

Fall 11-2021

Transfer-Learned Pruned Deep Convolutional Neural Networks for Efficient Plant Classification in Resource-Constrained Environments

Martinson Ofori

Follow this and additional works at: <https://scholar.dsu.edu/theses>



Part of the [Artificial Intelligence and Robotics Commons](#), [Databases and Information Systems Commons](#), [Data Science Commons](#), [Other Computer Sciences Commons](#), and the [Theory and Algorithms Commons](#)



**TRANSFER-LEARNED PRUNED DEEP
CONVOLUTIONAL NEURAL NETWORKS FOR
EFFICIENT PLANT CLASSIFICATION IN
RESOURCE-CONSTRAINED ENVIRONMENTS**

A dissertation submitted to Dakota State University in partial fulfillment of the requirements
for the degree of

Doctor of Science

in
Information Systems

November 2021

By
Martinson Quarshie Ofori

Dissertation Committee:

Omar El-Gayar, Ph.D. (Chair)

Cherie Noteboom, Ph.D.

Austin O'Brien, Ph.D.



DISSERTATION APPROVAL FORM

This dissertation is approved as a credible and independent investigation by a candidate for the Doctor of Science in Information Systems degree and is acceptable for meeting the dissertation requirements for this degree. Acceptance of this dissertation does not imply that the conclusions reached by the candidate are necessarily the conclusions of the major department or university.

Student Name: Martinson Q. Ofori

Dissertation Title: TRANSFER-LEARNED PRUNED DEEP CONVOLUTIONAL NEURAL NETWORKS FOR EFFICIENT PLANT CLASSIFICATION IN RESOURCE-CONSTRAINED ENVIRONMENTS

Dissertation Chair/Co-Chair: Omar El-Gayar

Date: November 18, 2021

Committee member: Cherie Bakker Nettekoom

Date: November 20, 2021

Committee member: Arshad Ali

Date: November 20, 2021

ACKNOWLEDGMENT

A big thank you goes to my chair, Dr. Omar El-Gayar, who dedicated his time and knowledge to guide and supervise me throughout this doctoral study. I would also like to thank my committee members – Dr. Cherie Noteboom and Dr. Austin O’Brien. Their questions suggestions, and comments improved this dissertation. I acknowledge Dr. Dave Bishop for his support and good advice throughout my stay at DSU. Thank you all for making this journey possible.

ABSTRACT

Traditional means of on-farm weed control mostly rely on manual labor. This process is time-consuming, costly, and contributes to major yield losses. Further, the conventional application of chemical weed control can be economically and environmentally inefficient. Site-specific weed management (SSWM) counteracts this by reducing the amount of chemical application with localized spraying of weed species.

To solve this using computer vision, precision agriculture researchers have used remote sensing weed maps, but this has been largely ineffective for early season weed control due to problems such as solar reflectance and cloud cover in satellite imagery. With the current advances in artificial intelligence, past research on weed detection in SSWM has used a large deep convolutional neural network (DCNN) for weed detection. These models are, however, computationally expensive and prone to overfitting on smaller datasets. Consequently, although DCNNs have shown continuous accuracy improvements in research settings, they remain relatively unused for practical purposes in precision agriculture due to their large number of parameters and the difficulty to implement on resource-constrained devices.

Accordingly, this research investigated the use of model compression to reduce complexity and increase the efficiency of DCNNs in low-resource conditions. The proposed approach involves stacking two pre-trained DCNN models – Xception and DenseNet – to reduce the effect of performance degradation during the model compression process. A performance evaluation of the resulting XD-Ensemble indicated that the model outperformed both state-of-the-art DCNNs and a lightweight EfficientNet-B1 model in a resource-constrained environment in terms of prediction accuracy, model size, and inference speed. The current study

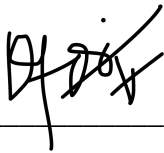
contributes to enhancing viability while minimizing the environmental footprint of agricultural technologies as well as maximizing their production efficiency.

DECLARATION

I hereby certify that this dissertation constitutes my own product, that where the language of others is set forth, quotation marks so indicate, and that appropriate credit is given where I have used the language, ideas, expressions, or writings of another.

I declare that the dissertation describes original work that has not previously been presented for the award of any other degree of any institution.

Signed,

A handwritten signature in black ink, appearing to read 'M. Ofori', is written above a horizontal line.

Martinson Q. Ofori

TABLE OF CONTENTS

DISSERTATION APPROVAL FORM	II
ACKNOWLEDGMENT	III
ABSTRACT	IV
DECLARATION	VI
TABLE OF CONTENTS	VII
LIST OF TABLES.....	XI
LIST OF FIGURES.....	XII
CHAPTER 1.....	1
INTRODUCTION	1
1.1 BACKGROUND OF THE PROBLEM	1
1.2 STATEMENT OF THE PROBLEM	4
1.3 OBJECTIVES OF THE PROJECT.....	5
1.4 OUTLINE OF THE DISSERTATION.....	6
CHAPTER 2.....	8
LITERATURE REVIEW	8
2.1 PRECISION AGRICULTURE.....	8
2.2 ARTIFICIAL INTELLIGENCE	10
2.2.1 <i>Machine Learning, Neural Networks, and Deep Learning</i>	11
2.2.2 <i>Convolutional Neural Networks</i>	13
2.2.3 <i>State-of-the-Art Model Architectures</i>	14
2.2.4 <i>Transfer Learning</i>	16
2.2.5 <i>Ensemble Learning</i>	18
2.2.6 <i>Model Compression</i>	19

2.3 ML FOR SSWM.....	20
2.3.1 <i>Computer Vision and ML for SSWM</i>	21
2.3.2 <i>Deep Learning for SSWM</i>	21
2.4 RESEARCH GAP.....	26
CHAPTER 3.....	28
RESEARCH METHODOLOGY.....	28
3.1 RATIONAL FOR THE DESIGN SCIENCE RESEARCH METHODOLOGY.....	28
3.2 PROPOSED ARTIFACT.....	29
3.2.1 <i>S1: Pre-Training using Transfer Learning</i>	30
3.2.2 <i>S2: Model Ensemble</i>	30
3.2.3 <i>S3: Model Compression</i>	33
3.3. PROPOSED EXPERIMENTS AND MODEL EVALUATION.....	33
3.3.1 <i>Dataset</i>	33
3.3.2 <i>Data Preparation</i>	36
3.3.3 <i>Training Conditions and Computational Resources</i>	37
3.3.4 <i>Evaluation</i>	39
CHAPTER 4.....	41
EVALUATION AND ANALYSIS.....	41
4.1. MODEL ACCURACY MEASURES.....	41
4.1.1 <i>Accuracy on the Plant Seedling Dataset</i>	41
4.1.2 <i>Accuracy on the Leafsnap Dataset</i>	43
4.1.3 <i>Accuracy on the Open Plant Phenotyping Dataset</i>	45
4.2. MODEL SIZE MEASUREMENTS.....	47
4.2.1 <i>Model Sizes for Plant Seedling Classification</i>	47
4.2.2 <i>Model Sizes for Leafsnap Classification</i>	48
4.2.3 <i>Model Sizes for Open Plant Phenotype Classification</i>	49
4.3. MODEL RUNTIME EVALUATION.....	49

4.3.1 <i>Experimental Runtime - Plant Seedling Classification</i>	49
4.3.2 <i>Experimental Runtime - Leafsnap Classification</i>	50
4.3.3 <i>Experimental Runtime - OPPD Classification</i>	51
4.4. ADDITIONAL EVALUATION	53
CHAPTER 5	57
DISCUSSION	57
5.1 RESULT SUMMARY AND COMPARISON WITH PAST RESEARCH	57
5.2 IMPLICATIONS FOR AGRICULTURAL PRACTICE	59
5.3 ADDITIONAL IMPLICATIONS AND HYPOTHETICAL USE-CASES FOR THE FUTURE	60
5.3.1 <i>Pre-Production: Seed sorting</i>	60
5.3.2 <i>Production: Diseases and pest identification</i>	61
5.3.3 <i>Post-Production: Harvesting and Distribution</i>	61
5.3.4 <i>Applications beyond farming</i>	62
CHAPTER 6	63
CONCLUSION	63
6.1 GENERAL SUMMARY	63
6.2 PRINCIPAL FINDINGS.....	64
6.3 STUDY LIMITATIONS AND RECOMMENDATIONS FOR FUTURE RESEARCHERS	65
REFERENCES	66
APPENDIX A: KOTLIN FOR ANDROID IMPLEMENTATION OF RUNTIME TESTS	87
TENSORFLOW LITE ANDROID APP - CLASSIFIER CLASS ADAPTED FOR DISSERTATION	87
UNIT TEST FOR LOGGING RUNTIME.....	88
APPENDIX B: PRECISION, RECALL, AND F₁-SCORE PER CLASS	89
PLANT SEEDLING DATASET	89
LEAFSNAP DATASET.....	89

OPEN PLANT PHENOTYPING DATASET.....93

APPENDIX C: PUBLICATIONS FROM THIS DISSERTATION95

LIST OF TABLES

Table 1. Examples of state-of-the-art DCNNs and their performance on the ImageNet dataset ^a	6
Table 2. Precision agriculture definitions over the years	9
Table 3. Computer vision and machine learning techniques for weed detection.....	21
Table 4. Recent applications of deep learning for weed detection	24
Table 5. Training conditions per dataset	37
Table 6. Experimental result – Plant Seedling dataset.....	42
Table 7. Experimental result – leafsnap dataset.....	44
Table 8. Experimental result – OPPD dataset.....	46
Table 9. Model size measurements ^a	48
Table 10. Runtime experiments – Plant Seedling dataset	50
Table 11. Runtime experiments – Leafsnap dataset.....	51
Table 12. Runtime experiments – OPPD dataset.....	52
Table 13. Precision, recall, and f_1 metrics for XD-Ensemble	55

LIST OF FIGURES

Figure 1. The flow of the dissertation	7
Figure 2. Relationship between artificial intelligence (AI) and its subfields. Source: Cai et al. (2020)	11
Figure 3. Structure of a simple neural network. Source: DeepAI.com	12
Figure 4. Comparing (a) traditional machine learning and (b) deep learning (J. Wang et al., 2018).....	13
Figure 5. Structure of the AlexNet CNN. Source: (X. Han et al., 2017)	14
Figure 6. The stacking approach used to ensemble the DCNN models	18
Figure 7. The three-stage model compression pipeline: pruning, quantization, and Huffman coding. Source: (S. Han et al., 2016).....	19
Figure 8. The proposed approach for a robust DCNN artifact.....	29
Figure 9. Snapshot of the stacked XD-Ensemble consisting of an Xception (left-base) and DenseNet121 (right-base) learners.....	32
Figure 10. Samples from the Plant Seedling Dataset	34
Figure 11. Samples from the Leafsnap Dataset	35
Figure 12. Samples from the OPPD Dataset.....	36
Figure 13. Training and testing steps	38
Figure 14. XD-Ensemble Validation Accuracy and Loss on Plant Seedling Dataset.	43
Figure 15. XD-Ensemble Validation Accuracy and Loss on Leafsnap Dataset	45
Figure 16. XD-Ensemble Validation Accuracy and Loss on OPPD Dataset.....	47
Figure 17. Comparing model accuracy vs compressed size (TFLite) per dataset.....	54

Figure 18. Grad-CAM visualization XD-Ensemble indicating model localization of stem and leaves as important regions for prediction 56

CHAPTER 1

INTRODUCTION

1.1 Background of the Problem

Early season plant growth is essential to agronomic production (López-Granados, 2011). The first few weeks of cropping are the most important time to eliminate competition between food crops and weeds for water and nutrients. Research shows that effective weed control at this stage is essential to increased yield in some crops (Patel et al., 2018). Traditionally, weed control has been done using manual labor, a process that has proven to be time-consuming, costly, and a major contributor to yield losses (Gianessi, 2009).

The preferred method for managing weeds in the last 50 years has been chemical weed killers, such as herbicides and pesticides, as they can kill up to 99% of targeted weeds (Bastiaans et al., 2008; H. Wang et al., 2018). However, the continuous employment of chemical weed killers undermines the United Nations' Sustainable Development Goals (SDG). Specifically, Goal 12 targets sound management of chemical release to the environment and harnessing technology for more sustainable production (UN, 2015). Often, the drawback and most criticized aspect of chemical weed control as a current cropping practice is their apparent harmfulness to the environment. Chemicals like alachlor, ametryn, and atrazine used in commercial pesticide products have been found near and in water resources adjacent to cropping areas due to their persistence and low biodegradability (Furtado et al., 2019; Moura et al., 2018; Tian et al., 1999). As a result, there are strong reasons to demand safer cropping systems with minimal environmental consequences.

In the last few decades, gains made to reduce the environmental effect of cropping systems have resulted in a substantial acceptance of agricultural information technology (AIT) – sometimes called smart farm technology (SFT) or variable rate technology (VRT) – as part of everyday agricultural practices (Balafoutis et al., 2017; Kernecker et al., 2020; Y. Wang et al., 2019; Wolfert et al., 2014a). These technologies have resulted in an agricultural paradigm known as Precision Agriculture (PA). Since its introduction in the 1980s, PA – defined as a practice that manages the spatial and temporal variability associated with agricultural soil, crops, and livestock for improved performance and sustainability with the aid of technology and green information systems (Balafoutis et al., 2017; Dedrick, 2010; Kernecker et al., 2020; Y. Wang et al., 2019; Wolfert et al., 2014b) – has made significant progress towards improving the sustainability of agriculture (Robert, 2002). Using methods such as site-specific weed management (SSWM), the practice can reduce the environmental impact of weed management through precise weed treatments that follow a four-step cyclical process consisting of 1) weed monitoring or detection, 2) management planning for action on weeding, 3) execution of the weed control method and 4) evaluation of performance (López-Granados, 2011). Therefore, as the foundation and precondition for SSWM, the importance of weed detection to the practice cannot be understated.

Commercial producers and researchers of SSWM equipment have sought methods to minimize the environmental effects of herbicide applications by prescribing weed killers according to their density and species. SSWM relies on several sensing technologies for weed detection. The most effective ones can be grouped into two main categories: *aerial remote sensing* and *ground-based* methods (López-Granados, 2011; Wang et al., 2019). While the aerial methods are effective for map-based SSWM in large areas (Wang et al., 2019), they suffer from several drawbacks such as their inability to detect small variations in reflectivity of seedlings, the need for higher resolution

images when weeds are distributed in small patches, the interference in detection caused by the reflectivity of soil background, and the fact that they are largely non-real-time (López-Granados, 2011; Thorp & Tian, 2004; Wang et al., 2019).

In this sense, proximal sensing has been posited as the best method for site-specific weed control especially ones that rely on ground-based computer vision (Gerhards, 2010). Even then, the problem for such systems is that at the early stages of plant growth, crops and weeds are almost indistinguishable in traditional pixel-based classification systems, and as such, rule-based methods that rely on edge detection for leaf shape and texture recognition are used (Golzarian & Frick, 2011). Feature extraction in such systems is also not robust enough to be generalized to different farming scenarios.

The recent resurgence of artificial intelligence (AI) and machine learning (ML) has resulted in phenomenal results in various problem domains. An ML technique known as *deep* convolutional neural networks (DCNN) has been successful because they learn to distinguish complex inherent patterns within images often difficult to observe otherwise (Simonyan & Zisserman, 2014). The success of the AlexNet in the ImageNet Large Scale Visual Recognition Challenge 2012 – achieving a top-5 test error rate of 15.3% as compared to 26.2% achieved by the second-best entry (Krizhevsky et al., 2012) –has resulted in a substantial increase in the body of research that employs DCNNs across several disciplines and industries. For SSWM, past research has successfully employed DCNNs to distinguish various crops in different growth stages using different DCNN models and methods (Ashqar et al., 2019; Dyrmann et al., 2016; Milioto et al., 2017; Pantazi et al., 2017; Sørensen et al., 2017; Xinshao & Cheng, 2015).

1.2 Statement of the Problem

Although the acceptance of technology in farming has been promising, PA suffers from a slow adoption rate (Lowenberg-DeBoer & Erickson, 2019). Failure to adopt agricultural technology has been attributed to concerns about complexity and high investment costs (Kerneck et al., 2020). Especially for most rural dwellers and small-scale farmers, the cost of buying and servicing both hardware and software can be a significant challenge that hampers adoption (Misaki et al., 2018; Saidu et al., 2017). Sustainable technology adoption must not affect farm profitability and efficiency (Tey & Brindal, 2012; Wolfert et al., 2017). As such, there is a persistent need to pursue definitive ways to maintain or lower costs associated with improving current systems with new technology.

While DCNNs are a good solution to increasing the accuracy of SSWM equipment, they fall short in terms of sustainability. They are complex and have high computational and energy demands. Increasingly powerful hardware systems are being developed to aid DCNN implementation but contribute to the cost of their commercial acceptance and use. The type of weed control systems used by these practices are often resource-constrained (Steward et al., 2019). Hence, the cost of replacing smaller edge, mobile, and resource-constrained devices with powerful hardware could be a barrier to their adoption and has profound implications for practice (Santos et al., 2020; A. Wang et al., 2019).

Consequently, a new set of mobile-sized models have been developed specifically for resource-limited systems and inference on the edge. Lightweight models such as the MobileNet, ShuffleNet, ANTNets, and some versions of the EfficientNets were developed specifically to address the issue of power efficiency and portability of DCNNs for embedded platforms (Khan et al., 2019). However, as Table 1 indicates, a comparison of these lightweight models with state-of-

the-art large DCNN models shows a lower performance as compared to the larger models; with very few exceptions including the much older VGGs 16 and 19 DCNNs. For example, the EfficientNets study which investigated how scaling network depth, width, and resolution affects model performance demonstrated that scaling some or all of these parameters resulted in exponentially better performance by compromising the model size and number of parameters (Tan & Le, 2019). According to Tan & Le (2019), the Efficientnets B0 and B1 at 29MB and 31MB achieve 77.3% and 79.2% Top-1 classification accuracy (CA) on the ImageNet dataset: a result that is close or better than some versions of the ResNet and DenseNet state-of-the-art models. However, that result also pales in comparison to the 84.3% and 84.4% Top-1 CA of the EfficientNet-B6 (166MB) and the EfficientNet-B7 (256MB) respectively.

An approach known as model compression can reduce the complexity and size of a DCNN to just a fraction of the original model. Techniques such as weight pruning in model compression can effectively scale down the resource requirements of DCNN models. The caveat is, in some instances, there is a performance degradation in the resulting less complex model (S. Han et al., 2016; Lammie et al., 2019; McCool et al., 2017).

1.3 Objectives of the Project

This study proposes a method of increasing the resource efficiency of DCNN models for ground-based plant classification systems. Using three different publicly available plant datasets in various phenological growth stages, the specific objectives of the current study are to:

- 1) Explore the potential of state-of-the-art DCNN architectures and *transfer learning* for weed detection.
- 2) Propose stacking two pre-trained DCNN models to reduce the effect of performance degradation in model compression.

- 3) Investigate the use of model compression to iteratively prune insignificant model weights and reduce computational cost and increase efficiency in low-resource conditions.
- 4) Evaluate how the proposed method compares to state-of-the-art DCNNs in a resource-constrained environment.

Table 1. Examples of state-of-the-art DCNNs and their performance on the ImageNet dataset^a

Model	Top-1 Accuracy	Size (MB)	Time (ms) per inference step (CPU)	Time (ms) per inference step (GPU)
NASNetLarge	0.825	343	344.51	19.96
Xception	0.79	88	109.42	8.06
ResNet152V2	0.78	232	107.5	6.64
InceptionV3	0.779	92	42.25	6.86
DenseNet201	0.773	80	127.24	6.67
ResNet152	0.766	232	127.43	6.54
ResNet101	0.764	171	89.59	5.19
DenseNet169	0.762	57	96.4	6.28
ResNet50V2	0.76	98	45.63	4.42
DenseNet121	0.75	33	77.14	5.38
ResNet50	0.749	98	58.2	4.55
NASNetMobile	0.744	23	27.04	6.7
VGG19	0.713	549	84.75	4.38
VGG16	0.713	528	69.5	4.16
MobileNetV2	0.713	14	25.9	3.83
MobileNet	0.704	16	22.6	3.44
EfficientNetB7	-	256	1578.9	61.62
EfficientNetB6	-	166	958.12	40.45
EfficientNetB5	-	118	579.18	25.29
EfficientNetB4	-	75	308.33	15.12
EfficientNetB3	-	48	139.97	8.77
EfficientNetB2	-	36	80.79	6.5
EfficientNetB1	-	31	60.2	5.55
EfficientNetB0	-	29	46	4.91

^a Source: <https://keras.io/api/applications/>; Excludes performance accuracy of EfficientNets which have been evaluated under different conditions in (Tan & Le, 2019);

1.4 Outline of the dissertation

This dissertation is organized as follows. Chapter 1 presents a general introduction, states the specific problem, introduces the objective, and presents the research approach employed in this

dissertation. Chapter 2 discusses the background of the study and provides a comprehensive survey of pertinent literature. Chapter 3 presents the design science methodology, presents the proposed approach, and the expected evaluation criteria.

The results and discussion are presented in Chapters 4 and 5, respectively. Chapter 6 concludes the dissertation. In these chapters, the dissertation will emphasize the contribution of the findings, implications for the use of information systems in fostering sustainable development and will provide suggestions for future research. Figure 1 below demonstrates the flow of the research. Solid arrows depict the normal flow and dotted red arrows show an alternative flow for readers familiar with the foundational concepts used in this study.

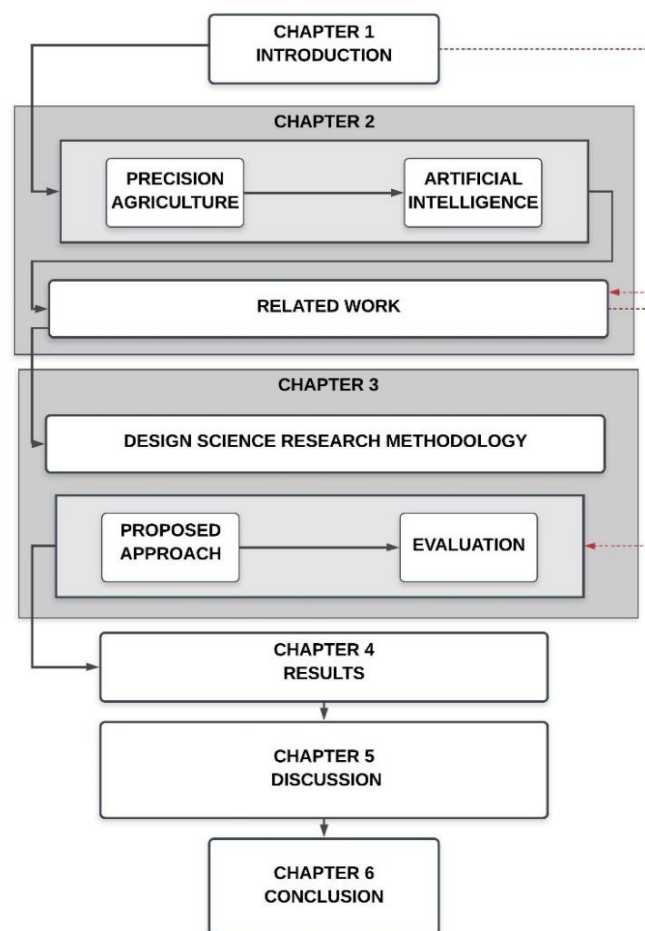


Figure 1. The flow of the dissertation

CHAPTER 2

LITERATURE REVIEW

This chapter presents foundational information for the approach used in this dissertation, as well as a summary and understanding of relevant literature. The chapter starts with an introduction of the PA management practices; establishes the basic concepts governing artificial intelligence; and concludes with the related works that employ DCNN for PA.

2.1 Precision Agriculture

Agriculture has grown from the labor-intensive manual practice of the past to a technology-friendly one that employs satellite technology, drones, robotics, big data, artificial intelligence, and other modern technology to provide selective rather than homogenous treatment for farm management (Aubert et al., 2012). A definition for PA, as used in this dissertation has been established in the prior section, but it should be noted that PA has been defined in several ways since its inception. Some of these definitions have been summarized in Table 2 below.

The practice has been established in the literature as inextricably linked to sustainability. Bongiovanni and Lowenberg-DeBoer (2004) in their review discuss this relationship at length and show that spatial management employed in PA can reduce environmental impact on sensitive areas while still maintaining profitability. In past research, this idea of profitability has been touted as a major pull factor for farmers to adopt PA (Wolfert et al., 2017). Even though some AITs in PA have been adopted just as fast as any other technology in history and provide the best sustainability gains, their use has rarely exceeded 20% of farms (Lowenberg-DeBoer & Erickson, 2019). The criticism is that farmers may be convinced of the idea of AITs but not their value. However, PA,

associated technologies, and their role in sustainability and climate change remain a recurrent theme in news and social media (Lakshmi & Corbett, 2020), and could be key to meeting future food demands and ensuring sustainable agriculture (Clercq et al., 2018; Walter et al., 2017; Wolfert et al., 2014a). Therefore, it is important to continuously improve these technologies and enhance their value.

Table 2. Precision agriculture definitions over the years

Reference	Definition
(Pierce et al., 1994)	Involves the variable management of soils and crops according to localized conditions within a field.
(Robert et al., 1995)	Information and technology-based agricultural management system to identify, analyze, and manage site-soil spatial and temporal variability within fields for optimum profitability, sustainability, and protection of the environment.
(Stafford, 1996)	Targeting of inputs to arable crop production according to crop requirements on a localized basis.
(Lowenberg-DeBoer & Swinton, 1997)	Information technology applied to agriculture.
(Olson, 1998)	The application of a holistic management strategy that uses information technology to bring data from multiple sources to bear on decisions associated with agricultural production, marketing, finance, and personnel.
(Pierce & Nowak, 1999)	The application of technologies and principles to manage spatial and temporal variability associated with all aspects of agricultural production for the purpose of improving crop performance and environmental quality.

(Robert, 2000)	It is rather an information revolution, made possible by new technologies that result in a higher level, a more precise farm management system.
(Plant, 2001)	Is the management of agricultural crops at a spatial scale smaller than that of the whole field.

2.2 Artificial Intelligence

The concept and real-world application of artificial intelligence have existed since circa 1950. The concern for AI has been developing systems that act as rational agents taking the best possible action in any given situation (Russell & Norvig, 2010). Cockburn et al. (2018) groups AI applications into three interrelated separate streams: *symbolic systems* such as natural language processing and image recognition: have been successfully applied at various levels but also been criticized for their inability to be scaled towards commercial solutions; *robotics*: used heavily for industrial automation; and the *learning approach*: that given some inputs can predict the presence of particular physical or logical events. These streams are not so far from the sub-specialties proposed by Michael Mills of Neota Logic (Martin, 2016). Similar to the streams proposed by Cockburn et al. (2018), AI is divided into robotics, cognition tools such as Natural Language Processing, and Vision systems used to capture and synthesize text, voice, and images. ML is a subset of AI consisting of several different techniques such as neural networks (NN) and deep learning (DL) used to find previously unknown relationships in data (Cai et al., 2020). Figure 2 shows the relations between AI, ML, NN, and DL.

By employing ML techniques, AI provides the needed analytic capabilities that support Data-Driven Decisions (DDD) (Bengio, 2009, 2013; Najafabadi et al., 2015). As the name implies, a data-driven decision is one made by using up-to-date data to analyze the facts in data and draw conclusions.

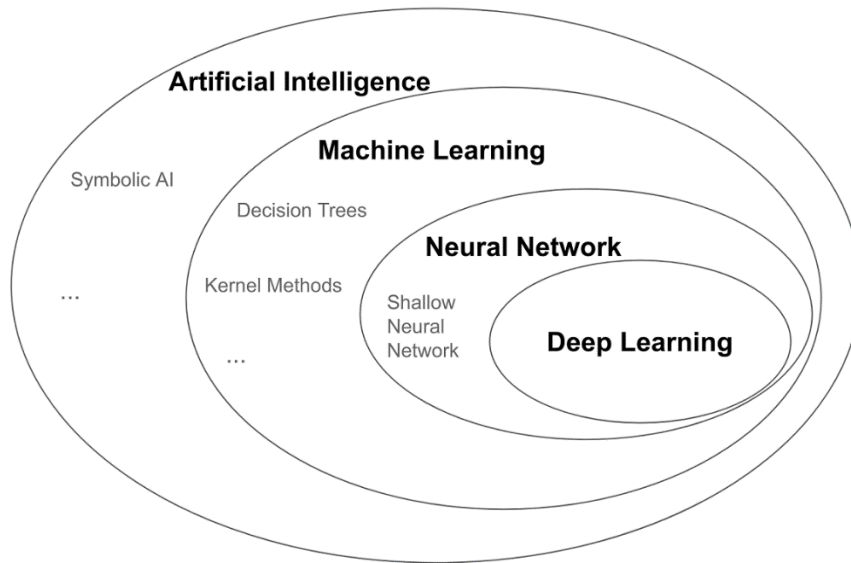


Figure 2. Relationship between artificial intelligence (AI) and its subfields. Source: Cai et al. (2020)

2.2.1 Machine Learning, Neural Networks, and Deep Learning

Machine Learning (ML) methods are a class of AI algorithms that allow computers to learn without being explicitly programmed. Several ML algorithms have been developed in the past. They can be grouped by their learning styles into supervised, unsupervised, semi-supervised, or reinforcement learning algorithms. They could also be categorized based on their similarity to each other in task performance such as instance-based (e.g. k-Nearest and Support Vector Machines), regression (e.g. Linear and Logistic regressions), decision trees (e.g. Classification and Regression Tree and Random Forests), clustering (such as k-Means and Expectation Maximization), Neural Networks (e.g. Multilayer Perceptron and Back-Propagation) or Deep Learning (e.g. Recurrent Neural Networks and Convolutional Neural Network) (Alpaydin, 2004; Bishop, 2006; Liakos et al., 2018; Mohri et al., 2018).

The outcome of ML is often to assist and/or emulate human decision-making. Earlier ML research for computer vision tasks centered around the development of improved features to be

fed into ML models. For instance, one of the most common approaches for object detection was to craft features using edge or gradient orientation approaches such as the Histogram of Oriented Gradients (HOG) (Dalal & Triggs, 2005). The resulting feature vector is then fed into a Support Vector Machine (SVM) algorithm to enable accurate classification.

Neural Networks (NN) refer to the ML algorithms modeled after the human brain to recognize patterns in data. Figure 3 shows a simple structure of a fully connected neural network. Neural networks learn by processing labeled input data supplied during training into feature vectors needed to return the correct output. When supplied with enough labeled training data, a NN model can process new and unseen inputs and successfully predict accurate results. This process is facilitated by hidden layers – a set of mathematical functions designed to produce an output specific to an intended result –located between the input and output of the algorithm which applies weights to the inputs and directs them through an activation function as an output. There are several types of NNs: feed-forward neural networks, recurrent neural networks, convolutional neural networks (CNN), etc. This study focuses on the application of CNNs which is explained in the ensuing section.

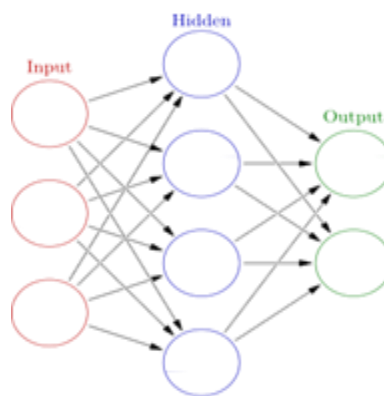


Figure 3. Structure of a simple neural network. Source: DeepAI.com

Deep learning (DL) refers to *deep* NN algorithms designed to exploit unknown structures that exist in data. As demonstrated in Figure 5, they are unlike the traditional ML that requires hand-crafted features based on human analysis to be fed into a generic ML algorithm classification. DL algorithms are designed to automatically learn multiple levels of representation (Bengio, 2012). In this way, DL algorithms uncover representations at multiple levels, with higher-level learned features expressed in terms of lower-level features (Bengio, 2013). That is to say, DL learns by abstracting features across hidden layers starting from low-level features (such as corners and edges) to high-level features (such as textures and shapes) allowing the mapping of input unto some output classification. While experts have not agreed on where shallow learning ends and deep learning begins (Schmidhuber, 2015), it is generally accepted that an ML model such as a CNN is considered *deep* if there is at least one hidden layer between the input and output layer.

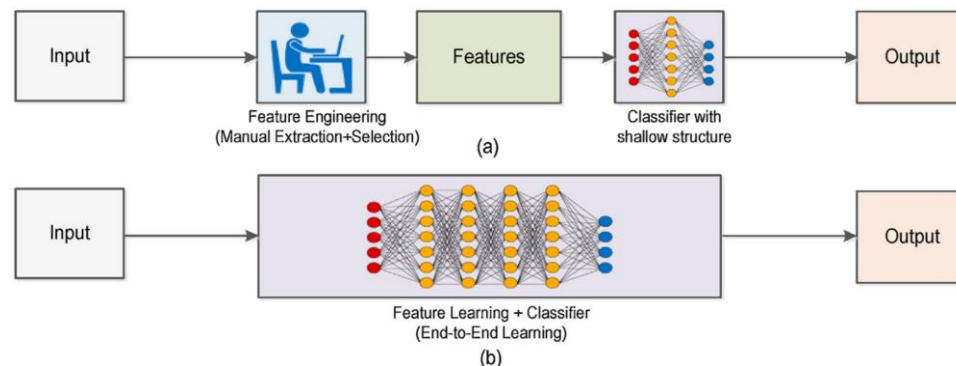


Figure 4. Comparing (a) traditional machine learning and (b) deep learning (J. Wang et al., 2018)

2.2.2 Convolutional Neural Networks

A CNN refers to a type of feed-forward back-propagation neural network aimed at processing data in the form of multiple arrays (LeCun et al., 2015). Data that CNNs process exceptionally well includes audio (1D), image (2D), and video (3D) data types. Images, for example, are mostly composed of three 2D arrays that represent the pixel intensities of the color

channels present in an RGB color image. Therefore, CNNs learn from image data by expanding them into arrays represented by the pixel intensities (0 to 255) of each point in the image, and through the process of deconvolution, separates the image into thousands of relevant features which are selected and aggregated into recognizable patterns viable for distinguishing new and unseen images. A structure of a simple CNN is depicted in Figure 5 using the AlexNet CNN architecture.

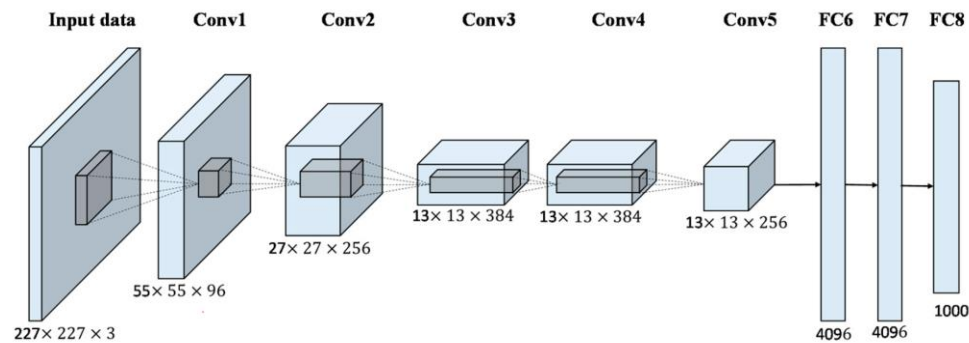


Figure 5. Structure of the AlexNet CNN. Source: (X. Han et al., 2017)

2.2.3 State-of-the-Art Model Architectures

As indicated earlier in Table 1, several DCNN models have been developed in the literature. Khan et al. (2019) summarized some of these DCNNs according to their architectural properties:

Spatial Exploitation Based – These kinds of networks take advantage of spatial filters to improve the performance of the network. The **VGG** is a popular DCNN network that replaced previous large filters with a smaller set of 3×3 filters and pushing depth to 16 and 19 layers will be used (Simonyan & Zisserman, 2014). The VGG won second place in the ImageNet Challenge 2014 classification track.

Depth Based – The basic assumption for these networks is that the deeper the network, the better it performs as it improves feature representations. **Inception** is a model introduced to

increase the depth and width of a network while ensuring computational cost remains low (Szegedy, Vanhoucke, et al., 2015; Szegedy, Wei Liu, et al., 2015). The InceptionV3 was the 1st runner-up of the 2015 ImageNet image classification challenge.

Width Based Multi-Connection – Instead of the traditional focus on the depth of a network, these models increase the width to improve learning. The **Xception** network introduced by Chollet (2017) does this by using depth-wise separable convolutions. It is an extreme version of the Inception network that maps the spatial correlations for each output channel separately, and then performs a pointwise convolution (1x1) to capture cross-channel correlation. The Xception is known to perform better than the Inception on ImageNet.

Depth and Multi-Path Based – The **ResNet** which won the ImageNet 2015 challenge in image classification, detection, and localization, as well as Winner of MS COCO 2015 detection, and segmentation uses both depth and multiple connections (He et al., 2015). It is a very deep network that learns the residual representation functions instead of learning the signal representations directly.

Multi-Path Based – To reduce the problem of performance degradation, gradient vanishing, or explosion problems, these networks connect one layer to another by skipping some intermediate layers while still allowing the flow of information across the layers through multiple paths or shortcuts connections. The **DenseNet** connects each layer to every other layer in a feed-forward fashion such that feature maps of all preceding layers are used as input to subsequent ones (Huang et al., 2018).

Lightweight - The *EfficientNet* DCNN developed by Tan & Le (2019) is a group of DCNN models created through neural architecture search to achieve an optimal network in all dimensions (width, depth, and resolution) that are more efficient. These, comparatively, lightweight models

have achieved better accuracy than past models. The main building block of EfficientNets is the mobile inverted bottleneck MBConvs introduced in MobileNetV2 (Sandler et al., 2019). MBConv blocks are made up of layers that expand then compress channels so that fewer channels are skip-connected. They employ depthwise and pointwise separable convolutions to reduce the number of trainable parameters by up to a factor of k^2 . The authors also employ squeeze-and-excitation (SE) optimization to improve performance by giving weight to channels instead of treating them equally. SE blocks return an output of $1 \times 1 \times \text{channel}$. Further, instead of the normal ReLU, the authors employ a swish activation to avoid information loss.

2.2.4 Transfer Learning

ML models, and in extension DCNNs, are often trained and tested with data taken from a single domain where the feature space and probability distribution are the same (or at least similar) (Pan & Yang, 2010). As such, and in the case of image datasets, ML models achieve the best result when both the training and the test images consist of the same number of classes, captured under similar conditions using a similar setup. However, ML models like DCNNs require many samples of training data to perform well on a classification task. Unfortunately, high-quality labeled data containing several samples of plant images are generally unavailable to researchers.

In such cases, a DL technique known as Transfer Learning (TL) can be useful. TL allows the use of representations learned from previous data to solve problems in new datasets (Pan & Yang, 2010; Yosinski et al., 2014). TL is valuable when data unavailability is a problem (such as the case with plant datasets) as it allows the domain, tasks, and distributions used in training to be different from those used in testing (Pan & Yang, 2010; Yosinski et al., 2014). TL is motivated by the fact that humans apply previously learned knowledge to solve new problems faster (Pan &

Yang, 2010). Hence, in humans and, in this context, machines, learning to identify an apple could make the task of recognizing oranges easier.

However, for TL to be successful, three main assumptions must be made 1) similar tasks/domain, 2) similar data distributions, and 3) a suitable model. When these conditions are not sufficiently satisfied, TL could degrade the performance of a DL model resulting in a scenario known as *negative transfer learning* (Pan & Yang, 2010; Rosenstein et al., 2005; Z. Wang et al., 2019; Zhang et al., 2021). A recent survey by Zhang et al. (2021) demonstrates negative TL as an active research area where various theoretical and statistical approaches have been proposed especially with regards to secure transfer and domain similarity estimation approaches (for example feature statistics, test performance, and fine-tuning).

As mentioned in Zhang et al. (2021), the fine-tuning approach employed in (Agrawal et al., 2014; Donahue et al., 2013) has proved beneficial when adapting a deep learning model from a source domain (such as the ImageNet dataset) to a different target domain (in this case plant datasets). Fine-tuning takes some weights from a DCNN model trained with a bigger base dataset and repurposes (or transfers) the learned features to a new model with a smaller target dataset. Specifically, the approach trains a base model using the base dataset, freezes the first *several* layers of the base model (consisting of generic features), and then re-trains the remaining layers with randomly initialized weights using the target dataset (to acquire the target-specific features) (Yosinski et al., 2014). Intuitively, this works because ML models have *generic* features near the input while the *domain-specific* features lie much deeper in the model (Yosinski et al., 2014). This approach is also effective for combating overfitting (where a large DCNN model learns all the nuances in the training data and generalizes poorly to new and unseen data).

High-level ML frameworks such as Keras make available pre-trained models from successful DCNNs such as the VGG, ResNet, Inception, and DenseNet models (Chollet & others, 2015). The gold standard is to pre-train these models on massive general-purpose image datasets such as ImageNet (<http://www.image-net.org/>).

2.2.5 Ensemble Learning

The principle behind using an ensemble strategy is to allow the models to generalize better using a combination of individual predictions from two models or more models (Rokach, 2019). It allows the final model to be characterized by the accuracy of each model as well as the diversity of predictions that may be present due to architectural differences. Generally, regardless of each model's performance, a model ensemble combines several models in a manner such that each model contributes an equal vote to the final prediction. A variation of this approach exists that allows the final prediction to be a result of a weighted average of each ensemble members' predictions. This allows well-performing models to contribute more and less-well-performing models to contribute less. A third variant of the model ensemble approach is known as the stacked generalization ensemble (Ting & Witten, 1999; Wolpert, 1992). This approach which is demonstrated in Figure 6, uses two or more base models – also called base learners or level 0 models – and combines their output as a basis for a meta-learner's predictions.

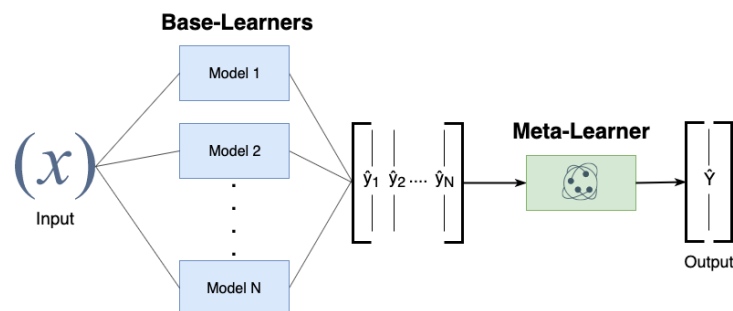


Figure 6. The stacking approach used to ensemble the DCNN models

2.2.6 Model Compression

Even though DL techniques such as TL often decrease training time and/or increase classification accuracy, DL models are known to be overparametrized and hence require significant computational resources. This is problematic in certain situations such as in PA where computational resources are limited. Model compression solves this problem by compacting models by about 35-50x the size of the base model (S. Han et al., 2016). Figure 7 demonstrates a model compression pipeline consisting of network pruning, quantization, and Huffman coding.

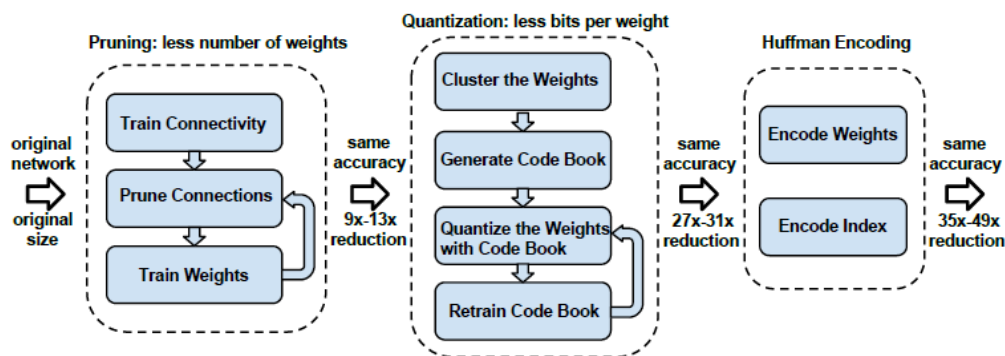


Figure 7. The three-stage model compression pipeline: pruning, quantization, and Huffman coding. Source: (S. Han et al., 2016)

Model pruning, which goes back to the 1990s (LeCun et al., 2015), refers to the biologically inspired algorithms that emphasize further changes to existing models to retain only the bare minimum information needed to achieve comparative accuracy to their base model (Kim et al., 2016; Molchanov et al., 2016; Zhu & Gupta, 2017). Pruning aims to reduce DCNN models by eliminating the redundancy and number of operations required for prediction. Further, quantization and weight sharing compress the pruned network by reducing the number of bits required to represent each weight, and Huffman coding ensures additional data compression. Compressing a DCNN model leads to a decrease in the number and complexity computations, as well as the

number of memory accesses for inference (the processing time for making a prediction) (Choudhary et al., 2020; Lalapura et al., 2021; Voghoei et al., 2018). The compressed model is more energy- and resource-efficient due to its smaller size and faster inference speed (S. Han et al., 2016). Successfully fitting a compressed model on an embedded or mobile device and performing inference at the edge (without a need to transmit data to an intermediary server) has some additional advantages. For example, in most embedded systems where compressed models have been implemented, training is performed offline; and only inference is run on the embedded device. In this case, the compressed model preserves user privacy and reduces the transmission cost (Lalapura et al., 2021; Voghoei et al., 2018). Further, the use of offline training (training once and deploying to several devices) reduces the resource requirement of the model as compared to continuous training (Chen et al., 2020). As demonstrated in Figure 6, all three compression techniques under the right conditions retain the prediction accuracy of the original model. Regardless, some studies have found that the pruning ratio affects the accuracy of the model (Fountsop et al., 2020; Rajaraman et al., 2020). In effect, some reduction in accuracy is possible depending on the percentage of the model's trainable weights that are pruned (Fountsop et al., 2020; Rajaraman et al., 2020; Tensorflow.org, 2021b).

2.3 ML for SSWM

Approaches used by researchers contain a range of ML techniques that can be applied to different problems along the value chain of agriculture from pre-production – such as activities that take place before actual planting occurs - to post-production – such as harvesting, supply chain, and marketing operations. A detailed description of the use of ML techniques in PA organized along the agricultural value chain is described in a prior study (El-Gayar & Ofori, 2020).

2.3.1 Computer Vision and ML for SSWM

In PA, weed detection using computer vision is not a novelty. It is an active research area that has evaded researchers over the years. Methods proposed by earlier studies used spectral imaging techniques available on drones (Goel et al., 2002; Gomez-Casero et al., 2010; Vioix et al., 2002). The aerial methods are most effective when spraying an entire field is required rather than the site-specific, or crop-specific, needs of precision and smart agriculture. Other studies that have used other ML techniques with hand-crafted features are listed in Table 3. Manual feature engineering can be cumbersome and are not ideal compared to recent methods.

Table 3. Computer vision and machine learning techniques for weed detection

Computer Vision Technique	Reference
Multispectral analysis	(Goel et al., 2002; Gomez-Casero et al., 2010; Vioix et al., 2002)
Logistic Regression	(Gutierrez et al., 2008)
Support Vector Machine	(Ahmed et al., 2012; Akbarzadeh et al., 2018; Bakhshipour et al., 2017; Binch & Fox, 2017)
K-Nearest Neighbor,	(Pallottino et al., 2018; Sabanci & Aydin, 2017)
Fuzzy Logic	(Sujaritha et al., 2017)
Fully Connected Artificial Neural Networks	(Bakhshipour & Jafari, 2018; Chantre et al., 2018; Pantazi et al., 2016, 2017; Torres-Sospedra & Nebot, 2014)

2.3.2 Deep Learning for SSWM

The literature shows that several studies continue to employ aerial-based methods for weed detection (dos Santos Ferreira et al., 2017; Farooq et al., 2019; Lammie et al., 2019; Milioto et al., 2017; Pantazi et al., 2016; Sørensen et al., 2017). Further, a recent review (Kamilaris & Prenafeta-

Boldú, 2018) found that all eight (8) deep learning papers for weed detection relied on satellite imagery-based remote sensing (Dyrmann et al., 2016; Milioto et al., 2017; Pantazi et al., 2017; Sørensen et al., 2017; Xinshao & Cheng, 2015) or unmanned aerial vehicles (dos Santos Ferreira et al., 2017; Farooq et al., 2019; Lammie et al., 2019). In terms of classification accuracy (CA), some of these studies have been extremely successful. For instance, Pantazi et al. (2017) employed hyperspectral imaging for crop and weed species recognition with their classifier able to identify 100% of crops. Specific to weed species recognition, however, the result varied between 31% and 98% in their mixture of Gaussians classifier; and 53% and 94% in their self-organizing maps classifier. Sørensen et al. (2017), on the other hand, identified weed classes growing among thistles with 97% CA. Similarly, Milioto et al. (2017) achieved up to 97.3% accuracy on two test sets of 10,000 sugarbeet UAV images using DCNNs.

Recent systematic reviews show that some studies have applied ground-based methods (Gikunda & Jouandeau, 2019; Kamilaris & Prenafeta-Boldú, 2018; Liakos et al., 2018; Moazzam et al., 2019; Santos et al., 2020; Zala & Patel, 2019)). A focused discussion of these DL applications in SSWM is presented below and summarized in Table 4.

Using ground-based methods, Lee et al. (2015) showed that despite the relatively lower precision of the AlexNet CNN in recognizing species from the LifeClef dataset [48% CA achieved by Reyes et al, (2015) using TL], the same model could achieve up to 99.50% CA on the 44-specie 90,000-image dataset sourced from the Royal Botanic Gardens in Kew, England. In this case, training from scratch employed in Lee et al. (2015) seems to deliver a significantly better result than TL employed in the former (Reyes et al., 2015). However, this could be due to the specific characteristics of the datasets used in the two studies. For example, in a different set of studies where adaptations of the VGG16 were used for early growth stage classification, Dyrmann et al.

(2016) managed 86.2% CA for the 22 species of plant seedlings while both Chavan & Nandedkar (2018) and Ashqar et al. (2019) found that using TL helped achieve CA of 93.64% and 99.48% respectively on a 12-class Plant Seedling dataset.

Some authors have demonstrated that combining different models can increase the robustness of the overall classification. For example, Xinshao & Cheng (2015) employed the PCANet, a CNN comprising of a principal component analysis network combined with a large margin classifier, to categorize 91 weed seed types found during the mixing of crop seeds. Their algorithm achieved 91% CA. Tang et al. (2017) achieved 92.89% accuracy using k-Means feature learning combined with a CNN for soybean classification. Such approaches, despite the performance gains, also increase the complexity of the models (Xinshao & Cheng, 2015).

On the other hand, some authors considered reducing DL model complexity (Lammie et al., 2019; McCool et al., 2017). McCool et al. (2017) found that their smaller AgNet model (0.25M parameters) could only achieve 85.9% CA as compared to an approach that incorporated a set of lightweight models with deep models. Only through this trade-off of memory and speed for accuracy were they able to achieve 90.3% CA but with an increased number of parameters (25M parameters). Lammie et al.'s (2019) attempts at model compression resulted in a smaller model that was 2.86 times faster but degraded CA by 1.17%.

Table 4. Recent applications of deep learning for weed detection

Reference	Problem Description	Dataset	DL model Used	Result% (Top 1 CA)	Ground-based Classification	Transfer Learning	Model Compression
(Xinshao & Cheng, 2015)	Classify 91 weed seed types	Data set of 3980 images containing 91 types of weed seeds	PCANet + LMC classifiers	90.96	x		
(Dyrmann et al., 2016)	Classify weed from crop species based on 22 different species in total	Data set of 10 413 images, taken mainly from BBCH 12–16 containing 22 weed and crop species at early growth stages	VGG16 (variation)	86.20	x		
(Sørensen et al., 2017)	Identify thistle in winter wheat and spring barley images	A total of 4500 images from 10, 20, 30, and 50m of altitude as captured by a Canon PowerShot G15 camera	DenseNet	97.00	x		
(Lee et al., 2015)	Recognize different plant species	Data set of 44 classes	AlexNet	99.60	x		
(Reyes et al., 2015)	Recognize different plant species	91 759 images	AlexNet	48.60	x	x	
(Fawakherji et al., 2019)	Pixel wise segmentation of Sunflower using CNN	500 images	Deep-Plant	90.00	x		
(Knoll et al., 2018)	Image-Based Convolutional Neural Networks for Carrot classification	500 images	CNN (authors do not name)	93.00	x		
(McCool et al., 2017)	Image-Based Convolutional Neural Networks for Carrot classification	60 images	AgNet	88.90	x		x
(Tang et al., 2017)	K-means feature learning accompanied by CNN for Soybean classification	820 Soybean images	CNN (authors do not name)	92.89	x		
(Milioto et al., 2017)	CNN based Semantic Segmentation of Sugar beet	10,000 images	CNN (authors do not name)	94.74			
(Andrea et al., 2017)	Image-Based Convolutional Neural Networks for Maize classification	2835 maize and 880 weed images	cNET	92.08	x		

(Chavan & Nandedkar, 2018)	Classification of 12 plant species	5544 images	AgroAVNET (hybrid model of AlexNet and VGG)	93.64	x	x	
(dos Santos Ferreira et al., 2017)	Weed detection and classification in soybean crops	400 crop images captured by the authors with UAV	CaeNet (CAFFE FW)	98.00			
(Farooq et al., 2019)	Weed detection and classification by spectral band analysis	200 Hyperspectral images with 61 bands	MatConvnet	94.72			
(Lammie et al., 2019)	Accelerate a DL approach with FPGA for weed classification with 8 classes	18 000 weed images from the DeepWeedX dataset	DenseNet	90.08			x
(Sabzi et al., 2018)	Identification of potato plants and three different weeds	4000 potato seedlings	Hybrid Artificial Neural Networks – Harmony Search algorithm (ANN-HS)	98.38	x		
(Ashqar et al., 2019)	Classification of 12 plant species	5608 images	VGG16	99.48	x	x	

2.4 Research Gap

Despite the successes that demonstrate the potential of DL for site-specific weed management, the producers of PA equipment have left DCNNs relatively underutilized. A criticism of DL and DCNN models is primarily their complexity resulting in the constant need for computing power which requires them to be run on high-end computers or graphical processing units – CPUs and GPUs (Santos et al., 2020). An added disadvantage to this high computing power requirement is that it results in high power consumption to make predictions – which is ineffective for sustainable farming (Santos et al., 2020; J. Wang et al., 2018).

Various literature reviews on the subject of agricultural information technology adoption for PA (Lowenberg-DeBoer & Swinton, 1997; Moazzam et al., 2019; Wolfert et al., 2017) have demonstrated that farmers are often concerned with their bottom-line, which makes the cost of technology a key issue when developing equipment. (Lowenberg-DeBoer et al., 2019), in their analysis of the economics of robots and automation, found that although switching from conventional mechanization to automated systems could have positive ripple effects on the whole farm, such a shift in on-farm mechanics will only gain traction if new systems can prove their cost-effectiveness. Similarly, Ofori and El-Gayar (Ofori & El-Gayar, 2020b), in their survey of social media posts, found that reducing the cost and complexity of agricultural information technologies could result in the uptake of technology and the adoption of precision agriculture.

Hence, for commercial farm equipment producers (and ultimately farmers) to accept and adopt DL systems for precision agriculture, research that introduces less complex models to reduce the demand for computing resources is required. To an extent, lightweight and mobile DCNN models have been able to solve the complexity problem but some work is needed to increase their accuracy to the levels of state-of-the-art DCNNs demonstrated in Table 1. In effect, reducing

model complexity should not result in performance degradation. This study proposes an approach to reducing the complexity of DCNN models for increased efficiency in ground-based plant classification systems. The proposed model stacks two DL models to counteract the expected effect of performance degradation in a compressed model and performs an empirical benchmark of performance runtimes in ss resource-constrained environments.

From a theoretical perspective, the research demonstrates the potential of leveraging transfer learning, model compression, and ensemble learning to reduce the complexity (and thus the resource demands) of the resultant model while still maintaining classification performance that is comparable to full-size models. By reducing model complexity, the proposed method can also have implications for practice as it decreases the demands for computational resources and supporting technology infrastructure, thus contributing to the improved likelihood of adoption in resources-constrained environments such as precision agriculture

CHAPTER 3

RESEARCH METHODOLOGY

This section introduces the research methodology employed in this dissertation. A brief discussion on Design Science research methodology (DSRM) and how it fits the context of the study is presented. This is followed by a description of the artifact proposed in this study.

3.1 Rational for the Design Science Research Methodology

Generally, IS research – which deals with the study of artificial phenomena such as information systems – is comprised of two complementary paradigms: behavioral and design science research (Hevner et al., 2004; March & Smith, 1995). The approach of the behavioral science method is geared towards theorizing and justifying, that is to say, understanding and predicting phenomena. On the other hand, the design approach is a problem-solving methodology that centers on the study, development, and performance of information systems artifacts. These innovative artifacts are meant to contribute new knowledge to the body of scientific evidence. DSRM offers an appropriate and comprehensive framework to exploit the design process, as an opportunity to learn and further understand a problem domain while contributing to the body of knowledge (Blakey et al., 2008; Vaishnavi et al., 2004).

Accordingly, this study ensured that it meets the standards of DSRM by covering all seven (7) DSRM guidelines for information systems research posited by Hevner et al. (2004). Specifically, the study followed the DSRM process model proposed by Peffers et al. (2007) in identifying the issues with the resource intensiveness of DL in SSWM and defining specific objectives for a suitable solution. Subsequently, the study designed an artifact using DL and

DCNN techniques. The development was done by applying state-of-the-art techniques to compress proposed DL models while increasing their generalizability and accuracy. To demonstrate the effectiveness, the study evaluated the proposed model using various plant datasets to assess the fitness of the model for practice and benchmarked the model runtime in a resource-constrained environment. This dissertation and the various publications made from serve as the means to communicate the findings of the work, and in extension, contribute to the body of knowledge of IS research.

3.2 Proposed Artifact

The discussion presented on the state-of-the-art DCNNs in the previous section highlights the effects model compression could have on model accuracy, and to an extent why there is a general lack of their application to SSWM. In this research, a novel method is proposed to realize an artifact for plant recognition. The proposed approach comprises three steps as shown in Figure 8 and detailed in the following sub-sections.

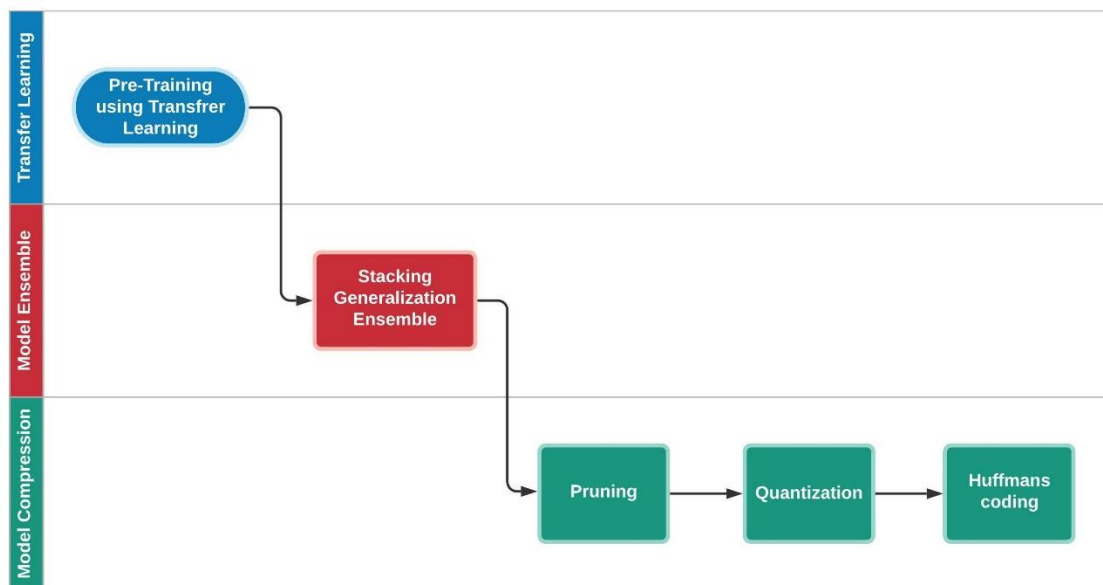


Figure 8. The proposed approach for a robust DCNN artifact

The novelty of the approach lies in the combination of hybrid ensemble deep learning with model compression techniques to compensate for the expected performance degradation in a compressed model.

3.2.1 S1: Pre-Training using Transfer Learning

As step one, several state-of-the-art DCNN models were considered based on their architectural properties as explained using the distinctions proposed by Khan et al. (2019) and as summarized in the literature review. These models were pre-trained with transfer learning. This strategy, which was discussed earlier, allowed retraining higher-level portions of the network while keeping the lower levels. The higher levels are chosen for retraining as they are known to contain features more specific to the original dataset for which they were trained (Yosinski et al. 2014). Additionally, the fine-tuning approach is used to combat negative transfer learning. In this case, the optimal models for each layer were chosen based on past work in this area (Ofori & El-Gayar, 2020a, 2021):

- **VGG16:** block4_conv2
- **InceptionV3:** conv2d_51
- **DenseNet121:** conv4_block 12 1_conv
- **Xception:** block10_sepconv 1_act
- **ResNet152V2:** conv4_block 14 2_conv
- **EfficientNet-B1:** block5a_expand_conv

3.2.2 S2: Model Ensemble

In step two, the stacked generalization model ensemble approach was used for combining models. This approach was considered because it has shown superior performance to the weighted ensemble and other forms of ensemble approaches in the past. It achieves the best result when,

instead of class labels, the class probability predictions of the base models are used as input to the meta-learner (Ting & Witten, 1999). It has the added advantage of one-shot inference as compared to other ensemble models where a prediction has to be performed for each model in the ensemble.

In this dissertation, the stacking ensemble model was developed based on the results of pre-training. The ensemble was formed by combining the two models from the previous step that gave the best trade-off between model accuracy and size. As per the results reported in the following chapter, these two models were the DenseNet and Xception models stacked into the new artifact as the *XD-Ensemble*. Figure 9 is an illustration of the XD-Ensemble captured using the Netron application (<https://netron.app/>). It demonstrates the Xception (as the base learner on the left of the network) and the DenseNet (as the base learner on the right of the network). These two base learners have been concatenated as an input into the meta-learner. In this case, transfer learning was used such that the base learners were frozen and the meta-learner was trained to predict results based on predictions from both models (taken from each base learner's softmax layer). The meta-learner consisted of three fully connected layers with the final softmax layer used for making predictions. It had the following dimensions:

- **Concatenation layer:** This layer which takes as input a list of tensors and returns a single tensor was used to combine the output from the two base models.
- **Dense layer:** Two densely connected NN layers of size 512 and 256 with ReLU – rectified linear activation function. A final dense softmax layer was added for prediction which has a size specific to each dataset.

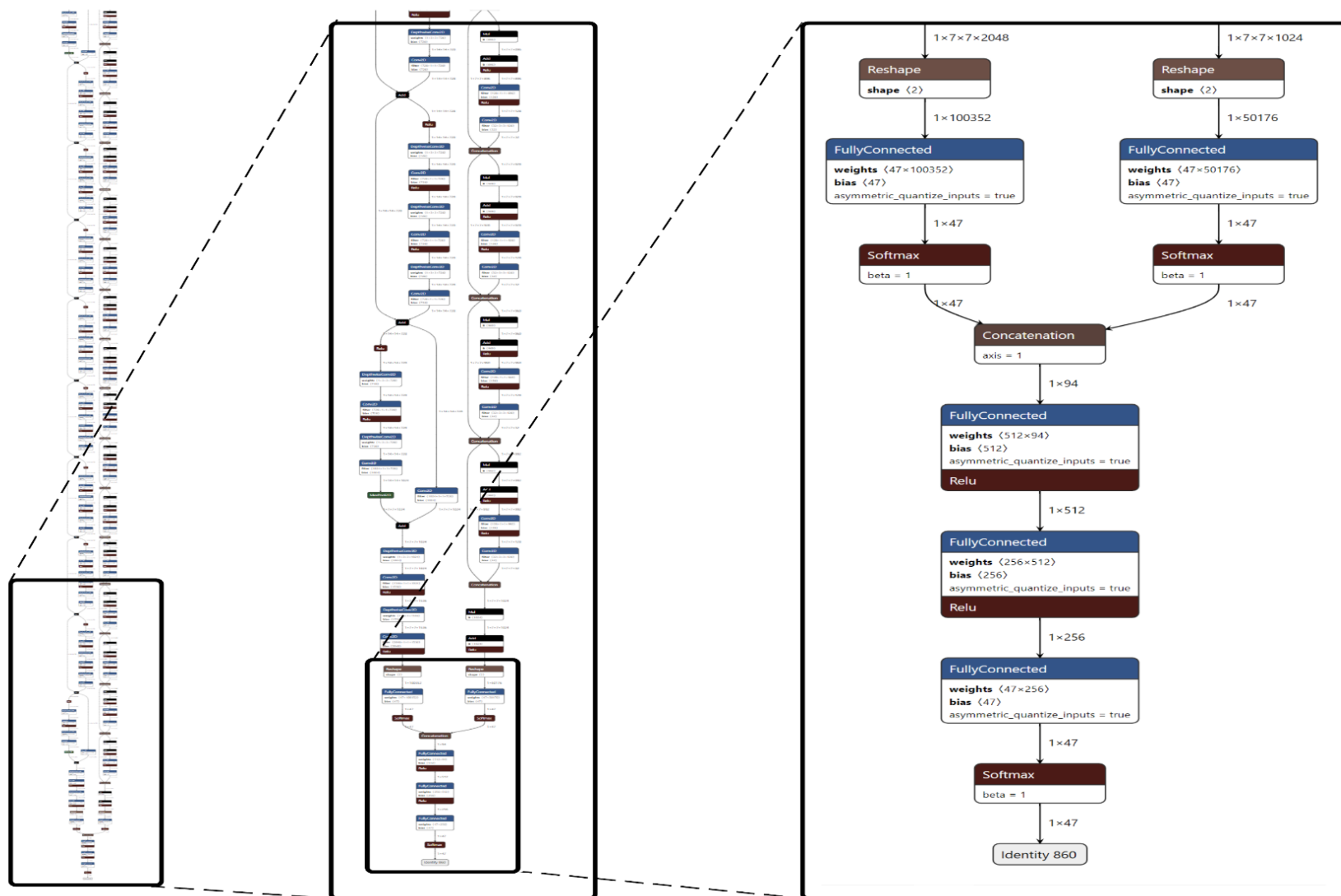


Figure 9. Snapshot of the stacked XD-Ensemble consisting of an Xception (left-base) and DenseNet121 (right-base) learners

3.2.3 S3: Model Compression

In step three, the resulting meta-learner was subjected to deep compression using S. Han et al.'s (2016) three-step pipeline: Pruning, Quantization, and Huffman encoding. Pruning to mask out redundant weights; Quantization to reduce the number of bits needed to store each weight through weight sharing such that only the effective weights (“codebook”) and resulting indices would be stored, and Huffman encoding to reduce the number of bits needed to represent the weights in the codebook.

In this study, model training was started at 50% sparsity with a target of 80% sparsity by the end of training. Following this, post-training dynamic range quantization and Huffman’s encoding were applied to reduce CPU and hardware accelerator latency, processing, power, and model size were performed (Tensorflow.org, 2021a). This was done during conversion to the TFLite format by reducing the number of bits needed to store each weight and compressing the resulting model in a lossless format .The TFLite format is part of the TensorFlow framework optimized for inference on the edge in devices such as mobile, Internet of Things (IoT), and embedded systems to measure. TFLite facilitates inference at the edge from five key areas: “latency (there's no round-trip to a server), privacy (no personal data leaves the device), connectivity (internet connectivity is not required), size (reduced model and binary size) and power consumption (efficient inference and a lack of network connections)” (Tensorflow.org, 2021c).

3.3. Proposed Experiments and Model Evaluation

3.3.1 Dataset

Plant Seedling Dataset. Giselsson et al. (2017) introduced the public image database for benchmarking plant seedling classification aimed at ground-based weed or species spotting (<https://vision.eng.au.dk/plant-seedlings-dataset/>). The dataset consists of 5,539 images of

approximately 960 unique plants belonging to 12 species at several growth stages. The plants were grown indoors in Styrofoam boxes and images were captured over 20 days. As overlapping plant leaves are minimal at the onset of plant growth, where most weed control such as broadcast spraying is undertaken, the images were captured in non-overlapping mode. Also, to avoid errors that may occur in pixel-based segmentation algorithms, plants were grown in soil that is covered in small stones. Six samples from the dataset are presented in Figure 10.

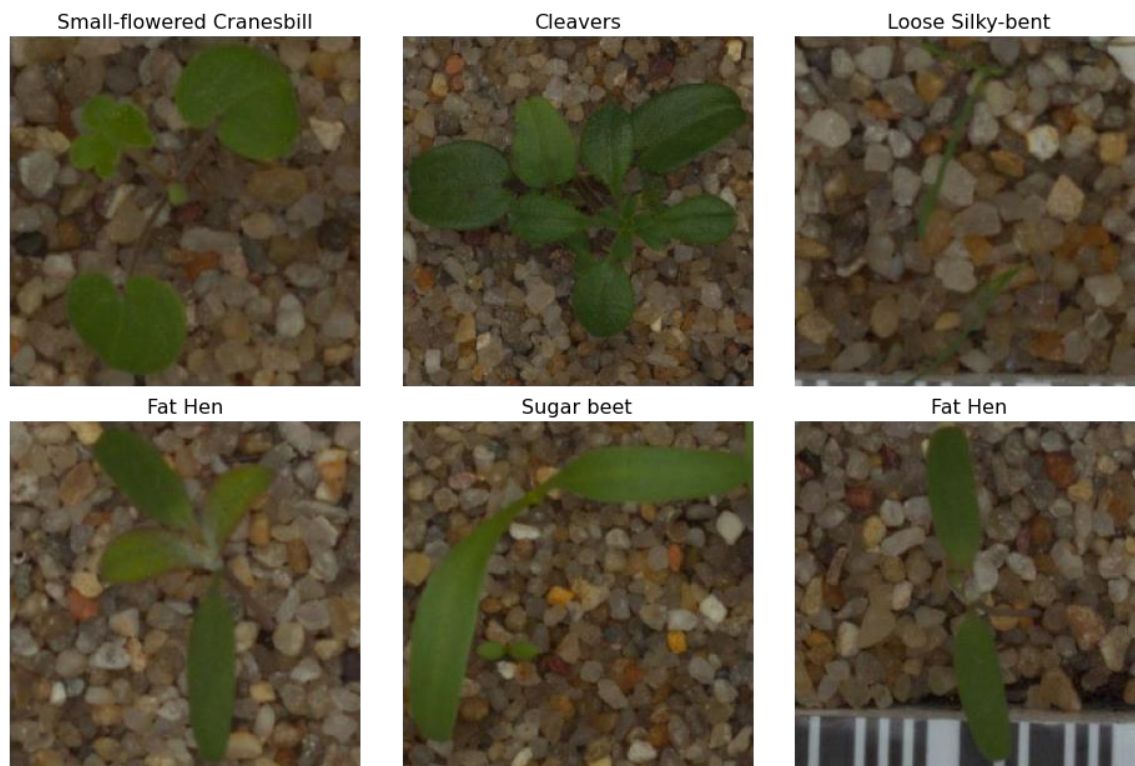


Figure 10. Samples from the Plant Seedling Dataset

Leafsnap Dataset. This dataset of plant species contains crops gathered across variable environmental conditions and light settings (Kumar et al., 2012). The dataset contains 30,866 RGB images of plant leaves of over 185 trees found in the Northeastern United States and Canada (<http://leafsnap.com/dataset/>). The images, as demonstrated in Figure 11, were captured by a non-

profit group (*Finding Species*) using mobile devices and all descriptions of the 185 plant species are botanist-curated.

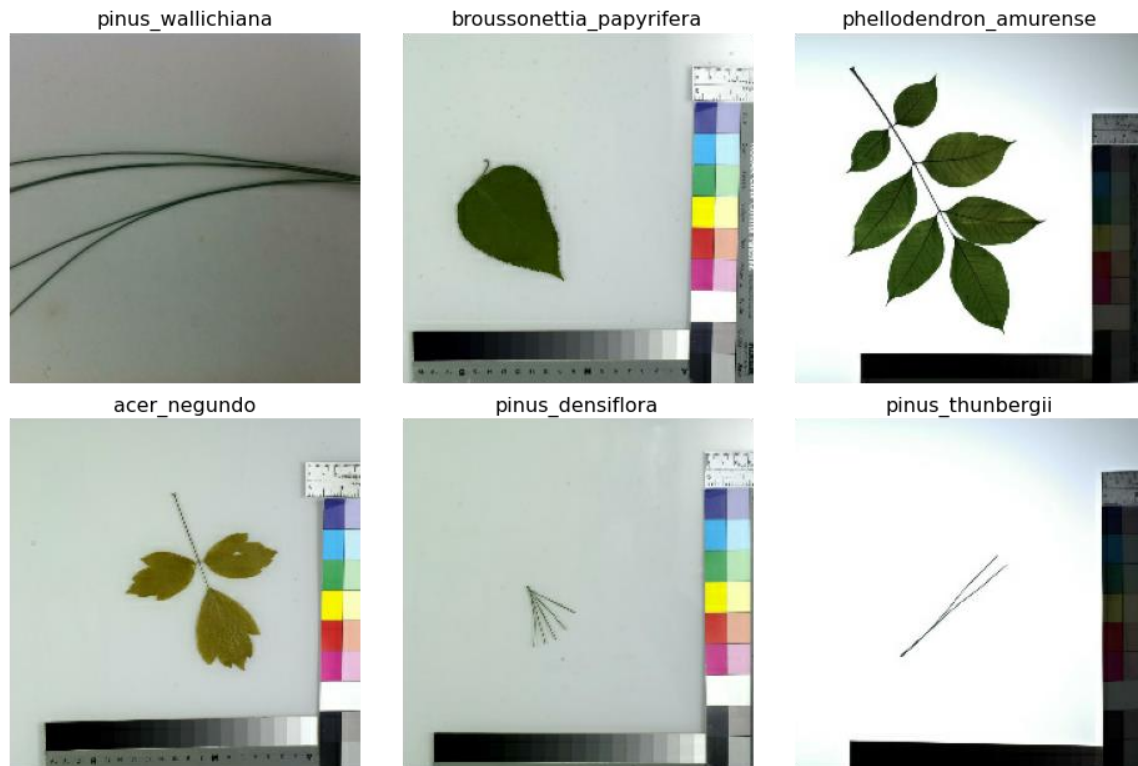


Figure 11. Samples from the Leafsnap Dataset

Open Plant Phenotyping Database (OPPD). The dataset authored by Leminen Madsen et al. (2020) contains over 315,038 plant objects, representing 64,292 individual plants from 47 different species (<https://vision.eng.au.dk/open-plant-phenotyping-database/>). Each species is cultivated under three different growth conditions, to provide a high degree of diversity in terms of visual appearance. For plant species classification as employed in this dissertation, the Monocotyledonous plants (PPPM) consisting of 9,365 images and Dicotyledonous plants (PPDD) consisting of 37,990 samples were excluded from the classification task to prevent the introduction of noise from the lower taxonomy levels during model training. Overall, 47 plant

species (264,899 plant images) were used for evaluating the models. Some examples from the dataset are shown in Figure 12.

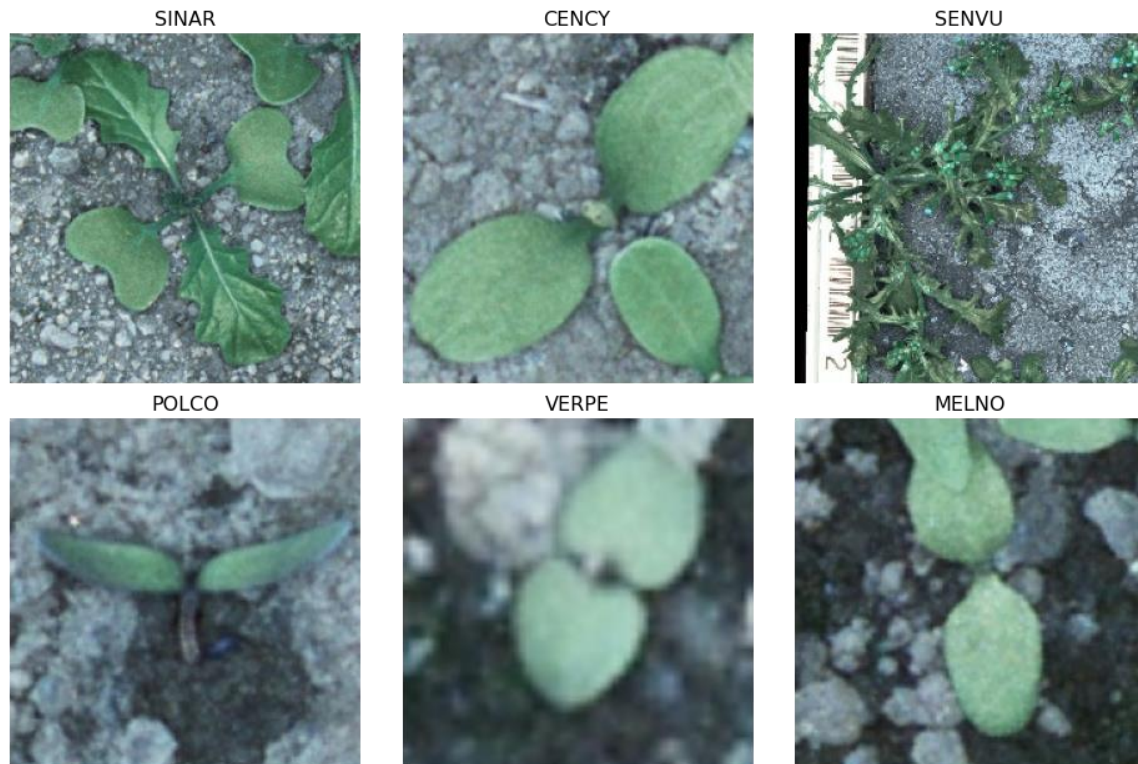


Figure 12. Samples from the OPPD Dataset

3.3.2 Data Preparation

Image resizing. All images were resized to 224x224 pixels to ensure the same aspect ratio.

Normalization of pixel values. Pixel values range from 0 to 255. However, for DL projects, it is good practice to maintain values between 0 and 1. This is done to ensure that all the pixels have similar data distribution. Pixel normalization aids the convergence of neural networks. To achieve this normalization, all pixel values were divided by the largest pixel value: 255.

Data augmentation. Since plants do not grow in a single orientation and images could be captured from different angles, image augmentation was performed using random horizontal and vertical flips (one in two chance for each). Additionally, random image saturation to modify the

depth and/or intensity of colors present within each image was applied to simulate bad lighting or image quality. A random value between 0 and 2 was used as a saturation factor.

3.3.3 Training Conditions and Computational Resources

All DCNN model training carried out in this research was conducted using the Python programming language and implemented using the Keras high-level API and a TensorFlow backend (Abadi et al., 2016; Chollet & others, 2015). Table 5 presents a summary of the different training conditions applied to each dataset. Random seed for both TensorFlow and NumPy was set at the beginning of training. The random seed allowed each model to be trained with the same distribution of datasets, augmented images, and more.

Table 5. Training conditions per dataset

Dataset	No. of Images			Classes	Mini Batch Sizes	Epochs
	Total	Training	Test			
Plant Seedling	5539	4,524	1,015	12	32	125
Leafsnap	30,866	24,705	6,161	185	64	65
OPPD	264,899	212,063	52,836	47	128	35

The development environment was set up on the cloud using Google Colab Pro's Tensor Processing Unit v2 (TPU). The initial learning rate was set at 0.0001. As shown in Table 5, different batch sizes and the number of epochs were experimented with: the larger the dataset the lower the number of epochs, and the higher the batch sizes. This serves two needs, it encouraged 1) experimentation of different training scenarios and 2) judicious use of expensive training hardware. The datasets were divided into 80% training and 20% test sets. It is worth noting that for each dataset, all models were trained using the same parameters – training and test images, mini-batch sizes, and number of epochs.

To demonstrate that the approach delivers consistent results, a k-fold cross-validation approach was used where the training dataset D , was randomly divided into k number of mutually exclusive folds (subsets): $S_1, S_2, S_3, \dots, S_k$. Consequently, the approach represented visually in Figure 13 and summarized below was used to train and validate each model:

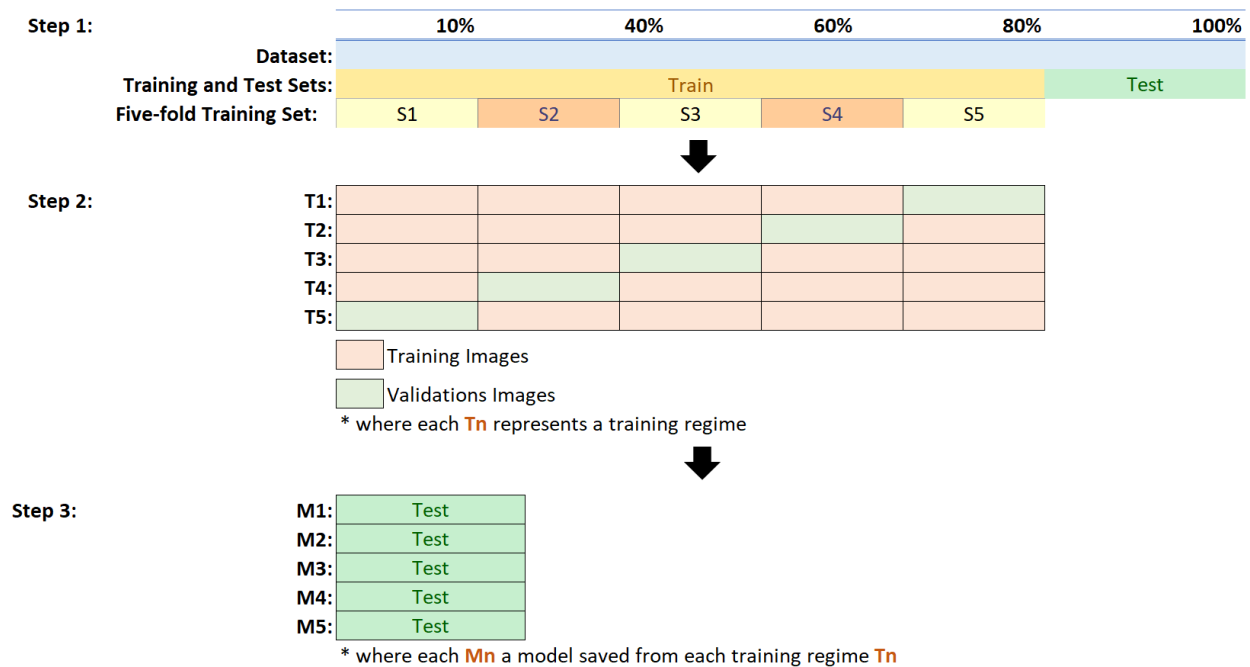


Figure 13. Training and testing steps

Step 1. A five-fold cross-validation approach was used where the training images (80% of the original dataset) were randomly divided into five different sets (S_1, S_2, S_3, S_4 , and S_5).

Step 2. Five training regimes (T_1, T_2, T_3, T_4 , and T_5) were run for all the models where every single model was trained on four out of the five sets with the fifth used as a holdout set to validate the model performance. During training, the performance of the model on each holdout set was used to tune the learning rate such that the initial rate of 0.0001 was decreased by a factor of 0.5 after every 3 continuous epochs where the validation accuracy on the holdout set did not improve.

Step 3. Once training was complete, each model had five variations ($M_1, M_2, M_3, M_4,$ and $M_5.$) to be used for evaluation. Each variation was evaluated against the test set. The specific evaluations carried out are described below.

3.3.4 Evaluation

Evaluation of the proposed ensemble model was carried out against state-of-the-art DCNN models which had been fine-tuned with transfer learning as presented in section 3.2.1. Similar to the proposed artifact, the last dense layer of each state-of-the-art model meant to predict 1,000 ImageNet classes was replaced with a new dense layer consisting of the number of classes in each dataset. The layers chosen for fine-tuning were the same as those used in pre-training and outlined earlier. This means about two-thirds of each model was frozen and the remaining layers were fine-tuned.

Overall, the proposed model was evaluated against six models – five state-of-the-art models: DenseNet121, Inception V3, ResNet152V2, VGG16, and Xception, as well as a mobile-sized DCNN with only 7.8M parameters: EfficientNet-B1. For accuracy evaluation, Top-1 accuracy in percentage was measured. In the state-of-the-art models, accuracy evaluation was performed twice to account for both the fine-tuned models without pruning (referred to as vanilla in the result) and with pruning (referred to as pruned in results). For the pruned models, the same pruning parameters were used such that training started at 50% sparsity with a target of 80% sparsity by the end of training.

Similarly, model size in megabytes (MB) was recorded. First, the vanilla models (M_1 to M_5 per model) were saved in the TFLite format. In the current study, the model metadata was added to populate mandatory information such as the class labels, for inference, no further modifications were made to the state-of-the-art models.

For the prediction runtime measurements, the models were initially benchmarked in the Google Colab development environment. The models were then converted to TensorFlow Lite (TFLite) to be benchmarked on a low-end android device: a Google Pixel 3 with a Qualcomm Snapdragon 845 chipset. The benchmark made use of TFLite’s image classification android application (<https://www.tensorflow.org/lite/guide/android>). This implementation of DCNNs on android using the TFLite Java API was adapted to suit the purposes of the current study (see Appendix A). Unit test cases were written to log the inference times of predicting 50 different input images using the TFLite models saved from each of the 5 folds. In effect, the average runtime of each model was collated by averaging the result of 250 prediction instances.

Additional Performance Benchmarks:

Additional performance evaluation was carried out on the XD-Ensemble model to demonstrate robustness. The evaluation metrics calculated were: *Precision* (P_c), *Recall* (R_c), and *Mean Weighted Average f_1 -scores* (S) as shown in equations 1-4 below:

$$\text{Precision:} \quad P_c = \frac{TP_c}{TP_c + FP_c} \quad (1)$$

$$\text{Recall:} \quad R_c = \frac{TP_c}{TP_c + FN_c} \quad (2)$$

$$\text{F1-scores:} \quad f_{1,c} = 2 \frac{P_c \cdot R_c}{TP_c + FN_c} \quad (3)$$

$$\text{Mean Weighted F1-scores:} \quad avg_{weighted}(f_1) = \sum_{c=1}^C \frac{N_c}{N} \cdot f_{1,c} \frac{P_c \cdot R_c}{TP_c + FN_c} \quad (4)$$

Where TP_c , FP_c , and FN_c denote True positives, False positives, and False negatives for class c respectively. P_c is class-specific precision and R_c is the class-specific recall. N denotes the total number of samples. N_c is the number of samples of class c and C is the total number of classes.

CHAPTER 4

EVALUATION AND ANALYSIS

This chapter presents a detailed experimental evaluation of the approach that is proposed and described in Chapter 3. Using the three datasets, the ensuing discussion presents the proposed evaluation benchmarks as pseudo-objectives to validate the practicality of the approach by showing a) accuracy measures to demonstrate that the proposed approach achieves prediction accuracy comparable to state-of-the-art DCNN models in similar training conditions (Figures 14-16), b) model size measurements showing the compression achieved by employing the proposed approach, and c) runtime benchmarks to demonstrate that the approach can achieve faster inference times compared to state-of-the-art DCNN models.

4.1. Model Accuracy Measures

4.1.1 Accuracy on the Plant Seedling Dataset

As demonstrated in Table 6, the first experiment employed the Plant Seedling dataset consisting of 12 plant seedlings and weed classes. When state-of-the-art models were fine-tuned with TL, the models delivered a consistent performance baseline that averaged $92.44\% \pm 1.70$. The DenseNet delivered the best performance with an average of $93.95\% \pm 0.34$, followed by the Xception at $93.62\% \pm 0.36$. The rest were the Inception at $93.32\% \pm 0.47$, the ResNet at $92.00\% \pm 0.48$, and the VGG at $89.30\% \pm 1.10$.

In a second scenario where the same models were pruned during training, there was a drop in prediction accuracy of $\approx 2\%$ with an average of $90.38\% \pm 2.23$. The pruning schedule affected the DenseNet the least with only an average drop of 0.06%. However, the remaining models had a

more dramatic accuracy loss: The ResNet lost 0.59%, the VGG lost 2.3%, the Xception lost 3.35% and the Inception model experienced the highest loss in accuracy: 4.26%.

Comparatively, the EfficientNet lightweight model's accuracy was 93.97%. Although this was higher than the accuracy of the state-of-the-art models, the proposed XD-Ensemble had an even better accuracy of 94.33 ± 0.63 with a minimum of 93.30% in the first fold and a maximum of 95.07% in the fifth fold.

Table 6. Experimental result – Plant Seedling dataset

Model	Version	Accuracy					Avg (μ)	Std (σ)
		M ₁	M ₂	M ₃	M ₄	M ₅		
DenseNet121	Pruned	0.94680	0.93202	0.93990	0.93005	0.94581	0.93892	0.00688
	Vanilla	0.93300	0.94089	0.94286	0.93990	0.94089	0.93951	0.00339
Inception V3	Pruned	0.88867	0.87783	0.88867	0.90837	0.88966	0.89064	0.00987
	Vanilla	0.93695	0.93596	0.93596	0.92414	0.93300	0.93320	0.00472
ResNet152V2	Pruned	0.91034	0.91330	0.91626	0.91330	0.91724	0.91409	0.00245
	Vanilla	0.91429	0.91527	0.92709	0.92020	0.92315	0.92000	0.00480
VGG16	Pruned	0.87882	0.85616	0.88966	0.86502	0.87389	0.87271	0.01148
	Vanilla	0.88276	0.89655	0.90936	0.89754	0.87882	0.89300	0.01102
Xception	Pruned	0.89655	0.90345	0.90837	0.90246	0.90246	0.90266	0.00376
	Vanilla	0.93498	0.93103	0.93498	0.94187	0.93793	0.93616	0.00360
EfficientNet-B1		0.93757	0.93600	0.94035	0.94222	0.94244	0.93972	0.00255
XD-Ensemble		0.93300	0.93990	0.94778	0.94483	0.95074	0.94325	0.00625

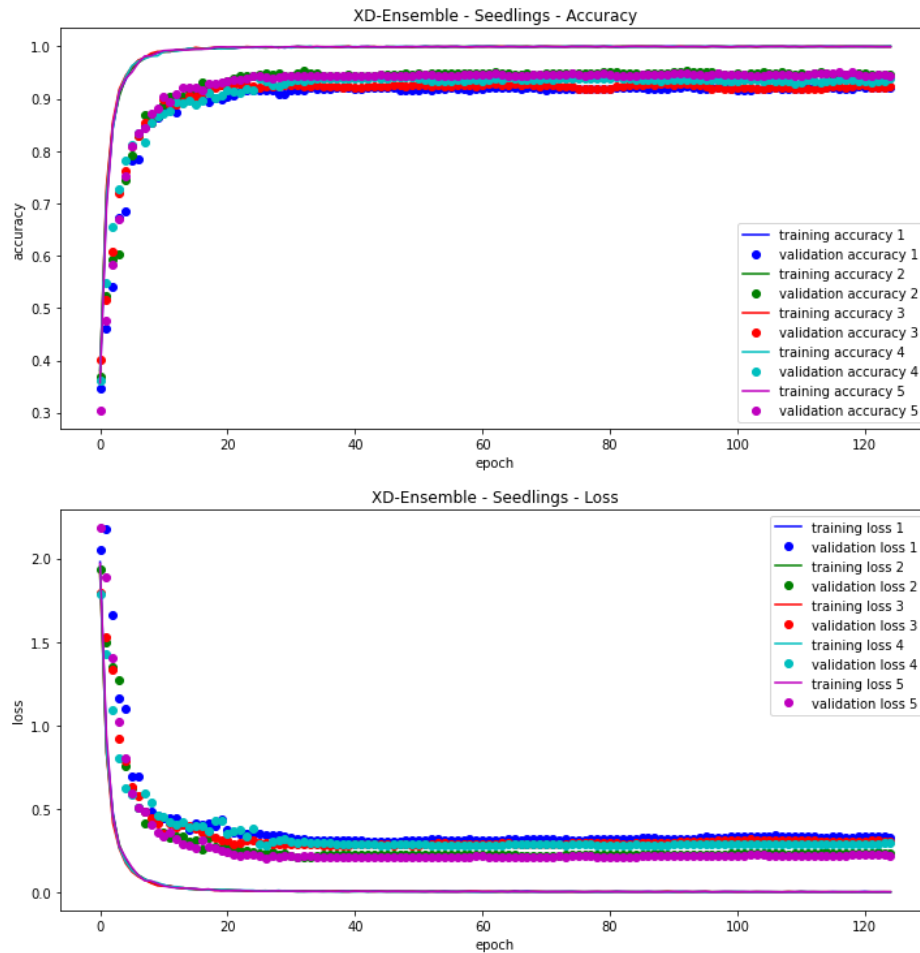


Figure 14. XD-Ensemble Validation Accuracy and Loss on Plant Seedling Dataset

4.1.2 Accuracy on the Leafsnap Dataset

Table 7 shows experiments conducted with the Leafsnap dataset which consists of 185 tree species as identified by photographs of their leaves. The full-sized models averaged $94.83\% \pm 0.01$. The highest accuracy was attained by the DenseNet with an average of $96.06\% \pm 0.11$, this was followed closely by the Xception at $95.60\% \pm 0.11$, the ResNet at $94.92\% \pm 0.12$, the Inception at $94.42\% \pm 0.14$, and finally the VGG at $83.08\% \pm 0.24$.

Pruning models resulted in a $\approx 1.5\%$ drop in overall model accuracy at $93.41\% \pm 1.52$. Once again, the DenseNet had the least average drop in prediction accuracy of 0.39% . The prediction

accuracy of the ResNet model dropped by 0.86%. The remaining three models, VGG, Xception, and Inception, also dropped 1.70%, 1.97%, and 2.50% respectively.

In this case, EfficientNet had an average accuracy of 95.195 and the XD-Ensemble achieved an accuracy of 95.97 ± 0.01 . In this set of results, the XD-Ensemble was 0.09% lower than the best performing DenseNet model. However, that result was also 0.30% higher than the best performing pruned model.

Table 7. Experimental result – leafsnap dataset

Model	Version	Accuracy					Avg (μ)	Std (σ)
		M ₁	M ₂	M ₃	M ₄	M ₅		
DenseNet121	Pruned	0.95358	0.95958	0.95764	0.95536	0.95715	0.95666	0.00205
	Vanilla	0.96056	0.96251	0.95894	0.96072	0.96023	0.96059	0.00114
Inception V3	Pruned	0.92371	0.92745	0.91966	0.91690	0.91365	0.92027	0.00488
	Vanilla	0.94725	0.94416	0.94335	0.94546	0.94595	0.94524	0.00136
ResNet152V2	Pruned	0.94483	0.94127	0.94317	0.93378	0.93994	0.94060	0.00379
	Vanilla	0.94774	0.94839	0.95114	0.94871	0.94985	0.94916	0.00120
VGG16	Pruned	0.90927	0.92258	0.91316	0.91219	0.91186	0.91381	0.00457
	Vanilla	0.93150	0.93280	0.92858	0.92745	0.93361	0.93079	0.00239
Xception	Pruned	0.93232	0.93605	0.93524	0.93848	0.93897	0.93621	0.00241
	Vanilla	0.95553	0.95796	0.95618	0.95472	0.95536	0.95595	0.00111
EfficientNet-B1		0.95510	0.95274	0.95192	0.95067	0.94932	0.95195	0.00195
XD-Ensemble		0.95803	0.96786	0.96397	0.94873	0.96624	0.95965	0.00697

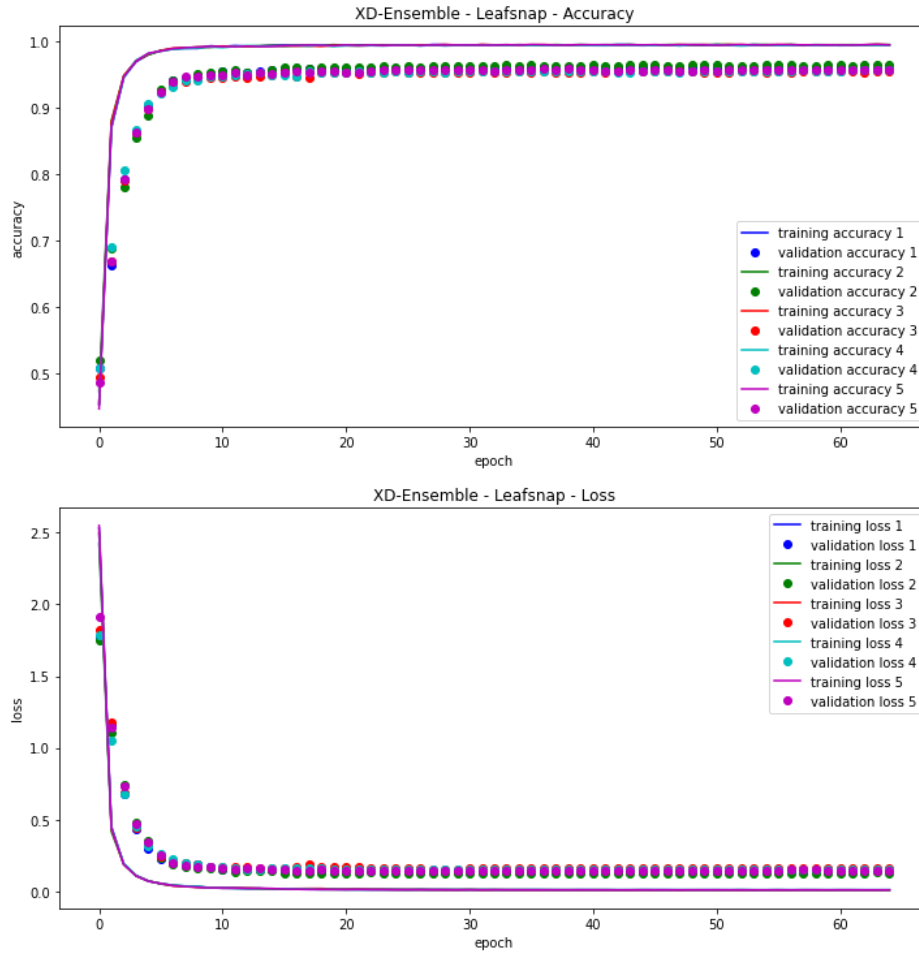


Figure 15. XD-Ensemble Validation Accuracy and Loss on Leafsnap Dataset

4.1.3 Accuracy on the Open Plant Phenotyping Dataset

In the last experiment on model accuracy, the OPPD consisting of 47 individual plant seedlings was utilized. Table 8 is a summary of the experimental results achieved per model in each of the five folds. The average accuracy of the full-sized models was $95.70\% \pm 1.08$. The model with the highest accuracy on this dataset was the Xception, $96.46\% \pm 0.14$, beating the performance of the DenseNet by 0.04%. The Inception and ResNet models achieved similarly high performance distinguishing the classes of this dataset with an average accuracy of $96.02\% \pm 0.20$, and $96.01\% \pm 0.11$ respectively. The worst accuracy for a vanilla model was the VGG16 recorded at $93.56\% \pm 0.07$.

The pruned models had $\approx 1.8\%$ lower accuracy than their vanilla versions with a drop in overall model accuracy to $93.89\% \pm 2.75$. The DenseNet had the highest sparse accuracy of $96.15\% \pm 0.06$, followed by $95.59\% \pm 0.12$ achieved by the ResNet, then $95.37\% \pm 0.11$ achieved by the Xception, $93.73\% \pm 0.22$ by the Inception, and $88.63\% \pm 0.10$ by the VGG.

The proposed XD-Ensemble achieved an accuracy of $97.02\% \pm 0.07$ which was 0.56% higher than the maximum average accuracy achieved by the best performing model on this dataset: the full-sized Xception model. Additionally, that result was also higher than the EfficientNet's 96.65% average prediction accuracy. The minimum accuracy of the XD-Ensemble was 96.92% on fold 1 and the maximum accuracy of 97.12% was recorded on fold 3.

Table 8. Experimental result – OPPD dataset

Model	Version	Accuracy					Avg (μ)	Std (σ)
		M ₁	M ₂	M ₃	M ₄	M ₅		
DenseNet121	Pruned	0.962526	0.960803	0.961333	0.961276	0.961674	0.961522	0.00057
	Vanilla	0.962715	0.963983	0.964835	0.964002	0.965592	0.964225	0.00096
Inception V3	Pruned	0.934458	0.939984	0.935063	0.939265	0.937827	0.937319	0.00221
	Vanilla	0.960444	0.959857	0.961977	0.96209	0.956677	0.960209	0.00197
ResNet152V2	Pruned	0.954898	0.955106	0.955012	0.956583	0.95787	0.955894	0.00116
	Vanilla	0.959441	0.960765	0.96192	0.95874	0.959441	0.960061	0.00114
VGG16	Pruned	0.88646	0.885211	0.886857	0.887823	0.885306	0.886331	0.00098
	Vanilla	0.935764	0.935782	0.934458	0.935612	0.936558	0.935635	0.00067
Xception	Pruned	0.954368	0.954822	0.954406	0.953043	0.951756	0.953679	0.00113
	Vanilla	0.965024	0.966898	0.964494	0.964083	0.962696	0.964639	0.00137
EfficientNet-B1		0.96658	0.96751	0.96603	0.96879	0.96366	0.96652	0.00171
XD-Ensemble		0.96917	0.96957	0.97053	0.97116	0.97074	0.96917	0.00074

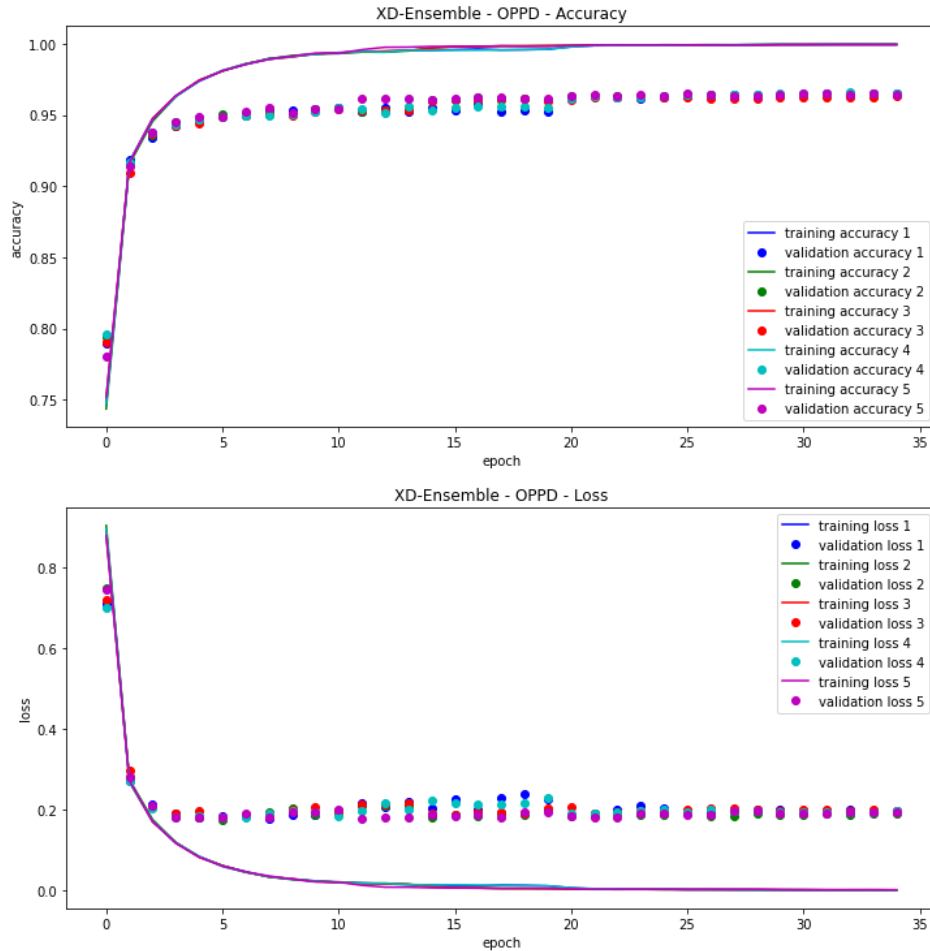


Figure 16. XD-Ensemble Validation Accuracy and Loss on OPPD Dataset

4.2. Model Size Measurements

4.2.1 Model Sizes for Plant Seedling Classification

When trained to classify the Plant Seedling dataset, the TFLite model sizes of the state-of-the-art models were 226.6MB for the ResNet, 85.44MB for the Inception, 83.91MB for the Xception 57.29MB for the VGG, and 29.94MB for the DenseNet. The EfficienNet model was also 27.71MB in mobile form while the XD-Ensemble measured 28.9MB.

Table 9. Model size measurements ^a

Model	Dataset	Mobile Size (TFLite)		
		Avg (μ)	Max	Min
VGG16 ^b	Seedlings	57.290	57.290	57.290
	Leafsnap	73.847	73.847	73.847
	OPPD	60.640	60.640	60.639
Xception ^b	Seedlings	83.914	83.914	83.913
	Leafsnap	150.141	150.141	150.141
	OPPD	97.312	97.313	97.312
ResNet152V2 ^b	Seedlings	226.607	226.608	226.605
	Leafsnap	292.834	292.835	292.832
	OPPD	240.004	240.004	240.004
InceptionV3 ^b	Seedlings	85.436	85.437	85.434
	Leafsnap	119.226	119.226	119.224
	OPPD	92.272	92.273	92.270
DenseNet121 ^b	Seedlings	29.941	29.942	29.939
	Leafsnap	62.008	62.009	62.007
	OPPD	35.594	35.594	35.593
EfficientNet-B1 ^b	Seedlings	27.708	27.710	27.708
	Leafsnap	69.102	69.102	69.100
	OPPD	36.083	36.082	36.084
XD-Ensemble	Seedlings	28.895	28.895	28.893
	Leafsnap	54.989	54.990	54.987
	OPPD	35.009	35.009	35.007

^a Size in megabytes and standard deviation < 0.01

^b Size measure resulting from converting the vanilla model to TFLite format

4.1.2 Model Sizes for Leafsnap Classification

Compared to the other datasets, the models trained to classify images from the Leafsnap dataset had the highest model sizes due to the greater number of classes in the dataset. In a similar order, the ResNet had the highest size of 292.83MB. The remaining models were measured as 150.14MB for the Xception, 119.23MB for the Inception, 73.85MB for the VGG, and 62MB for the DenseNet. The EfficientNet maintained its comparatively smaller size at 69.1MB and the compressed stacked XD-Ensemble measured 54.98MB.

4.1.3 Model Sizes for Open Plant Phenotype Classification

Models used in the OPPD dataset classification followed the same model size trends measured in the two previous experiments. The ResNet was recorded at 240MB, the Xception was 97.31MB, followed by the Inception, VGG, and DenseNet at 92.27MB, 60.64MB, and 35.6MB respectively. The lightweight EfficientNet was 36.08MB and the proposed XD-Ensemble measured 35.01MB in HDF5. Table 9 presents a summary of the model size measurement.

4.3. Model Runtime Evaluation

4.3.1 Experimental Runtime - Plant Seedling Classification

The runtime benchmarks were generally dependent on the model rather than the platform. While the ResNet and VGG were noticeably faster on the cloud (608ms and 600ms) than on the android device (703ms and 1003ms), the Xception model which is the second-largest model in terms of its byte size was only, on average, a fraction of a millisecond faster on the cloud (291.81ms) than when it was ported to the mobile device (292.32ms). The Inception and DenseNet models demonstrated slightly better runtimes on the mobile device with runtimes of 182ms and 196ms, as opposed to the cloud benchmark results of 190ms and 253ms respectively. Meanwhile, the EfficientNet lightweight model was benchmarked at 75ms and 57ms in the cloud and the android environments respectively.

Comparatively, the XD-Ensemble demonstrated remarkably improved runtimes on the mobile device after model compression such that it was on average $\approx 8x$ faster (58ms) than the benchmark results of the uncompressed XD-Ensemble model on the cloud CPU (486ms) and about $\approx 4x$ faster than the fastest benchmark results of the mobile models. A summary of the results are presented in Table 10 below

Table 10. Runtime experiments – Plant Seedling dataset

Model	Platform	Avg (μ)	Max	Min	Std (σ)
DenseNet121	Colab	253.219	256.417	250.232	2.490
	Android	196.480	211.000	195.000	2.729
InceptionV3	Colab	190.936	191.883	189.719	0.768
	Android	182.900	194.000	181.000	1.803
ResNet152V2	Colab	608.244	612.116	598.582	4.914
	Android	703.340	708.000	702.000	1.518
VGG16	Colab	599.813	611.086	590.319	6.624
	Android	1003.102	1022.000	988.000	7.181
Xception	Colab	291.811	293.254	290.485	0.957
	Android	292.320	298.000	291.000	1.476
EfficientNet-B1	Colab	74.520	79.975	71.757	2.276
	Android	57.480	60.000	55.000	1.410
XD-Ensemble	Colab	486.226	489.196	483.387	2.307
	Android	57.860	61.000	57.000	0.775

4.3.2 Experimental Runtime - Leafsnap Classification

As shown in Table 11, the second set of benchmark results on the Leafsnap dataset was similar to the results achieved in the first set of experiments described above. The Xception, ResNet, and VGG were again faster on the cloud than on the mobile device with a benchmark result of 297ms, 611ms, and 600ms (on the cloud); and 298ms, 714ms, and 1007ms (in the android environment), respectively. The DenseNet and Inception were also benchmarked at 259ms and 200ms (on the cloud) and 199ms and 186ms (on the android device), respectively. Measured in both environments, the EfficientNet recorded 82ms and 65ms average runtimes.

The XD-Ensemble remained on average $\approx 8x$ faster (62ms) in the resource-constrained android environment than its full-sized version on the cloud (496ms). As compared to the first set of experiments, there was a slight degradation in the performance on this dataset but the model remained $\approx 3x$ faster than the fastest benchmark results of the mobile models.

Table 11. Runtime experiments – Leafsnap dataset

Model	Platform	Avg (μ)	Max	Min	Std (σ)
DenseNet121	Colab	259.239	260.603	256.676	1.476
	Android	198.660	202.000	197.000	1.088
InceptionV3	Colab	199.590	223.213	192.922	11.824
	Android	186.080	192.000	184.000	1.495
ResNet152V2	Colab	610.886	621.787	589.157	11.862
	Android	713.720	828.000	703.000	17.375
VGG16	Colab	599.699	601.234	597.644	1.186
	Android	1006.700	1019.000	992.000	5.991
Xception	Colab	297.295	301.453	288.982	4.405
	Android	298.200	306.000	296.000	1.887
EfficientNet-B1	Colab	82.486	92.610	75.748	4.980
	Android	65.420	71.000	66.000	1.783
XD-Ensemble	Colab	495.503	497.461	494.031	1.143
	Android	62.160	64.000	61.000	0.731

4.3.3 Experimental Runtime - OPPD Classification

The last set of experiments using the OPPD dataset remained in trend with the previous experiments. The Xception was benchmarked at 293ms on the cloud and 294ms on android, the ResNet was 602ms as against 795ms, and the VGG was again faster on the cloud at 598ms than

on android at 1097ms. The DenseNet had a runtime of 256ms on the cloud and 196ms on android, and Inception had 192ms and 185ms runtime on the cloud and android devices, respectively. Last, the lightweight EfficientNet recorded the following runtimes: 89ms in the cloud and 65ms in the android environment.

As expected, the XD-Ensemble was slower (486ms) in its full-sized version than with compression (60ms). This means the compressed model continued to outperform its full-sized version by $\approx 8x$ and was faster than the fastest model by $\approx 3x$. Table 12 summarizes this result.

Table 12. Runtime experiments – OPPD dataset

Model	Platform	Avg (μ)	Max	Min	Std (σ)
DenseNet121	Colab	255.865	257.678	252.118	1.953
	Android	196.460	200.000	195.000	0.964
InceptionV3	Colab	192.466	195.329	190.973	1.661
	Android	185.260	238.000	182.000	7.621
ResNet152V2	Colab	601.971	614.984	580.865	14.228
	Android	794.460	1064.000	707.000	131.197
VGG16	Colab	598.464	612.210	587.895	7.863
	Android	1097.200	1534.000	997.000	168.170
Xception	Colab	293.078	295.666	290.231	1.773
	Android	294.360	298.000	293.000	0.995
EfficientNet-B1	Colab	88.582	95.102	83.675	3.614
	Android	65.260	66.000	63.000	1.117
XD-Ensemble	Colab	486.126	489.143	481.528	2.496
	Android	60.240	62.000	59.000	0.618

4.4. Additional Evaluation

Figure 17 below presents a visual representation of how the proposed model compares to the other state-of-the-art models. It summarizes the relationship between the model accuracy and size. In this case, the runtime has been omitted since the results show that it correlates with model size. For an unbiased comparison, the results were normalized by rescaling the results of each metric to a range between 0 and 1 using equation 5 below:

$$\text{Normalization:} \quad y = (x - \min) / (\max - \min) \quad (5)$$

Where the minimum and maximum values relate to the value x being normalized. Figure 17 demonstrates that ResNet and VGG16 are not on the Pareto efficient frontier with respect to size and accuracy compared to EfficientNet, DenseNet, and XD-Ensemble. Additionally, the proposed model performed better than the others on one out of three datasets (the OPPD dataset, which is also the largest dataset). There was a tradeoff to be made regarding size and accuracy for the other two data sets. Overall, the results demonstrate that the proposed approach could be beneficial for agriculture and mobile inference.

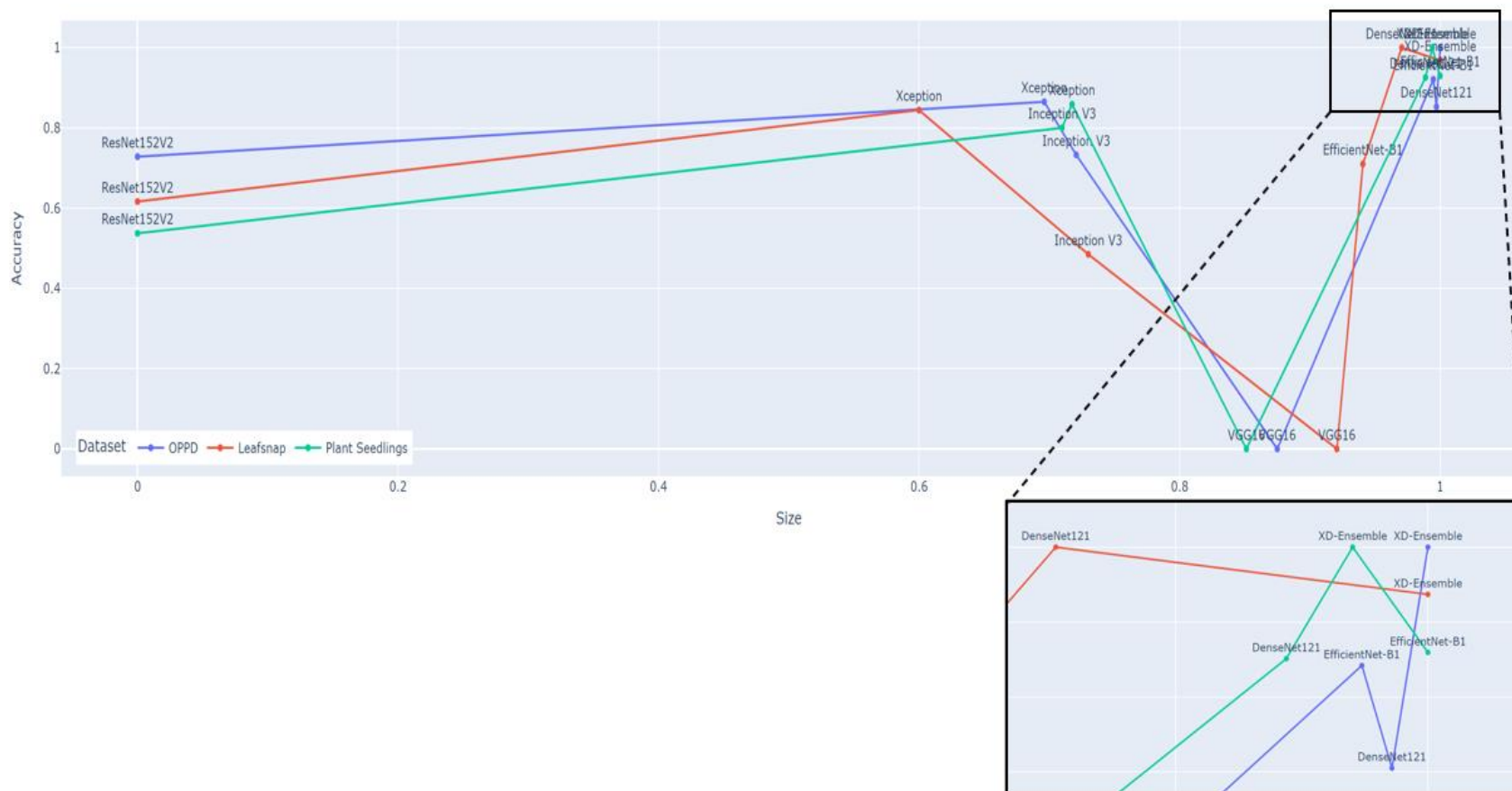


Figure 17. Comparing model accuracy vs compressed size (TFLite) per dataset

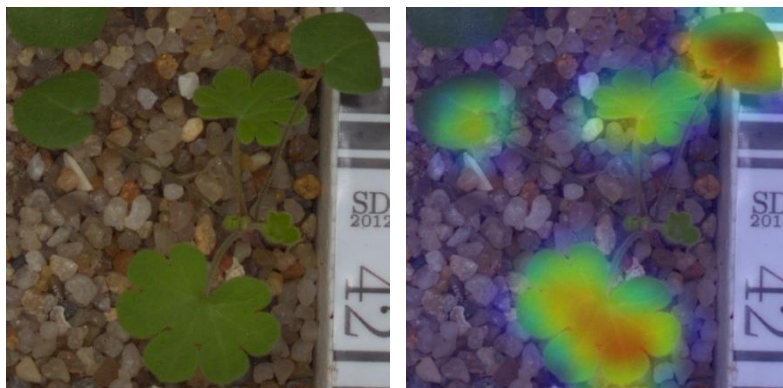
Two additional sets of experiments were conducted on the XD-Ensemble to demonstrate its validity and usability. The first set of experiments proved the robustness of the model by assessing a) the ratio of class predictions that truly belong to the class (Precision), the ratio of class accurate predictions out of the total samples in the dataset (Recall), and the balance between the two (f_1 -score). Table 13 summarizes this result. The raw result per class is presented in Appendix B.

Table 13. Precision, recall, and f_1 metrics for XD-Ensemble

Dataset	No of Images	No of Classes	Precision		Recall		f1-Score	
			Macro	Weighted	Macro	Weighted	Macro	Weighted
Plant Seedling	1015	12	0.9475	0.9480	0.9337	0.9488	0.9377	0.9461
Leafsnap	6161	185	0.9697	0.9675	0.9695	0.9664	0.969	0.9664
Open Plant Phenotyping	52836	47	0.9715	0.9707	0.9497	0.9706	0.9598	0.9705

The use of transfer learning with pre-trained models – models originally trained on the ImageNet dataset which contains general real-world images and are made up of varying shapes, colors, and hues as compared to plant datasets made up mainly of green plants and brown gravel background – there is a possibility that the XD-Ensemble be making predictions based on some leaked information in the datasets. To demonstrate that there was no negative transfer occurring in the models, the second set of experiments sought to visually investigate how the models were making predictions. The Gradient-weighted Class Activation Map (Grad-Cam), which uses the gradient of a target class on the final convolutional layer to produce a coarse localization map of important regions used in predicting the class to visualize and validate the areas in the images used by the model for prediction was used (Selvaraju et al., 2020). It was realized that the model was indeed identifying objects based on the correct patterns. For example, rather than identifying the background information, especially parts of the box or labels that leaked into the image dataset,

the model established its predictions on the plant leaves and stems. Figure 18 shows sample classes from all three datasets and the localization maps generated by Grad-CAM.



(a) Plant Seedling: Small-flowered cranesbill identified by the model based on leaf shape



(b) Leafsnap: *Platanus occidentalis* (sycamore) identified by the model based on leaf edges



(c) POLLA (pale smartweed) identified by the model based on clumps

Figure 18. Grad-CAM visualization XD-Ensemble indicating model localization of stem and leaves as important regions for prediction

CHAPTER 5

DISCUSSION

The current study proposed a method to decrease the complexity of DCNN models for ground-based plant classification systems. This chapter presents the underlying meaning of this research by delving into the results and presenting possible use-cases where the proposed approach could be beneficial.

5.1 Result Summary and Comparison with Past Research

While the ensuing discussion focuses mainly on the result from the prediction accuracy assessment for comparison with prior research, it also considers the Pareto efficient trade-off between accuracy and size/speed.

In the first case – the Plant Seedling dataset – the vanilla models achieved a prediction accuracy of 92.4% which was comparable to the results of prior studies that employed the same dataset (Alimboyong & Hernandez, 2019; Ofori & El-Gayar, 2020a, 2021; Rahman et al., 2020). As expected, compressing these state-of-the-art models resulted in about a 2% decrease in prediction accuracy. This was also similar to the result of a prior study using the VGG16 model – trained over 100 epochs on the Plant Seedling dataset with a 90% pruning ratio and post-training quantization applied (Fountsop et al., 2020). The XD-Ensemble, at 95% prediction accuracy, presented a performance that was better than the state-of-the-art models. In addition to the accuracy assessment, it is clear from Figure 17 that the XD-Ensemble and EfficientNet were on the Pareto efficient frontier for this dataset. In effect, while the results for the other models were inferior to these two, it was clear that a trade-off existed between accuracy and size such that the XD-

Ensemble was more accurate (and larger) while the EfficientNet was smaller in size (and less accurate).

The Leafsnap dataset which consisted of over 60,000 weed classes also resulted in a similar outcome. The average prediction accuracy of 94.8% was comparable to prior research (Bodhwani et al., 2019; Ibrahim et al., 2018; Kala & Viriri, 2018; Pawara et al., 2017). The XD-Ensemble achieved an accuracy of 96% in response to a 1.5% drop in average prediction accuracy of the pruned-vanilla models. The DenseNet had a marginally better result of 96.05%. Further, according to Figure 17, the XD-Ensemble and DenseNet were on the Pareto efficient frontier for this dataset.

On the OPPD dataset, there were not enough studies to compare with except for the original study by Leminen Madsen et al. (2020) where the dataset was proposed. In that study, model accuracy was 77%. The state-of-the-art models employed in the current study demonstrated a performance baseline of 95.7%. On this dataset, model compression resulted in models with about a 1.8% lower prediction accuracy. The XD-Ensemble approach achieved the best overall average accuracy of 97%. Additionally, the XD-Ensemble was the Pareto optimal model when both accuracy and size metrics were considered.

By examining the Pareto efficient frontier, it is evident that the XD-Ensemble performed particularly well on tasks involving the two largest datasets - the Leafsnap and OPPD datasets. The increased number of classes and training image samples likely combined to provide marginal improvements in the result on these two datasets as compared to the state-of-the-art models. This demonstrates that the combination of three cutting-edge DCNN techniques – transfer learning, model compression, and ensemble learning – could be valuable to inference on the edge. Especially because each technique is engineered to deliver benefits that can enhance and underpin the generalizability of the result. Transfer learning, where a model trained on one task can be ported

to another task, offers opportunities for reducing overfitting and ensuring robust results in the face of limited training data (Bengio, 2012; Ofori & El-Gayar, 2021; Pan & Yang, 2010). Model compression offers additional benefits for reducing the size of the DL models, which means an equivalent reduction in both energy consumption and inference time (Choudhary et al., 2020; Fountsop et al., 2020; S. Han et al., 2016; Zhu & Gupta, 2017). Last, ensemble learning improves classification performance by combining two architecturally different DL models into a single model to improve prediction accuracy (Rokach, 2019; Ting & Witten, 1999; Torres-Sospedra & Nebot, 2014; Wolpert, 1992).

5.2 Implications for Agricultural Practice

Data in agriculture, as with other industries, is relevant now more than ever. This is because, years of research have gone into developing many types of sensors for recording agronomically relevant parameters, collecting enough data through the IoT devices and sensor networks, developing farm management systems, devising AI-driven farm machinery, and more. It is believed that there are over 75 million agricultural IoT devices currently in use; by 2050 an average farm could be generating up to 4.1 million data points daily as compared to 190,000 data points generated in 2014 (Clercq et al., 2018). These IoT and Cloud computing systems are generating and collecting unprecedented amounts of useful *big data* (Clercq et al., 2018; El-Gayar & Ofori, 2020).

Despite the apparent usefulness of data, there are several cross-cutting issues regarding cost and complexity, especially for smallholder farmers in developing countries (Misaki et al., 2018; Saidu et al., 2017). With the massive amounts of data being captured, most of which are fluid and hierarchically distributed, it is required that analysis of the data take as little time as possible and give better profit margins as compared to human-made management decisions. Thus,

capturing and making real-time decisions could mean moving decision points from servers and cloud services to IoT and mobile devices. Research, such as the current one, leads efforts aimed at faster and more efficient methods of retrieving some value from data such as farm imagery. It steers the conversation from the drawbacks of DL (for example the need for large amounts of data, longer training times, and expensive computers) to inference implemented directly on cheaper IoT and mobile devices. Continuous and significant improvements to speed up DL techniques and reduce their resource needs have implications for yield improvements and competitive pricing (El-Gayar & Ofori, 2020; Kernecker et al., 2020).

5.3 Additional Implications and Hypothetical Use-Cases for the Future

In principle, given relevant data and parameters to learn from, machines can perform most tasks with the help of AI and ML. The future of AI for farming lies in its very definition, that given any situation will act as a rational agent to take the best action possible (Russell & Norvig, 2010). Mobile-sized and lightweight DL applications present several interesting scenarios for PA and Green IS. The findings of the dissertation demonstrate that implementing DCNNs on the edge does not have to be computationally expensive: only a few MBs of memory and a split second are needed for inference. This section explores some interesting hypotheticals where the current study could be beneficial.

5.3.1 Pre-Production: Seed sorting

Fresh seeds for planting often contain impurities such as leaves, branches, and stones that need to be separated but current machinery can find this process difficult especially at faster sorting velocities (Buus et al., 2011; Kaliniewicz et al., 2021). Due to the characteristics of seeds themselves (for example seed density, surface texture, and shape), a solution that might work for certain seeds might not generalize well to others. Delegating the image analysis process to a

lightweight DL model such as the XD-Ensemble could increase both prediction accuracy and inference speed for identifying optimal seed varieties for planting.

5.3.2 Production: Diseases and pest identification

The current research has focused on weed detection. However, studies have shown that DL models are also effective for plant disease and pest identification (Liu & Chahl, 2021; Suresh et al., 2019; Too et al., 2019). Suresh et al., (2019) discussed the need for reducing the number of parameters, as well as training time using methods like TL. This could allow for instant detection of crop diseases and pest infestation. The proposed model serves this need while still outperforming the models employed in prior studies, albeit on different datasets. On a larger scale, additional data from satellite imagery could corroborate the onset of disease such as blight, as well as swarms of insects infesting nearby farms even before agronomists perform farm inspections.

5.3.3 Post-Production: Harvesting and Distribution

There are use-cases for improving the robotic harvesting process using embedded devices (Horng et al., 2020). Additionally, there is an apparent lack of transparency regarding data ownership and privacy between technology providers and farmers which has led to perception issues that fuel farmers' reluctance to wholly embrace PA (Sykuta, 2016; Wiseman et al., 2019; Wolfert et al., 2017). Lightweight models, such as the XD-Ensemble, present an opportunity for further advancing privacy-preserving for increased inference on the edge (Lalapura et al., 2021; Lammie et al., 2019; Voghoei et al., 2018). Further, mobile-based DL applications could even go beyond computer vision applications to provide optimal pricing and distribution prediction with data that never leaves the farmers' device (Zhao, 2021).

5.3.4 Applications beyond farming

The onset of the COVID-19 pandemic has introduced new ways of relating and an added reliance on technology to maintain social distancing. Further, mobile applications have become ubiquitous in current contexts and present several opportunities for DL and DCNN tasks beyond agricultural settings. Botanical gardens, arboretums, and even amateur bird watchers could increase foot traffic for self-guided tours using virtual plant and animal species classification applications. Augmented reality has also gained popularity in recent times. Applications such as Snapchat, Facebook, and Instagram implement innovative *smart camera* features for entertainment value. Such features implemented on the edge presents opportunities to improve prediction time by reducing data usage bandwidth. Additionally, edge processing preserves the privacy of the user since no image or video data will be sent to application servers for processing. These are just two examples of an unlimited number of use-cases where efficient DL mobile applications could be beneficial.

CHAPTER 6

CONCLUSION

This chapter presents a summary of the dissertation. The limitations encountered in the study and propositions for future research are outlined.

6.1 General Summary

The race for better chemical agents with higher biodegradability and lower environmental persistence continues unabated. Computer vision equipment used in SSWM should be able to capture images and distinguish between food crops and weeds quickly and efficiently, especially at the onset of plant growth, where lax weed control could result in up to 100% yield loss. As such, and given the success of DCNNs, ensuring their applicability to farming scenarios will represent a huge milestone for Precision Agriculture and Green IS. However, the drawback of DL and other machine learning tasks is in their requirement for huge amounts of data for training which has a direct impact on both energy consumption and computing power of the infrastructure involved.

This study proposed a DCNN approach for plant seedling classification and weed detection using a set of techniques for reducing the hardware requirements of resource-constrained systems while keeping accuracy at par with full-sized state-of-the-art DCNNs. The approach employed three stages to devise a sparse network: transfer learning, model compression; and a model stacking ensemble to determine the appropriate combination of model weights that deliver the best accuracy. The proposed XD-Ensemble outperformed the baseline average prediction accuracy of state-of-the-art models on all three datasets.

6.2 Principal Findings

At the start of the current study, the specific objectives set were to reduce computational cost and increase the efficiency of DL models in low-resource conditions, to introduce a new DL architecture, and to evaluate this new model against well-established DCNN models. These objectives were motivated by the need for robust AI applications to support sustainable development. Thus, in the face of resource-constrained agricultural devices, the current study introduced a novel concept for plant recognition tasks that exploited existing DL techniques such as transfer learning; model compression; and ensemble learning.

Often, prediction accuracy, and a few other metrics, is the standard evaluation method for DL models, but for IoT, mobile devices, and embedded systems inference speed, model size, and power efficiency is a challenge that requires innovative solutions. However, the literature points to a relationship between model accuracy and model compression, such that pruning models could result in decreased accuracy (Fountsop et al., 2020; Rajaraman et al., 2020). Theoretically, the approach proposed in this dissertation aimed to demonstrate that combining transfer learning and ensemble learning could resolve the performance degradation associated with model compression. The results indicated that the proposed XD-Ensemble delivered a performance at a high level, on par or better than the state-of-the-art, and also outperformed the lightweight EfficientNet-B1 model with regards to prediction runtime and model accuracy. Specifically, in each of the pseudo-objectives employed in Chapter 4, the model achieved 1) better accuracy than both the state-of-the-art and lightweight models in two out of three datasets and was only marginally *out-predicted* by the DenseNet on one dataset, 2) smaller model size as compared to the vanilla models after compression; except for the first dataset where the lightweight EfficientNet was smaller by 1MB,

and 3) faster inference time than all state-of-the-art models in the resource-constrained android device; only the EfficientNet model recorded a faster prediction runtime on one dataset.

Thus, the approach presented in the current study will be beneficial to ground-based weed detection equipment used for SSWM systems and contribute to minimizing the environmental footprint of AITs while maximizing production efficiency. Model compression and ensemble methods have been applied independently in several past studies. This research proves that they can be combined to a great effect. Although the context for the research is limited to computer vision use cases in PA and Green IS, the proposed method could be applied to similar computer vision tasks, as well as machine learning systems that employ other forms of data, in resource-constrained environments commonly encountered in other industries such as healthcare and telecommunication.

6.3 Study Limitations and Recommendations for Future Researchers

With regards to the use of transfer learning, the number of layers to freeze for each model is a hyperparameter that can be tuned. Although the dissertation leaned on past work that experimented with this particular parameter, it is still a potential limitation since other researchers could achieve varying results depending on which layers are chosen for fine-tuning.

Additionally, the research has used different plant species captured both in-field and lab settings to demonstrate the robustness of the proposed DCNN model but further analysis using different forms of data such as satellite imagery, video streams, as well as numerical and time-series data should be investigated. In addition, evaluation on different types of resource-constrained devices such as embedded systems and micro-controllers could prove useful. Further, an analysis of the lifetime cost and energy savings of employing this approach has not been measured and could warrant further investigation.

REFERENCES

- Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., ... Zheng, X. (2016). TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *ArXiv:1603.04467 [Cs]*. <http://arxiv.org/abs/1603.04467>
- Agrawal, P., Girshick, R., & Malik, J. (2014). Analyzing the Performance of Multilayer Neural Networks for Object Recognition. *ArXiv:1407.1610 [Cs]*. <http://arxiv.org/abs/1407.1610>
- Ahmed, F., Al-Mamun, H. A., Bari, A. S. M. H., Hossain, E., & Kwan, P. (2012). Classification of crops and weeds from digital images: A support vector machine approach. *CROP PROTECTION*, *40*, 98–104. <https://doi.org/10.1016/j.cropro.2012.04.024>
- Akbarzadeh, S., Paap, A., Ahderom, S., Apopei, B., & Alameh, K. (2018). Plant discrimination by Support Vector Machine classifier based on spectral reflectance. *COMPUTERS AND ELECTRONICS IN AGRICULTURE*, *148*, 250–258. <https://doi.org/10.1016/j.compag.2018.03.026>
- Alimboyong, C. R., & Hernandez, A. A. (2019). An Improved Deep Neural Network for Classification of Plant Seedling Images. *2019 IEEE 15th International Colloquium on Signal Processing Its Applications (CSPA)*, 217–222. <https://doi.org/10.1109/CSPA.2019.8696009>
- Alpaydin, E. (2004). *Introduction to Machine Learning*. The MIT Press.
- Andrea, C.-C., Mauricio Daniel, B. B., & Jose Misael, J. B. (2017). Precise weed and maize classification through convolutional neuronal networks. *2017 IEEE Second Ecuador Technical Chapters Meeting (ETCM)*, 1–6. <https://doi.org/10.1109/ETCM.2017.8247469>

- Ashqar, B. A. M., Abu-Nasser, B. S., & Abu-Naser, S. S. (2019). *Plant Seedlings Classification Using Deep Learning*. 3(1), 8.
- Aubert, B. A., Schroeder, A., & Grimaudo, J. (2012). IT as enabler of sustainable farming: An empirical analysis of farmers' adoption decision of precision agriculture technology. *Decision Support Systems*, 54(1), 510–520. <https://doi.org/10.1016/j.dss.2012.07.002>
- Bakhshipour, A., & Jafari, A. (2018). Evaluation of support vector machine and artificial neural networks in weed detection using shape features. *COMPUTERS AND ELECTRONICS IN AGRICULTURE*, 145, 153–160. <https://doi.org/10.1016/j.compag.2017.12.032>
- Bakhshipour, A., Jafari, A., Nassiri, S. M., & Zare, D. (2017). Weed segmentation using texture features extracted from wavelet sub-images. *BIOSYSTEMS ENGINEERING*, 157, 1–12. <https://doi.org/10.1016/j.biosystemseng.2017.02.002>
- Balafoutis, A. T., Beck, B., Fountas, S., Tsiropoulos, Z., Vangeyte, J., van der Wal, T., Soto-Embodas, I., Gómez-Barbero, M., & Pedersen, S. M. (2017). Smart Farming Technologies – Description, Taxonomy and Economic Impact. In S. M. Pedersen & K. M. Lind (Eds.), *Precision Agriculture: Technology and Economic Perspectives* (pp. 21–77). Springer International Publishing. https://doi.org/10.1007/978-3-319-68715-5_2
- Bastiaans, L., Paolini, R., & Baumann, D. T. (2008). Focus on ecological weed management: What is hindering adoption? *Weed Research*, 48(6), 481–491. <https://doi.org/10.1111/j.1365-3180.2008.00662.x>
- Bengio, Y. (2009). Learning Deep Architectures for AI. *Foundations and Trends® in Machine Learning*, 2(1), 1–127. <https://doi.org/10.1561/22000000006>
- Bengio, Y. (2012). *Deep Learning of Representations for Unsupervised and Transfer Learning*. 21.

- Bengio, Y. (2013). Deep Learning of Representations: Looking Forward. In A.-H. Dediu, C. Martín-Vide, R. Mitkov, & B. Truthe (Eds.), *Statistical Language and Speech Processing* (Vol. 7978, pp. 1–37). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-39593-2_1
- Binch, A., & Fox, C. W. (2017). Controlled comparison of machine vision algorithms for Rumex and Urtica detection in grassland. *Computers and Electronics in Agriculture*, *140*, 123–138. <https://doi.org/10.1016/j.compag.2017.05.018>
- Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.
- Blakey, P., Atkins, C., & Crump, B. (2008). *Using Design Research to Improve Data Modelling Performance among Novice End Users*. 10.
- Bodhwani, V., Acharjya, D. P., & Bodhwani, U. (2019). Deep Residual Networks for Plant Identification. *Procedia Computer Science*, *152*, 186–194. <https://doi.org/10.1016/j.procs.2019.05.042>
- Bongiovanni, R., & Lowenberg-Deboer, J. (2004). Precision Agriculture and Sustainability. *Precision Agriculture*, *5*(4), 359–387. <https://doi.org/10.1023/B:PRAG.0000040806.39604.aa>
- Buus, O. T., Jørgensen, J. R., & Carstensen, J. M. (2011). Analysis of Seed Sorting Process by Estimation of Seed Motion Trajectories. In A. Heyden & F. Kahl (Eds.), *Image Analysis* (Vol. 6688, pp. 273–284). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-21227-7_26
- Cai, S., Bileschi, S., & Nielsen, E. (2020). *Deep Learning with JavaScript: Neural networks in TensorFlow.js* (1st edition). Manning Publications.

- Chantre, G. R., Vigna, M. R., Renzi, J. P., & Blanco, A. M. (2018). A flexible and practical approach for real-time weed emergence prediction based on Artificial Neural Networks. *BIOSYSTEMS ENGINEERING*, *170*, 51–60. <https://doi.org/10.1016/j.biosystemseng.2018.03.014>
- Chavan, T. R., & Nandedkar, A. V. (2018). AgroAVNET for crops and weeds classification: A step forward in automatic farming. *Computers and Electronics in Agriculture*, *154*, 361–372. <https://doi.org/10.1016/j.compag.2018.09.021>
- Chen, Y., Zheng, B., Zhang, Z., Wang, Q., Shen, C., & Zhang, Q. (2020). Deep Learning on Mobile and Embedded Devices: State-of-the-art, Challenges, and Future Directions. *ACM Computing Surveys*, *53*(4), 1–37. <https://doi.org/10.1145/3398209>
- Chollet, F. (2017). Xception: Deep Learning with Depthwise Separable Convolutions. *ArXiv:1610.02357 [Cs]*. <http://arxiv.org/abs/1610.02357>
- Chollet, F. & others. (2015). *Keras*. <https://keras.io>
- Choudhary, T., Mishra, V., Goswami, A., & Sarangapani, J. (2020). A comprehensive survey on model compression and acceleration. *Artificial Intelligence Review*, *53*(7), 5113–5155. <https://doi.org/10.1007/s10462-020-09816-7>
- Clercq, M. D., Vats, A., & Biel, A. (2018). Agriculture 4.0: The Future of Farming Technology. *World Government Summit*, 30.
- Cockburn, I., Henderson, R., & Stern, S. (2018). *The Impact of Artificial Intelligence on Innovation* (No. w24449). National Bureau of Economic Research. <https://doi.org/10.3386/w24449>
- Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, *1*, 886–893 vol. 1. <https://doi.org/10.1109/CVPR.2005.177>

- Dedrick, J. (2010). Green IS: Concepts and Issues for Information Systems Research. *Communications of the Association for Information Systems*, 27(1). <https://doi.org/10.17705/1CAIS.02711>
- Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., & Darrell, T. (2013). DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition. *ArXiv:1310.1531 [Cs]*. <http://arxiv.org/abs/1310.1531>
- dos Santos Ferreira, A., Matte Freitas, D., Gonçalves da Silva, G., Pistori, H., & Theophilo Folhes, M. (2017). Weed detection in soybean crops using ConvNets. *Computers and Electronics in Agriculture*, 143, 314–324. <https://doi.org/10.1016/j.compag.2017.10.027>
- Dyrmann, M., Karstoft, H., & Midtiby, H. S. (2016). Plant species classification using deep convolutional neural network. *Biosystems Engineering*, 151, 72–80. <https://doi.org/10.1016/j.biosystemseng.2016.08.024>
- El-Gayar, O., & Ofori, M. (2020). Disrupting Agriculture: The Status and Prospects for AI and Big Data in Smart Agriculture. In M. Strydom & S. Buckley (Eds.), *AI and Big Data's Potential for Disruptive Innovation*. IGI Global. <https://doi.org/10.4018/978-1-5225-9687-5.ch007>
- Farooq, A., Hu, J., & Jia, X. (2019). Analysis of Spectral Bands and Spatial Resolutions for Weed Classification Via Deep Convolutional Neural Network. *IEEE Geoscience and Remote Sensing Letters*, 16(2), 183–187. <https://doi.org/10.1109/LGRS.2018.2869879>
- Fawakherji, M., Youssef, A., Bloisi, D., Pretto, A., & Nardi, D. (2019). Crop and Weeds Classification for Precision Agriculture Using Context-Independent Pixel-Wise Segmentation. *2019 Third IEEE International Conference on Robotic Computing (IRC)*, 146–152. <https://doi.org/10.1109/IRC.2019.00029>

- Fountsop, A. N., Ebongue Kedieng Fendji, J. L., & Atemkeng, M. (2020). Deep Learning Models Compression for Agricultural Plants. *Applied Sciences*, *10*(19), 6866. <https://doi.org/10.3390/app10196866>
- Furtado, R., Azevedo, E., & Motheo, A. (2019). Electrochemical degradation of aqueous alachlor and atrazine: Products identification, lipophilicity, and ecotoxicity. *Eclética Química Journal*, *44*, 12. <https://doi.org/10.26850/1678-4618eqj.v44.1SI.2019.p12-25>
- Gerhards, R. (2010). Spatial and Temporal Dynamics of Weed Populations. In E.-C. Oerke, R. Gerhards, G. Menz, & R. A. Sikora (Eds.), *Precision Crop Protection—The Challenge and Use of Heterogeneity* (pp. 17–25). Springer Netherlands. https://doi.org/10.1007/978-90-481-9277-9_2
- Gianessi, L. P. (2009). Solving Africa's weed problem: Increasing crop production & improving the lives of women. *Aspects of Applied Biology*, *No.96*, 9–23.
- Gikunda, P. K., & Jouandeu, N. (2019). State-of-the-Art Convolutional Neural Networks for Smart Farms: A Review. In K. Arai, R. Bhatia, & S. Kapoor (Eds.), *Intelligent Computing* (Vol. 997, pp. 763–775). Springer International Publishing. https://doi.org/10.1007/978-3-030-22871-2_53
- Giselsson, T. M., Jørgensen, R. N., Jensen, P. K., Dyrmann, M., & Midtiby, H. S. (2017). A Public Image Database for Benchmark of Plant Seedling Classification Algorithms. *ArXiv:1711.05458 [Cs]*. <http://arxiv.org/abs/1711.05458>
- Goel, P. K., Prasher, S. O., Patel, R. M., Smith, D. L., & DiTommaso, A. (2002). Use of Airborne Multi-Spectral Imagery for Weed Detection in Field Crops. *TRANSACTIONS OF THE ASAE*, *45*, 9.

- Golzarian, M., & Frick, R. (2011). Classification of images of wheat, ryegrass and brome grass species at early growth stages using principal component analysis. *Plant Methods*, 7, 28. <https://doi.org/10.1186/1746-4811-7-28>
- Gomez-Casero, M. T., Castillejo-Gonzalez, I. L., Garcia-Ferrer, A., Pena-Barragan, J. M., Jurado-Exposito, M., Garcia-Torres, L., & Lopez-Granados, F. (2010). Spectral discrimination of wild oat and canary grass in wheat fields for less herbicide application. *AGRONOMY FOR SUSTAINABLE DEVELOPMENT*, 30(3), 689–699. <https://doi.org/10.1051/agro/2009052>
- Gutierrez, P. A., Lopez-Granados, F., Pena-Barragan, J. M., Jurado-Exposito, M., & Hervás-Martínez, C. (2008). Logistic regression product-unit neural networks for mapping *Ridolfia segetum* infestations in sunflower crop using multitemporal remote sensed data. *COMPUTERS AND ELECTRONICS IN AGRICULTURE*, 64(2), 293–306. <https://doi.org/10.1016/j.compag.2008.06.001>
- Han, S., Mao, H., & Dally, W. J. (2016). *Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding*. 14.
- Han, X., Zhong, Y., Cao, L., & Zhang, L. (2017). Pre-Trained AlexNet Architecture with Pyramid Pooling and Supervision for High Spatial Resolution Remote Sensing Image Scene Classification. *Remote Sensing*, 9(8), 848. <https://doi.org/10.3390/rs9080848>
- He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition. *ArXiv:1512.03385 [Cs]*. <http://arxiv.org/abs/1512.03385>
- Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). *Design Science in Information Systems Research*. 32.

- Hornig, G., Liu, M., & Chen, C. (2020). The Smart Image Recognition Mechanism for Crop Harvesting System in Intelligent Agriculture. *IEEE Sensors Journal*, 20(5), 2766–2781. <https://doi.org/10.1109/JSEN.2019.2954287>
- Huang, G., Liu, Z., van der Maaten, L., & Weinberger, K. Q. (2018). Densely Connected Convolutional Networks. *ArXiv:1608.06993 [Cs]*. <http://arxiv.org/abs/1608.06993>
- Ibrahim, Z., Sabri, N., & Isa, D. (2018). Multi-maxpooling Convolutional Neural Network for Medicinal Herb Leaf Recognition. *Proceedings of The 6th IIAE International Conference on Intelligent Systems and Image Processing 2018*, 327–331. <https://doi.org/10.12792/icisip2018.060>
- Kala, J. R., & Viriri, S. (2018). Plant Specie Classification using Sinuosity Coefficients of Leaves. *Image Analysis & Stereology*, 37(2), 119. <https://doi.org/10.5566/ias.1821>
- Kaliniewicz, Z., Anders, A., Markowski, P., Tylek, P., & Owoc, D. (2021). Analysis of the physical properties of spindle seeds for seed sorting operations. *Scientific Reports*, 11(1), 13625. <https://doi.org/10.1038/s41598-021-93166-z>
- Kamilaris, A., & Prenafeta-Boldú, F. X. (2018). Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture*, 147, 70–90. <https://doi.org/10.1016/j.compag.2018.02.016>
- Kernecker, M., Knierim, A., Wurbs, A., Kraus, T., & Borges, F. (2020). Experience versus expectation: Farmers’ perceptions of smart farming technologies for cropping systems across Europe. *Precision Agriculture*, 21, 34–50. <https://doi.org/10.1007/s11119-019-09651-z>
- Khan, A., Sohail, A., Zahoor, U., & Saeed, A. (2019). A Survey of the Recent Architectures of Deep Convolutional Neural Networks. *Artificial Intelligence Review*. <https://doi.org/10.1007/s10462-020-09825-6>

- Kim, Y.-D., Park, E., Yoo, S., Choi, T., Yang, L., & Shin, D. (2016). *Compression of deep convolutional neural networks for fast and low power mobile applications*. 16.
- Knoll, F. J., Czymmek, V., Poczihoski, S., Holtorf, T., & Hussmann, S. (2018). Improving efficiency of organic farming by using a deep learning classification approach. *Computers and Electronics in Agriculture*, 153, 347–356.
<https://doi.org/10.1016/j.compag.2018.08.032>
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In F. Pereira, C. J. C. Burges, L. Bottou, & K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems 25* (pp. 1097–1105). Curran Associates, Inc. <http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf>
- Kumar, N., Belhumeur, P. N., Biswas, A., Jacobs, D. W., Kress, W. J., Lopez, I. C., & Soares, J. V. B. (2012). Leafsnap: A Computer Vision System for Automatic Plant Species Identification. In A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, & C. Schmid (Eds.), *Computer Vision – ECCV 2012* (Vol. 7573, pp. 502–516). Springer Berlin Heidelberg.
https://doi.org/10.1007/978-3-642-33709-3_36
- Lakshmi, V., & Corbett, J. (2020, January 7). *How Artificial Intelligence Improves Agricultural Productivity and Sustainability: A Global Thematic Analysis*.
<https://doi.org/10.24251/HICSS.2020.639>
- Lalapura, V. S., Amudha, J., & Satheesh, H. S. (2021). Recurrent Neural Networks for Edge Intelligence: A Survey. *ACM Computing Surveys*, 54(4), 1–38.
<https://doi.org/10.1145/3448974>

- Lammie, C., Olsen, A., Carrick, T., & Rahimi Azghadi, M. (2019). Low-Power and High-Speed Deep FPGA Inference Engines for Weed Classification at the Edge. *IEEE Access*. <https://doi.org/10.1109/ACCESS.2019.2911709>
- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. <https://doi.org/10.1038/nature14539>
- Lee, S. H., Chan, C. S., Wilkin, P., & Remagnino, P. (2015). Deep-plant: Plant identification with convolutional neural networks. *2015 IEEE International Conference on Image Processing (ICIP)*, 452–456. <https://doi.org/10.1109/ICIP.2015.7350839>
- Leminen Madsen, S., Mathiassen, S. K., Dyrmann, M., Laursen, M. S., Paz, L.-C., & Jørgensen, R. N. (2020). Open Plant Phenotype Database of Common Weeds in Denmark. *Remote Sensing*, *12*(8), 1246. <https://doi.org/10.3390/rs12081246>
- Liakos, K., Busato, P., Moshou, D., Pearson, S., & Bochtis, D. (2018). Machine Learning in Agriculture: A Review. *Sensors*, *18*(8), 2674. <https://doi.org/10.3390/s18082674>
- Liu, H., & Chahl, J. S. (2021). Proximal detecting invertebrate pests on crops using a deep residual convolutional neural network trained by virtual images. *Artificial Intelligence in Agriculture*, *5*, 13–23. <https://doi.org/10.1016/j.aiia.2021.01.003>
- López-Granados, F. (2011). Weed detection for site-specific weed management: Mapping and real-time approaches. *Weed Research*, *51*, 1–11. <https://doi.org/10.1111/j.1365-3180.2010.00829.x>
- Lowenberg-DeBoer, J., & Erickson, B. (2019). Setting the Record Straight on Precision Agriculture Adoption. *Agronomy Journal*, *111*(4), 1552. <https://doi.org/10.2134/agronj2018.12.0779>

- Lowenberg-DeBoer, J., Huang, I. Y., Grigoriadis, V., & Blackmore, S. (2019). Economics of robots and automation in field crop production. *Precision Agriculture*. <https://doi.org/10.1007/s11119-019-09667-5>
- Lowenberg-DeBoer, J., & Swinton, S. M. (1997). Economics of Site-Specific Management in Agronomic Crops. In F. J. Pierce & E. J. Sadler (Eds.), *ASA, CSSA, and SSSA Books* (pp. 369–396). American Society of Agronomy, Crop Science Society of America, Soil Science Society of America. <https://doi.org/10.2134/1997.stateofsitespecific.c16>
- March, S. T., & Smith, G. F. (1995). Design and natural science research on information technology. *Decision Support Systems*, *15*(4), 251–266. [https://doi.org/10.1016/0167-9236\(94\)00041-2](https://doi.org/10.1016/0167-9236(94)00041-2)
- Martin, K. (2016, April 27). *How will artificial intelligence affect legal practice?* | Thomson Reuters. Answers On. <https://blogs.thomsonreuters.com/answerson/artificial-intelligence-legal-practice/>
- McCool, C., Perez, T., & Upcroft, B. (2017). Mixtures of Lightweight Deep Convolutional Neural Networks: Applied to Agricultural Robotics. *IEEE Robotics and Automation Letters*, *2*(3), 1344–1351. <https://doi.org/10.1109/LRA.2017.2667039>
- Milioto, A., Lottes, P., & Stachniss, C. (2017). Real-Time Blob-Wise Sugar Beets Vs Weeds Classification for Monitoring Fields Using Convolutional Neural Networks. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, *IV-2/W3*, 41–48. <https://doi.org/10.5194/isprs-annals-IV-2-W3-41-2017>
- Misaki, E., Apiola, M., Gaiani, S., & Tedre, M. (2018). Challenges facing sub-Saharan small-scale farmers in accessing farming information through mobile phones: A systematic literature

- review. *The Electronic Journal of Information Systems in Developing Countries*, 84(4), e12034. <https://doi.org/10.1002/isd2.12034>
- Moazzam, S. I., Khan, U. S., Tiwana, M. I., Iqbal, J., Qureshi, W. S., & Shah, S. I. (2019). A Review of Application of Deep Learning for Weeds and Crops Classification in Agriculture. *2019 International Conference on Robotics and Automation in Industry (ICRAI)*, 1–6. <https://doi.org/10.1109/ICRAI47710.2019.8967350>
- Mohri, M., Rostamizadeh, A., & Talwalkar, A. (2018). *Foundations of machine learning* (Second edition). The MIT Press.
- Molchanov, P., Tyree, S., Karras, T., Aila, T., & Kautz, J. (2016). *Pruning Convolutional Neural Networks for Resource Efficient Inference*. <https://arxiv.org/abs/1611.06440v2>
- Moura, M. A. M., Oliveira, R., Jonsson, C. M., Domingues, I., Soares, A. M. V. M., & Nogueira, A. J. A. (2018). The sugarcane herbicide ametryn induces oxidative stress and developmental abnormalities in zebrafish embryos. *Environmental Science and Pollution Research*, 25(14), 13416–13425. <https://doi.org/10.1007/s11356-017-9614-0>
- Najafabadi, M. M., Villanustre, F., Khoshgoftaar, T. M., Seliya, N., Wald, R., & Muharemagic, E. (2015). Deep learning applications and challenges in big data analytics. *Journal of Big Data*, 2(1), 1. <https://doi.org/10.1186/s40537-014-0007-7>
- Ofori, M., & El-Gayar, O. (2020a). Towards Deep Learning for Weed Detection: Deep Convolutional Neural Network Architectures for Plant Seedling Classification. *AMCIS 2020 Proceedings*. https://aisel.aisnet.org/amcis2020/sig_green/sig_green/3
- Ofori, M., & El-Gayar, O. (2020b). Drivers and challenges of precision agriculture: A social media perspective. *Precision Agriculture*. <https://doi.org/10.1007/s11119-020-09760-0>

- Ofori, M., & El-Gayar, O. (2021). An Approach for Weed Detection Using CNNs And Transfer Learning. *Hawaii International Conference on System Sciences*, 10.
- Olson, K. D. (1998). *Precision Agriculture: Current Economic And Environmental Issues*. 1685-2016–137073, 10. <https://doi.org/10.22004/ag.econ.14487>
- Pallottino, F., Menesatti, P., Figorilli, S., Antonucci, F., Tomasone, R., Colantoni, A., & Costa, C. (2018). Machine Vision Retrofit System for Mechanical Weed Control in Precision Agriculture Applications. *Sustainability*, 10(7), 2209. <https://doi.org/10.3390/su10072209>
- Pan, S. J., & Yang, Q. (2010). A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10), 1345–1359. <https://doi.org/10.1109/TKDE.2009.191>
- Pantazi, X. E., Moshou, D., & Bravo, C. (2016). Active learning system for weed species recognition based on hyperspectral sensing. *Biosystems Engineering*, 146, 193–202. <https://doi.org/10.1016/j.biosystemseng.2016.01.014>
- Pantazi, X. E., Tamouridou, A. A., Alexandridis, T. K., Lagopodi, A. L., Kashefi, J., & Moshou, D. (2017). Evaluation of hierarchical self-organising maps for weed mapping using UAS multispectral imagery. *Computers and Electronics in Agriculture*, 139, 224–230. <https://doi.org/10.1016/j.compag.2017.05.026>
- Patel, T. U., Vihol, K. J., Thanki, J. D., Gudaghe, N. N., & Desai, L. J. (2018). Weed and nitrogen management in direct-seeded rice. *Indian Journal of Weed Science*, 50(4), 320. <https://doi.org/10.5958/0974-8164.2018.00069.2>
- Pawara, P., Okafor, E., Surinta, O., Schomaker, L., & Wiering, M. (2017). Comparing Local Descriptors and Bags of Visual Words to Deep Convolutional Neural Networks for Plant Recognition: *Proceedings of the 6th International Conference on Pattern Recognition Applications and Methods*, 479–486. <https://doi.org/10.5220/0006196204790486>

- Peffer, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, 24(3), 45–77. <https://doi.org/10.2753/MIS0742-1222240302>
- Pierce, F. J., & Nowak, P. (1999). Aspects of Precision Agriculture. In D. L. Sparks (Ed.), *Advances in Agronomy* (Vol. 67, pp. 1–85). Academic Press. [https://doi.org/10.1016/S0065-2113\(08\)60513-1](https://doi.org/10.1016/S0065-2113(08)60513-1)
- Pierce, F. J., Robert, P. C., & Mangold, G. (1994). Site Specific Management: The Pros, the Cons, and the Realities. *Proceedings of the Integrated Crop Management Conference*, 11934214. <https://doi.org/10.31274/icm-180809-454>
- Plant, R. E. (2001). Site-specific management: The application of information technology to crop production. *Computers and Electronics in Agriculture*, 30(1), 9–29. [https://doi.org/10.1016/S0168-1699\(00\)00152-6](https://doi.org/10.1016/S0168-1699(00)00152-6)
- Rahman, N. R., Hasan, Md. A. M., & Shin, J. (2020). Performance Comparison of Different Convolutional Neural Network Architectures for Plant Seedling Classification. *2020 2nd International Conference on Advanced Information and Communication Technology (ICAICT)*, 146–150. <https://doi.org/10.1109/ICAICT51780.2020.9333468>
- Rajaraman, S., Siegelman, J., Alderson, P. O., Folio, L. S., Folio, L. R., & Antani, S. K. (2020). Iteratively Pruned Deep Learning Ensembles for COVID-19 Detection in Chest X-Rays. *IEEE Access*, 8, 115041–115050. <https://doi.org/10.1109/ACCESS.2020.3003810>
- Reyes, A. K., Caicedo, J. C., & Camargo, J. (2015). Fine-tuning Deep Convolutional Networks for Plant Recognition. *CLEF*.
- Robert, P. C. (2000). Site-specific Management for the Twenty-first Century. *HortTechnology*, 10(3), 444–447. <https://doi.org/10.21273/HORTTECH.10.3.444>

- Robert, P. C. (2002). Precision agriculture: A challenge for crop nutrition management. In W. J. Horst, A. Bürkert, N. Claassen, H. Flessa, W. B. Frommer, H. Goldbach, W. Merbach, H.-W. Olf, V. Römheld, B. Sattelmacher, U. Schmidhalter, M. K. Schenk, & N. v. Wirén (Eds.), *Progress in Plant Nutrition: Plenary Lectures of the XIV International Plant Nutrition Colloquium: Food security and sustainability of agro-ecosystems through basic and applied research* (pp. 143–149). Springer Netherlands. https://doi.org/10.1007/978-94-017-2789-1_11
- Robert, P. C., Rust, R. H., & Larson, W. E. (1995). Proceedings of site-specific management for agricultural systems: Second international conference, March 27-30, 1994 Thunderbird Hotel, 2201 East 78th St., Minneapolis, MN: conducted by the Department of Soil Science and Minnesota Extension Service, University of Minnesota. *2nd International Conference on Site-Specific Management for Agricultural Systems., Minneapolis, Minn.(USA), 1994.*
- Rokach, L. (2019). *Ensemble Learning: Pattern Classification Using Ensemble Methods*. World Scientific Publishing Co Pte Ltd.
- Rosenstein, M. T., Marx, Z., Kaelbling, L. P., & Dietterich, T. G. (2005). *To Transfer or Not To Transfer*. 4.
- Russell, S. J., & Norvig, P. (2010). *Artificial intelligence: A Modern Approach* (3rd ed). Prentice Hall.
- Sabancı, K., & Aydın, C. (2017). Smart Robotic Weed Control System for Sugar Beet. *JOURNAL OF AGRICULTURAL SCIENCE AND TECHNOLOGY*, 19(1), 73–83.
- Sabzi, S., Abbaspour-Gilandeh, Y., & García-Mateos, G. (2018). A fast and accurate expert system for weed identification in potato crops using metaheuristic algorithms. *Computers in Industry*, 98, 80–89. <https://doi.org/10.1016/j.compind.2018.03.001>

- Saidu, A., Clarkson, A. M., Adamu, S. H., Mohammed, M., & Jibo, I. (2017). Application of ICT in Agriculture: Opportunities and Challenges in Developing Countries. *International Journal of Computer Science and Mathematical Theory*, 3(1), 11.
- Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2019). MobileNetV2: Inverted Residuals and Linear Bottlenecks. *ArXiv:1801.04381 [Cs]*. <http://arxiv.org/abs/1801.04381>
- Santos, L., Santos, F. N., Oliveira, P. M., & Shinde, P. (2020). Deep Learning Applications in Agriculture: A Short Review. In M. F. Silva, J. Luís Lima, L. P. Reis, A. Sanfeliu, & D. Tardioli (Eds.), *Robot 2019: Fourth Iberian Robotics Conference* (Vol. 1092, pp. 139–151). Springer International Publishing. https://doi.org/10.1007/978-3-030-35990-4_12
- Schmidhuber, J. (2015). Deep Learning in Neural Networks: An Overview. *Neural Networks*, 61, 85–117. <https://doi.org/10.1016/j.neunet.2014.09.003>
- Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2020). Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization. *International Journal of Computer Vision*, 128(2), 336–359. <https://doi.org/10.1007/s11263-019-01228-7>
- Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. *ArXiv:1409.1556 [Cs]*. <http://arxiv.org/abs/1409.1556>
- Sørensen, R. A., Rasmussen, J., Nielsen, J., & Jørgensen, R. N. (2017). Thistle detection using convolutional neural networks. *2017 Efitra Wcca Congress - European Conference Dedicated To the Future Use of Ict*, 161–162.

- Stafford, J. V. (1996). Essential Technology for Precision Agriculture. In *Proceedings of the Third International Conference on Precision Agriculture* (pp. 593–604). John Wiley & Sons, Ltd. <https://doi.org/10.2134/1996.precisionagproc3.c74>
- Steward, B., Gai, J., & Tang, L. (2019). The use of agricultural robots in weed management and control. In J. Billingsley (Ed.), *Burleigh Dodds Series in Agricultural Science* (pp. 161–186). Burleigh Dodds Science Publishing. <https://doi.org/10.19103/AS.2019.0056.13>
- Sujaritha, M., Annadurai, S., Satheeshkumar, J., Sharan, S. K., & Mahesh, L. (2017). Weed detecting robot in sugarcane fields using fuzzy real time classifier. *COMPUTERS AND ELECTRONICS IN AGRICULTURE*, *134*, 160–171. <https://doi.org/10.1016/j.compag.2017.01.008>
- Suresh, G., Gnanaprakash, V., & Santhiya, R. (2019). Performance Analysis of Different CNN Architecture with Different Optimisers for Plant Disease Classification. *2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)*, 916–921. <https://doi.org/10.1109/ICACCS.2019.8728282>
- Sykuta, M. E. (2016). Big Data in Agriculture: Property Rights, Privacy and Competition in Ag Data Services. *International Food and Agribusiness Management Review Special Issue*, *19(A)*, 18.
- Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2015). Rethinking the Inception Architecture for Computer Vision. *ArXiv:1512.00567 [Cs]*. <http://arxiv.org/abs/1512.00567>
- Szegedy, C., Wei Liu, Yangqing Jia, Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. *2015 IEEE Conference on*

- Computer Vision and Pattern Recognition (CVPR)*, 1–9.
<https://doi.org/10.1109/CVPR.2015.7298594>
- Tan, M., & Le, Q. V. (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *ArXiv:1905.11946 [Cs, Stat]*. <http://arxiv.org/abs/1905.11946>
- Tang, J., Wang, D., Zhang, Z., He, L., Xin, J., & Xu, Y. (2017). Weed identification based on K-means feature learning combined with convolutional neural network. *Computers and Electronics in Agriculture*, *135*, 63–70. <https://doi.org/10.1016/j.compag.2017.01.001>
- Tensorflow.org. (2021a). *Post-training quantization | TensorFlow Model Optimization*.
https://www.tensorflow.org/model_optimization/guide/quantization/post_training
- Tensorflow.org. (2021b). *Pruning in Keras example | TensorFlow Model Optimization*.
https://www.tensorflow.org/model_optimization/guide/pruning/pruning_with_keras
- Tensorflow.org. (2021c). *TensorFlow Lite*. TensorFlow. <https://www.tensorflow.org/lite/guide>
- Tey, Y. S., & Brindal, M. (2012). Factors influencing the adoption of precision agricultural technologies: A review for policy implications. *Precision Agriculture*, *13*(6), 713–730.
<https://doi.org/10.1007/s11119-012-9273-6>
- Thorp, K., & Tian, L. F. (2004). A Review on Remote Sensing of Weeds in Agriculture. *Precision Agriculture*, *5*, 477–508. <https://doi.org/10.1007/s11119-004-5321-1>
- Tian, L., Reid, J. F., & Hummel, J. W. (1999). Development of a Precision Sprayer for Site-Specific Weed Management. *Transactions of the ASAE*, *42*(4), 893–900.
<https://doi.org/10.13031/2013.13269>
- Ting, K. M., & Witten, I. H. (1999). Issues in Stacked Generalization. *Journal of Artificial Intelligence Research*, *10*, 271–289. <https://doi.org/10.1613/jair.594>

- Too, E. C., Yujian, L., Njuki, S., & Yingchun, L. (2019). A comparative study of fine-tuning deep learning models for plant disease identification. *Computers and Electronics in Agriculture*, *161*, 272–279. <https://doi.org/10.1016/j.compag.2018.03.032>
- Torres-Sospedra, J., & Nebot, P. (2014). Two-stage procedure based on smoothed ensembles of neural networks applied to weed detection in orange groves. *BIOSYSTEMS ENGINEERING*, *123*, 40–55. <https://doi.org/10.1016/j.biosystemseng.2014.05.005>
- UN. (2015, September 10). *Sustainable consumption and production*. <https://www.un.org/sustainabledevelopment/sustainable-consumption-production/>
- Vaishnavi, V., Kuechler, W., & Petter, S. (2004). Design Science Research in Information Systems. *January, 20*, 2004.
- Vioix, J.-B., Douzals, J.-P., Truchetet, F., Assémat, L., & Guillemin, J.-P. (2002). Spatial and Spectral Methods for Weed Detection and Localization. *EURASIP Journal on Advances in Signal Processing*, *2002*(7), 793080. <https://doi.org/10.1155/S1110865702204072>
- Voghoei, S., Hashemi Tonekaboni, N., Wallace, J. G., & Arabnia, H. R. (2018). Deep Learning at the Edge. *2018 International Conference on Computational Science and Computational Intelligence (CSCI)*, 895–901. <https://doi.org/10.1109/CSCI46756.2018.00177>
- Walter, A., Finger, R., Huber, R., & Buchmann, N. (2017). Opinion: Smart farming is key to developing sustainable agriculture. *Proceedings of the National Academy of Sciences*, *114*(24), 6148–6150. <https://doi.org/10.1073/pnas.1707462114>
- Wang, A., Zhang, W., & Wei, X. (2019). A review on weed detection using ground-based machine vision and image processing techniques. *Computers and Electronics in Agriculture*, *158*, 226–240. <https://doi.org/10.1016/j.compag.2019.02.005>

- Wang, H., Liu, W., Zhao, K., Yu, H., Zhang, J., & Wang, J. (2018). Evaluation of weed control efficacy and crop safety of the new HPPD-inhibiting herbicide-QYR301. *Scientific Reports*, 8(1), 7910. <https://doi.org/10.1038/s41598-018-26223-9>
- Wang, J., Ma, Y., Zhang, L., Gao, R. X., & Wu, D. (2018). Deep learning for smart manufacturing: Methods and applications. *Journal of Manufacturing Systems*, 48, 144–156. <https://doi.org/10.1016/j.jmsy.2018.01.003>
- Wang, Y., Jin, L., & Mao, H. (2019). Farmer Cooperatives' Intention to Adopt Agricultural Information Technology—Mediating Effects of Attitude. *Information Systems Frontiers*, 21. <https://doi.org/10.1007/s10796-019-09909-x>
- Wang, Z., Dai, Z., Póczos, B., & Carbonell, J. (2019). Characterizing and Avoiding Negative Transfer. *ArXiv:1811.09751 [Cs, Stat]*. <http://arxiv.org/abs/1811.09751>
- Wiseman, L., Sanderson, J., Zhang, A., & Jakku, E. (2019). Farmers and their data: An examination of farmers' reluctance to share their data through the lens of the laws impacting smart farming. *NJAS - Wageningen Journal of Life Sciences*, 90–91, 100301. <https://doi.org/10.1016/j.njas.2019.04.007>
- Wolfert, S., Ge, L., Verdouw, C., & Bogaardt, M.-J. (2017). Big Data in Smart Farming – A review. *Agricultural Systems*, 153, 69–80. <https://doi.org/10.1016/j.agry.2017.01.023>
- Wolfert, S., Goense, D., & Sorensen, C. A. G. (2014a). A Future Internet Collaboration Platform for Safe and Healthy Food from Farm to Fork. *2014 Annual SRII Global Conference*, 266–273. <https://doi.org/10.1109/SRII.2014.47>
- Wolfert, S., Goense, D., & Sorensen, C. A. G. (2014b). A Future Internet Collaboration Platform for Safe and Healthy Food from Farm to Fork. *2014 Annual SRII Global Conference*, 266–273. <https://doi.org/10.1109/SRII.2014.47>

- Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, 5(2), 241–259.
[https://doi.org/10.1016/S0893-6080\(05\)80023-1](https://doi.org/10.1016/S0893-6080(05)80023-1)
- Xinshao, W., & Cheng, C. (2015). Weed seeds classification based on PCANet deep learning baseline. *2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, 408–415. <https://doi.org/10.1109/APSIPA.2015.7415304>
- Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. (2014). How transferable are features in deep neural networks? *ArXiv:1411.1792 [Cs]*. <http://arxiv.org/abs/1411.1792>
- Zala, C., & Patel, P. (2019). *A Survey on Applications of Deep Learning in Agriculture*. 07(06), 9.
- Zhang, W., Deng, L., Zhang, L., & Wu, D. (2021). A Survey on Negative Transfer. *ArXiv:2009.00909 [Cs, Stat]*. <http://arxiv.org/abs/2009.00909>
- Zhao, H. (2021). Futures price prediction of agricultural products based on machine learning. *Neural Computing and Applications*, 33(3), 837–850. <https://doi.org/10.1007/s00521-020-05250-6>
- Zhu, M., & Gupta, S. (2017). *To prune, or not to prune: Exploring the efficacy of pruning for model compression*. <https://arxiv.org/abs/1710.01878v2>

APPENDIX A: KOTLIN FOR ANDROID IMPLEMENTATION OF RUNTIME TESTS

TensorFlow Lite Android App - Classifier ClassAdapted For Dissertation

```

/** Runs inference and returns the classification results. */
public AbstractMap.Entry<List<Recognition>, Long> recognizeImage(final Bitmap bitmap,
                                                             int sensorOrientation) {
    // Logs this method so that it can be analyzed with systrace.
    Trace.beginSection( sectionName: "recognizeImage");

    Trace.beginSection( sectionName: "loadImage");
    long startTimeForLoadImage = SystemClock.uptimeMillis();
    inputImageBuffer = loadImage(bitmap, sensorOrientation);
    long endTimeForLoadImage = SystemClock.uptimeMillis();
    Trace.endSection();
    Log.v(TAG, msg: "Timecost to load the image: " + (endTimeForLoadImage - startTimeForLoadImage));

    // Runs the inference call.
    Trace.beginSection( sectionName: "runInference");
    long startTimeForReference = SystemClock.uptimeMillis();
    try {
        Thread.sleep( millis: 2);
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
    tflite.run(inputImageBuffer.getBuffer(), outputProbabilityBuffer.getBuffer().rewind());
    long endTimeForReference = SystemClock.uptimeMillis();
    Trace.endSection();

    // Gets the map of label and probability.
    Map<String, Float> labeledProbability =
        new TensorLabel(labels, probabilityProcessor.process(outputProbabilityBuffer))
            .getMapWithFloatValue();
    Trace.endSection();

    return new AbstractMap.SimpleEntry<>(getTopKProbability(labeledProbability),
                                       (endTimeForReference - startTimeForReference));
}

```

Unit Test for Logging Runtime

```
@Test
public void logClassificationResults() throws IOException {
    ClassifierActivity activity = rule.getActivity();
    Classifier classifier =
        Classifier.create(activity, Model.QUANT_XD_ENS, Device.CPU, numThreads: 1);

    for (int i = 0; i < INPUTS.length; i++) {
        String imageFileName = INPUTS[i];

        Bitmap input = loadImage(imageFileName);

        AbstractMap.Entry<List<Recognition>, Long> recog =
            classifier.recognizeImage(input, sensorOrientation: 0);
        List<Recognition> result = recog.getKey();
        long inferenceTime = recog.getValue();

        for (Recognition actual : result) {
            if (i > 1) {
                Log.v( tag: "Android-Inference-Log", msg: "Iter:," + (i-1) + ",Predicted:," +
                    actual.getTitle() + ",Time:," + inferenceTime);
            }
        }
    }
}
```

APPENDIX B: PRECISION, RECALL, AND F₁-SCORE PER CLASS

Plant Seedling Dataset

	precision	recall	f1-score	support
Maize	1.0000	1.0000	1.0000	47
Shepherd's Purse	0.9636	0.9636	0.9636	55
Scentless Mayweed	0.9741	0.9576	0.9658	118
Common Chickweed	0.9914	0.9914	0.9914	116
Black-grass	0.7353	0.4717	0.5747	53
Common wheat	0.9400	0.9592	0.9495	49
Cleavers	1.0000	0.9710	0.9853	69
Loose Silky-bent	0.8148	0.9496	0.8771	139
Charlock	0.9873	1.0000	0.9936	78
Small-flowered Cranesbill	0.9921	1.0000	0.9960	126
Fat Hen	1.0000	0.9688	0.9841	96
Sugar beet	0.9710	0.9710	0.9710	69
accuracy			0.9488	1015
macro avg	0.9475	0.9337	0.9377	1015
weighted avg	0.9480	0.9488	0.9461	1015

Leafsnap Dataset

	precision	recall	f1-score	support
prunus_yedoensis	0.9310	0.9643	0.9474	28
quercus_rubra	1.0000	0.9130	0.9545	23
populus_deltoides	1.0000	1.0000	1.0000	41
taxodium_distichum	1.0000	1.0000	1.0000	35
pinus_densiflora	0.8889	1.0000	0.9412	24
syringa_reticulata	0.9474	0.9474	0.9474	19
tilia_europaea	0.9667	0.9355	0.9508	31
prunus_serrulata	0.9130	0.9545	0.9333	22
magnolia_stellata	0.8529	0.9062	0.8788	32
populus_grandidentata	0.9730	1.0000	0.9863	36
halesia_tetraptera	0.9394	0.9394	0.9394	33
betula_populifolia	1.0000	1.0000	1.0000	26
chamaecyparis_thyoides	1.0000	0.9655	0.9825	29
ulmus_glabra	0.9667	0.9667	0.9667	30
morus_alba	1.0000	0.9706	0.9851	34
paulownia_tomentosa	0.9677	0.9677	0.9677	31
acer_saccharinum	0.9792	1.0000	0.9895	47
malus_baccata	0.8718	1.0000	0.9315	34

styrax_japonica	0.9767	0.9545	0.9655	44
betula_jacqemontii	1.0000	1.0000	1.0000	27
juglans_nigra	1.0000	0.9667	0.9831	30
quercus_muehlenbergii	0.7576	0.5682	0.6494	44
fraxinus_americana	1.0000	1.0000	1.0000	23
eucommia_ulmoides	0.9767	1.0000	0.9882	42
pinus_pungens	1.0000	0.9688	0.9841	32
aesculus_flava	0.9615	1.0000	0.9804	25
ailanthus_altissima	0.9524	1.0000	0.9756	20
pseudolarix_amabilis	0.9600	0.9600	0.9600	25
larix_decidua	1.0000	0.9286	0.9630	28
quercus_alba	1.0000	1.0000	1.0000	42
quercus_velutina	1.0000	1.0000	1.0000	33
chionanthus_virginicus	1.0000	0.9608	0.9800	51
fraxinus_pennsylvanica	1.0000	1.0000	1.0000	25
pinus_rigida	0.9667	0.9667	0.9667	30
quercus_stellata	1.0000	1.0000	1.0000	35
crataegus_crus-galli	1.0000	1.0000	1.0000	20
malus_coronaria	0.9412	0.9143	0.9275	35
prunus_subhirtella	0.7111	0.8421	0.7711	38
magnolia_denudata	0.8889	0.8889	0.8889	36
acer_ginnala	1.0000	0.9583	0.9787	24
magnolia_virginiana	1.0000	1.0000	1.0000	39
quercus_montana	1.0000	1.0000	1.0000	50
styrax_obassia	0.9474	0.9000	0.9231	20
prunus_sargentii	0.7162	0.8548	0.7794	62
cedrus_libani	0.9394	1.0000	0.9688	31
quercus_macrocarpa	1.0000	1.0000	1.0000	22
pinus_koraiensis	0.9118	0.9118	0.9118	34
prunus_virginiana	0.7164	0.7164	0.7164	67
pinus_bungeana	0.9630	0.9630	0.9630	27
acer_negundo	0.9756	1.0000	0.9877	40
pinus_taeda	0.9756	0.9524	0.9639	42
cedrus_deodara	0.9474	1.0000	0.9730	18
tilia_americana	0.9688	1.0000	0.9841	31
pinus_resinosa	0.9706	1.0000	0.9851	33
pinus_flexilis	0.8750	0.9655	0.9180	29
platanus_occidentalis	1.0000	1.0000	1.0000	41
pinus_parviflora	1.0000	0.9429	0.9706	35
ulmus_parvifolia	1.0000	1.0000	1.0000	36
magnolia_grandiflora	1.0000	1.0000	1.0000	30
cornus_mas	0.9667	0.9667	0.9667	30
morus_rubra	1.0000	1.0000	1.0000	24
acer_campestre	1.0000	1.0000	1.0000	31
gymnocladus_dioicus	1.0000	1.0000	1.0000	24
pinus_nigra	1.0000	0.9750	0.9873	40
quercus_bicolor	1.0000	1.0000	1.0000	26
oxydendrum_arboreum	0.9667	1.0000	0.9831	29
tsuga_canadensis	0.9722	1.0000	0.9859	35
crataegus_phaenopyrum	1.0000	1.0000	1.0000	29
carpinus_caroliniana	0.8947	1.0000	0.9444	34
ulmus_rubra	0.9701	0.9701	0.9701	67
quercus_marilandica	1.0000	1.0000	1.0000	42
asimina_triloba	0.9762	0.9535	0.9647	43

malus_floribunda	1.0000	1.0000	1.0000	37
diospyros_virginiana	0.9273	1.0000	0.9623	51
evodia_daniellii	0.9545	1.0000	0.9767	21
malus_angustifolia	0.9444	0.8947	0.9189	19
betula_nigra	1.0000	1.0000	1.0000	21
cryptomeria_japonica	1.0000	1.0000	1.0000	33
abies_nordmanniana	1.0000	1.0000	1.0000	31
robinia_pseudo-acacia	1.0000	1.0000	1.0000	33
chamaecyparis_pisifera	0.9688	1.0000	0.9841	31
acer_saccharum	1.0000	1.0000	1.0000	34
cercis_canadensis	1.0000	1.0000	1.0000	41
carpinus_betulus	0.9667	1.0000	0.9831	29
ulmus_pumila	0.9808	0.9808	0.9808	52
pyrus_calleryana	0.9688	1.0000	0.9841	31
aesculus_hippocastamon	1.0000	1.0000	1.0000	24
ostrea_virginiana	1.0000	0.9388	0.9684	49
pinus_strobus	0.9286	1.0000	0.9630	26
salix_babylonica	1.0000	1.0000	1.0000	34
metasequoia_glyptostroboides	0.9762	1.0000	0.9880	41
ulmus_americanus	0.9487	1.0000	0.9737	37
ulmus_procera	1.0000	0.9259	0.9615	27
pinus_virginiana	0.7838	0.7250	0.7532	40
abies_concolor	0.9778	0.9778	0.9778	45
ptelea_trifoliata	0.9792	0.9792	0.9792	48
malus_pumila	0.9744	0.9744	0.9744	39
betula_lenta	0.9615	0.9615	0.9615	26
prunus_pensylvanica	1.0000	0.9259	0.9615	27
quercus_coccinea	0.9459	1.0000	0.9722	35
ficus_carica	1.0000	1.0000	1.0000	35
quercus_virginiana	1.0000	1.0000	1.0000	25
sassafras_albidum	0.9697	0.9697	0.9697	33
toona_sinensis	1.0000	1.0000	1.0000	16
magnolia_macrophylla	1.0000	1.0000	1.0000	28
picea_abies	1.0000	1.0000	1.0000	22
catalpa_speciosa	0.9250	0.9737	0.9487	38
koelreuteria_paniculata	1.0000	1.0000	1.0000	32
pinus_cembra	0.9667	0.9062	0.9355	32
aesculus_pavi	0.9737	0.9737	0.9737	38
cladrastis_lutea	1.0000	1.0000	1.0000	36
pinus_thunbergii	0.9565	0.9565	0.9565	23
quercus_falcata	0.8889	0.8889	0.8889	9
malus_hupehensis	0.9655	0.9655	0.9655	29
quercus_nigra	1.0000	1.0000	1.0000	31
liquidambar_styraciflua	1.0000	1.0000	1.0000	31
cornus_kousa	0.9630	0.9630	0.9630	27
fagus_grandifolia	0.9714	1.0000	0.9855	34
phellodendron_amurense	1.0000	1.0000	1.0000	27
pinus_sylvestris	1.0000	1.0000	1.0000	34
magnolia_soulangiana	1.0000	1.0000	1.0000	8
carya_ovata	1.0000	0.9286	0.9630	42
tilia_cordata	0.9231	0.9474	0.9351	38
quercus_shumardii	1.0000	1.0000	1.0000	25
cornus_florida	1.0000	0.9773	0.9885	44
stewartia_pseudocamellia	0.9310	1.0000	0.9643	27

ilex_opaca	1.0000	1.0000	1.0000	43
tilia_tomentosa	0.8846	1.0000	0.9388	23
amelanchier_arborea	0.9524	0.8333	0.8889	24
magnolia_acuminata	1.0000	0.9412	0.9697	34
juniperus_virginiana	1.0000	0.9583	0.9787	24
platanus_acerifolia	0.9697	1.0000	0.9846	32
corylus_colurna	1.0000	1.0000	1.0000	16
salix_matsudana	1.0000	1.0000	1.0000	37
carya_glabra	0.8519	1.0000	0.9200	23
aesculus_glabra	1.0000	0.9714	0.9855	35
nyssa_sylvatica	0.8966	0.9630	0.9286	27
acer_palmatum	1.0000	1.0000	1.0000	47
quercus_robur	1.0000	1.0000	1.0000	33
pinus_wallichiana	0.9706	0.9429	0.9565	35
catalpa_bignonioides	1.0000	0.9038	0.9495	52
broussonettia_papyrifera	0.9726	0.9467	0.9595	75
carya_cordiformis	0.9756	1.0000	0.9877	40
albizia_julibrissin	1.0000	1.0000	1.0000	23
maclura_pomifera	0.9753	0.9753	0.9753	81
quercus_cerris	1.0000	1.0000	1.0000	32
crataegus_viridis	1.0000	0.9545	0.9767	44
zelkova_serrata	1.0000	0.9706	0.9851	34
carya_tomentosa	1.0000	0.9286	0.9630	28
celtis_tenuifolia	1.0000	0.9697	0.9846	33
quercus_palustris	0.9688	0.9394	0.9538	33
populus_tremuloides	0.9375	1.0000	0.9677	30
magnolia_tripetala	0.9000	0.8710	0.8852	31
acer_griseum	0.9697	1.0000	0.9846	32
betula_alleghaniensis	1.0000	1.0000	1.0000	28
prunus_serotina	0.9565	0.9565	0.9565	23
picea_orientalis	1.0000	0.9744	0.9870	39
fraxinus_nigra	0.9744	0.9500	0.9620	40
crataegus_pruinosa	0.9677	1.0000	0.9836	30
amelanchier_laevis	1.0000	1.0000	1.0000	29
quercus_imbricaria	0.9714	0.9444	0.9577	36
acer_platanoides	1.0000	0.9600	0.9796	25
staphylea_trifolia	1.0000	0.8974	0.9459	39
liriodendron_tulipifera	1.0000	0.9737	0.9867	38
celtis_occidentalis	0.9615	0.9615	0.9615	26
crataegus_laevigata	1.0000	1.0000	1.0000	34
acer_rubrum	1.0000	1.0000	1.0000	73
salix_nigra	0.9773	1.0000	0.9885	43
chionanthus_retusus	0.9688	0.9394	0.9538	33
cedrus_atlantica	1.0000	0.9756	0.9877	41
castanea_dentata	1.0000	1.0000	1.0000	27
juglans_cinerea	0.9565	0.9362	0.9462	47
cercidiphyllum_japonicum	1.0000	0.9615	0.9804	26
acer_pseudoplatanus	1.0000	1.0000	1.0000	21
quercus_michauxii	1.0000	0.9688	0.9841	32
pinus_peucea	1.0000	0.9722	0.9859	36
quercus_phellos	1.0000	1.0000	1.0000	20
gleditsia_triactanthos	1.0000	1.0000	1.0000	43
ginkgo_biloba	1.0000	1.0000	1.0000	24
picea_pungens	1.0000	1.0000	1.0000	27

salix_caroliniana	1.0000	1.0000	1.0000	33
quercus_acutissima	1.0000	0.9600	0.9796	25
acer_pensylvanicum	1.0000	1.0000	1.0000	31
amelanchier_canadensis	0.9310	0.9310	0.9310	29
pinus_echinata	0.9333	0.9032	0.9180	31
accuracy			0.9664	6161
macro avg	0.9697	0.9695	0.9690	6161
weighted avg	0.9675	0.9664	0.9664	6161

Open Plant Phenotyping Dataset

	precision	recall	f1-score	support
LOLMU	0.9593	0.9246	0.9416	663
THLAR	0.9731	0.9791	0.9761	1291
MATIN	0.9723	0.9793	0.9758	3047
STEME	0.9776	0.9838	0.9807	1733
POLCO	0.9897	0.9796	0.9846	294
MYOAR	0.9723	0.9259	0.9485	607
POLPE	0.9838	0.9806	0.9822	619
APESV	0.8466	0.7610	0.8015	544
GALAP	0.9873	0.9541	0.9705	327
FUMOF	1.0000	0.7241	0.8400	29
VERPE	0.9809	0.9814	0.9811	1932
CHEAL	0.9578	0.9835	0.9705	1270
PLALA	0.9241	0.9267	0.9254	723
BRNN	0.9685	0.9771	0.9728	786
BROST	0.9558	0.8757	0.9140	346
CENCY	0.9964	0.9822	0.9893	845
ALOMY	0.9284	0.9011	0.9145	475
EPHHE	1.0000	0.8824	0.9375	34
CAPBP	0.9732	0.9821	0.9776	3293
RUMCR	0.9701	0.9796	0.9748	1225
SOLNI	0.9811	0.9460	0.9632	1204
SONOL	0.9868	0.9825	0.9846	914
CHYSE	0.9843	0.9436	0.9635	266
ANGAR	0.9730	0.9771	0.9751	2141
SENVU	0.9732	0.9799	0.9765	1296
CIRAR	0.9764	0.9414	0.9585	307
VERAR	0.9705	0.9898	0.9801	5692
LAPCO	1.0000	0.9368	0.9673	253
PAPRH	0.9862	0.9933	0.9898	4620
VICHI	0.9870	0.9813	0.9841	854
MATCH	0.9731	0.9488	0.9608	1601
LYCAR	1.0000	0.9737	0.9867	76
SONAS	0.9490	0.8532	0.8986	109
ARTVU	0.9471	0.9283	0.9376	1060
GERMO	0.9960	0.9893	0.9926	748
PLAMA	0.9719	0.9679	0.9699	966
POLAV	0.9805	0.9557	0.9679	474
VIOAR	0.9866	0.9858	0.9862	1195

POAAN	0.8917	0.9578	0.9236	1874
POLLA	0.9941	0.9826	0.9883	688
EROCI	0.9918	0.9791	0.9854	863
URTUR	0.9659	0.9795	0.9726	1560
AVEFA	0.9158	0.8995	0.9076	617
SINAR	0.9892	0.9753	0.9822	849
MELNO	0.9840	0.9805	0.9823	1131
EPHPE	0.9911	0.9711	0.9810	1141
CONAR	0.9960	0.9724	0.9841	254
accuracy			0.9706	52836
macro avg	0.9715	0.9497	0.9598	52836
weighted avg	0.9707	0.9706	0.9705	52836

APPENDIX C: PUBLICATIONS FROM THIS DISSERTATION

- Ofori, M., El-Gayar, O., O'Brien, A., & Noteboom, C. (2022). A Deep Learning Model Compression and Ensemble Approach for Weed Detection. *Hawaii International Conference on System Sciences*, 9.
- Ofori, M., & El-Gayar, O. (2021). An Approach for Weed Detection Using CNNs And Transfer Learning. *Hawaii International Conference on System Sciences*, 10.
- Ofori, M., & El-Gayar, O. (2020a). Towards Deep Learning for Weed Detection: Deep Convolutional Neural Network Architectures for Plant Seedling Classification. *AMCIS 2020 Proceedings*. https://aisel.aisnet.org/amcis2020/sig_green/sig_green/3
- Ofori, M., & El-Gayar, O. (2020b). Drivers and challenges of precision agriculture: A social media perspective. *Precision Agriculture*. <https://doi.org/10.1007/s11119-020-09760-0> El-Gayar, O., & Ofori, M. (2020). Disrupting Agriculture: The Status and Prospects for AI and Big Data in Smart Agriculture. In M. Strydom & S. Buckley (Eds.), *AI and Big Data's Potential for Disruptive Innovation*. IGI Global. <https://doi.org/10.4018/978-1-5225-9687-5.ch007>
- Ofori, M., & El-Gayar, O. (2019). The State and Future of Smart Agriculture: Insights from mining social media. *2019 IEEE International Conference on Big Data (Big Data)*, 5152–5161. <https://doi.org/10.1109/BigData47090.2019.9006587>