and new results are presented in Section 6.3. These models constitute the baseline for the models that use data fusion.

The training process of these algorithms involved repeated k-fold cross validation (k=2), a commonly used procedure to evaluate ML/DL models on datasets of limited size. This resampling technique without replacement is applied to obtain an error rate as independent as possible of the train-validation split of the dataset, because each sample is part of the training and validation sets only once. Given the small sample size, we performed repeated (5 times) cross-validation experiments to reduce variance in the results.

The same models and network architecture presented in the aforementioned paper were used here, but different regularization hyperparameters were chosen via nested cross validation within each fold. Specifically, we used a 5-fold split in this case.

To summarize, the 200 samples in our dataset were split in two groups (outer loop) for the cross-validation procedure. Each of these two groups was split into 5 groups (inner loop) in order to tune the hyperparameters used in the outer loop. This process was repeated 5 times. The results presented in Section 6.3 correspond to the mean values of this process.

Support Vector Regression [7] describes SVM as widely used learning algorithms, that have the optimization objective of maximizing the margin, which is the hyperplane that separates the training samples that are closest to it. These samples are called support vectors. A parameter ϵ defines the epsilon-tube inside of which no penalty is associated in the training loss function. There is also a regularization parameter C that controls the misclassification penalty. To deal with non-linear data, SVM implements the so-called kernel trick that transforms the data into a higher dimensional feature space via a mapping function. The kernel trick has a γ parameter associated to it, that indicates how soft the decision boundary would be.

Although SVM is initially defined for classification, it can be applied to regression problems as well. In this case we try to define how much error is acceptable and the model finds a hyperplane to fit the data. SVR provides an estimate of the output as a non-linear function of the inputs.

The kernel function plays a crucial role in transforming inputs into a higher dimensional space and we used the non-linear RBF kernel. After applying nested cross-validation, we found that the best parameters for our model were: $\gamma = 1$, $C = 1000$, and $\epsilon = 1$.

Random Forest RF is an effective classification and regression method that involves ensemble learning of multiple decision trees. A decision tree is a simple concept in which the model learns a series of questions that define cut-off values for the different features in the dataset. It starts at the root and splits the data on the feature that generates the largest information gain, in an iterative process that finishes when the leaves are pure or at a predefined depth [7]. 50 trees with maximum depth of 20 were found to be the best hyperparameters after the tuning process.

Artificial Neural Network ANN is a concept roughly based upon the understanding of how the human brain works, and it was first introduced in the 1940's [7]. Its basic element is a processing unit called neuron that connects to other neurons and can be arranged in layers in order to fit complex functions. The connections between neurons have weights associated to them, which are the learnable parameters. These weights are updated simultaneously every pass of the training set, or epoch, and this is performed by an optimization algorithm that calculates the gradient of a loss function. The final goal is to minimize the loss function.

An ANN with three hidden layers (48, 48, and 24 neurons) was implemented [5]. In this study, we used dropout regularization, which randomly drops units and their connections from the neural network during training. After nested cross-validation, the dropout used to train the final model is 10%. Some important training parameters are the optimizer and loss function (Adam and MSE, respectively). The model was trained for 1000 epochs per fold, using a batch size of 20.

DL Models - Data Fusion

An important aspect of learning is diversity. For example, [99] demonstrates that their method for cross-modal learning improves robustness to outliers and generalization capabilities. In our study, to use all the data captured by the sensors, data fusion was implemented by means of two-headed models. One of these heads uses the tabular data while the other uses the sequential time-series data. Every sample or data point is constituted by those two components and the network produces a unique prediction for that given sample.

The sequential data contains more information (features) compared to the number of samples, and it is also dimensionally incompatible with the tabular part; therefore, it had to be processed by a different model capable of extracting the useful information.

Two similar versions of the two-headed model were tested: CNN- and LSTM-based. Both models use an architecture similar to the ANN network implemented for the averaged data, without reusing the weights, and this constitutes the first head. The other head is different for each model and is based on a previous paper [14], where CNN and LSTM networks show good results on time series data.

The training process of these algorithms was similar to the ML case, since it involved repeated k-fold cross validation, although in this case k=10 and the process was repeated 5 times. We did not use any regularization technique and no hyperparameters were tuned, so we did not use nested cross-validation in this case.

CNN-based Model A common DL model is the CNN, which is a kind of neural network that uses the convolution operation in place of general matrix multiplication in at least one layer, and is specialized in the processing of data that contains grid-like patterns, such as in the case of time series (1-D grid) [8].

A CNN is usually implemented as a stacked architecture with some convolutional and pooling layers interchanged. Fig. 6.3 shows the two-headed CNN-based model in which the left side is a two-layer one-dimensional CNN and the right side is the ANN based on the traditional ML model already used. In the lower part of the figure, there is a fully connected regressor that generates the final predictions based on the information extracted by the two upper branches.

The CNN branch (including the regressor) is based on the popular LeNet-5 architecture originally introduced by [29], and it has two layers with a depth of 20 and 30, with kernels of size 3 and 2, similar to what is implemented by [14]. From the reference architecture, we tried to keep things as unchanged as possible, so only the dimensions (from 2 to 1) and the final layer (multiclass

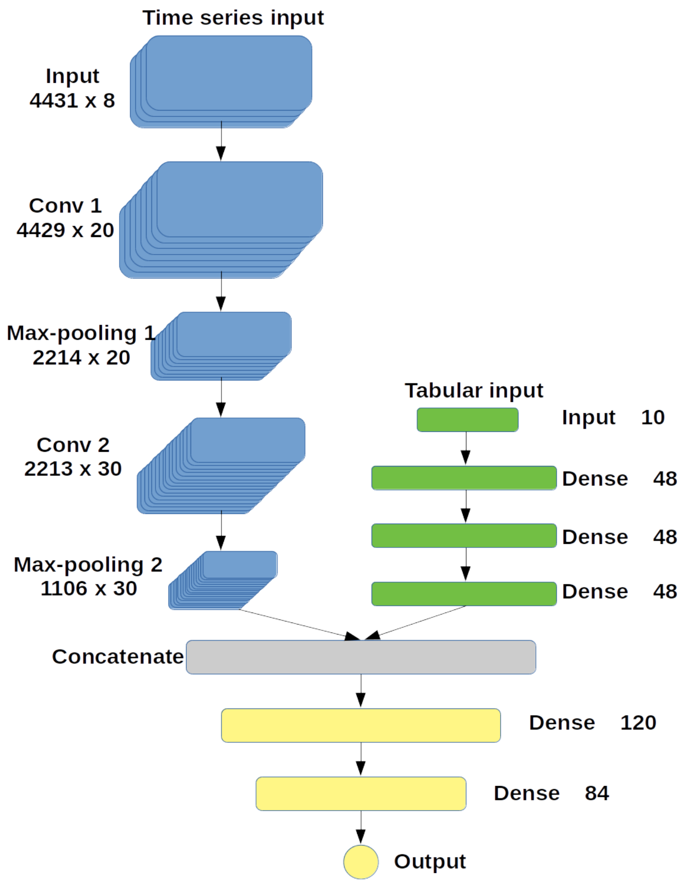


Figure 6.3: Architecture of the CNN-based two-headed network implemented to train the mixed dataset (data fusion)

classification to regression) were modified. The complete model was trained for 15,000 epochs per fold, with a batch size of 20, and Adadelta was used as optimizer.

LSTM-based Model RNN are a kind of network for processing sequential data, mainly because they are able to scale to much longer sequences than other networks [8]. They are able to achieve this task by keeping information from the past. LSTM is a common implementation of RNN, and it controls this flow of information by means of three gates: forget, input, and output. One downside of this network is that it involves more operations and therefore demands more computational resources.

In the LSTM implementation, the main difference with respect to the CNN model is that we replace the convolutional-pooling blocks by LSTM layers with 20 and 30 units to process the time-series data. This model was trained for 1500 epochs per fold and Adam was used as optimizer.

All models were implemented in Python 3.7, using Scikit Learn 0.22 for SVR and RF, and Tensorflow 2.0 for the other models. They were trained on a dedicated server with 64 Intel(R) Xeon(R) Gold 6130 CPU's at 2.10GHz, 1.6 TB of RAM, and a Nvidia Tesla V100-PCIE-16GB

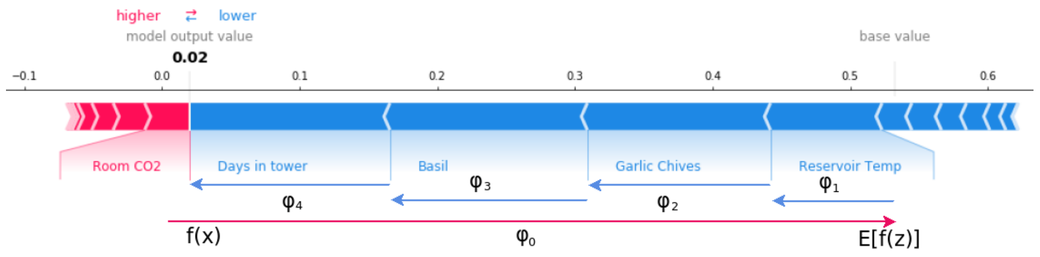


Figure 6.4: Example of how SHAP values make the output go from the expected ($E[f(z)]$) to the predicted ($f(x)$) value, related to the features in our dataset

GPU.

6.2.6 Model Interpretability

Physical models are based on known relationships between inputs and outputs, thus interpretability is inherent to them. On the other hand, statistical models (including AI) result from the observed data and learned weights. To authors in [34], this means that the relationship between inputs and outputs is unknown and, in most cases, impossible to interpret, so that they are usually referred to as black box models. While some ML models such as RF are interpretable, more complex models (such as the two-headed models in this chapter) are more challenging when their predictions need to give meaningful interpretations; therefore, researchers have tried to generate prediction models that can handle a wide variety of AI models.

For this work, interpretability is relevant to understand the features that affect the final yield of a crop. To accomplish this task, the SHAP approach was used. This is a game theory technique originally introduced in [36] that treats each feature as a player in a game, where that player (feature) contributes to increase or decrease the predicted value.

SHAP uses the expected value over the analyzed dataset as the base value (predicted value if no feature is known), and the effects of each feature are then subsequently added. The sum of these effects adds up to the predicted value. Fig. 6.4 shows an example of the representation of SHAP values for a randomly chosen example in our dataset that indicates how these add up to the predicted value. Fig. 6.4 is also an example of how SHAP values produce interesting visualizations, first introduced in [95].

The original model is a function of the training dataset; therefore, according to [34], SHAP values fully depend on it and they can be applied to the training or validation sets. This work uses the entire dataset. After finding the best hyperparameters for our models, we have retrained them using all the 200 samples in the dataset. These retrained models are the ones used for interpretability, although their prediction results are not reported because they would likely overfit the training set, and therefore would be over optimistic.

SHAP belongs to the additive feature attribution methods, which assign an individual effect to each feature and sum the effects of all feature attributions to approximate the output of the original model [36]. In the case of our sequential dataset, we obtain a SHAP value for each of the time steps in our data, which corresponds to the effect that each of those time steps has in the final prediction. Using SHAP, it is possible to obtain a global interpretation based on aggregations of SHAP values [96]; thus, we sum all the feature attributions for each one of our data samples to find an aggregate SHAP value for each of our features, per data point.

The SHAP feature importance is defined as the mean of the absolute SHAP values across the data [96, 34], and that is used in this chapter to obtain the features that contribute the most to yield prediction. Additionally, since we can aggregate SHAP values to obtain global interpretations, we construct the crop type feature that includes the effects of the six crop types, to simplify the information described by the individual features, and that is presented in Tables 6.3 and 6.4, and Fig. 6.5.

SHAP is implemented in different versions depending on the kind of original model that is used for generating the approximation. In the case of our work, we have used the Tree Explainer for the RF model, the Kernel Explainer for the SVR and ANN models, the Deep explainer for the CNN-based and the Gradient Explainer for the LSTM-based models.

6.3 Results and Discussion

There are four main results of this work: (i) AI models for yield prediction, (ii) interpretation of the predictions, (iii) optimization of the models, and (iv) impact of the features in yield prediction.

6.3.1 Yield Prediction

Results of the regression process are summarized in Tables 6.1 and 6.2 (models that use all features). Table 6.1 focuses on the traditional ML models and includes the results presented in Chapter 3, and as in the case of that Chapter, RF and ANN models show good performance. However, RF shows better generalization capabilities due to its best scores in the validation dataset. These tables also include the time that it took to train the final model in each case, in the server with the specifications mentioned in Section 6.2.5.

Table 6.2 shows the results of the CNN- and LSTM-based models that use the mixed dataset, with the CNN-based being the best performing model when measuring both the error metrics, MAE and MSE, and the coefficient of determination R^2, for the training and validation datasets. It is also worth noticing the difference in training times between the ML and DL models. Although the time to train the final model is in the order of seconds (or fractions of a second) for the traditional ML models (Table 6.1), the better performance of DL models justify the time it takes to train a single model that is in the order of one or two hours (Table 6.2).

Table 6.1: Performance metrics for the training and validation sets, for averaged data (best metrics are in bold)

ML model	MSE		MAE		R^2		Training time -
	train	val	train	val	train	val	final model
SVR[a]	N/A	N/A	N/A	N/A	0.86	0.45	N/A
SVR[b]	81.536	219.938	**3.832**	8.401	0.799	0.493	0:00:00.003
SVR[c]	98.301	214.011	4.540	8.361	0.766	0.507	**0:00:00.003**
RF[a]	N/A	N/A	N/A	N/A	0.9	0.62	N/A
RF[b]	53.872	148.435	4.200	7.309	0.869	0.661	0:00:00.125
RF[c]	52.573	**141.366**	4.051	**6.924**	0.873	**0.678**	0:00:00.053
ANN[a]	N/A	N/A	N/A	N/A	**0.91**	0.67	N/A
ANN[b]	**50.578**	174.299	4.038	7.266	0.879	0.604	0:00:43.397
ANN[c]	60.371	177.270	4.298	7.273	0.853	0.609	0:00:43.680

[a] Results presented in Chapter 3.
[b] Model trained using all features.
[c] Model trained using a reduced number of features (selected by SHAP).

Table 6.2: Performance metrics for the training and validation sets, for mixed data (best metrics are in bold)

ML model	MSE		MAE		R^2		Training time -
	train	val	train	val	train	val	final model
CNN[a]	**38.847**	**62.441**	**3.181**	**4.537**	**0.909**	**0.752**	**0:49:15.973**
CNN[b]	39.825	65.672	3.279	4.713	0.907	0.742	2:15:33.616
LSTM[a]	50.557	72.815	3.933	4.853	0.881	0.735	1:18:07.740
LSTM[b]	48.517	72.141	3.887	4.874	0.886	0.738	1:33:30.207

[a] Model trained using all features.
[b] Model trained using a reduced number of features (selected by SHAP).

These results confirm our hypothesis that ML/DL models can extract the useful information required to predict yield in our aeroponic system, and that using the data fusion process improves this prediction. Therefore, it is possible and even useful to feed the DL models with the data that comes from the sensors without averaging as we did with the traditional ML models.

6.3.2 Interpreting Predictions

One of the main goals of this chapter is to find which features are more relevant for yield prediction. Fig. 6.5 shows the plots of the most contributing features in the best models for both cases: RF and CNN.

Note the change in contributions of the sequential data, which have relative smaller average

Table 6.3: Most important 5 features found by SHAP for the models trained on the averaged dataset

	SVR	**RF**	**ANN**	**Selected features**
1	Days in tower	Days in tower	Days in tower	Days in tower
2	Harvest number	Reservoir temperature	Reservoir pH	Harvest number
3	Reservoir TDS	Room CO_2	Room CO_2	Reservoir pH
4	Room light level	Harvest number	Room humidity	Room CO_2
5	Reservoir pH	Room humidity	Reservoir temperature	Reservoir temperature
6	Room CO_2	Reservoir pH	Harvest number	
7	Reservoir temperature	Room VPD	Days in tray	
8	Room temperature	Room temperature	Room light level	
9	Days in tray	Reservoir TDS	Room temperature	
10	Room VPD	Room light level	Room VPD	
11	Room humidity	Days in tray	Grow number	
12	Grow number	Grow number	Reservoir TDS	

impact on Fig. 6.5a compared to Fig. 6.5b. This indicates that DL models extract more information from the sequential data than the pure average used in the tabular dataset.

6.3.3 Optimizing the Models

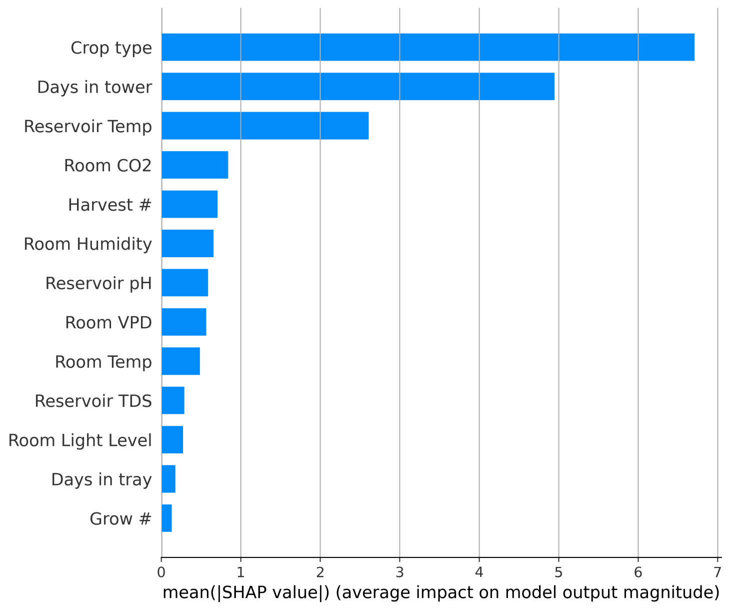
The models were retrained with the top five contributing features for each case. Tables 6.3 and 6.4 show these features for the two datasets. The same architectures and training procedure as in the original models were used, to make a fair comparison.

The results of training the models with reduced features are shown in Tables 6.1 and 6.2 (reduced features), showing similar or better performance metrics for both cases, especially for the validation sets. This is evidence of the relevance of those features for our problem.

Analyzing Tables 6.3 and 6.4 together, it is evident the importance of two features: Room CO_2 and Reservoir Temperature. SHAP values for these variables rate them as the most important for both datasets, so they should be considered as highly relevant in case of automating the production process.

6.3.4 Impact of Features on Yield Prediction

SHAP values give useful insights on how the features influence yield predictions. Fig. 6.6a, resulting from the RF model trained with the selected features, shows a positive impact of days in tower in the model output because low values in this feature (blue dots) cause lower predicted SHAP values (left side of the SHAP value scale), while reservoir temperature and room CO_2 (the most relevant



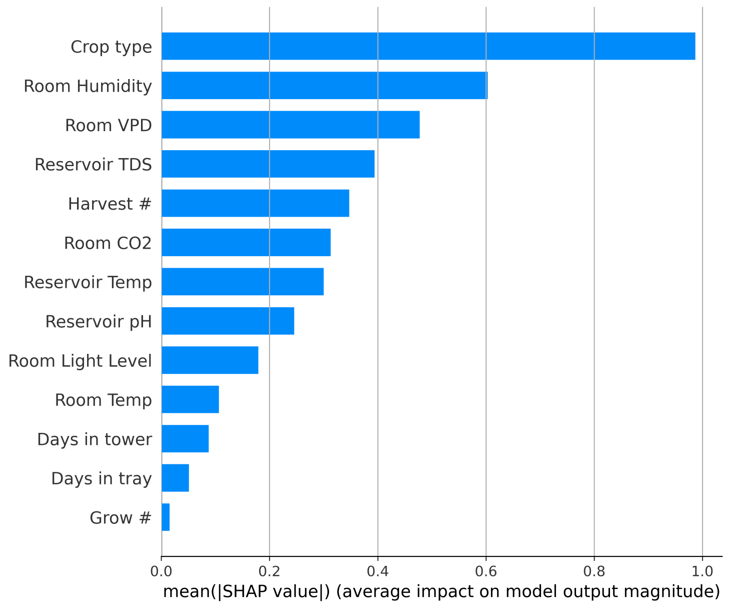


Figure 6.5: Contributions of individual features to yield prediction using the best model for each case. Top: Averaged data (a). Bottom: Mixed data (b)

Table 6.4: Most important 5 features found by SHAP for the models trained on the mixed dataset (if relevant for both datasets, selected features are in bold)

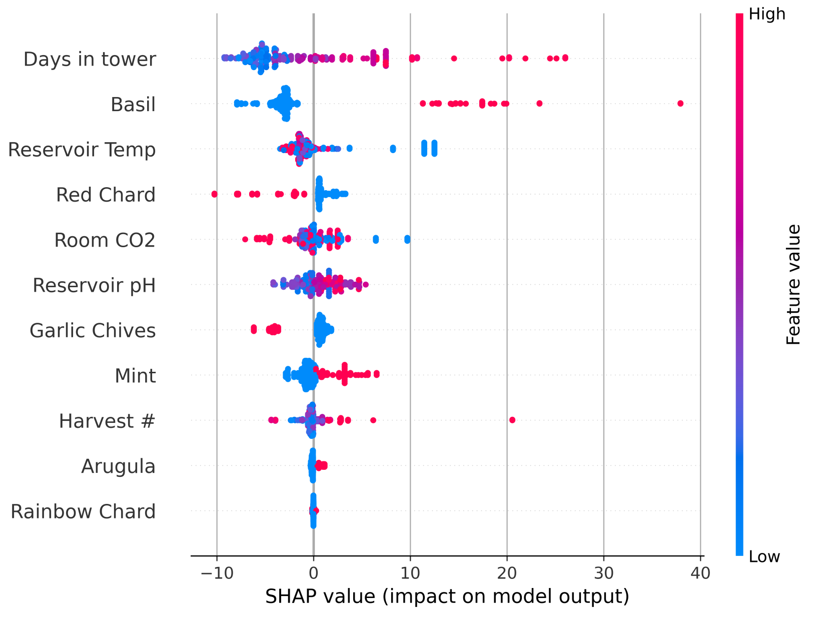
	CNN	LSTM	Selected features
1	Room humidity	Reservoir TDS	Room humidity
2	Room VPD	Room CO_2	Room VPD
3	Reservoir TDS	Room VPD	Reservoir TDS
4	Harvest number	Room humidity	**Room CO_2**
5	Room CO_2	Reservoir temperature	**Reservoir temperature**
6	Reservoir temperature	Room temperature	
7	Reservoir pH	Reservoir pH	
8	Room light level	Room light level	
9	Room temperature	Days in tower	
10	Days in tower	Grow number	
11	Days in tray	Harvest number	
12	Grow number	Days in tray	

features in Tables 6.3 and 6.4) have the opposite effect. Furthermore, Fig. 6.6b, the result of the CNN model, shows that reservoir TDS has a positive impact in the final predictions (i.e. higher feature values imply higher SHAP values), while reservoir temperature and room CO_2 show more complex relationships. In the first case, higher reservoir temperatures have little or no effect on the model outputs and lower temperatures impact the model either increasing or decreasing the SHAP values. For room CO_2, the relationship is the opposite.

The impact that these two features have on the model output should be further explored. While the effect of CO_2 in plant growth is known in general [100], and also the impact of the temperature of the roots has been studied before [101]; it would be important to further study these results, possibly designing new experiments for this particular case.

Besides SHAP, there are other recently proposed methods for analyzing the interpretability of a model, such as LORE [102], which is only defined for tabular data, or LEFTIST [103], for time series data. Therefore, these models would not fit in our work due to the mixed nature of our dataset, but they are nonetheless worth evaluating for future works. It would also be interesting to explore dimensionality reduction approaches such as clustering [97, 104], although they might not provide information regarding the individual features and their impact on model output, as it is the case of this chapter.

Also, with the development of precision farming, sensor analytics will be more relevant in agricultural applications. Authors in [105] expect that in the near future farms will integrate wireless sensor networks (WSN) and intelligent systems to increase and improve production. These



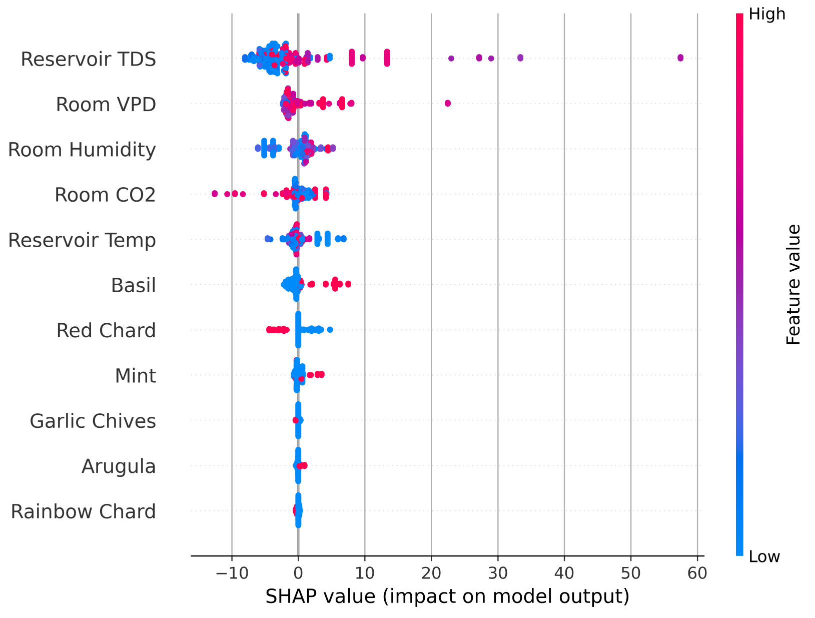


Figure 6.6: Impact of each feature in the final predictions using the best reduced model for each case. Top: Averaged data - RF model (a). Bottom: Mixed data - CNN model (b)

data could also be used by decision makers to manage agriculture in higher levels, according to [106]. This means that AI analysis tools, concepts and procedures will become even more relevant in the near future.

6.4 Conclusions

This interdisciplinary work presented a number of yield prediction models for aeroponic crops in a controlled environment, based on the environmental variables measured in the production system. 200 samples covering 6 different crops were used, and the presented models reached a coefficient of determination value R^2=0.752 for the validation dataset in the best case (CNN-based model).

The contribution of this chapter is twofold; first, results indicate that the use of data fusion is possible and even improves yield prediction in aeroponics, allowing the use of the data gathered by the sensors for the AI models, without the need for a complex preprocessing scheme. Second, the SHAP approach has found which features are more relevant for yield prediction of six different crops.

Two versions of the dataset were implemented and tested. First, a purely tabular (averaged values) set for which the best performing model is the RF. Second, a mixed dataset that includes the full sequential time-series data acquired by the sensors. In this case, the CNN-based model shows better results.

SHAP analysis was applied to the trained models, which identified two main features in our data: Room CO_2 and Reservoir Temperature. SHAP values also provided important information for re-training the models with reduced features, producing similar or better performance metrics, which is evidence of their relevance. SHAP values also give useful insights of how these features influence the predictions. Results indicate that there is positive correlation between days in tower and yield, and that reservoir TDS has a positive impact in the final yield, while reservoir temperature and room CO_2 present more complex impacts.

Results also show the potential for implementing a yield prediction model as the first step towards the full automation of an aeroponic crop production system that would require less human intervention with a higher profit margin. For future works, it would be interesting to evaluate other recently proposed methods for model interpretability, such as LORE or LEFTIST, especially considering the development of precision farming where sensor analytics and intelligent systems will be used to increase and improve production.

7 Improving the Detection of Explosives in a MOX Chemical Sensors Array with LSTM Networks[6]

This chapter continues to explore the idea of selecting relevant features for model optimization. In this particular case, it deals with taking only the initial time-steps of a sequence in order to make a prediction. Different ML and DL models are implemented and tested. In the end, a light-weight model, that could speed up the classification task, is implemented into an embedded device. By using only the initial part of the sequences, the number of trainable parameters (weights) is reduced in one order of magnitude when using 30 seconds instead of 300, for the LSTM-based model. This model was implemented and tested on a Raspberry Pi 3 board, which could be used as an on-site processing element directly connected to the data acquisition card, which would be an important part of an accurate, portable, fast and low cost e-nose device for the detection of explosives. Thus, this chapter constitutes an example of the whole proposed process, starting with the implementation of customized AI models, going through model optimization, and finally arriving to the implementation on an edge device.

Entities throughout the world face the problem of detecting hidden explosives, where human and canine inspection might not be a viable solution. Therefore, it is important to develop fast, reliable, and portable integrated inspection systems by means of automated methods, such as electronic noses. The goal of the work presented here is to develop an accurate, fast and light-weight machine/deep learning classification model to be used in a MOX chemical sensors array (electronic nose), in order to identify explosive substances.

For this chapter, 140 samples were taken, combining TNT or gunpowder with either soap or toothpaste, or acquiring raw samples of those substances in amounts ranging from 0.1 g to 2 g. For the classification problem, among the different options in machine learning techniques, five models were evaluated. The implemented LSTM version of a LeNet-5 based network, classifies accurately the compounds in 100% of the cases when using only 30 seconds from the 360 obtained by the sensor array per each sample. The results of this work indicate that the proposed LSTM-based

[6]The content of this chapter is originally published in IEEE Sensors Journal [14]. The manuscript has been reformatted for inclusion in this thesis.

Julio Torres-Tello: Conceptualization, Methodology, Software, Validation, Data Curation, Writing - Original Draft, Visualization. Ana V. Guaman: Validation, Investigation, Data Curation, Resources, Writing - Review & Editing. Seok-Bum Ko: Resources, Writing - Review & Editing, Supervision, Project administration.

deep learning model could be easily implemented into an embedded system.

7.1 Introduction

There have been many terrorist activities in past years around the world, in such different places like Brussels, Abuja, Boston, London, Madrid, Moscow, Mumbai, etc. [107] which have had impacts on how security is managed worldwide. Therefore, illegal trafficking of explosives is a real challenge to civil security, and law enforcement entities throughout the world face the problem of detecting them where human and canine inspection is at least difficult and expensive [108]. Explosive substances are often mixed with other compounds, so that they can be transported and marketed.

There are well known accurate techniques for the detection of chemical substances, such as gas chromatography-mass spectrometry (GC-MS), ion-mobility spectrometry (IMS) [109, 110, 111], arrays of high-selective sensors; however, e-noses are usually cheaper, smaller, and convenient for rapid detection [18, 108, 112, 113]. Works in literature use metal oxide (MOX or MOS) sensors for e-nose implementations or at least as a reference technique [18, 19, 112, 113, 114, 115], and in some cases specifically for explosive detection [116, 117, 118, 119], and [120] for example, used PCA to discriminate 1g of different explosives (TNT, RDX and PETN).

Chemical sensing is challenging because it tries to emulate the complex biological olfactory system, which is composed by hundreds of receptors and neurons [121]. The substances that generate volatile organic compounds (VOC) can be identified by humans and animals from past odor experiences because data remain into their memory as synaptic weights [122].

One of the important points when analyzing data in order to detect an explosive, is the development of a model that is accurate and fast. Currently, the development of new techniques open new possibilities for the development of such models. DL is emerging as an important solution for many classification problems. As a simple example, a search for the terms *deep learning* and *classification* in the IEEE Xplore database provides 12,344 results since 1996; however, its application in electronic noses (e-nose) is just starting (we obtained only two results [19, 123] published on journals out of the 3,662 e-nose / chemical-sensor-array related works in that database), and the search for *deep learning* and *electronic noses* gave only 6 results in the Scopus database. In any case, DL is starting to be adopted as a solution for some problems that are common in e-noses, such as accurate classification [6, 122, 30, 113, 114, 115, 124, 125], sensor drift correction [126], fast detection [127, 128], navigation [112], and even trying to replace dogs as explosive seekers [108].

Furthermore we look for a light-weight model that could be eventually implemented into an embedded device, so our algorithms try to keep a low number of layers (and parameters), in contrast to deeper and more complex models found in literature, such as [30] that implements a CNN network with 38 layers. However, we have found that [6] implements a gas identification algorithm based in the popular CNN architecture known as LeNet-5 [29], which is one of the first successful implementations of DL. The aforementioned work is used as the reference model for our implementation.

The work described in this chapter elaborates over an e-nose presented on [121, 129, 130]. Researchers in [121, 129] used the e-nose for the detection of dinitrotoluene, vinegar, ethanol, trinitrotoluene, gunpowder in double base and alcohol, for which they used simple ML models such as PCA, Linear Discriminant Analysis (LDA), and kNN, obtaining 86.66% discrimination accuracy. Similar results are reported in [130].

In this work, we develop not only an accurate classifier for the detection of substances containing explosives, but also one that could speed up the classification task. As it will be explained in the data collection procedure, each sample in our dataset consists of the response of six chemical sensors during six minutes, and that is the time length that we try to reduce.

Therefore, in this chapter we evaluate the possibility of using a ML/DL algorithm that could give accurate results in less time, which would be important in order to design future experiments with limited data-collection times. The main purpose of the work presented here is to develop an accurate, fast and light-weight classification model to be used in an e-nose in order to identify explosive substances (Gunpowder and trinitrotoluene in raw state or mixed with other substances).

The algorithms used in this study will be described in Section 7.2, along with the e-nose prototype. Section 7.3 discusses the results obtained, and section 7.4 presents some conclusions.

7.2 Materials and Methods

7.2.1 The Electronic Nose Prototype

The e-nose is an analytical device for sensing VOC, attempting to mimic the biological olfactory system. It is composed by three functional blocks as shown in Fig. 7.1.

The sensing block is made up of a sampling system, a sensing chamber, and a pneumatic system. The pneumatic system is composed of an air pump of 3 lpm (litres per minute), three electrovalves of 12VDC with a maximum caudal of 31 lpm, and pipes with a diameter of 4mm. The whole system caudal is limited to 1.1 lpm, mainly in order to guarantee a proper sweep of the VOC explosive through the vials where the samples are placed. The electrovalves allow switching from two scenarios: (i) pumping VOC through the sampling system into the sensing chamber and (ii) cleaning the system by allowing clean air to pass through the whole system eliminating VOC content.

The sampling system is based on a headspace methodology to foster the VOC generation, and it mainly has an air pump and two vials: one with 2 ml of ethanol and one with the explosive at different weights. In essence, the pump makes air flow through the first vial in order to push the

VOC of ethanol into the second vial, where it gets mixed with the VOC of the explosive substance. This mixture of ethanol and explosives is what activates the sensors.

An array of six MOX chemical sensors from the FIGARO family (two of them are T-GS2610, two T-GS822, and the others are T-GS825 and T-GS826) composes the sensing chamber. MOX gas sensors such as the ones used in this work contain dense packs of granular material of N-type metal oxide (tin oxide (SnO_2), zinc oxide (ZnO), etc.), or P-type metal oxide (copper oxide (CuO), chromium oxide (Cr_2O_3)). The gas sensing mechanism is based on the oxidation-reduction reaction [131]. A voltage divider is used as a signal conditioning circuit where the load resistance was set up to maximize sensitive sensor response.

The sensor chamber has a volume of 0.5 l, and a PID temperature controller was set up to avoid external environmental changes that might affect the behavior of the sensors. The reference temperature was set to 29°C during the whole experiments, with 0.05% steady-state error, and the sensing chamber temperature was kept for 12 minutes before any data-collection experiment took place. The data collection algorithm sampled the output of the sensors at 1 Hz, and it was implemented on an Arduino UNO board.

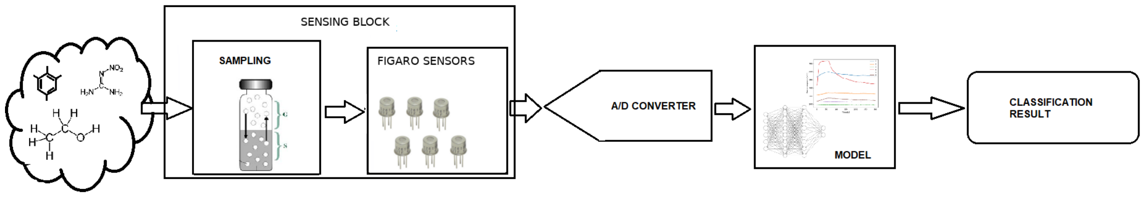


Figure 7.1: Block diagram of an Electronic Nose prototype used in this work.

7.2.2 Data collection

Each data-collection experiment lasted 360 seconds, of which VOC was pumped from the sampling system to the sensing chamber during the first minute, and the cleaning system was activated for the remaining five. The data files were finally bounded to five minutes (300 s) for practical purposes because some experiments finished the cleaning process earlier.

Raw TNT and gunpowder measurements were taken in amounts of 0g, 0.1g, 0.2g, 0.3g, 0.4g, 0.5g, 0.7g, 1g, 1.5g, 2g. Besides, TNT and gunpowder were mixed with 1g of soap and 1g of toothpaste for testing the device under real scenarios, considering that these items could be used by criminals to camouflage the explosives. For the collection of each of these samples, we used 2 ml of ethanol, using headspace methodology as it was previously explained. Also, we used ethanol only as blank of the system. Random experimentation was carried out to avoid any memory effect that could influence the results.

The obtained database consists of 140 experiments where 44 come from TNT, 37 are gunpowder, while 6 of them were only soap and 6 only toothpaste, and 47 measurements were blanks. Table 7.1

Table 7.1: General description of the database created for this work.

Substance	Number of experiments		Label
TNT	32	44	Explosive
TNT + soap	6		
TNT + toothpaste	6		
Gunpowder	29	37	
Gunpowder + soap	4		
Gunpowder + toothpaste	4		
Soap	6	59	Non-explosive
Toothpaste	6		
Ethanol (blank of the system)	47		

shows a detailed explanation of the database, but notice that in the end the classification problem is reduced to identify explosive and nonexplosive samples.

7.2.3 Data Pre-processing

Data augmentation

A small number of samples is usually avoided in order not to overfit the data [6, 115]. Since only 140 samples were obtained, mainly due to the difficulty of acquiring larger amounts of explosives, we implemented data augmentation techniques. As a first step, we noticed that the important information contained in the response curves of the sensors was in the low frequency range; therefore, a moving average (k = 5, and k = 10 window sizes) filtering process was performed in order to obtain new (augmented) samples. This process resulted in the generation of a dataset with 420 samples (three times the size of the original dataset).

The next step was based on the data augmentation process described in [6], in which translation of the feature matrices is performed. We rotated the vectors containing the response of each of the sensors, as shown in Fig. 7.2. This procedure increased the number of samples by six, giving us a total of 2,520 samples to work with. This data augmentation technique considers that there is no useful correlation between the order of the vectors and the physical distribution of the sensors, and this procedure will even allow the model to generalize better in the sense that it will be robust against the specific response of a sensor in a given physical location.

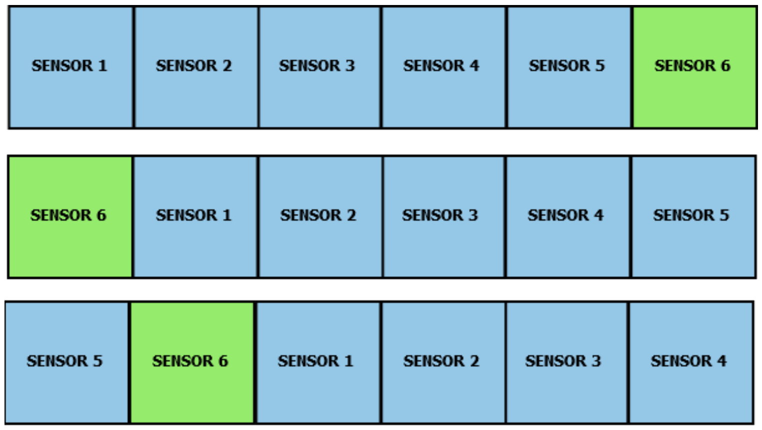


Figure 7.2: Translation of the features, in this case by rotating the vectors containing the response of each of the six chemical sensors.

Mean normalization and feature scaling

A randomly selected sample is shown in Fig. 7.3, where we can see the lack of a common baseline for the different sensors and the difference in amplitude among signals. Therefore, signals went through a pre-processing phase consisting of feature scaling and mean normalization in such a way that all features would be in the same scale and have approximately zero mean, obtaining signals similar to the ones shown in Fig. 7.4. This figure is also useful to see the responses of only one sensor to different stimuli (pure substances as an example). This, along with the literature that supports the use of MOX sensors for the detection/classification of gasses [18, 19, 112, 113, 114, 115, 116, 117, 118, 119, 120], was used as evidence that classification between explosives and non-explosives was possible for our case.

It is important to mention that this pre-processing scheme was performed on the already time-reduced samples (it will be clearly explained in the following sections), in order to keep the idea of speeding up the explosive detection.

7.2.4 The Machine Learning Algorithms

After reviewing the literature and analyzing our options, five different algorithms were implemented and evaluated in this work: A kNN, an SVM, an RF, a CNN, and an LSTM network. The three former algorithms correspond to classical ML techniques and were tested in order to constitute a baseline for the two DL models (CNN and LSTM based), which are the real focus of this work. There are other implementations of neural networks having application in sensor data, such as fully connected networks [114] and auto-encoders [18, 113]; each one being used for different tasks; however, we believe CNN and LSTM better fulfill our requirements.

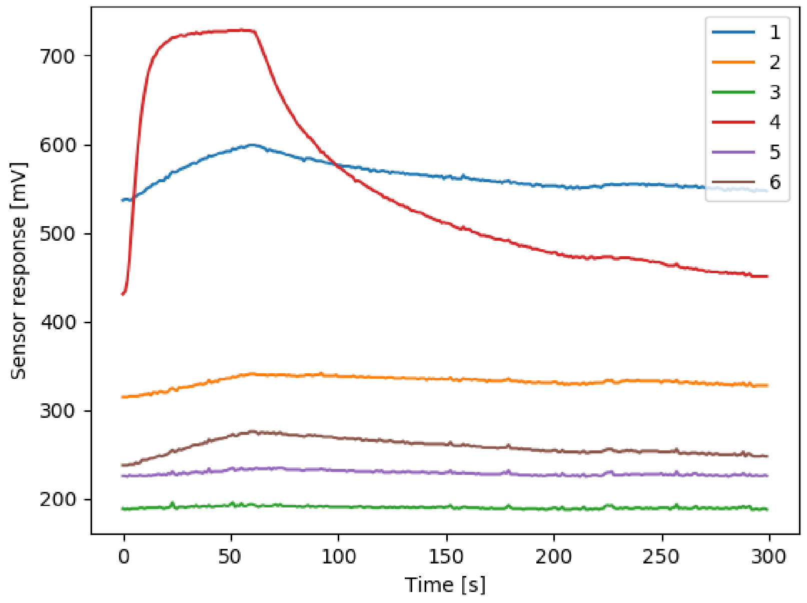


Figure 7.3: Plot of the response of the six chemical sensors in a randomly selected sample.

k Nearest Neighbors

kNN was chosen for the classification task because it was implemented in the works [121, 129, 130] that used the previous version of this e-nose. This is a simple algorithm that chooses the class of the sample we want to classify, among the k nearest neighbors (samples), by majority vote. One downside of this method is that the model needs to memorize the whole training set in order to be able to make a decision, which makes it poorly scalable.

Support Vector Machine

This algorithm is widely used for classification tasks [7], due to its good performance in terms of training speed and unique solution. In our literature review we found that this is a popular model for detection and classification of substances in e-noses. For example, [126] uses an SVM with RBF kernel as baseline model, and it is mentioned and/or used as a reference model for other works as well [30, 113, 122, 124, 127, 128].

Random Forest

This is another algorithm popular in similar tasks. For example [20] uses RF for the identification of liquors, [132] uses it for the optimization of a gas sensor array, and [127] implements it as a baseline model in their gas recognition system.

RF is an ensemble model that combines decision trees in order to generate a strong learner,

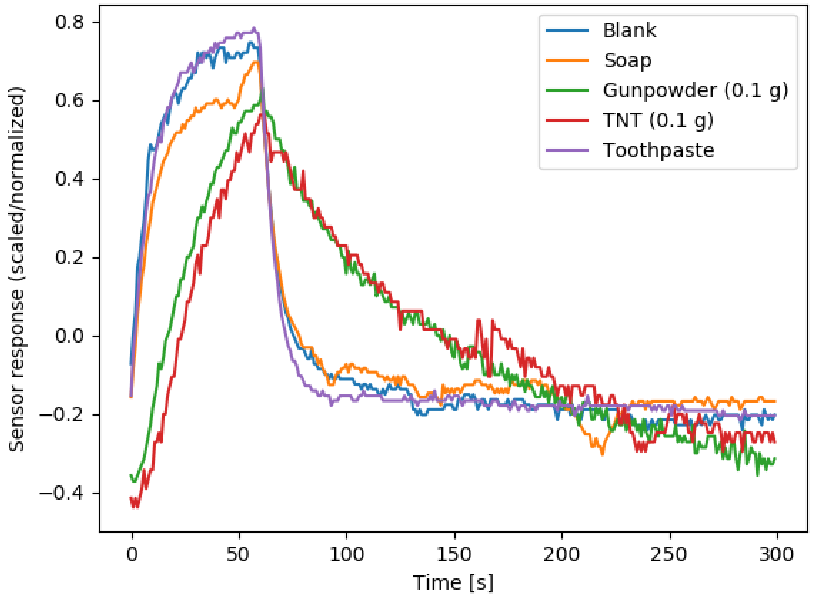


Figure 7.4: Normalized and scaled response of one sensor after preprocessing, for different substances.

finally classifying by majority vote. A decision tree is a classifier that learns questions (thresholds) that separate classes in an iterative process.

Convolutional Neural Network

A very popular algorithm in DL is CNN, given the good results it provides in many areas. The main idea behind a CNN is the implementation of multiple layers of feature detectors, that automatically identify patterns within the data. In order to learn those patterns, convolutional layers use a sliding filter (kernel) that covers the whole sample (input feature map) and generates an output feature map that captures the important characteristics that are finally used by a fully connected classifier.

CNNs have been designed for processing visual data [30], which requires two dimensional kernels; however, they can have any number of dimensions [4]. One dimensional kernels are suitable for gas recognition problems [125]. In the case of our study, we consider that this will capture the independent response of each sensor to the stimuli; therefore, we have adopted 1D kernels for the implementation of our CNN.

Many architectures of CNNs have been proposed and used in the last years, and the first successful implementation was the so called LeNet-5 [29], that was used to classify handwritten digits. LeNet-5 has been used as the basis for the gas classification structure proposed in [6], which is in turn the basis for the classifier presented in this work.

The CNN architecture presented in [6] got 98.67% accuracy in its implemented task, which

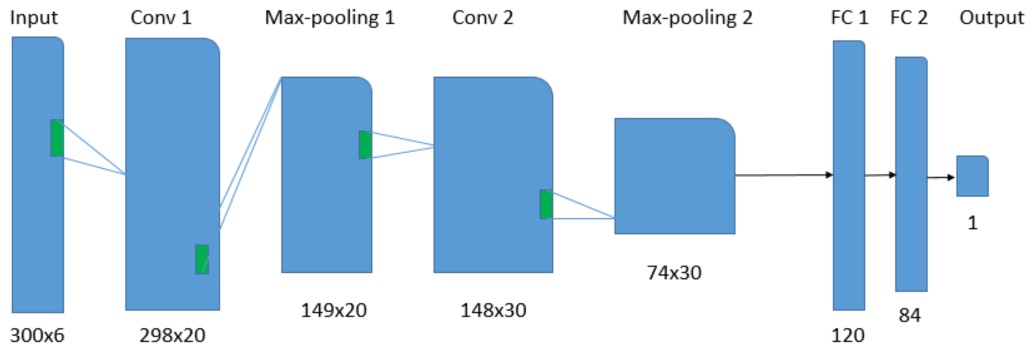


Figure 7.5: Implemented 1D LeNet-5 CNN, taking [6] as reference.

corresponded to a three-class gas identification using an e-nose constituted by 12 chemical sensors acquiring data during 8 minutes. We can see that the general task is similar to the one being addressed in our study, since we have a six-sensor e-nose that acquires samples during six minutes, and outputs two classes. The general architecture of the network in [6] has two conv layers of dimensions 10x10x20 and 4x4x30, with kernels of size 3x3 and 2x2 respectively, intercalated with two max-pooling layers of dimensions 5x5x20 and 2x2x30 with kernels of size 2x2. Finally, it has two dense layers of size 120 and 84.

Fig. 7.5 shows the CNN implemented in this chapter. We have tried to keep as many characteristics as possible from our reference model, such as the number and depth of the layers and the kernels; however, due to the differences in the dataset, the dimensions of width and height are different. Understanding of the dimensions and shape of datasets corresponding to gas sensors is often the cause for new proposed network architectures, such as the reference model [6], or the one in [115], or ultimately the one proposed here.

We have chosen to implement 1D convolutions, so we have one less axis in our tensors. The use of 1D convolutions means that the network uses the same filter for each of the individual six channels at its input, but the output is just the class (explosive or not) to which each sample belongs. Also, we use 300 seconds of data per sample (in the initial implementation), and this caused that instead of having 10x10x20 as dimensions of the first layer, we would have 300x20, and therefore the following layers will also be different given that they come as result of pooling-conv operations.

Just like in the reference model implemented in [6], some of the parameters used are the kernel size for the conv layers set to 3 and 2 (layers 1 and 3 respectively), pooling kernel of size 2, the activation function is ReLU for all layers except the last one which is a sigmoid function because it is a binary classifier, and no padding. Here it is worth noticing that in our first trials we kept the same vector size of the kernels as in the reference model (i.e. 3x3=9, and 2x2=4), but keeping the size of the single dimension (3 and 2 respectively) gave more consistent results.

Long Short Term Memory

LSTM has emerged as an effective and scalable model for several learning problems related to sequential data. Previous methods for addressing these problems have either been specifically designed towards a particular case or did not scale well [32]. In literature, there are some publications that validate the general idea of LSTM networks for their use in e-noses but only a few implemented cases. [124] mentions that their proposed LSTM prediction model has a strong applicability and high accuracy in the concentration identification of gas mixtures. [127] also introduces the idea of including one LSTM layer as a component of their model used for fast gas recognition, although details of the implementation, architecture and results are limited.

RNNs, such as LSTM, are special neural networks in which memory effect takes an important role. In these networks a state of what the network has seen so far is kept and used during the learning process [1]. This memory of the past is what is important in time-series datasets such as ours. LSTM is a special implementation of the general RNNs, that deals with the vanishing gradient problem, by means of three gates: forget, input, and output; which are described by equations (7.1), (7.2), and (7.3) [8].

The forget gate controls the self-loop weights, and a unit $f_i^{(t)}$ for the time step t and cell i, is defined by (7.1).

$$f_i^{(t)} = \sigma\left(b_i^f + \sum_j U_{i,j}^f x_j^{(t)} + \sum_j W_{i,j}^f h_j^{(t-1)}\right), \tag{7.1}$$

where b^f, U^f, and W^f are the biases, input weights, and recurrent weights for those forget gates, in that order; and x^t and h^t are the current input and hidden layer vectors, respectively. Note that the final value of the gate is set to be between 0 and 1, due to the sigmoid activation function. Accordingly, the input $g_i^{(t)}$ and output $q_i^{(t)}$ gates are defined in a similar fashion, but with their own parameters.

$$g_i^{(t)} = \sigma\left(b_i^g + \sum_j U_{i,j}^g x_j^{(t)} + \sum_j W_{i,j}^g h_j^{(t-1)}\right), \tag{7.2}$$

$$q_i^{(t)} = \sigma\left(b_i^0 + \sum_j U_{i,j}^0 x_j^{(t)} + \sum_j W_{i,j}^0 h_j^{(t-1)}\right). \tag{7.3}$$

In this work, we have taken the model shown in Fig. 7.5 and we have modified it by replacing each conv-pooling block by an LSTM layer of the same depth, as shown in Fig. 7.6. The resulting network architecture has two LSTM layers (one per conv-pooling block), and the two dense layers from the original model. The idea behind replacing the convolutional operation with LSTM cells is that the later would be more effective in capturing the time-series behavior of the response of the chemical sensors (given their dependence on t, as shown in equations (7.1), (7.2), and (7.3)), and therefore it was expected that this architecture would provide a better accuracy when dealing with limited time-related data.The activation functions used are the same as for the CNN version.

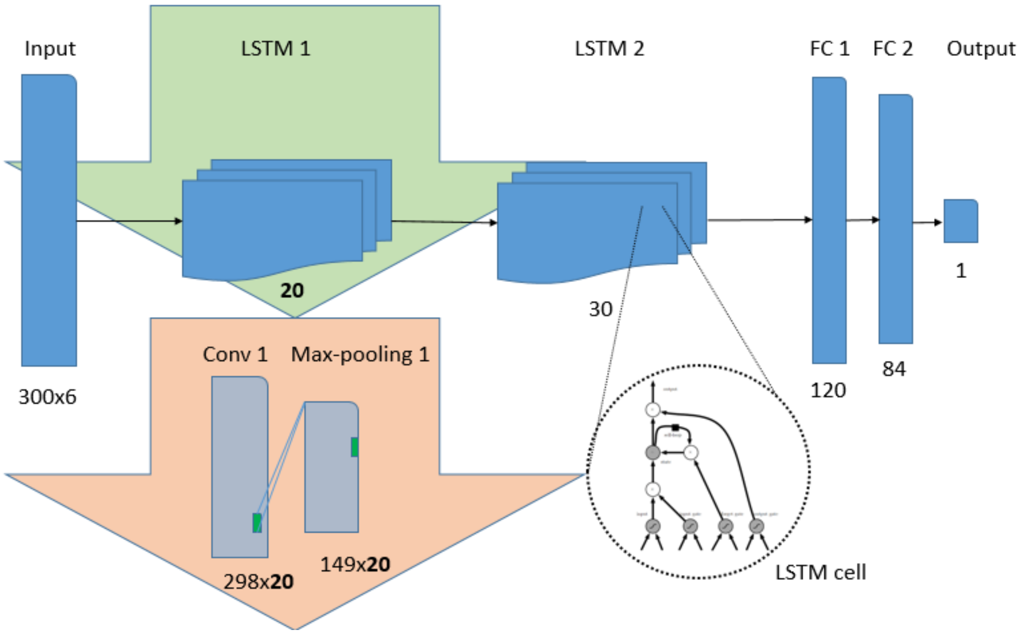


Figure 7.6: Conv-max pooling blocks being replaced by LSTM layers with the same depth. LSTM cells are the basic component of these layers, instead of being single neurons.

7.2.5 The training process

In order to tune the hyperparameters of the different models used in this work, and to obtain a true error rate, we have implemented a nested cross-validation process for our algorithm selections [7]. Fig. 7.7 shows the 5x2 nested cross-validation process, in which the inner loop is used for hyperparameter tuning, and when the best parameters have been found they are used for training/testing the model in a 5-fold cross validation fashion, which is a resampling technique without replacement in which each sample in the dataset is part of the training and testing set only once.

kNN, SVM and RF

Due to the nature of these three ML models, the vectors containing the response of the sensors had to be first concatenated one after the other, in order to be fed into the models. No other additional processing (dimensionality reduction, feature extraction, etc.) was performed, in order to make a fair comparison with the DL models.

The kNN algorithm was implemented and tested using the previously mentioned process. During the grid search the number of neighbors was chosen among 5, 7 and 9.

We implemented and tested an SVM model with the same criteria, but the hyperparameters tested during the grid search were the regularization ones C (1, 10) and gamma (0.01, 1). Regu-

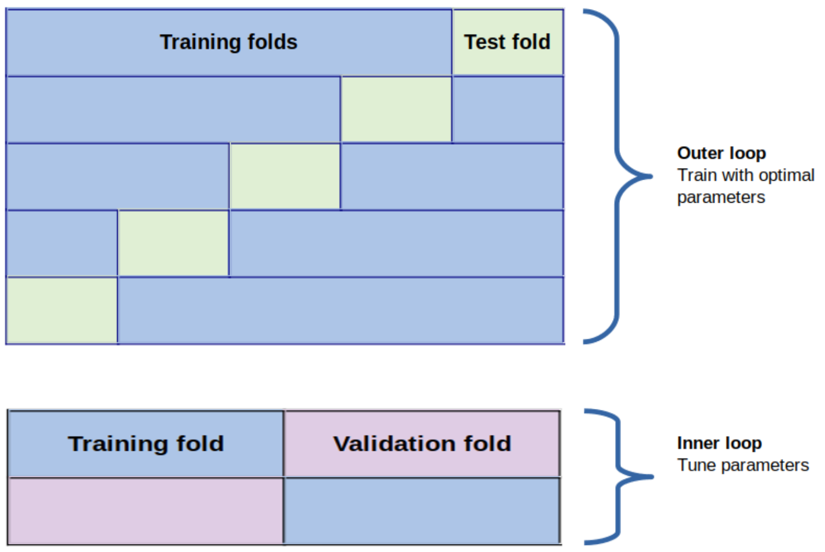


Figure 7.7: The 5x2 nested cross validation process used for this work.

larization is important in order to find a model that does not overfit, or fails to generalize over the test set. Our SVM model used an RBF kernel.

Finally, we implemented a RF algorithm, in which the tuned hyperparameters where the number of trees (10, 20, 30) and the depth (5, 10). One important difference between RF and the previous two ML models is that the former does not require data normalization, given the nature of the algorithm [7].

These three ML models were implemented using Scikit Learn on Python.

CNN

The general structure of this neural network is shown in Fig. 7.5. During the training process L2 regularization was implemented in the conv layers, and dropout set at 10% for the conv and dense layers to avoid overfitting. Adam was used as the optimization algorithm with a decay rate of 1e-7, and binary cross entropy as the loss function because it is a binary classifier. A batch size of 50 was chosen, and the model was trained for at most 500 epochs each fold.

For the grid search process the tuned hyperparameters were learning rate (1e-4, 3e-3) and L2 regularization (0.005, 0.01).

LSTM

Fig. 7.6 shows the structure of this network. In order to avoid overfitting and to create a robust model, a dropout of 10% is used for every dense layer. Adadelta was used as the optimization algorithm and binary cross entropy was used as the loss function. A batch size of 50 was chosen, and the model was trained for at most 500 epochs each fold.

Regarding the grid search process, the recurrent dropout rate in the LSTM layers was the only hyperparameter being tuned, between 20 and 30%.

Both CNN and LSTM networks were implemented using the Keras framework with TensorFlow backend. The training process took place on an Nvidia Tesla V100-PCIE-16GB GPU.

7.2.6 Speeding up classification

As it was already explained in the experimental procedure, each sample in our dataset consists of the response of six chemical sensors during 300 seconds. For the fast classification approach, we consider matrices of Nx6 elements, where N is the time-length of our samples and it varies from 300 down to 20. This process also allowed us to have smaller models for smaller sizes of the inputs, which leads to a lighter model that is easier to implement in an embedded system.

The idea of the fast classification is that we may not need to collect the whole response of the chemical sensors (six minutes), but that an accurate classification could be achieved with less time-related data. For this purpose, data has been restricted to five minutes, then four, three, and so on. There are some works in literature that try to make a fast detection of substances [21, 127, 128, 133] that implement a similar concept for e-noses using CNN or mixed networks; however, they do not show results for other kind of networks or algorithms, or any other comparison. In any case, the results of the speeding-up process are also dependent on the dataset, and a comparison between our results and the ones of those papers would not be fair.

7.3 Tests and Results

In order to evaluate the results of this work, we have divided this section into three parts: one dedicated to the classifiers when using the dataset with all the time-related values (N = 300), one that focuses on the iterative restrictions in time (N = 240, 180, ...), and finally the results of implementing the best performing model into a low-cost portable computer.

As a point of reference for all those results, we introduce Table 7.2, which contains the accuracy and the mean AUC (area under the curve). Initially we considered presenting also true and false positive rates in order to make clear the evaluation of unbalanced classes, but due to the high accuracy obtained with the two DL models, we have omitted this part.

7.3.1 Classification at 300 seconds

The results of evaluating all the classifiers using the whole dataset are shown in the first row of Table 7.2. There we find the accuracy for the five models when using the 300 seconds response data from the six chemical sensors. Results correspond to the test datasets in a 5-fold testing fashion.

Table 7.2: Classification results for the five implemented models, for different time-restricted data (5 down to 1 minute for all models, and 30 and 20 seconds for the LSTM-based implementation, due to its good accuracy with less data than other models).

Model	**kNN**		**SVM**		**RF**		**LeNet-CNN**		**LeNet-LSTM**	
N[s]	Acc.(%)	AUC	Acc.(%)	AUC	Acc.(%)	AUC	Acc.(%)	AUC	Acc.(%)	AUC
300	57.34	0.60	63.69	0.67	60.16	0.61	100	0.99	100	0.99
240	57.70	0.59	65.75	0.69	59.33	0.59	99.84	0.99	100	0.99
180	59.64	0.58	61.83	0.66	59.17	0.60	98.53	0.99	100	0.99
120	56.94	0.56	60.44	0.58	57.78	0.56	77.94	0.85	100	0.99
60	59.33	0.56	57.62	0.57	57.38	0.55	60.24	0.55	100	0.99
30	-	-	-	-	-	-	-	-	100	0.99
20	-	-	-	-	-	-	-	-	72.94	0.77

Table 7.2 shows that all the ML models (kNN, SVM and RF) fail to classify the samples, even when using the whole dataset. Among these three models, SVM achieves the better accuracy with almost 64%.

In the aforementioned table, we can see that the two DL models classify accurately the compounds containing explosives in 100% of the cases; therefore the receiver operating characteristic (ROC) curves have not been plotted.

7.3.2 Performance of the models with less time-related data

While data was being reduced, the algorithms were modified accordingly, but we tried to keep them as close as possible to the original implementation. Of course, the dimensions of the dataset and therefore of the hidden layers (of the DL models) had to be changed, and a training process was performed independently for each case.

The only important modification involves the shape of the input data. The size of the samples (Nx6) needs to be changed for each chosen N. As a result, the number of training parameters changed for every N (Table 7.3), generating each time lighter versions of the DL models. The direct consequence of this modification is the reduction of the number of weights that need to be trained. This is important if we want to deploy the resulting solution in a device with limited resources.

Table 7.2 shows the restriction steps followed in our work, one minute at a time (and 30 and 20 seconds at the end for the LSTM-based model).The LSTM-based model is the one that keeps its good accuracy by much longer than the other models; in our experiments it proved to work well with only 30 seconds, and that is the main result of this work.

Table 7.3: Number of training parameters of the DL models, varying according to the restriction in the data length.

	Number of training parameters	
N[s]	**LeNet-CNN**	**LeNet-LSTM**
300	278,379	1,098,649
180	170,379	666,649
60	62,379	234,649
30	-	126,649

Another important result presented in Table 7.3, is that both DL models show similar number of trainable parameters at the point in which they still have a good accuracy (180 s for CNN and 30 s for LSTM). Also, the number of trainable parameters (weights) is reduced in one order of magnitude when using 30 seconds instead of 300, for the LSTM-based model.

7.3.3 Inference on an embedded system

In order to validate the hypothesis that the model was small enough to be implemented into a portable device, the LSTM model obtained for the 30-second classification problem was implemented and tested on a Raspberry Pi 3 board, which could be used as an on-site processing element directly connected to the Arduino UNO acquisition card. For this purpose, TensorFlow and Keras were installed in this device and we run the inference model over the 140 original samples.

The results of this experiment showed that the inference model was run for all the 140 samples in 2.017 seconds, which means it would take 14.4 ms per sample. If we consider that in order to detect an explosive we need 30 to 60 s for the cleaning process of the e-nose as mentioned in section II, and 30 s of data acquisition, the whole process could take up to 90 s, plus the aforementioned inference time which would be negligible for our case. This is a real improvement if we consider that the original situation required of around 6 minutes.

As a first step, we implemented these classification models to see whether or not the collected data would be enough to detect the explosives, and the results of our experiments support this idea. Our results also indicate that the proposed DL model could be easily implemented into an embedded system, becoming a key component of an accurate, portable, fast, and low cost e-nose device for the detection of explosives.

7.4 Conclusion

This work presents a testing procedure to determine the accuracy of an e-nose consisting of an array of six chemical sensors, when detecting explosive substances in a challenging situation, and when the available time related data is being restricted. Five algorithms were implemented: kNN, SVM, RF, CNN and LSTM. The two DL models resulted in an accuracy of 100% when using 5-minute data, and kept an accuracy of at least 98% with as little as 180 s. However, with less time-related data, the LSTM classifier kept that accuracy with only 30 s. The implemented models tried to be relatively small, and the final LSTM model has only ∼127,000 training parameters.

These results indicate that the proposed DL model could be easily implemented into an embedded system, which would be an important part of an accurate, portable, fast and low cost e-nose device for the detection of explosives. However, by enhancing the prototype characteristics in both hardware and software, better results could be achieved, which could lead to the implementation of a prototype with real application in security systems, such as automated airport scanning.

We believe that the time series nature of our dataset, which is the main target of recurrent networks such as LSTM, allows this model to get the best results when time becomes restricted, as in our experiments. That trait might become less relevant when more time-related data is available and so we have that CNN performs great under those circumstances. On favour of SVM we must say that it shows a better performance than the other two ML models, and so it becomes a worth implementing model in similar applications at least as a baseline

For future works, it would be interesting to test the LSTM classifier in a real time operation, and also calibrate the model with more samples and in higher concentrations. This implementation could also use optimized techniques for fast sampling, such as the ones mentioned in [21] and [133]. These future works should be designed in such a way that the whole experiment involves reduced data-collection times in order to discard any hardware limitations that might have been not perceptible here, and if they achieve good results, a final model capable of on-line learning should be deployed.

8 Optimizing a Multispectral-Images-Based DL Model, Through Feature Selection, Pruning and Quantization[7]

This chapter elaborates on the previous ones, continuing with the implementation of DL models for specific applications, and selecting the features that contribute the most to the predictions in order to create smaller and faster models. It also includes pruning and quantization, which are model optimization techniques that further improve the task of creating models for embedded applications. To build edge computing solutions, we need real-time, portable systems, which can be embedded in low-performance hardware. Thus, the main goal of this chapter is to demonstrate that after applying some optimization techniques, a DL model can be implemented on a Raspberry Pi 4 board, which could in turn be mounted on or near a data gathering devices (such as a drone), which could become a tool in the implementation of efficient food production strategies.

The inclusion of technology in agriculture is highly relevant given the increasing global demand for food and our growing population. This chapter focuses on an application for the analysis of multispectral images of wheat fields, and how they could be used to predict yield in a way that is resource-efficient and time convenient. Thus, our main goal is to optimize a deep learning model already proposed in the literature, through feature selection, pruning, and quantization, to be efficient enough that it could be deployed on a computer with limited resources. The main results of this work show that the size of the model was reduced by almost 94%, its inference time was almost 73% faster compared to the original model, while reducing its performance by 19% (still better than what was found in literature). This could be an important step towards the deployment of edge intelligence for plant phenotyping.

[7]The content of this chapter is currently accepted for publication in ISCAS 2022 [12]. The manuscript has been reformatted for inclusion in this thesis.
Julio Torres-Tello: Conceptualization, Methodology, Software, Validation, Data Curation, Writing - Original Draft, Visualization. Seok-Bum Ko: Resources, Writing - Review & Editing, Supervision, Project administration.

8.1 Introduction

With the increasing global demand for healthy and fresh foods [11], and the growing human population and urbanization [134], the inclusion of technology in agriculture is of utmost importance [13]. So far, we mainly rely upon experienced agronomists and farmers, whose decisions are based on empirical experience [134]. The use of new kinds of sensors, data, and tools could improve their decisions (e.g., when and where to use nutrients and water) and therefore the food production.

The main technologies used in this study are RS, DL, and edge computing. RS tries to alleviate issues related to conventional data collection techniques, which are usually costly, time-consuming, labour-intensive, and invasive [13, 134]. DL is widely employed in RS in fields like agriculture, especially due to the growing amount of drone and satellite images being collected [10, 13], reducing labour costs [22]. Edge computing refers to the analysis of data on low-power devices, which allows to reduce data transmission and ensure distributed intelligence [134].

UAVs are a common phenotyping platform in RS because they facilitate a fast, high-throughput, and high-quality image collection process of large fields at different crop stages [10], or as it is the case of this work, MSI, which provide additional information regarding the reflectance spectrum of an object [10]. Furthermore, UAVs or drones exemplify the notion of mobile edge computing [135], with low-performance devices that can provide real-time results.

One of the most common DL models is the CNN, which uses the convolution operation in place of general matrix multiplication in at least one layer, and is specialized in the processing of data that contains patterns, such as images, and it is usually implemented as a stacked architecture with some convolutional and pooling layers interchanged [11]. Although they have achieved great progress in terms of prediction capabilities, their disadvantage is their large number of parameters, slow inference speed, and large memory footprints, which make them difficult to be deployed on farms, in remote areas, or edge devices [136]. Thus, one aspect of current CNN design focuses on miniaturization and compression, intending to produce small networks that achieve the effect of large ones [39].

Edge computing refers to the deployment of computation closer to data sources (edge), such as mobile phones, wearable devices and IoT nodes. This approach is useful to alleviate issues with high costs, latency, energy consumption, privacy, and scalability [38, 137]. To build edge computing solutions, we need real-time, portable systems, which can be embedded in low-performance hardware [138], and to make these edge devices more intelligent, it is necessary to consider DL [38, 139], what is sometimes referred to as *edge intelligence* [14, 135, 137]. A common edge device [14, 22, 38, 39, 135, 137, 138, 139] that is used to implement DL solutions is the Raspberry Pi, which is the one we use in this study.

CNNs rely on a large number of multiplications, making their use expensive in terms of computational complexity and hardware [140], but there are some ways to optimize (reduce) the size of a DL model, such as feature selection, pruning and quantization. Feature selection is an important process in the analysis of high dimensional data [13], and it basically refers to the use of only some of the features in the data, which should speed up the training process and should generate DL models of smaller size, with minimum reduction in performance. In this chapter, we implement the results of [10] (Chapter 4), for feature selection, to test if the use of only the features identified in that paper are enough for yield prediction in wheat crops.

Feature selection is only the first step in our optimization workflow. The other two steps, commonly found in the literature, are pruning and quantization [22, 38, 39, 135, 136, 137, 138, 140, 141]. The aim of pruning is to delete redundant parameters [139] to find equivalent sparse models, either by compressing a pre-trained model or by inducing sparsity during training [135]. While pruning focuses on reducing the number of non-zero parameters, it can be used along with other techniques, such as quantization [141]. Parameter quantization is a conversion technique to reduce the size of a model with minimal degradation in its performance [38], reducing the number of bits that represent a number while maintaining the highest possible precision. A typical example is to transform a 32-bit floating point arithmetic number (float32) to an integer format, such as int32, but usually to an 8-bit number (int8) [136, 138]. In general, model compression offers the benefit of reducing the total number of energy-intensive memory accesses, at the time that improves the inference time through an effectively higher memory bandwidth for fetching model parameters [141].

Therefore, the main goal of this chapter is to optimize a DL model designed to predict wheat yield from MSI, through feature selection, pruning, and quantization, in order to be efficient enough that it could be deployed on a computer with low resources. This would be an important step towards the implementation of edge intelligence for plant phenotyping.

This chapter is organized as follows: Section 8.2 describes the dataset, the implementation, and the training process of the DL models, along with the optimization steps and implementation on the Raspberry Pi. Section 8.3 presents the results achieved by our models and comments on their meaning. Finally, Section 8.4 presents the conclusions of the chapter.

8.2 Material and Methods

This section describes the steps that we have followed in order to implement and test the proposed model. All the code that implements these steps is publicly available on GitHub[a].

[a]https://github.com/juliotorrest/MSI_DL_optimization

8.2.1 Description of the dataset and reference model

As it was previously mentioned, this study re-implements the solution presented in Chapter 4 [10], so we have used the same dataset and the VGG16-based model as our reference. Despite being an older CNN architecture, VGG is still used and implemented as part of many recent studies [10, 22, 136, 139, 140], mainly due to its straightforward implementation, which makes it simple to modify and analyze.

Regarding the dataset [10], it comprises a collection of 192 MSI, each containing information of 12 dates and 5 spectral bands, and their respective target value, which is the yield of the wheat crops. The MSI correspond to 4 fields that contain 48 images (plots) each, distributed into 3 replicates, as shown in Fig. 8.1.

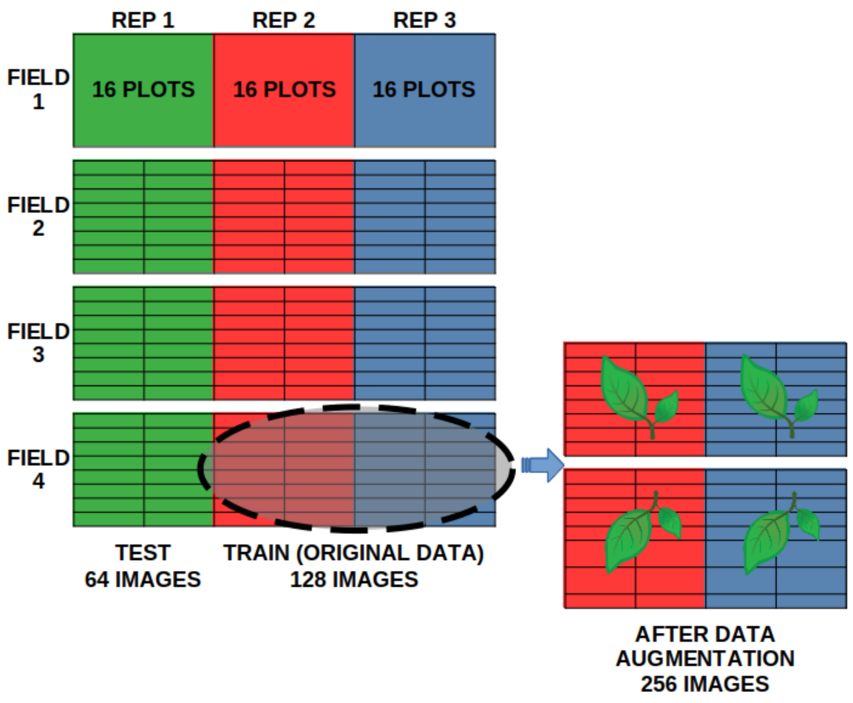


Figure 8.1: Distribution of the MSI samples in the dataset.

As for the DL model (Fig. 8.2), we took the model presented in Chapter 4 as our reference and re-implemented it using more filters in the CNN. The original model uses 5 in the first layer, and this study uses 64.

8.2.2 Training the DL model

The first step for training and optimizing our models was the train/test separation. Following what is done in Chapter 4, here we implemented a 3-fold cross-validation process, in which each fold uses 1 replicate for testing and the other 2 for training the model. This is clearly shown in Fig. 8.1, where we can also see that the replicates used for training first go through a data augmentation process in which all the images are rotated 180 degrees [10, 136]. In the end, the dataset had 256 samples for training (used for train and validation) and 64 for testing, per fold.

Once we had our dataset ready, we implemented the neural network shown in Fig. 8.2. This network, as well as the upcoming optimized versions, were all implemented using Tensorflow 2.4.1 (Tensorflow 1 is used in Chapter 4) on Python 3.7.7, and trained and tested on a Tesla V100 Nvidia GPU, with the only exception of the tests on the Raspberry Pi.

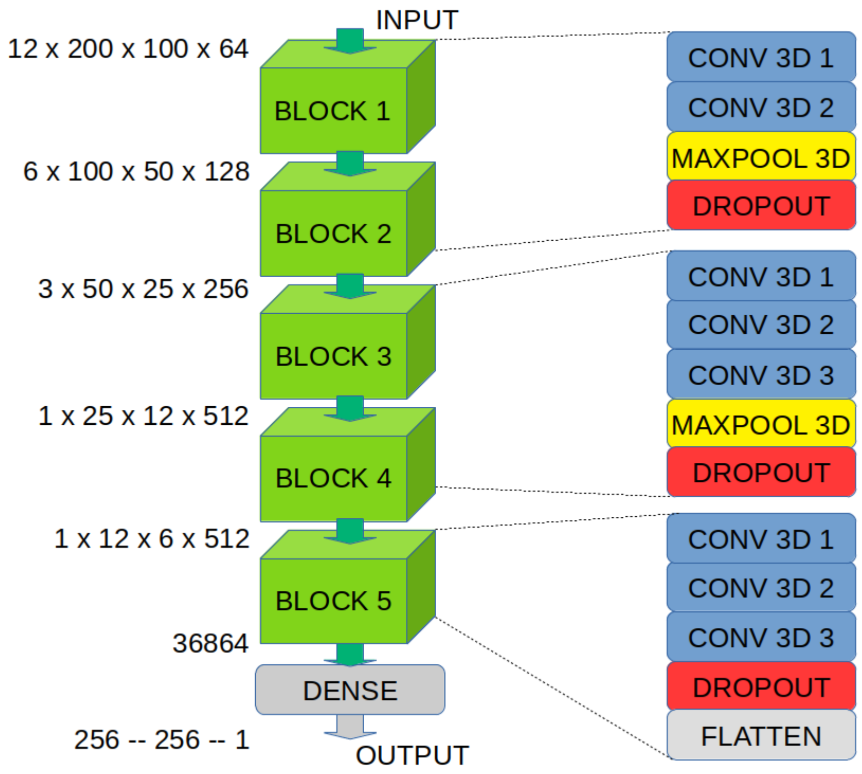


Figure 8.2: VGG-based architecture implemented as the initial model.

Since we have used the general architecture presented in [10], for this work, we only tuned the dropout hyperparameter. For this, we used a grid search process with 10, 25, and 33% as our options, using a 3-fold cross-validation scheme. Each fold was trained for 1000 epochs, using a batch size of 8, Adadelta as the optimizer, and MAPE as the loss function.

8.2.3 Feature Selection

One of the main results of Chapter 4 is the identification of salient features that are proposed as either the best dates to fly the drone that collects the MSI, or the most relevant spectral bands that might be used for data analysis. In this work, we propose the use of those features for the optimization of the DL model. Optimization, in this case means the implementation of a smaller DL model as a result of using fewer features in the dataset, while keeping an acceptable performance (at least similar to the reference model).

The aforementioned features correspond to dates 5, 7, 11, and 12, and red and red-edge bands. Therefore, we have extracted only those 4 and 2 features from the 12 and 5 originally included in the dataset. We tried to train the DL model shown in Fig. 8.2 with this new dataset, but due to the reduced dimensions we had to either pad the images with zeros, or as in the case of the implemented model, reduce the number of convolutional-pooling blocks. The DL model for reduced features is presented in Fig. 8.3, where we have discarded two blocks, which in itself represents less parameters in the network.

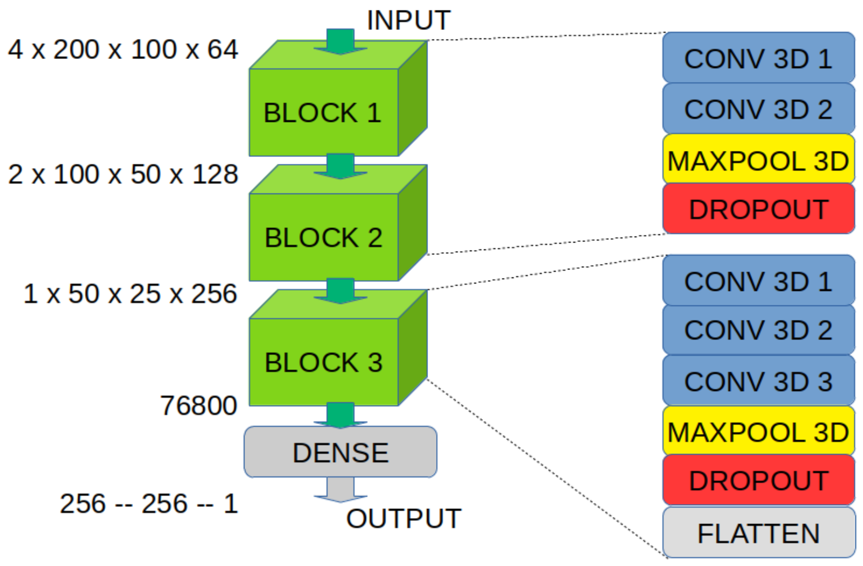


Figure 8.3: DL model that uses only the selected features.

As for the implementation and training process, we followed exactly the same steps and using the same resources that were described in the case of the model that uses all the features.

8.2.4 Pruning

The main idea behind pruning is that weights below a threshold are removed from the network [37], in which the sparsity is gradually increased for a number of pruning steps [141]. For this process, we used only one model and one replicate for testing. We have chosen the model and replicate that yielded the best metrics (replicate 2 as will be explained in Section 8.3). In this case, we defined the initial and final sparsities as 10 and 40%, respectively. Finally, the neural network was re-trained because the neural network obtained after pruning has changed [37, 136, 139].

8.2.5 Quantization

As the final optimization step, we needed to compress the model to reduce its size through quantization, which reduces the precision of the numbers and works well on pruned networks [37]. For this, we used the *TFLiteConverter* optimization tool included in Tensorflow. This generates a *.tflite* model of reduced size that could be implemented on embedded devices, for inference purposes [22, 38, 39]. The Raspberry Pi is a good candidate for this study due to its good community support and its ability to handle TensorFlow and TensorFlow Lite frameworks [38], which is useful for our experiments.

8.2.6 Implementation on a Raspberry Pi 4

The Raspberry Pi 4 Model B is the latest model of its range of computers. Its key features include a 64-bit quad-core ARMv8 processor, 8 GB of RAM, wired and wireless connectivity, implemented on an 85x56 mm board that requires a 5V/3A DC input source [142].

The main challenge here was running Tensorflow 2.4.1 on this computer, since only version 1 is officially supported. For this, first, we had to install a 64-bit Operating System (by default the Raspberry comes with a 32-bit OS), then Python 3 and a compatible C++ compiler. These tools were needed for the installation of Tensorflow from an appropriate wheel[b].

When the required software was working properly, we could load the models (original and optimized) and compare their performance on this device.

8.3 Results

The main results of this work are summarized in Table 8.1. In this table we have used the coefficient of determination (R^2) as the metric to evaluate the performance of our models, and to be able to compare to Chapter 4. To compare the reduction in size we have used the size of the model file and the number of trainable parameters. Finally, we have used the time it would take to generate the predictions over one test dataset (64 samples), in order to evaluate the latency that this step would add to any real-life solution that would use the DL model presented here.

For the R^2 values is important to mention that since we used cross-validation, the first row in Table 8.1 presents the average value for those 3 folds, while the second row presents only the performance of the best fold among the 3 (fold 2 in our case). We have chosen to do so, to compare the results of this work with Chapter 4, but also because we just needed one DL model for the optimization step. Arguably, we could have retrained another DL model with all the data for this second step (using the best hyperparameter), but that would have made difficult to compare our results with the reference work, Chapter 4 [10].

Table 8.2 reports the metrics obtained for each one of the 3 folds, using the original model (Fig. 8.2). From this table, we can see that when using replicate 2 for testing (1 and 3 for training) is

[b]https://github.com/Qengineering/TensorFlow-Raspberry-Pi_64-bit
[c]https://github.com/tensorflow/tensorflow/issues/46569

Table 8.1: Comparison of the metrics for the reference, trained, and optimized models.

Metric/ Model	**Reference work** [10]	**Using all features**	**After feature selection**	**After pruning and quantization**
R^2 *average*	0.676	0.922	0.789	NA
R^2 *best model*	NA	0.965	0.82	0.779
Model size [MB]	122.5	643.9	299.3	40.5
Parameters[M]	10	53	25	25[a]
Inference time [MM:SS]	NA	07:43.03	NA	02:05.56

a Approximate value according to Tensorflow documentation [c]

when we obtain the lowest MAE and MSE, as well as the highest R^2. Thus, we use this replicate as our reference for the optimization processes.

Table 8.2: Results per fold and average, obtained with the original model.

	Rep 1	**Rep 2**	**Rep 3**	**Average**
MAE	154.441	118.772	228.262	167.158
MSE	50116.111	26169.805	89120.239	55135.385
R^2	0.925	0.965	0.875	0.922

The same process was followed for the dataset with reduced features, and the resulting metrics are reported in Table 8.3. Once again, we see that the best results are obtained with replicate 2.

Table 8.3: Results per fold and average, obtained with the model that uses the selected features.

	Rep 1	**Rep 2**	**Rep 3**	**Average**
MAE	305.001	271.301	276.818	284.373
MSE	161998.089	135891.118	148181.174	148690.127
R^2	0.756	0.820	0.792	0.789

The model trained with reduced features was re-trained for pruning purposes. Finally, this model was compressed using quantization. We can now compare the performance of this model against the original and the one with reduced features. We can see that a 40% pruning reduced the performance of the model, but it is still in acceptable values. Table 8.1 shows that the R^2 metric went down from 0.965 to 0.779 (a decrease of 19%) when applying this optimization process, which is still better than the 0.676 presented in Chapter 4. Fig. 8.4 shows the final predictions of the optimized model over the rep 2 test set.

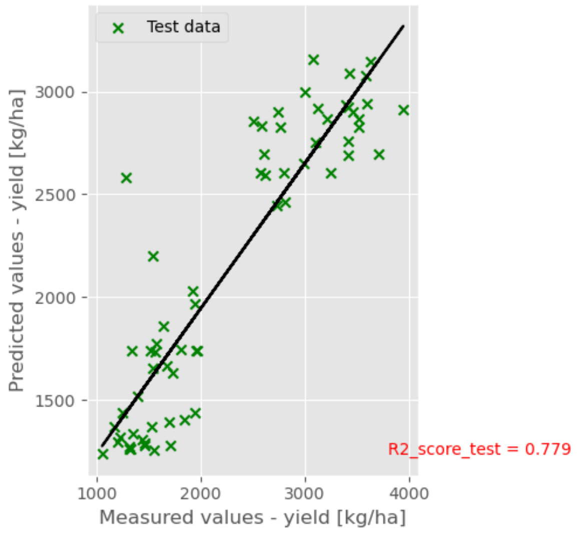


Figure 8.4: Results of the predictions made by the final model, for the test dataset.

On the other hand, we have that the size of the model went down from 643.9 to 40.5 MB, a decrease of 93.7%, and around 3 times smaller than the model in [10]. Also, the number of parameters was reduced by 52.8%, from 53 to 25 million.

Finally, when performing the inference tests on the Raspberry Pi, the time it took to go through all the samples in the replicate 2 test set (64 images) was 72.88% faster when using the optimized model, at an average of 1.96 seconds per MSI.

8.4 Conclusions

In this chapter, we have presented a complete optimization workflow for a DL model that predicts wheat yield for MSI. The main conclusion is that such a model could be implemented on a portable device to provide a drone with edge intelligence capabilities.

The results here obtained are aligned to the ones found in literature that show the important reduction in size and inference times of DL models when using pruning and quantization, while maintaining good performance [22, 38, 141].

On the feature selection side, here we have only used the results of Chapter 4, but there are other recently proposed techniques, such as SHAP (Chapters 5 and 6) that are worth exploring and that could lead to better outcomes. From the results of this work, we can see that this is the step in which most of the prediction capabilities were affected, with fewer gains in terms of optimization.

Another future work would be the implementation and comparison of these models on field-programmable gate arrays (FPGAs), and deploying them in real farm environments [136].

The results presented here show that it is possible to add intelligence to edge devices, which could become a tool in the implementation of efficient food production strategies towards the goal of feeding millions of people with limited resources.

Part V

Conclusion

9 Conclusions and Future Research

9.1 Summary and Conclusions

The main focus and motivation of this thesis have been the opportunities for inclusion of new technologies (such as AI) in agriculture, which still presents a lot of opportunities and needs for innovation. Therefore, this thesis studies a way to use different data sources mainly obtained from agricultural sites, to implement custom AI models able to predict some variables of interest. Then, it explores a number of mechanisms to optimize the AI models for deployment of intelligent algorithms closer to the edge, where data is being generated. Furthermore, this approach also contributes to the understanding of which features are more relevant for the predictions generated by the DL models.

Based on the concepts discussed throughout this thesis, some solutions are provided in order to first implement custom models for the particular tasks being addressed, and then for the optimization of some of those models. This optimization is mainly based on a feature selection approach, but finally complemented with pruning and quantization. The implementation of some of these models on an edge-like device, demonstrates the feasibility of this approach. Thus, the main contributions of this thesis are on the general workflow for custom model optimization, as well as in the proposed scheme for feature selection based on a model interpretability approach.

Table 9.1 presents a brief summary regarding the contribution of each chapter to the specific goals of this work. The main objective of proposing a scheme (Fig. 1.1) to develop, implement, and optimize custom AI models for their deployment on edge devices is fulfilled by all chapters in one way or another, so it has not been included on the table.

In order to construct the final scheme, this thesis introduces new concepts progressively in each chapter, in such a way that they elaborate onto the previous ones. Chapter 3 presents a yield prediction model for aeroponic crops in a controlled environment, based on the environmental variables that can be controlled and/or measured in the production system, that are used to train different ML/DL models, whose results are improved by means of an ensemble of the two best models (DNN and RF). Therefore, this is a first approach for using AI in the application of interest.

As a next step, Chapter 4 shows that a DL model is capable of finding useful features within a MSI dataset for yield prediction purposes, using the SBS approach. Also this analysis found the best timing for wheat yield prediction. This chapter proposes the utilization of optimal features as

Table 9.1: Contribution of each chapter to the specific objectives of this thesis.

PARTS AND CHAPTERS		SPECIFIC OBJECTIVES: Validate that AI generates models for studied applications.	SPECIFIC OBJECTIVES: Propose and implement tools for model optimization.	SPECIFIC OBJECTIVES: Demonstrate implementation on edge-like devices.
Part II Custom ML Models	**Ch. 3** Ensemble Learning in Aeroponics	✓		
Part III Feature Selection in DL	**Ch. 4** Identifying Useful Features in MSI	✓	✓	
	Ch. 5 A Novel Approach to Identify Spectral Bands	✓	✓	
Part IV Model Optimization & Edge Intelligence	**Ch. 6** Interpretability of AI Models in Aeroponics	✓	✓	
	Ch. 7 Improving Detection with LSTM	✓	✓	✓
	Ch. 8 Feature Selection, Pruning & Quantization		✓	✓

a methodology to improve yield prediction.

Elaborating on the previous results and ideas, Chapter 5 focuses on two key aspects: (i) finding an accurate DL model for the prediction of moisture content of canola and wheat crops, based on HSI; and (ii) rating the importance of spectral bands in those predictions, using a novel approach based on model explainability/interpretability analysis. As for the first goal, the final ensemble model (that as a novelty, uses a pointwise convolution to convert the 150 bands of HSI into 3, easier for the use of pre-trained models) obtained better results than what was found in literature, and reference methods such as PLSR and the NDVI vegetation index. Regarding the assessment of spectral bands, this chapter proposes for the first time, the use of a model interpretability approach that by explaining the predictions of an AI model, it can point out the spectral bands that contribute to generate predictions. So far, in literature, this approach has not been used for supervised band selection in a regression model that uses HSI and DL. Importantly, these results, when compared to PCA, validate the idea that using SHAP obtains the spectral bands that are important for this task. In both cases, the most relevant features are located on the NIR part of the spectrum, which is widely used in moisture measurement of agricultural products and vegetation analysis. Moreover, unlike other approaches such as PCA, in which the information is extracted with disregard of the application or specific need, the proposed approach generates explanations of the trained model, focused on a target, a feature, or a subset of the HSI dataset.

Building mainly on Chapters 3 and 5, Chapter 6 presents yield prediction models for aeroponics, with two main contributions: (i) the use of data fusion, which improves predictions, allowing the use of data gathered by the sensors for AI models, without the need for complex pre-processing; and (ii) the use of SHAP values to find which features are more relevant for yield prediction in aeroponics. For this, two versions of the dataset were implemented and tested, a purely tabular and a mixed dataset that includes the full sequential time-series data acquired by the sensors. A CNN-based model produced the best results. As a second step, SHAP analysis was applied to the trained models, which identified the salient features. SHAP values also provided important information for re-training the models with reduced features, producing similar or better performance metrics, which is evidence of their relevance. This implementation validates the idea of using SHAP for model optimization. In the interpretability side, SHAP values also give useful insights of how these features influence the predictions.

Finally arriving to the implementation of optimized models into edge-like devices, Chapter 7 presents a testing procedure to determine the accuracy of an e-nose consisting of an array of six chemical sensors, for which a number of ML/DL algorithms were implemented (kNN, SVM, RF, CNN and LSTM), trying to keep a relatively small number of parameters. The best results are obtained with an LSTM model, which is finally implemented and tested on a Raspberry Pi 3 board. Complementing this work, Chapter 8 presents a complete optimization workflow that includes feature selection, pruning, and quantization, for a DL model that predicts wheat yield for MSI. The final model is implemented on a portable device (Raspberry Pi 4), which could provide a drone or other mobile device with edge intelligence capabilities.

The results presented here show that it is possible to add intelligence to edge devices.

9.2 Limitations

It is important to acknowledge that this work has some limitations that set some boundaries to the results that have been achieved, and provided that they could be overcome, this might represent a variety of opportunities to propose and develop future research and applications.

The main limitations of this thesis, as it has been mentioned along the document, are in the data collection process. In general, data collected from agricultural settings tend to be slow to acquire and highly dependent on variables that cannot be controlled (such as the weather, or mobility restrictions). Thus, this work could be improved by the inclusion of data from other geographic regions as well as new seasons. Data generated by satellites is another opportunity for inclusion of new datasets, that has not been explored in this work. The use of satellite imagery would require a previous agreement with farmers that would be willing to share the yield information from their farms. However, such scheme for data collection could imply the continuous improvement of predictions, that could foster the adoption of AI solutions in farms.

9.3 Future Research

Although this thesis tries to be a self-contained work, encompassing all the aspects required to go from AI model design to implementation on an edge device, there are some aspects that could be further studied, analyzed, and improved. Furthermore, it is almost impossible to keep the pace with all the new developments in the fields of AI, IoT, edge computing, hardware and software tools, etc., which opens the field for new discussions and proposals. This work tries to fill some gaps and to propose some ideas that hopefully will be useful for future researchers in the development of new technologies and solutions. This subsection proposes some future works.

The work described in Chapter 3 draws the main path to be followed in order to have a yield prediction tool that would eventually require less human intervention. However, in order to have a robust model that can be deployed in a production stage, more data should be collected and organized in such a way that it does not present the issues addressed in this work. This would allow to generate more accurate and self-explanatory results, that could be easily implemented in a final control system. Importantly, the collection of more data is recommended for all the works described in this thesis, since it has been limited in all cases. Even though solutions like data augmentation and the use of pre-trained models have been used to diminish the effects of the limited number of samples, new strategies could be explored, not only collecting more data, but also possibly integrating other datasets.

Moving forward to the feature selection stages, the findings about some spectral bands as salient features could provide a direction to utilize and develop new vegetation indices for canola and wheat phenotyping. The research presented in Chapters 4 and 5 proved that using less spectral bands would also allow the use software tools designed to handle only three channels; for example, a pre-trained VGG network using green, red and red-edge. However, despite some initial proposals like the use of pointwise convolutions to adapt the number of channels, we believe that there is still room for improvement on those approaches which could be matter of a future work, and they could be very beneficial for many fields in which non-standard samples are being acquired. These works also give us guidance relative to data collection, and could be used for implementing more efficient data collection strategies and in the design of sensors tailored for a specific application that would acquire images with only the useful spectral bands.

Considering the model explainability/interpretability approach, for future works, it would be interesting to evaluate other recently proposed methods, especially considering the development of precision farming where sensor analytics and intelligent systems will be used to increase and improve production. Furthermore, since this thesis proposes the use of local explanation models, the evaluation of global models should be further studied as well.

Finally, regarding to the edge intelligence applications, the results of this thesis indicate that the proposed models can be easily implemented into embedded systems, which would be an important part of an accurate, portable, fast and low cost device for data collection. However, the characteristics of the implementation in both hardware and software can be improved, and thus better results could be achieved. This could lead to the implementation and real-life evaluation of prototypes with application in security systems, aeroponic farms, and HSI collection units.

In general, those future works should be designed in such a way that the whole experiment involves the collection of samples with only the desired characteristics, so that the whole process can be evaluated, in order to discard any hardware limitations that might have been not perceptible here, and if they achieve good results, a final model capable of on-line learning should be deployed.

References

[1] F. Chollet, *Deep Learning with Python*, 1st ed. Shelter Island, New York: Manning Publications, Dec. 2017.

[2] D. McCarthy, P. Rutten, and C. Kanaracus, "Increasing Intelligence at the Edge with AI," International Data Corporation, Whitepaper, Jan. 2022. [Online]. Available: https://resources.nvidia.com/en_us_fleet_command/en-us-fleet-command/increating-intelligence-edge-ai?lb-mode=preview

[3] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, 2nd ed. Beijing, China; Sebastopol, CA: O'Reilly Media, Oct. 2019.

[4] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," *arXiv:1603.07285 [cs, stat]*, Mar. 2016.

[5] J. Torres-Tello, S. Venkatachalam, L. Moreno, and S.-B. Ko, "Ensemble Learning for Improving Generalization in Aeroponics Yield Prediction," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, Spain, Oct. 2020, pp. 1–5.

[6] G. Wei, G. Li, J. Zhao, and A. He, "Development of a LeNet-5 Gas Identification CNN Structure for Electronic Noses," *Sensors*, vol. 19, no. 1, p. 217, Jan. 2019.

[7] S. Raschka, *Python Machine Learning, 1st Edition.* Birmingham Mumbai: Packt Publishing, Sep. 2015.

[8] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning.* Cambridge, Massachusetts: The MIT Press, Nov. 2016.

[9] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient Processing of Deep Neural Networks: A Tutorial and Survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, Dec. 2017.

[10] J. Torres-Tello and S.-B. Ko, "Identifying Useful Features in Multispectral Images with Deep Learning for Optimizing Wheat Yield Prediction," in *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2021, pp. 1–5.

[11] J. Torres-Tello and S.-B. Ko, "Interpretability of Artificial Intelligence models that use data fusion to predict yield in Aeroponics," *Journal of Ambient Intelligence and Humanized Computing*, Sep. 2021.

[12] J. Torres-Tello and S.-B. Ko, "Optimizing a Multispectral-Images-Based DL model, through feature selection, pruning and quantization," in *2022 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2022, In press.

[13] J. W. Torres-Tello, S. Ko, K. D. Singh, and S. J. Shirtliffe, "A novel approach to identify the spectral bands that predict moisture content in canola and wheat," *Biosystems Engineering*, vol. 210, pp. 91–103, Oct. 2021.

[14] J. Torres-Tello, A. V. Guamán, and S.-B. Ko, "Improving the detection of explosives in a MOX chemical sensors array with LSTM networks," *IEEE Sensors Journal*, vol. 20, no. 23, pp. 14 302–14 309, 2020.

[15] Z. Khan, V. Rahimi-Eichi, S. Haefele, T. Garnett, and S. J. Miklavcic, "Estimation of vegetation indices for high-throughput phenotyping of wheat using aerial imaging," *Plant Methods*, vol. 14, no. 1, p. 20, Mar. 2018.

[16] I. A. Lakhiar, G. Jianmin, T. N. Syed, F. A. Chandio, N. A. Buttar, and W. A. Qureshi, "Monitoring and Control Systems in Agriculture Using Intelligent Sensor Techniques: A Review of the Aeroponic System," 2018.

[17] Agriculture and Agri-Food Canada, "Canada: Outlook for Principal Field Crops, 2020-07-17," Jul. 2020. [Online]. Available: https://www.agr.gc.ca/eng/crops/reports-and-statistics-data-for-canadian-principal-field-crops/canada-outlook-for-principal-field-crops-2020-07-17/?id=1595253011859

[18] P. Zhu, Y. Zhang, Y. Chou, and Y. Gu, "Recognition of the storage life of mitten crab by a machine olfactory system with deep learning," *Journal of Food Process Engineering*, vol. 42, no. 7, p. e13095, 2019.

[19] G.-J. Jong, Hendrick, Z.-H. Wang, K.-S. Hsieh, and G.-J. Horng, "A Novel Feature Extraction Method an Electronic Nose for Aroma Classification," *IEEE Sensors Journal*, vol. 19, no. 22, pp. 10 796–10 803, Nov. 2019.

[20] Q. Li, Y. Gu, and N. Wang, "Application of Random Forest Classifier by Means of a QCM-Based E-Nose in the Identification of Chinese Liquor Flavors," *IEEE Sensors Journal*, vol. 17, no. 6, pp. 1788–1794, Mar. 2017.

[21] P. Qi, Q. Meng, Y. Jing, Y. Liu, and M. Zeng, "A Bio-Inspired Breathing Sampling Electronic Nose for Rapid Detection of Chinese Liquors," *IEEE Sensors Journal*, vol. 17, no. 15, pp. 4689–4698, Aug. 2017.

[22] X. Chen and J. Sheng, "The Design of Detector to Illegal Ingredients of Kitchen Waste Based on Embedded Device," *Journal of Physics: Conference Series*, vol. 1802, no. 3, p. 032009, Mar. 2021.

[23] N. Corporation, "Top Considerations for Deploying AI at the Edge," NVIDIA Corporation, USA, Technical Overview, Jun. 2021. [Online]. Available: https://resources.nvidia.com/en_us_fleet_command/en-us-fleet-command/considerations-for-deploying-ai-at-the-edge-technical-overview?lb-mode=preview

[24] R. Pallas, *Sensores y Acondicionadores de Señal*, 4th ed. Spain: Alfaomega - Marcombo, Feb. 2008.

[25] S. S. Virnodkar, V. K. Pachghare, V. C. Patil, and S. K. Jha, "Remote sensing and machine learning for crop water stress determination in various crops: A critical review," *Precision Agriculture*, vol. 21, no. 5, pp. 1121–1155, Oct. 2020.

[26] D. G. Manolakis, R. B. Lockwood, and T. W. Cooley, *Hyperspectral Imaging Remote Sensing: Physics, Sensors, and Algorithms.* Cambridge, United Kingdom: Cambridge University Press, Oct. 2016.

[27] Z. Zhang, M. Liu, X. Liu, and G. Zhou, "A New Vegetation Index Based on Multitemporal Sentinel-2 Images for Discriminating Heavy Metal Stress Levels in Rice," *Sensors (Basel, Switzerland)*, vol. 18, no. 7, Jul. 2018.

[28] W. Xie, J. Lei, J. Yang, Y. Li, Q. Du, and Z. Li, "Deep Latent Spectral Representation Learning-Based Hyperspectral Band Selection for Target Detection," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 58, no. 3, pp. 2015–2026, Mar. 2020.

[29] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[30] P. Peng, X. Zhao, X. Pan, and W. Ye, “Gas Classification Using Deep Convolutional Neural Networks,” *Sensors*, vol. 18, no. 1, p. 157, Jan. 2018.

[31] Z. Han, J. Zhao, H. Leung, K. F. Ma, and W. Wang, “A Review of Deep Learning Models for Time Series Prediction,” *IEEE Sensors Journal*, pp. 1–1, 2019.

[32] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, “LSTM: A Search Space Odyssey,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2222–2232, Oct. 2017.

[33] B. Mittelstadt, C. Russell, and S. Wachter, “Explaining Explanations in AI,” in *Proceedings of the Conference on Fairness, Accountability, and Transparency*, ser. FAT* ’19. New York, NY, USA: Association for Computing Machinery, Jan. 2019, pp. 279–288.

[34] L. S. Carlsson, P. B. Samuelsson, and P. G. Jönsson, “Interpretable Machine Learning—Tools to Interpret the Predictions of a Machine Learning Model Predicting the Electrical Energy Consumption of an Electric Arc Furnace,” *Steel Research International*, 2020.

[35] J. Padarian, A. B. McBratney, and B. Minasny, “Game theory interpretation of digital soil mapping convolutional neural networks,” *SOIL*, vol. 6, no. 2, pp. 389–397, Aug. 2020.

[36] S. M. Lundberg and S.-I. Lee, “A Unified Approach to Interpreting Model Predictions,” in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 4765–4774.

[37] S. Han, H. Mao, and W. J. Dally, “Deep Compression: Compressing Deep Neural Network with Pruning, Trained Quantization and Huffman Coding,” in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2016.

[38] I. N. K. Wardana, J. W. Gardner, and S. A. Fahmy, “Optimising Deep Learning at the Edge for Accurate Hourly Air Quality Prediction,” *Sensors*, vol. 21, no. 4, p. 1064, Jan. 2021.

[39] Q. Wang and Z. Wang, “Research on Deploying the Deeplearning Models with Embedded Devices,” in *2019 IEEE 9th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*, Jul. 2019, pp. 1337–1341.

[40] M. K. van Ittersum, K. G. Cassman, P. Grassini, J. Wolf, P. Tittonell, and Z. Hochman, “Yield gap analysis with local to global relevance—A review,” *Field Crops Research*, vol. 143, pp. 4–17, Mar. 2013.

[41] I. A. Lakhiar, J. Gao, T. N. Syed, F. A. Chandio, and N. A. Buttar, “Modern plant cultivation technologies in agriculture under controlled environment: A review on aeroponics,” *Journal of Plant Interactions*, vol. 13, no. 1, pp. 338–352, Jan. 2018.

[42] S. Chandra, S. Khan, B. Avula, H. Lata, M. H. Yang, M. A. Elsohly, and I. A. Khan, “Assessment of total phenolic and flavonoid content, antioxidant properties, and yield of aeroponically and conventionally grown leafy vegetables and fruit crops: A comparative study,” *Evidence-Based Complementary and Alternative Medicine: eCAM*, vol. 2014, 2014.

[43] National Aeronautics and Space Administration, “Innovative Partnerships Program Spinoff,” 2006. [Online]. Available: https://www.nasa.gov/pdf/164449main_spinoff_06.pdf

[44] A. X. Wang, C. Tran, N. Desai, D. B. Lobell, and S. Ermon, “Deep Transfer Learning for Crop Yield Prediction with Remote Sensing Data,” in *COMPASS ’18*, 2018.

[45] M. Guerif, M. Launay, and C. Duke, "Remote sensing as a tool enabling the spatial use of crop models for crop diagnosis and yield prediction," in *IGARSS 2000. IEEE 2000 International Geoscience and Remote Sensing Symposium. Taking the Pulse of the Planet: The Role of Remote Sensing in Managing the Environment. Proceedings (Cat. No.00CH37120)*, vol. 4, Jul. 2000, pp. 1477–1479 vol.4.

[46] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015.

[47] J. Schmidhuber, "Deep Learning in Neural Networks: An Overview," *Neural Networks*, vol. 61, pp. 85–117, Jan. 2015.

[48] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct. 2001.

[49] V. Vapnik, *The Nature of Statistical Learning Theory*, 2nd ed., ser. Information Science and Statistics. New York: Springer-Verlag, 2000.

[50] R. M. Bilder, F. W. Sabb, T. D. Cannon, E. D. London, J. D. Jentsch, D. S. Parker, R. A. Poldrack, C. Evans, and N. B. Freimer, "Phenomics: The systematic study of phenotypes on a genome-wide scale," *Neuroscience*, vol. 164, no. 1, pp. 30–42, Nov. 2009.

[51] O. A. Montesinos-López, A. Montesinos-López, J. Crossa, G. de los Campos, G. Alvarado, M. Suchismita, J. Rutkoski, L. González-Pérez, and J. Burgueño, "Predicting grain yield using canopy hyperspectral reflectance in wheat breeding data," *Plant Methods*, vol. 13, no. 1, p. 4, Jan. 2017.

[52] J. Delegido, J. Verrelst, C. M. Meza, J. P. Rivera, L. Alonso, and J. Moreno, "A red-edge spectral index for remote sensing estimation of green LAI over agroecosystems," *European Journal of Agronomy*, vol. 46, pp. 42–52, Apr. 2013.

[53] K. A. Al-Gaadi, A. A. Hassaballa, E. Tola, A. G. Kayad, R. Madugundu, B. Alblewi, and F. Assiri, "Prediction of Potato Crop Yield Using Precision Agriculture Techniques," *PLOS ONE*, vol. 11, no. 9, p. e0162219, Sep. 2016.

[54] C. Yang, J. H. Everitt, J. M. Bradford, and D. Murden, "Airborne Hyperspectral Imagery and Yield Monitor Data for Mapping Cotton Yield Variability," *Precision Agriculture*, vol. 5, no. 5, pp. 445–461, Oct. 2004.

[55] X. Zhou, H. B. Zheng, X. Q. Xu, J. Y. He, X. K. Ge, X. Yao, T. Cheng, Y. Zhu, W. X. Cao, and Y. C. Tian, "Predicting grain yield in rice using multi-temporal vegetation indices from UAV-based multispectral and digital imagery," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 130, pp. 246–255, Aug. 2017.

[56] S. Khaki and L. Wang, "Crop Yield Prediction Using Deep Neural Networks," *Frontiers in Plant Science*, vol. 10, 2019.

[57] H. Russello, "Convolutional neural networks for crop yield prediction using satellite images," Ph.D. dissertation, IBM Center for Advanced Studies, Jun. 2018.

[58] J. Torres-Tello and P. Fuenmayor-Viteri, "Infocentros, a key factor for the deployment of e-agriculture in Ecuador," in *2017 Fourth International Conference on eDemocracy eGovernment (ICEDEG)*, Apr. 2017, pp. 189–194.

[59] Y. Ampatzidis and V. Partel, "UAV-Based High Throughput Phenotyping in Citrus Utilizing Multispectral Imaging and Artificial Intelligence," *Remote Sensing*, vol. 11, no. 4, p. 410, Jan. 2019.

[60] P. Yu, M. Huang, M. Zhang, and B. Yang, "Optimal Wavelength Selection for Hyperspectral Imaging Evaluation on Vegetable Soybean Moisture Content during Drying," *Applied Sciences*, vol. 9, no. 2, p. 331, Jan. 2019.

[61] A. Benelli, C. Cevoli, L. Ragni, and A. Fabbri, "In-field and non-destructive monitoring of grapes maturity by hyperspectral imaging," *Biosystems Engineering*, vol. 207, pp. 59–67, Jul. 2021.

[62] H.-C. Wang, "General Deep Learning segmentation process used in Remote Sensing images," in *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLIII-B2-2020. Copernicus GmbH, Aug. 2020, pp. 1289–1296.

[63] T. Jiang and X. J. Wang, "Hyperspectral Images classification based on fusion features derived from 1D and 2D Convolutional Neural Network," in *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLII-3-W10. Copernicus GmbH, Feb. 2020, pp. 335–341.

[64] Jiang, Zhan, Wang, and Ko, "Retinal blood vessel segmentation using fully convolutional network with transfer learning." *Computerized Medical Imaging and Graphics : the Official Journal of the Computerized Medical Imaging Society*, vol. 68, pp. 1–15, Apr. 2018.

[65] K. J. Chae, G. Y. Jin, S. B. Ko, Y. Wang, H. Zhang, E. J. Choi, and H. Choi, "Deep Learning for the Classification of Small ($\leq$2 cm) Pulmonary Nodules on CT Imaging: A Preliminary Study," *Academic Radiology*, vol. 27, no. 4, pp. e55–e63, Apr. 2020.

[66] R. D. Castro-Zunti, J. Yépez, and S.-B. Ko, "License plate segmentation and recognition system using deep learning and OpenVINO," *IET Intelligent Transport Systems*, vol. 14, no. 2, pp. 119–126, Jan. 2020.

[67] E. Pan, X. Mei, Q. Wang, Y. Ma, and J. Ma, "Spectral-spatial classification for hyperspectral image based on a single GRU," *Neurocomputing*, vol. 387, pp. 150–160, Apr. 2020.

[68] B. Niu, J. Lan, Y. Shao, and H. Zhang, "A Dual-Branch Extraction and Classification Method Under Limited Samples of Hyperspectral Images Based on Deep Learning," *Remote Sensing*, vol. 12, no. 3, p. 536, Jan. 2020.

[69] J. Nalepa, M. Myller, and M. Kawulok, "Training- and Test-Time Data Augmentation for Hyperspectral Image Segmentation," *IEEE Geoscience and Remote Sensing Letters*, vol. 17, no. 2, pp. 292–296, Feb. 2020.

[70] K. Lee, H. J. Chang, J. Choi, B. Heo, A. Leonardis, and J. Y. Choi, "Motion-aware ensemble of three-mode trackers for unmanned aerial vehicles," *Machine Vision and Applications*, vol. 32, no. 3, p. 54, Mar. 2021.

[71] L. Wang, J. Yan, L. Mu, and L. Huang, "Knowledge discovery from remote sensing images: A review," *WIREs Data Mining and Knowledge Discovery*, vol. 10, no. 5, p. e1371, 2020.

[72] W.-H. Su and D.-W. Sun, "Potential of hyperspectral imaging for visual authentication of sliced organic potatoes from potato and sweet potato tubers and rapid grading of the tubers according to moisture proportion," *Computers and Electronics in Agriculture*, vol. 125, pp. 113–124, Jul. 2016.

[73] Y. Lu, W. Saeys, M. Kim, Y. Peng, and R. Lu, "Hyperspectral imaging technology for quality and safety evaluation of horticultural products: A review and celebration of the past 20-year progress," *Postharvest Biology and Technology*, vol. 170, p. 111318, Dec. 2020.

[74] B. Fang, Y. Bai, and Y. Li, "Combining Spectral Unmixing and 3D/2D Dense Networks with Early-Exiting Strategy for Hyperspectral Image Classification," *Remote Sensing*, vol. 12, no. 5, p. 779, Jan. 2020.

[75] Y. Ding, Y. Zhu, J. Feng, P. Zhang, and Z. Cheng, "Interpretable spatio-temporal attention LSTM model for flood forecasting," *Neurocomputing*, vol. 403, pp. 348–359, Aug. 2020.

[76] K. D. Singh, H. S. N. Duddu, S. Vail, I. Parkin, and S. J. Shirtliffe, "UAV-Based Hyperspectral Imaging Technique to Estimate Canola (Brassica napus L.) Seedpods Maturity," *Canadian Journal of Remote Sensing*, vol. 0, no. 0, pp. 1–15, Feb. 2021.

[77] N. Azmi, L. M. Kamarudin, A. Zakaria, D. L. Ndzi, M. H. F. Rahiman, S. M. M. S. Zakaria, and L. Mohamed, "RF-Based Moisture Content Determination in Rice Using Machine Learning Techniques," *Sensors*, vol. 21, no. 5, p. 1875, Jan. 2021.

[78] A. H. Elmetwalli and A. N. Tyler, "Estimation of maize properties and differentiating moisture and nitrogen deficiency stress via ground – Based remotely sensed data," *Agricultural Water Management*, vol. 242, p. 106413, Dec. 2020.

[79] Government of Saskatchewan, "Agriculture and Agri-Value — Key Economic Sectors." [Online]. Available: https://www.saskatchewan.ca/business/investment-and-economic-development/key-economic-sectors/agriculture-and-agri-value

[80] M. Imani and H. Ghassemian, "An overview on spectral and spatial information fusion for hyperspectral image classification: Current trends and challenges," *Information Fusion*, vol. 59, pp. 59–83, Jul. 2020.

[81] P. Wu, Z. Cui, Z. Gan, and F. Liu, "Three-Dimensional ResNeXt Network Using Feature Fusion and Label Smoothing for Hyperspectral Image Classification," *Sensors*, vol. 20, no. 6, p. 1652, Jan. 2020.

[82] Z. Xiong, Y. Yuan, and Q. Wang, "AI-NET: Attention Inception Neural Networks for Hyperspectral Image Classification," in *IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium*, Jul. 2018, pp. 2647–2650.

[83] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017, pp. 1800–1807.

[84] H. Abdi and L. J. Williams, "Partial Least Squares Methods: Partial Least Squares Correlation and Partial Least Square Regression," in *Computational Toxicology: Volume II*, ser. Methods in Molecular Biology, B. Reisfeld and A. N. Mayeno, Eds. Totowa, NJ: Humana Press, 2013, pp. 549–579.

[85] J. Lu, D. Cheng, C. Geng, Z. Zhang, Y. Xiang, and T. Hu, "Combining plant height, canopy coverage and vegetation index from UAV-based RGB images to estimate leaf nitrogen concentration of summer maize," *Biosystems Engineering*, vol. 202, pp. 42–54, Feb. 2021.

[86] J.-A. Jiang, M.-S. Liao, T.-S. Lin, C.-K. Huang, C.-Y. Chou, S.-H. Yeh, T.-T. Lin, and W. Fang, "Toward a higher yield: A wireless sensor network-based temperature monitoring and fan-circulating system for precision cultivation in plant factories," *Precision Agriculture*, vol. 19, no. 5, pp. 929–956, Oct. 2018.

[87] Z. Jiang, J. Yepez, S. An, and S. Ko, "Fast, accurate and robust retinal vessel segmentation system," *Biocybernetics and Biomedical Engineering*, vol. 37, no. 3, pp. 412–421, Jan. 2017.

[88] R. Castro-Zunti, K. J. Chae, Y. Choi, G. Y. Jin, and S.-b. Ko, "Assessing the speed-accuracy trade-offs of popular convolutional neural networks for single-crop rib fracture classification," *Computerized Medical Imaging and Graphics*, vol. 91, p. 101937, Jul. 2021.

[89] Y. Wang, E. J. Choi, Y. Choi, H. Zhang, G. Y. Jin, and S.-B. Ko, "Breast Cancer Classification in Automated Breast Ultrasound Using Multiview Convolutional Neural Network with Transfer Learning," *Ultrasound in Medicine and Biology*, vol. 46, no. 5, pp. 1119–1132, May 2020.

[90] Y. Wang, H. Zhang, K. J. Chae, Y. Choi, G. Y. Jin, and S.-B. Ko, "Novel convolutional neural network architecture for improved pulmonary nodule classification on computed tomography," *Multidimensional Systems and Signal Processing*, vol. 31, no. 3, pp. 1163–1183, Jul. 2020.

[91] F. Francis, P. L. Vishnu, M. Jha, and B. Rajaram, “IOT-Based Automated Aeroponics System,” in *Intelligent Embedded Systems*, ser. Lecture Notes in Electrical Engineering, D. Thalmann, N. Subhashini, K. Mohanaprasad, and M. S. B. Murugan, Eds. Singapore: Springer, 2018, pp. 337–345.

[92] B. D. Argo, Y. Hendrawan, and U. Ubaidillah, “A fuzzy micro-climate controller for small indoor aeroponics systems,” *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 17, no. 6, pp. 3019–3026, Dec. 2019.

[93] M. Mehra, S. Saxena, S. Sankaranarayanan, R. J. Tom, and M. Veeramanikandan, “IoT based hydroponics system using Deep Neural Networks,” *Computers and Electronics in Agriculture*, vol. 155, pp. 473–486, Dec. 2018.

[94] A. Goldstein, L. Fink, A. Meitin, S. Bohadana, O. Lutenberg, and G. Ravid, “Applying machine learning on sensor data for irrigation recommendations: Revealing the agronomist’s tacit knowledge,” *Precision Agriculture*, vol. 19, no. 3, pp. 421–444, Jun. 2018.

[95] S. M. Lundberg, B. Nair, M. S. Vavilala, M. Horibe, M. J. Eisses, T. Adams, D. E. Liston, D. K.-W. Low, S.-F. Newman, J. Kim, and S.-I. Lee, “Explainable machine-learning predictions for the prevention of hypoxaemia during surgery,” *Nature Biomedical Engineering*, vol. 2, no. 10, pp. 749–760, Oct. 2018.

[96] M. J. Ariza-Garzón, J. Arroyo, A. Caparrini, and M.-J. Segovia-Vargas, “Explainability of a Machine Learning Granting Scoring Model in Peer-to-Peer Lending,” *IEEE Access*, vol. 8, pp. 64 873–64 890, 2020.

[97] Z. Li, F. Nie, X. Chang, Y. Yang, C. Zhang, and N. Sebe, “Dynamic Affinity Graph Construction for Spectral Clustering Using Multiple Features,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 12, pp. 6323–6332, Dec. 2018.

[98] J. Torres-Tello, S. Venkatachalam, L. Moreno, and S.-B. Ko, “Yield measurements in aeroponics for six different crops,” *Mendeley Data*, no. V1, 2020.

[99] M. Luo, X. Chang, Z. Li, L. Nie, A. G. Hauptmann, and Q. Zheng, “Simple to complex cross-modal learning to rank,” *Computer Vision and Image Understanding*, vol. 163, pp. 67–77, Oct. 2017.

[100] H. C. Thompson, R. W. Langhans, A.-J. Both, and L. D. Albright, “Shoot and Root Temperature Effects on Lettuce Growth in a Floating Hydroponic System,” *Journal of the American Society for Horticultural Science*, vol. 123, no. 3, pp. 361–364, May 1998.

[101] L. H. Ziska, “Rising Atmospheric Carbon Dioxide and Plant Biology: The Overlooked Paradigm,” *DNA and Cell Biology*, vol. 27, no. 4, pp. 165–172, Mar. 2008.

[102] R. Guidotti, A. Monreale, F. Giannotti, D. Pedreschi, S. Ruggieri, and F. Turini, “Factual and Counterfactual Explanations for Black Box Decision Making,” *IEEE Intelligent Systems*, vol. 34, no. 6, pp. 14–23, Nov. 2019.

[103] M. Guillemé, V. Masson, L. Rozé, and A. Termier, “Agnostic Local Explanation for Time Series Classification,” in *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, Nov. 2019, pp. 432–439.

[104] Z. Li, F. Nie, X. Chang, L. Nie, H. Zhang, and Y. Yang, “Rank-Constrained Spectral Clustering With Flexible Embedding,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 12, pp. 6073–6082, Dec. 2018.

[105] S. Ivanov, K. Bhargava, and W. Donnelly, “Precision Farming: Sensor Analytics,” *IEEE Intelligent Systems*, vol. 30, no. 4, pp. 76–80, Jul. 2015.

[106] S. Liu, L. Guo, H. Webb, X. Ya, and X. Chang, "Internet of Things Monitoring System of Modern Eco-Agriculture Based on Cloud Computing," *IEEE Access*, vol. 7, pp. 37 050–37 058, 2019.

[107] INTERPOL, "Chemical and Explosives terrorism." [Online]. Available: https://www.interpol.int/en/Crimes/Terrorism/Chemical-and-Explosives-terrorism

[108] E. Stark, S. Hoover, A. DeCesare, and E. Barenholtz, "Medicine Has Gone to the Dogs: Deep Learning and Robotic Olfaction to Mimic Working Dogs," *IEEE Technology and Society Magazine*, vol. 37, no. 4, pp. 55–60, Dec. 2018.

[109] R. G. Ewing, D. A. Atkinson, G. A. Eiceman, and G. J. Ewing, "A critical review of ion mobility spectrometry for the detection of explosives and explosive related compounds," *Talanta*, vol. 54, no. 3, pp. 515–529, May 2001.

[110] M. Tabrizchi and V. Ilbeigi, "Detection of explosives by positive corona discharge ion mobility spectrometry," *Journal of Hazardous Materials*, vol. 176, no. 1-3, pp. 692–696, Apr. 2010.

[111] A. Zalewska, W. Pawłowski, and W. Tomaszewski, "Limits of detection of explosives as determined with IMS and field asymmetric IMS vapour detectors," *Forensic Science International*, vol. 226, no. 1-3, pp. 168–172, Mar. 2013.

[112] V. V. Krylov, "Odor Space Navigation Using Multisensory E-Nose," *Automation and Remote Control*, vol. 79, no. 1, pp. 167–179, Jan. 2018.

[113] W. Zhao, Q.-H. Meng, M. Zeng, and P.-F. Qi, "Stacked Sparse Auto-Encoders (SSAE) Based Electronic Nose for Chinese Liquors Classification," *Sensors*, vol. 17, no. 12, p. 2855, Dec. 2017.

[114] B. Szulczyński, K. Armiński, J. Namieśnik, and J. Gebicki, "Determination of Odour Interactions in Gaseous Mixtures Using Electronic Nose Methods with Artificial Neural Networks," *Sensors*, vol. 18, no. 2, p. 519, Feb. 2018.

[115] D. Wu, D. Luo, K.-Y. Wong, and K. Hung, "POP-CNN: Predicting Odor Pleasantness With Convolutional Neural Network," *IEEE Sensors Journal*, vol. 19, no. 23, pp. 11 337–11 345, Dec. 2019.

[116] B. Firtat, C. Moldovan, C. Brasoveanu, G. Muscalu, M. Gartner, M. Zaharescu, P. Chesler, C. Hornoiu, S. Mihaiu, C. Vladut, I. Dascalu, V. Georgescu, and I. Stan, "Miniaturised MOX based sensors for pollutant and explosive gases detection," *Sensors and Actuators B: Chemical*, vol. 249, pp. 647–655, Oct. 2017.

[117] L. A. Horsfall, D. C. Pugh, C. S. Blackman, and I. P. Parkin, "An array of WO3 and CTO heterojunction semiconducting metal oxide gas sensors used as a tool for explosive detection," *Journal of Materials Chemistry A*, vol. 5, no. 5, pp. 2172–2179, Jan. 2017.

[118] K. Konstantynovski, G. Njio, F. Börner, A. Lepcha, T. Fischer, G. Holl, and S. Mathur, "Bulk detection of explosives and development of customized metal oxide semiconductor gas sensors for the identification of energetic materials," *Sensors and Actuators B: Chemical*, vol. 258, pp. 1252–1266, Apr. 2018.

[119] W. J. Peveler, R. Binions, S. M. V. Hailes, and I. P. Parkin, "Detection of explosive markers using zeolite modified gas sensors," *Journal of Materials Chemistry A*, vol. 1, no. 7, pp. 2613–2620, Jan. 2013.

[120] K. Brudzewski, S. Osowski, and W. Pawlowski, "Metal oxide sensor arrays for detection of explosives at sub-parts-per million concentration levels by the differential electronic nose," *Sensors and Actuators B: Chemical*, vol. 161, no. 1, pp. 528–533, Jan. 2012.

[121] R. Triviño, D. Gaibor, J. Mediavilla, and A. V. Guamán, “Challenges to embed an electronic nose on a mobile robot,” in *2016 IEEE ANDESCON*, Oct. 2016, pp. 1–4.

[122] M. M. Rahman, C. Charoenlarpnopparut, and P. Suksompong, “Classification and pattern recognition algorithms applied to E-Nose,” in *2015 2nd International Conference on Electrical Information and Communication Technologies (EICT)*, Dec. 2015, pp. 44–48.

[123] Y. Cheng, K. Wong, K. Hung, W. Li, Z. Li, and J. Zhang, “Deep Nearest Class Mean Model for Incremental Odor Classification,” *IEEE Transactions on Instrumentation and Measurement*, vol. 68, no. 4, pp. 952–962, Apr. 2019.

[124] Q. Wang, T. Xie, and S. Wang, “Research on air Pollution Gases Recognition Method Based on LSTM Recurrent Neural Network and Gas Sensors Array,” in *2018 Chinese Automation Congress (CAC)*, Nov. 2018, pp. 3486–3491.

[125] Z. Wen, W. Ye, X. Zhao, and X. Pan, “A Novel 1D Deep Convolutional Neural Network Based Algorithm for Mixture Gases Recognition,” in *2018 IEEE 61st International Midwest Symposium on Circuits and Systems (MWSCAS)*, Aug. 2018, pp. 893–896.

[126] Q. Liu, X. Hu, M. Ye, X. Cheng, and F. Li, “Gas Recognition under Sensor Drift by Using Deep Learning,” *International Journal of Intelligent Systems*, vol. 30, no. 8, pp. 907–922, 2015.

[127] H. Zhang, W. Ye, X. Zhao, R. K. F. Teng, and X. Pan, “A Novel Convolutional Recurrent Neural Network Based Algorithm for Fast Gas Recognition in Electronic Nose System,” in *2018 IEEE International Conference on Electron Devices and Solid State Circuits (EDSSC)*, Jun. 2018, pp. 1–2.

[128] P. Qi, Q. Meng, and M. Zeng, “A CNN-based simplified data processing method for electronic noses,” in *2017 ISOCS/IEEE International Symposium on Olfaction and Electronic Nose (ISOEN)*, May 2017, pp. 1–3.

[129] P. López, R. Triviño, D. Calderón, A. Arcentales, and A. V. Guamán, “Electronic nose prototype for explosive detection,” in *2017 CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON)*, Oct. 2017, pp. 1–4.

[130] A. V. Guaman, P. Lopez, and J. Torres-Tello, “Multivariate Discrimination Model for TNT and Gunpowder Using an Electronic Nose Prototype: A Proof of Concept,” in *Information Technology and Systems*, ser. Advances in Intelligent Systems and Computing, Á. Rocha, C. Ferrás, and M. Paredes, Eds. Springer International Publishing, 2019, pp. 284–293.

[131] N. Ratchapakorn and Y. Ariyakul, “Development of a Low-cost Explosive Vapor Detector Using Metal Oxide Gas Sensors,” in *2017 21st International Computer Science and Engineering Conference (ICSEC)*, Nov. 2017, pp. 1–5.

[132] G. Wei, J. Zhao, Z. Yu, Y. Feng, G. Li, and X. Sun, “An Effective Gas Sensor Array Optimization Method Based on Random Forest*,” in *2018 IEEE SENSORS*, Oct. 2018, pp. 1–4.

[133] X. Yin, L. Zhang, F. Tian, and D. Zhang, “Temperature Modulated Gas Sensing E-Nose System for Low-Cost and Fast Detection,” *IEEE Sensors Journal*, vol. 16, no. 2, pp. 464–474, Jan. 2016.

[134] S. Nesteruk, D. Shadrin, V. Kovalenko, A. Rodríguez-Sánchez, and A. Somov, “Plant Growth Prediction through Intelligent Embedded Sensing,” in *2020 IEEE 29th International Symposium on Industrial Electronics (ISIE)*, Jun. 2020, pp. 411–416.

[135] P. Prakash, C. Murti, S. Nath, and C. Bhattacharyya, “Optimizing DNN Architectures for High Speed Autonomous Navigation in GPS Denied Environments on Edge Devices,” in *PRICAI 2019: Trends in Artificial Intelligence*, ser. Lecture Notes in Computer Science, A. C. Nayak and A. Sharma, Eds. Cham: Springer International Publishing, 2019, pp. 468–481.

[136] R. Wang, W. Zhang, J. Ding, M. Xia, M. Wang, Y. Rao, and Z. Jiang, “Deep Neural Network Compression for Plant Disease Recognition,” *Symmetry*, vol. 13, no. 10, p. 1769, Oct. 2021.

[137] R. Mohandas, M. Bhattacharya, M. Penica, K. Camp, and M. Hayes, “TensorFlow Enabled Deep Learning Model Optimization for enhanced Realtime Person Detection using Raspberry Pi operating at the Edge,” in *AICS*, 2020.

[138] S. Mikhaylevskiy, V. Chernyavskiy, V. Pavlishen, I. Romanova, and R. Solovyev, “Fast Emotion Recognition Neural Network for IoT Devices,” in *2021 International Seminar on Electron Devices Design and Production (SED)*, Apr. 2021, pp. 1–6.

[139] Y. Chen, C. Li, L. Gong, X. Wen, Y. Zhang, and W. Shi, “A deep neural network compression algorithm based on knowledge transfer for edge devices,” *Computer Communications*, vol. 163, pp. 186–194, Nov. 2020.

[140] J. Yepez and S.-B. Ko, “Stride 2 1-D, 2-D, and 3-D Winograd for Convolutional Neural Networks,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 4, pp. 853–863, Apr. 2020.

[141] M. Zhu and S. Gupta, “To prune, or not to prune: exploring the efficacy of pruning for model compression,” in *6th International Conference on Learning Representations, ICLR*, Canada, Apr. 2018.

[142] Raspberry Pi Foundation, “Raspberry Pi Documentation.” [Online]. Available: https://www.raspberrypi.com/documentation/