Unsupervised Representation Learning With Deep Convolutional Generative Adversarial Networks for Precision Agriculture

M.Sc. in Data Science 2023



Department of Computing, ATU Donegal, Port Road, Letterkenny, Co. Donegal, Ireland.

Unsupervised Representation Learning With Deep Convolutional Generative Adversarial Networks for Precision Agriculture

Author:

Supervised by: Dr. Shagufta Henna

A thesis submitted in partial fulfillment of the requirements for the

Master of Data Science

Submitted to Quality and Qualifications Ireland (QQI)  
Dearbhú Cáilíochta agus Cáilíochtaí Éireann September 2023

# 

# Declaration

I hereby certify that the material, which l now submit for assessment on the programmes of study leading to the award of Master of Science in Data Science, is entirely my own work and has not been taken from the work of others except to the extent that such work has been cited and acknowledged within the text of my own work. No portion of the work contained in this thesis has been submitted in support of an application for another degree or qualification to this or any other institution. I understand that it is my responsibility to ensure that I have adhered to ATU’s rules and regulations.

I hereby certify that the material on which I have relied on for the purpose of my assessment is not deemed as personal data under the GDPR Regulations. Personal data is any data from living people that can be identified. Any personal data used for the purpose of my assessment has been psudonymised and the data set and identifiers are not held by ATU. Alternatively, personal data has been anonymised in line with the Data Protection Commissioners Guidelines on Anonymisation.

I consent that my work will be held for the purposes of education assistance to future students and will be shared on the ATU Computing website (www.lyitcomputing.com) and Research THEA website (https://research.thea.ie/). I understand that documents once uploaded onto the website can be viewed throughout the world and not just in the Ireland. Consent can be withdrawn for the publishing of material online by emailing Thomas Dowling; Head of Department at thomas.dowling@lyit.ie to remove items from the ATU Computing website and by email emailing Denise McCaul; Systems Librarian at denise.mccaul@lyit.ie to remove items from the Research THEA website. Material will continue to appear in printed formats once published and as websites are public medium, LYIT cannot guarantee that the material has not been saved or downloaded.

Signature of Candidate:

Yadnesh Mankame Date: 29/08/2024

# Acknowledgments

First and foremost, I thank the Lord Almighty, to whom I owe my very life, for giving me this opportunity and giving me the strength to complete it successfully. It is with heartfelt gratitude and profound respect that I acknowledge the help, encouragement, and inspiration of my supervisor, Dr. Shagufta Henna. I would like to express my sincere thanks to Atlantic Technological University and to all the staff who have made it possible for me to pursue and complete this master's degree, and last but not least I would like to sincerely thank my family for always being there with love and support during my studies.

# Abstract

With a particular focus on soybean fields, the thesis investigates the use of DCGANs to address the difficulties associated with weed detection and classification in precision agriculture. The study highlights the drawbacks of conventional weed management techniques as well as the potential of DL more especially,  CNNs for accurate and efficient weed identification. CNNs, however, can be sensitive to class imbalance, where certain weed species are not included in the training data. They also frequently require large labeled datasets, which can be costly and time-consuming to acquire. The thesis suggests using DCGANs to create artificial weed images in order to get close to these problems. This will enhance the training dataset and make it easier for CNN to identify and categorize different types of weeds. With augmentation, the model achieves a near-perfect testing accuracy of 0.98 and a perfect training accuracy of 1.00, as opposed to 0.99 and 0.95, respectively, without augmentation. This demonstrates how well DCGAN-based data augmentation works to overcome the drawbacks of unbalanced datasets and improve CNN models' performance in weed detection for precision farming. In the future, technology will be crucial in guaranteeing food security and protecting the environment. The research described in this thesis advances AI-driven solutions for efficient and sustainable farming practices.

# Acronyms

|  |  |
| --- | --- |
| Acronym | Definition |
| AI | Artificial Intelligence |
| AL | Active Learning |
| CNNs | Convolutional Neural Networks |
| CV | Computer Vision |
| DL | Deep Learning |
| DCGAN | Deep Convolutional Generative Adversarial Network |
| DCNNs | Deep Convolutional Neural Networks |
| FRR | False Rejection Ratio |
| FAR | False Acceptance Ratio |
| GANs | Generative Adversarial Networks |
| GNO | GoogLe Net Overfeat |
| GLCM | Gray Level Co-occurrence Matrix |
| GPU | Graphics Processing Unit |
| GC | Google Colab |
| HSV | Hue Saturation Value |
| IoU | Intersection over Union |
| IoT | Internet of Things |
| KNN | k-Nearest Neighbors |
| LSTM | Long Short-Term Memory |
| MAP | Mean Average Precision |
| ML | Machine Learning |
| NN | Neural Networks |
| NDI | Normalized Difference Index |
| NIR | Near Infrared |
| NLP | Natural Language Processing |
| RNNs | Recurrent Neural Networks |
| RCNNs | Regional Convolutional Neural Networks |
| RF | Random Forest |
| RPA | Remotely Piloted Aircraft |
| R-FCN | Region-based Fully Convolutional Network |
| SMOTE | Synthetic Minority Over-sampling Technique |
| SPADE | Spatially Adaptive Denormalization |
| SL | Supervised Learning |
| SSWM | Site-Specific Weed Management |
| SLR | Systematic Literature Review |
| SVM | Support Vector Machine |
| TF | Transfer Learning |
| UAVs | Unmanned Aerial Vehicles |
| WSA | Watershed Segmentation Algorithm |
| WIR | Weed Infestation Rate |

# 

# Table of Contents

[Declaration 3](#_Toc175666435)

[Acknowledgements 4](#_Toc175666436)

[Abstract 5](#_Toc175666437)

[Acronyms 6](#_Toc175666438)

[Table of Contents 8](#_Toc175666439)

[Table of Figures 10](#_Toc175666440)

[1. Introduction 11](#_Toc175666441)

[1.1 Related Work 13](#_Toc175666442)

[1.2 Research Question 15](#_Toc175666443)

[1.3 Scope and Limitation 16](#_Toc175666444)

[1.4 Report Outline 16](#_Toc175666445)

[2. Literature Review 18](#_Toc175666446)

[3. Unsupervised Representation Learning With DCGAN for Weed Classification 32](#_Toc175666447)

[3.1. Proposed DCGAN-based Weed Classification 32](#_Toc175666448)

[3.1.1. Deep Convolutional GAN 32](#_Toc175666449)

[3.1.1.1 Generator 37](#_Toc175666449)

[3.1.1.2 Discriminator 40](#_Toc175666449)

[3.1.1.3 Data Extraction 44](#_Toc175666449)

[3.1.1.4 Classification Based on DCGAN Augmented Data 46](#_Toc175666449)

[4. Performance Evaluation and Analysis of Classification with DCGAN-based Data Augmentation 51](#_Toc175666447)

[4.1. Environment Setup 51](#_Toc175666462)

[4.1.1. Hardware 51](#_Toc175666463)

[4.1.2. Software Requirements 51](#_Toc175666464)

[4.1.3. Dataset 52](#_Toc175666465)

[4.1.4. Augmentation using DCGAN 52](#_Toc175666465)

[4.2. Performance Metrics 56](#_Toc175666466)

[4.2.1. Confusion matrix 56](#_Toc175666467)

[4.2.2. Accuracy 57](#_Toc175666468)

[4.2.3. F1 Score 57](#_Toc175666469)

[4.2.4. AUC ROC 59](#_Toc175666470)

[4.2.5. CNN Results with DCGAN Augmentation 60](#_Toc175666471)

[4.3. Performance Analysis of CNN with GAN Augmentation 60](#_Toc175666472)

[4.3.1. Image Augmentation using GAN 60](#_Toc175666471)

[4.3.2. GAN Model Design 62](#_Toc175666471)

[4.3.3. CNN based on DC-GAN Augmentations 64](#_Toc175666471)

[4.3.4. Result of GAN with CNN 65](#_Toc175666471)

[4.4. Performance Analysis without Augmentation 66](#_Toc175666473)

[4.5. Discussion 69](#_Toc175666473)

[5. Conclusion and Future Works 71](#_Toc175666474)

[5.1. Conclusions 71](#_Toc175666475)

[5.2. Future Works 73](#_Toc175666476)

# Table of Figures

[Figure 3.1.1 Framework for Weed Detection in Soybean Fields using a DCGAN and CNN 33](#_Toc429427538)

[Figure 3.1.1.1 Generator 38](#_Toc429427539)

[Figure 3.1.1.2 Discriminator 41](#_Toc429427539)

[Figure 3.1.1.3 CNN Model 47](#_Toc429427539)

[Figure 4.1.1 GPU Version 50](#_Toc429427539)

[Figure 4.1.4.1 Augmented Data 52](#_Toc429427539)

[Figure 4.1.4.2 Training of DCGAN 53](#_Toc429427539)

[Figure 4.1.4.3 Generator Code 53](#_Toc429427539)

[Figure 4.1.4.4 Discriminator Code 54](#_Toc429427539)

[Figure 4.1.4.5 CNN Model 55](#_Toc429427539)

[Figure 4.2.1 DCGAN Confusion Matrix 56](#_Toc429427539)

[Figure 4.2.2.1 DCGAN Test Accuracy 57](#_Toc429427539)

[Figure 4.2.2.2 DCGAN Train Accuracy 57](#_Toc429427539)

[Figure 4.2.3.1 F1 Score 58](#_Toc429427539)

[Figure 4.2.3.2 Classification Report 58](#_Toc429427539)

[Figure 4.2.4 AUC ROC Curves 59](#_Toc429427539)

[Figure 4.2.5 CNN Result with DCGAN Augmentation 60](#_Toc429427539)

[Figure 4.3.1.1 Before Augmentation 61](#_Toc429427539)

[Figure 4.3.1.2 After Augmentation 61](#_Toc429427539)

[Figure 4.3.2.1 Generator Summary 62](#_Toc429427539)

[Figure 4.3.2.2 Discriminator Summary 63](#_Toc429427539)

[Figure 4.3.2.3 Generator and Discriminator Loss 64](#_Toc429427539)

[Figure 4.3.3 CNN Model Summary 64](#_Toc429427539)

[Figure 4.3.4.1 GAN with CNN Confusion Matrix 65](#_Toc429427539)

[Figure 4.3.4.2 GAN with CNN Accuracy Score 66](#_Toc429427539)

[Figure 4.4.1 CNN Confusion Matrix 66](#_Toc429427539)

[Figure 4.4.2 CNN Accuracy Score 67](#_Toc429427539)

[Figure 5.1 Generator Model Code 77](#_Toc429427539)

[Figure 5.2 Initialize Generator Model Code 78](#_Toc429427539)

[Figure 5.3 Discriminator Model Code 78](#_Toc429427539)

[Figure 5.4 Initialize Discriminator Model Code. 78](#_Toc429427539)

[Figure 5.5 Dataset Extraction Code 79](#_Toc429427539)

[Figure 5.6 CNN Model Code 79](#_Toc429427539)

# Introduction

Precision Agriculture, or smart agriculture, is an innovative farming method that uses modern technologies in order to maximize resource efficiency, improve crop productivity, and reduce environmental impact. Using a combination of IoT, AI, ML, and big data analytics, this modern farming framework monitors and manages agricultural operations with an amount of precision that has never been achieved before. Some of the greatest challenges facing traditional farming are addressed by smart agriculture, such as resource inefficiency, weed and pest control, and the unpredictable weather patterns caused by climate change. One of the crucial areas where smart agriculture has advanced significantly is weed management. Weeds prevent crops' access to essential resources like sunlight, water, and nutrients, resulting in lower crop yields and higher production costs. Traditional weed-control techniques, which mainly rely on human labour and chemical herbicides, are frequently ineffective, time-consuming, and harmful to the environment. By utilizing modern technologies to enable more accurate and successful weed management strategies, smart agriculture provides creative answers to these problems. The use of DL models for weed detection and classification is one of the most promising uses of smart agriculture in weed management. DL, a branch of ML, uses multi-layered artificial NN to process and learn from huge amounts of data. DL models can be trained to recognize and classify various weed species in agricultural fields, allowing for more focused and effective weed management strategies.

The efficient use of important assets like water, fertilizers, and pesticides is the goal of smart agriculture. Farmers are able to provide real-time weather patterns, soil conditions, and crop health monitoring by utilizing technologies such as IoT sensors, satellite imagery, and data analytics. This reduces waste and its negative effects on the environment by enabling them to apply inputs exactly where and when they are needed. Precision systems for watering, for example, guarantee that crops receive precisely the right amount of water, saving water and increasing crop yields. The future prediction describes that there will be almost 10 billion people on the planet, which will result in a sharp increase in food demand. Smart agriculture boosts farm productivity to help meet this demand. With the use of   technologies like drones, automated machinery, and AI-powered analytics possible problems like pest infestations or nutrient shortages can be detected early on and treated in a timely manner to increase yields overall and prevent crop loss. The basic idea of smart agriculture is sustainability. Farmers may implement actions that reduce their environmental impact by utilizing insights derived from data. Precision farming techniques, for example, lower the risk of polluting soil and water by reducing the need for chemical inputs. Furthermore, differentiating crops and rotation are supported by smart agriculture, which improves soil health and biodiversity. Extreme weather, altered rainfall patterns, and unpredictable weather patterns are just a few of the ways that climate change harms agriculture. By allowing farmers to adjust to these changes, smart agriculture gives them the means to increase their climate durability. Farmers can modify their planting schedules or methods to water in response to weather deviations, for instance, by using advanced weather forecasting models. Furthermore, based on data breeding programs using smart systems may help in the development of crop varieties that are flexible to climate change. Decision-making at every level of farming is improved by the extensive information that smart agriculture gives farmers. Analysis of data is essential for choosing the right crops and for figuring out when to plant and harvest them. Farmers can now forecast possible outcomes based on past data and current conditions with the help of predictive modeling, which is made possible by the integration of AI and ML in agriculture. For farmers, the use of smart agriculture can have a major positive financial impact. Profit margins can be enhanced by smart farming methods by lowering input costs and raising efficiency. Furthermore, through automation and remote monitoring systems, precision agriculture can lower labour costs while allowing farmers to run their businesses more profitably. More financial stability and future possibilities for farmers are also made possible by the better productivity and crop yields linked to smart agriculture. As the world's population rises, one of the biggest challenges will be maintaining food security. In order to address this issue, smart agriculture is necessary because it makes methods of farming more productive and efficient. Smart agriculture contributes to a consistent supply of food by optimizing crop yields and making the best use of the resources. In addition, smart agriculture encourages the long-term sustainability of food production systems by minimizing the negative effects of farming on the environment. Smart agriculture generates huge volumes of data, which opens up possibilities for ongoing innovation. This data can be used by scientists and farmers to create new farming methods, a variety of crops, and improvements in technology that will increase agricultural sustainability and productivity. Continuous data collection and analysis also make it possible to improve current procedures, which eventually results in small increases in yield and productivity.

The important initial expense needed for technology and infrastructure is one of the biggest obstacles to the adoption of smart agriculture. Drones, GPS systems, IoT devices, and automated machinery are examples of advanced devices that are expensive. Several small and medium-sized farmers may find these expenses to be unaffordable. The ongoing costs of data management, software subscriptions, and maintenance also contribute to the financial pressure. Smaller farms are therefore often at a disadvantage when it comes to the adoption of precision agriculture, which is typically restricted to larger, more financially stable farming operations. High levels of technical expertise are needed to implement smart agriculture. Farmers must be able to operate and understand advanced technologies, analyze and interpret large amounts of data, and make decisions based on results from analysis. This is a difficult path to learning for many people. This problem is made worse by the lack of access to precision agriculture training and education, especially in developing nations where there may be low levels of technological education. Farmers might find it difficult to completely enjoy the rewards of smart agriculture if they lack the necessary skills and information. Data processing, analysis, and collection are essential components of smart agriculture. However, handling massive amounts of data comes with a lot of difficulties. Farmers must make sure that the information they gather is timely, accurate, and relevant. Sensitive data regarding agricultural procedures, crop yields, and resource usage may be at risk for cyberattacks or unauthorized access, raising concerns about data privacy and security. For farmers and agricultural companies, establishing strong data management systems and guaranteeing commitment to data protection regulations are essential but difficult tasks. Reliable internet connectivity is essential for smart agriculture to work, especially in rural areas where the majority of farming activities are conducted. Unfortunately, a lot of rural areas still don't have access to fast internet, which makes it challenging to use cloud-based platforms, IoT devices, and real-time data analytics. Farmers may encounter data transmission delays in places with insufficient connectivity, which reduces the value of precision farming instruments. To ensure that smart agriculture is widely adopted, it is necessary to address the lack of internet access in rural areas. Technologies for smart agriculture are frequently created and evaluated in particular environments. However, there are many variations in crop varieties, climatic conditions, and soil types found in actual farming environments. Precision farming tools' accuracy and performance may be impacted by this variability. When used in a different type of soil, a sensor calibrated for one type of soil may not produce precise results. It is still very difficult to adapt smart agriculture technologies to a variety of changing and varied environmental conditions. Although precision agriculture technologies can greatly increase productivity on large farms, it can be difficult to scale these technologies for use on smaller farms or in other agricultural situations. It can take a lot of time and money to customize smart agriculture solutions to fit different farm sizes, crop types, and cultural traditions. Furthermore, smaller farms may not see the same financial advantages from smart agriculture because of the decreased opportunity for return on investment due to their smaller scale of business operations.

Precision Agriculture,  makes it possible to apply pesticides and fertilizers precisely where they are needed, as instead of equally across entire fields. This minimizes both costs and environmental impact. In addition, early problem detection and prevention are made possible by real-time crop health and soil condition monitoring, which also increases overall productivity. In addition, farmers benefit financially from data-driven insights since they help in forecasting crop performance and preparing for market demands. Smart agriculture essentially maximizes the effectiveness of all resources and processes involved in crop production, transforming traditional farming into a more productive, sustainable, and efficient activity.

## Related Work

(Hasan et al. 2021) give an in-depth examination of DL techniques for agricultural weed detection and classification, emphasizing the difficulties in advising weeds from crops because of their similar visual characteristics. With the goal of providing insights into the development and potential of DL in automated weed management systems, the authors organize the literature into four main categories: evaluation metrics, DL techniques for weed detection, dataset preparation, and data acquisition. Because they can automatically extract features and handle sequential data, they highlight CNNs and RNNs, especially LSTM networks, which are the most effective architectures. (Fawakherji et al. 2020) present a novel method for improving crop/weed classification in precision agriculture by using GANs for data augmentation. By creating artificial images, their technique reduces the requirement for expensive, manually-labeled datasets and enhances CNN training. The study shows that segmentation accuracy is improved across different architectures when synthetic and real images are combined, but more testing is required for wider application. (Pai, Kamath, and Balachandra 2024) offer a thorough analysis of DL techniques for agricultural weed detection, highlighting how DL improves localization and detection and increases the accuracy of weed and crop identification. While pointing out challenges in scalability and practical application, the study examines the efficacy of CNNs, including DCNNs and RCNNs, and discusses integrating AI with drones for enhanced weed management. A new semi-supervised GAN framework is proposed by (Kerdegari et al. 2023) to handle multispectral image classification in precision agriculture, specifically for weed detection using UAVs. With just 50% of the data labeled, their model achieves a high F1 score of 0.85, effectively classifying weeds and crops. The study highlights how GANs can help overcome limited data challenges for precise farming applications. The current status of nearby and broad weed detection systems for SSWM has been evaluated by (Lopez-Granados 2011). The importance of multispectral and hyperspectral imaging for precise weed maps which are essential for precision farming is emphasized by the study. However, the technology's potential for accurate weed management is limited by high costs and technical difficulties, especially for smaller farms. An Internet of Things-based weed detection model that uses CNNs to classify weeds and crops in real-time is proposed by (Dankhara, Patel, and Doshi 2019). By using real-time data processing to improve weed identification accuracy, this strategy aims to reduce the use of herbicides. Although , the paper would benefit from a discussion of the difficulties associated with large-scale implementation in various agricultural conditions as well as specific performance metrics. To enhance weed detection in agriculture,(Kulkarni and Angadi 2019) present an innovative system combining DL, image processing, and IoT. Their system detects weeds among crops by utilizing a CNN model that is installed on a Raspberry Pi, with the goal of minimizing the use of harmful herbicides. In spite its effectiveness (85% accuracy), the study highlights that for wider applicability, larger datasets and improved model precision are required. A thorough literature review on DL, and more especially CNNs, for weed detection in agriculture was carried out by (Murad et al. in 2023). Since 2015, they have noticed a rise in research interest because of DL's picture recognition capabilities. The review differs traditional ML techniques with DL, emphasizing DL's rising prominence while acknowledging ML's ongoing significance. According to (Islam et al., 2021), weeds in Australian chili farms can be successfully identified using UAV images and ML algorithms (RF, SVM, and KNN). With accuracy rates of 96% and 94%, respectively, SVM and RF perform better than KNN. The study's small dataset and dependence on RGB images, however, point to the need for additional research using a variety of image types and DL techniques. (Umamaheswari et al. 2018) used TensorFlow's GNO model and GPU parallel image processing to create a real-time weed detection system. In terms of accuracy, recall, and precision, the system performs better than current techniques and has a high recall rate. The dependence on a single dataset and lack of a thorough discussion of the model architecture, however, point to the necessity of more generalization and robustness research. (Bento et al. 2023) classified weeds, cover crops, and coffee plants in a coffee plantation using an RPA equipped with a multispectral sensor and ML algorithms (SVM and RF). SVM was slightly superior to RF. The method, which reduces herbicide costs by approximately 92.68%, has potential for controlling weeds, but it is limited by its emphasis on particular weed species and coffee types. For DL techniques to be used and explored more widely, more investigation is required. Using RGB and RGB+NIR images, (Lottes et al. 2017) created a vision-based system for classifying crops and weeds in UAV images. Their multi-class RF classification method works well for weed species and sugar beets, and NIR data makes it even more effective. Even though the system is reliable, it would still be beneficial to evaluate it in a variety of scenarios and investigate DL techniques to improve accuracy and feature extraction. Ground-based sensor technologies, such as spectrophotometric, optoelectronic, fluorescence, and LiDAR/ultrasonic sensors, have been examined by (Peteinatos et al. 2013) for the purpose of weed detection. In order to improve weed management, they stress the importance of sensor integration and draw attention to the need for advancements in hardware, algorithms, and affordability. While thorough, the paper lacks new findings and recommends more research on the sustainability impacts and sensor efficacy for improved integration into farming practices. Using CNN models (ResNet50, MobileNetV2), (Razfar et al. 2022) suggest a DL-based weed detection system for soybean farms. Tested on a Raspberry Pi 4 and a powerful GPU, their exclusive 5-layer CNN model achieves 97.7% detection accuracy with minimal memory usage. The study emphasizes its potential for precision agriculture, but it also recommends more testing in various contexts and research into cutting-edge DL methods.

## Purpose of the Study and Research Questions

The purpose of this thesis is to explore the application of DCGANs for augmenting UAV-captured imagery and improving weed classification in precision agriculture using CNN. This research aims to leverage DCGANs to generate synthetic images that enhance the diversity and volume of training data available for CNN models. By focusing on how the inclusion of DCGAN-augmented data impacts the classification accuracy and robustness of CNNs, the study seeks to advance precision agriculture practices by providing more reliable and effective weed identification tools. Ultimately, this research contributes to developing advanced machine-learning techniques for agricultural applications, with a specific emphasis on enhancing the performance of weed classification systems.

1. How effectively can DCGANs generate synthetic UAV imagery that closely resembles real-world UAV-captured images in terms of visual and statistical similarity?

This question examines the capability of DCGANs to produce synthetic images that mirror the characteristics of actual UAV images, ensuring that the augmented data is realistic and suitable for training CNN models.

1. Can the inclusion of DCGAN-augmented synthetic UAV imagery improve the classification accuracy of CNN models for weed identification in precision agriculture?

This question investigates whether integrating synthetic data into the training process enhances the CNN models' ability to accurately classify weeds, leading to improved performance in real-world agricultural scenarios.

## Scope and Limitations

The use of GANs for precision agriculture 's data augmentation in weed detection is examined in this thesis. Using UAV imagery from soybean fields, it focuses on enhancing CNNs by integrating GANs to address class imbalance and dataset limitations. The study also looks at how well the model generalizes to different environments and whether it can be scaled up for small-scale farming. The research's limitations are acknowledged in multiple areas as a limited timestamp:

* Computational Resources: The need for significant GPU resources makes certain organizations, particularly those in developing nations, less accessible.
* Generalization: Because of regional variations, it is challenging to generalize DL models across a range of agricultural environments.
* Class Imbalance: Synthetic images produced by GANs might not adequately address the class imbalance, which could result in inaccurate models.
* Scalability: Difficulties in expanding the solution for small-scale farmers because of restricted access to essential technologies such as high-performance computing and UAV's.

## Report Outline

Chapter 2 – The chapter looks at the suggested approach for improving the accuracy of weed detection using a DCGAN for the creation of synthetic images, talks about data collection using UAVs and preprocessing techniques, addresses class imbalance using DCGAN, SMOTE, and class weight adjustments, and compares related DL models, focusing on their performance and challenges in agriculture.

Chapter 3 – An in-depth description of the DCGAN architecture is provided in this chapter, along with the focus on the functions of the Generator and Discriminator in creating artificial images for data augmentation, the methodical training of DL models using these increased datasets, evaluation metrics, and solutions to issues like computational costs and dataset limitations.

Chapter 4 – The hardware and software used, the procedures for applying DCGAN and CNN models to the dataset, the results of weed detection, a comparison of model performance with and without data augmentation, an analysis of key metrics, and a discussion of how artificial images improved performance in addition to pointed out limitations are all covered in the experimental setup and implementation.

Chapter 5 – In order to reduce the use of herbicides and increase crop yields, this chapter shows how using DL models and GANs significantly enhances weed detection in precision agriculture. Additional study is suggested to improve GAN architectures and solve current issues for wider application, which will ultimately lead to AI-driven advancements in farming methods.

# Literature Review

The basic concept of the suggested approach is to enhance the training dataset with artificial weed images created by a DCGAN, which will increase the classification accuracy of weed images. Two NN make up the DCGAN: a DISC that separates real images from synthetic ones and a generator that uses random noise to produce realistic weed images. The GEN increases the volume and diversity of the training data by producing higher-quality synthetic images through adversarial training. This augmentation method helps to enhance the model's generalization abilities by adding variations in backgrounds, lighting, and weed appearances. It is especially helpful for small datasets.

The DISC and the GEN are the two primary parts of the DCGAN architecture. The DISC's job is to distinguish between real and synthetic images, whereas the GEN's purpose is to produce synthetic weed images from a random noise vector. In the adversarial training process, the DISC aims to accurately distinguish between real and fake images, while the GEN tries to trick the DISC. Both networks are pushed by this competition to continuously enhance their performance.

The workflow's beginning is data collection. UAVs are used to take high-quality pictures of soybean fields. The ability to extract particular weed samples is made feasible by the detailed perspective of the fields. After that, the acquired photos are processed to normalize their dimensions and format so that the CNN and DCGAN classifiers can use them for training. Resizing images, converting them to tensors, and normalizing pixel values are crucial preprocessing steps.

In classification models, class imbalance is a common problem, especially for tasks involving the classification of weeds, since some weed types are frequently missing in the dataset. In order to tackle this problem, the suggested approach combines multiple strategies. To increase the number of instances of the minority classes in the training set, a DCGAN creates synthetic images for them in the first step of data augmentation. To produce a more balanced dataset, methods of resizing are also used, which include under sampling the majority classes and oversampling the minority classes.

In order to give minority classes greater weight during model training, the approach also involves modifying the class weights in the loss function. In addition, DCGANs are used together with synthetic data generation techniques like the SMOTE to increase the quantity and variety of data available for the missing classes. All of these strategies work together to increase the model's capacity to correctly identify any kind of weed, regardless of how frequently it appeared in the original dataset.

(Hasan et al.'s 2021) provides an in-depth analysis of DL methods for classifying and detecting weeds in photos. The main goal is to assess current DL-based methods used in agriculture, with a focus on weed identification, localization, and detection within crops. With their similar colors, textures, and shapes, weeds and crops can be difficult to distinguish from one another. This research deals with this problem. The authors have categorized the current writings into four main areas evaluation metrics, DL techniques for weed detection, dataset preparation, and data acquisition. This classification aims to offer an in-depth understanding of the development and possibilities of DL in automated weed management systems. The main purpose of the paper is to survey different DL models used for weed detection. The most popular and efficient NN architectures are highlighted as CNNs and RNNs. Because CNNs can automatically extract ordered features from images, they are widely used in spite of their reputation for better image analysis performance. The use of LSTM networks, a kind of RNN, which work well with sequential data and time-series information important to the agricultural industry, is also discussed in the review. The authors have described the steps involved in a typical DL-based weed detection system workflow, such as image acquisition, pre-processing, DL model classification, and model evaluation. The performance of DL models in comparison to conventional ML methods is evaluated in this paper. The authors notice that, in general, DL models perform better than conventional techniques, particularly when working with big datasets. The review points out that while manual feature extraction is necessary for traditional ML approaches, DL techniques are superior at automatic feature learning, which improves classification accuracy. However, the efficacy of these DL models frequently depends on availability of significant structured data sets. In the paper, various DL models are compared and their accuracy and robustness under a range of agricultural conditions are highlighted. Although (Hasan et al. 2021) offer a detailed evaluation of DL methods for weed identification, the paper's lack of data backing may come under heat. Although a number of DL models and their potential are covered in detail in the review, there are no real-world applications or experimental results to support the claims made. Furthermore, the paper highlights the technical features of DL techniques without sufficiently addressing the practical issues that arise in their application, such as the cost of acquiring data, the requirement for high computational resources, and the possible negative effects on the environment and the economy. In order to close the gap between conceptual potential and practical deployment in agricultural settings, future work could benefit from including case studies or real-world examples.

(Fawakherji et al. 2020) use GANs for data augmentation in their paper they present an innovative approach to improve crop/weed segmentation in precision farming. The difficulty of training DL models in particular, CNNs, which need large, expensive, and time-consuming analyzed datasets is discussed by the authors. Their method improves existing datasets without requiring additional manual labeling by creating synthetic images of crops and inserting them into actual agricultural situations. The GAN architecture used by the authors was created explicitly to produce artificial crop images. The GAN in use is a variation of the SPADE architecture, which uses semantically defined masks to produce photo-realistic images. This model is selected to improve CNNs' training for crop/weed segmentation by generating high-quality images that closely resemble actual crops. On the Sugar Beet 2016 dataset, the suggested approach is assessed using a number of segmentation architectures, including U-Net, SegNet, Bonnet, and U-Net-ResNet. IoU scores for the categories of ground, weed, and crop are used to evaluate the performance. Models trained on a combination of synthetic and real images always beat those trained on either real or synthetic images alone, according to the evaluation. This illustrates how crop/weed segmentation models become more accurate and more general when GAN-generated data is included. Across various architectures, the mixed dataset approach which uses both synthetic and real images—produced the highest IoU scores. For example, when trained with mixed datasets, the U-Net model significantly increased the IoU for crops (from 0.901 to 0.946) and weeds (from 0.294 to 0.553). Similar changes have been observed for additional architectures, suggesting that the artificial images function as an efficient improve to actual data. Although the paper offers a promising method for enhancing agricultural datasets, additional testing in a variety of crop types and environments would be beneficial to confirm the generality of the approach. Furthermore, the technique depends on the accuracy of the initial division faces, which could introduce biases if they are not generated accurately. Furthermore, even with the suggested approach's prevention of the computational cost of training GANs and producing synthetic images, widespread use in environments with limited resources remains challenging. Following studies projects might look into enhanced GAN models or substitute methods for data augmentation, which could further reduce these computational requirements.

The authors of the paper (Pai, Kamath, and Balachandra 2024) implement a comprehensive evaluation of DL methods for weed detection in agricultural environments. The study highlights how DL has transformed agriculture, particularly in the areas of localization and detection, solving the problem of weeds and crops having similar color, form, and texture. The goal of the study is to look at the state of DL based weed detection methods nowadays, highlighting how different DL techniques can enhance the accuracy and efficacy of crop weed localization, identification, and classification. The paper focuses on SL strategies that improve pre-trained models using large labeled datasets. It discusses various DL models that are commonly used in weed detection. In order to achieve high accuracy in image-based weed detection systems, the authors describe in detail the use of CNNs, including particular architectures like DCNNs and RCNNs The investigation also looks at how AI-driven systems can be integrated with advanced farming tools like drones and UAVs to improve weed management and detection. The main criteria used to evaluate these models is how well they differentiate between crops and weeds, taking into consideration various factors such as the quality of the image, colors, and the addition of non-imaging detectors. The performance of several DL models is compared by the authors, who point out that in controlled environments, models like DCNNs and RCNNs have provided positive results. However, because of things like climate change and the difficulty of telling different plant species apart, it is still difficult to use these models in practical agricultural settings. The paper by (Pai et al. 2024) covers DLs possibility of agricultural weed detection in great detail, but it doesn't address how scalable these models are for large-scale implementation. The authors fail to provide a clear roadmap for moving from testing environments to field-ready solutions, nor do they sufficiently address the computational resources needed for training and implementing these models in real-time applications. The paper would also benefit from a deeper investigation of the possible environmental effects and its financial stability of increasing use of these technologies in agricultural production.

In order to address the problem of classifying multispectral images in accurate farming, the study by (Kerdegari et al. 2023) suggests an inventive semi-supervised GAN framework. Using UAVs matches with multispectral cameras, the research focuses on weed growth detection. CNNs, one of the more traditional methods for classifying images, need large labeled datasets, which can be expensive and difficult to obtain. The authors suggest a semi-supervised GAN to overcome this, which produces realistic images that are used to train a DISC a multi-class classifier using only a small amount of labeled data. The GEN and the DISC are the two main parts of the semi-supervised GAN model that is presented in the paper. The GEN network mimics the distribution of actual multispectral data by creating realistic synthetic images from random noise. The DISC, a modified DCNN, performs the dual roles of a multi-class classifier by classifying pixels into distinct groups such as crop, weed, and background in addition to differentiating between real and fake images. Built on the basis of the DCGAN, the architecture uses a softmax activation function in place of the typical sigmoid function in the final layer to enable pixel-wise classification. This design seeks to increase classification accuracy even in the presence of limited labeled data. The multispectral photos of weeds and crops in the  dataset were used to evaluate the proposed model. The F1 score was used to compare various input channel configurations and different percentages (30%, 40%, and 50%) of labeled data in order to evaluate performance. With two channels (Red and NIR) and 50% labeled data, the semi-supervised GAN performed well in classification, as mentioned in the results, which showed an F1 score of roughly 0.85. The study also compared its results with an earlier method that trained a transmitted CNN using a fully labeled dataset. Comparable performance was demonstrated by the semi-supervised GAN approach, demonstrating its usefulness even with limited labeled data. Although the work by (Kerdegari et al. 2023) offers an innovative approach for multispectral image classification using a semi-supervised GAN, there are certain aspects of the study that could be improved. Although, the model's limited generalizability to other datasets or scenarios where the image characteristics differ significantly from those in the training set could be linked to its dependence on a small subset of labeled data. In addition, the study only considers a small number of multispectral channels, mostly the Red and NIR bands, thus eliminating to investigate the possible advantages of including other spectral bands. By using a wider range of spectral data and evaluating the model's performance in various agricultural contexts, future work could improve its stability.

In their research paper, (Xiaojun et al. 2021) present a new approach to the problems associated with DL models for conceptual image synthesis. The development and use of an innovative DCGAN architecture for producing high-quality images with accurate semantic details is the main focus of this paper. The authors have implemented an SPADE mechanism into the GAN framework with the goal of enhancing the spatial and semantic consistency of generated images. The model used in this study combines the SPADE method with a DCGAN architecture. Using SPADE layers to alter the feature maps according to the input conceptual designs, the GEN is intended to produce images based on conceptual segmentation maps. With this modification, the GEN can more effectively hold onto relevant data during the image production process. However, the DISC is trained to discriminate between synthetic and real images, which forces the GEN to generate outputs that are more realistic. The effectiveness of the suggested model is evaluated through the use of other qualitative evaluations and common metrics like IoU. The performance of the authors' model is compared with that of other models, such as RCNNs and traditional DCNNs. The outcomes show that the SPADE-DCGAN model performs better than earlier techniques, especially when it comes to maintaining high levels of consistency and detail across various logical regions of the generated images. The model's ability to accurately reconstruct semantic boundaries is shown by the use of IoU metrics, which represents a major improvement over initial models. The proposed SPADE-DCGAN model and other modern designs are examined in detail in this paper. It attracts attention to how well the SPADE-DCGAN performs in producing visually realistic and conceptually accurate images. According to the results, there is better semantic alignment between the generated images and the input classification maps when using the suggested model, as evidenced by its higher IoU scores. Also, qualitative comparisons indicate that when compared to other models like CNNs and LSTM-based networks, the SPADE-DCGAN generates images that are more logical, detailed, and show less conceptual error. Although the authors provide an attractive method for enhancing semantic representation of images, an additional review on more types of datasets would ensure generalizability. The model's scalability to larger or more complex datasets may be limited by the extra work introduced by dependence on SPADE, even though its success. Also, even though the improvements are clearly visible in terms of both quality and quantity, the paper does not fully investigate any potential drawbacks or errors of the model in specific situations, such as managing extremely complicated or overloaded situations.

For SSWM, (Lopez-Granados 2011) suggests evaluating the state of both closest and remote weed detection systems at the moment. The study investigates how these technologies might improve accurate farming by offering exact weed maps, which are necessary for effective weed management techniques. The author highlights the importance of using a multispectral and hyperspectral images as key instruments for weed detection at different phenological stages. The research , also highlight the difficulties in making these technologies widely available and affordable. The models of proximal and remote sensing that are used for weed detection are the main topic of this paper. To identify the spectral patterns of weeds, multispectral and hyperspectral sensors are used in remote sensing, which is based on satellite and aerial photographs. These signatures change according to various phenological stages, making it possible to identify weeds in crops. The model highlights how crucial spectral and spatial resolution are to these imaging methods. On the other hand, proximal sensing uses ground-based sensors installed on farm equipment to enable real-time weed management and detection. Many complexities are involved in the models talked about, including the requirement for high-resolution imagery in order to distinguish between different species of weeds and identify small patches of weeds. The capacity of hyperspectral imagery to identify minute variations in reflection is especially noteworthy, as this is essential for early weed detection. However, there are a number of important drawbacks, including the high expense of hyperspectral imaging systems and the difficulty of handling massive volumes of data. This paper compares the accuracy of various imaging techniques in order to evaluate how well these weed detection models perform. The results show that although hyperspectral imagery produces extremely accurate maps of weeds, particularly in the early and late stages of climate, its high costs make it impossible from a financial point of view. Even though it is less accurate, multispectral imagery is more practical and more affordable for larger-scale applications. The assessment indicates that although the existing technologies for remote sensing have a lot of promise, their high cost and complicated technical requirements prevent them from being widely used. A detailed review of the advantages and disadvantages of weed detection systems for SSWM is provided by (Lopez-Granados 2011). The technological developments in proximal and remote sensing, as well as their applications in precision agriculture, are simply highlighted in this paper. The lack of attention paid to the financial viability of these technologies for smaller farms, where the implementation of such expensive systems may not be feasible, is an important remarks, though. Also, although the technical aspects are covered in great detail in the paper, a deeper investigation into the possible environmental effects of these technologies beyond their financial and operational effects would be helpful. Although the research offers insightful information overall, it could be strengthened even more by addressing these more general effects.

In their research paper, (Dankhara, Patel, and Doshi 2019) present using the IoT to create a reliable weed detection model for modern agriculture. By employing IoT based intelligent robots that can exactly classify and distinguish between weeds and crops in real-time, the goal is to decrease the usage of dangerous herbicides. Based on image data that has been processed by a variety of SL classifiers, this classification aims to decrease human intervention and increase herbicide application quality. CNNs are the main method used by the model covered in the paper for weed detection. To achieve high accuracy in separating weeds from crops, RGB images of plants and soil are processed by CNNs. The model includes particular data that is kept on a server and is available in real time for use by robots that detect weed. Because of its capacity to manage the variability in plant-weed mixtures that vary with seasons, environments, and crops, the CNN-based method was selected. The accuracy and precision of the various weed detection techniques are compared in this paper. The provided part fails to have specific performance metrics however, the authors highlight the advantages of the CNN based approach in managing different soil types and environmental conditions. The use of IoT improves the performance of the CNN model by enabling real-time data processing and retrieval, which raises the accuracy of weed classification. The work by (Dankhara, Patel, and Doshi 2019) offers an effective way of decreasing the use of herbicides by utilizing CNNs and IoT, but it would benefit from an in-depth evaluation of how well the model works in various agricultural situations. Data from experiments showing the model's performance under different conditions and a discussion about possible drawbacks, such as the computational demands of real-time CNN processing and the difficulties of large-scale deployment in highly variable fields, could strengthen the paper.

The authors (Kulkarni and Angadi 2019) present an innovative approach that combines DL, image processing, and IoT to address the major problem of weed control in agriculture. The main goal is to create an intelligent weed detection system that uses a CNN to identify weeds among crops, minimizing the use of harmful herbicides and reducing their impact on the environment. The suggested system, which combines a Raspberry Pi, a camera for image capture, and a trained CNN to distinguish between crops and weeds, uses a CNN model for accurate weed detection. The first step in the process is data collection, which involves gathering and classifying photos of weeds and crops. After that, the data is used to train the CNN, which is split into training (70%) and validation (30%) sets. The CNN classifies the images after they have been divided using the WSA. Finally, the trained model is installed on a Raspberry Pi to process fresh photos, detect weeds, and provide farmers with separated information that will help them apply herbicides more accurately. A dataset consisting of 250 images for training and 20 images for testing was used to evaluate the effectiveness of the suggested system. With an average FRR of 7% and an average FAR of 2.6%, the system achieved an average accuracy of 85%. These metrics show that while there is a need for improvement, especially in lowering the FRR, the system is fairly effective at differentiating between crops and weeds. The suggested CNN-based approach offers a more automated and accurate solution compared to traditional methods, which frequently involve manual intervention or less advanced image processing techniques, according to a comparative analysis of existing weed detection methods provided in the paper. Though there are some areas where the research could be improved, (Kulkarni and Angadi's 2019) work represents an important step in the field of automated weed detection. Firstly, the model's ability to be applied to various crop varieties and environmental conditions may be limited by the small size of the dataset utilized for both training and evaluation. Increasing the size of the dataset and adding different images would probably improve the robustness of the model. Secondly, although the claimed accuracy is excellent, but it indicates an important error rate, especially in the case of the FRR of weed sections. To lower these errors, future research could concentrate on improving the CNN architecture or testing advanced image segmentation methods. In the end, an in-depth review of the weaknesses of the present methodology and possible future research avenues such as the addition of real-time processing capabilities and scalability to larger agricultural fields would improve the paper's overall quality.

A SLR on the use of DL methods, specifically CNNs, for weed detection in agriculture was carried out by (Murad et al. in 2023). Since 2015, there has been an important demand in research on this subject, according to the authors, mainly because of developments in DL and its potential for exact image-based recognition. The review's main objectives were to analyze trends, identify different types of weeds, compare the effectiveness of different DL techniques, and use algorithms. A survey of conventional ML methods for weed detection before DL took over was also included in the study. In spite of the encouraging results that DL particularly CNNs has shown in weed detection, the authors came to the conclusion that traditional ML algorithms still have potential and should be investigated further. In order to encourage better comparison and progress in this area, they also showed the importance of standardized image databases and evaluation metrics. The research doesn't carry out any unique experiments or suggest a new model. Rather, it conducts an extensive examination of the current state of research on DL based weed detection. The performance of different DL algorithms—mostly CNNs and their variations as documented in the reviewed papers is examined by the writers. The techniques used for image processing, the kind of algorithm (e.g., CNN, SVM, etc.), and performance metrics (e.g., accuracy, precision, recall, and F1-score) are used to classify the papers. The study also covers the kinds of crops and weeds that were used in the investigation, as well as the image methods (like RGB and NIR) and data collection techniques (like drones and robots) that were used. The analysis shows that weed detection has been effectively achieved using traditional ML algorithms (e.g., SVM and RF) and DL algorithms (e.g., CNNs). Although CNNs are used more often, the authors observe that the performance of ML algorithms is frequently comparable. They create attention to the importance of more investigation to clearly distinguish between ML and DL's performance in this field. The significance of developing standardized image datasets and evaluating standards is also highlighted in the paper in order to make possible more insightful comparisons between various methodologies. A great overview of the the latest in DL based weed detection is given by (Murad et al. 2023). A deeper discussion of each DL algorithm's drawbacks and difficulties, as well as how well-suited it is for various weed detection situations, would improve the review, however. A more critical examination of the stated performance metrics would also be beneficial to the paper, taking into factors that can greatly affect the outcomes, such as dataset size, image quality, and evaluation procedures.

According to research by (Islam et al. 2021), using pictures taken by UAVs, ML algorithms specifically, RF, SVM, and KNN can be used to detect weeds early in an Australian chili farm. Preprocessing UAV photos, extracting features like color indices and vegetation indices, and training ML models to identify pixels as weed, crop, or unwanted regions are the methods used in this study. Metrics such as recall, kappa coefficient, accuracy, and precision are used to assess the models performance. The results show that SVM and RF beat KNN (63%), achieving high accuracy (96% and 94%, respectively). The authors draw the conclusion that RF and SVM are useful and effective methods for weed identification in UAV images. The study has certain drawbacks, in spite of showing the potential of ML for weed detection. The fact that the dataset was restricted to a single chilli farm and was comparatively small could limit how useful the results can be. Furthermore, only RGB images are taken into account in this study; multispectral or hyperspectral images may offer additional information for weed discrimination. The authors notice these drawbacks and recommend more research investigating various image formats and DL techniques.

(Umamaheswari et al. 2018) present a real-time weed detection system that makes use of GPU parallel image processing. The system learns from labeled field images and uses a DL technique more accurately, the GNO model within the TensorFlow framework to identify weeds in fresh images. Metrics like precision, recall, and accuracy are used to assess the model's performance, and it is compared to a reference dataset. The results show that the suggested system achieves excellent recall and precision, significantly better than the current approach on some metrics. The system's durability and scalability to new weed types are linked by the authors to its ability to extract features directly from images without the need for manual labeling or segmentation. The system is suitable for real-time applications since GPU parallelization greatly speeds up the training process. Although the authors' method for real-time weed detection seems positive, it is not without mistakes. A single standard dataset, which might not accurately reflect the variety of actual agricultural scenarios, works as the evaluation's main backbone. In addition, the architecture of the model and hyperparameter tuning are not thoroughly discussed in the paper, which makes it challenging to replicate the results or understand the exact design decisions that impact the model's performance. It is necessary to conduct more research to determine the model's capacity for generalization and resilience to various weed species and environmental conditions.

The research done by (Bento et al. 2023) indicates locating and classifying the growth of weeds in a coffee plantation using a RPA fitted with a multispectral sensor. The study used ML algorithms, namely SVM and RF, to distinguish between subjected soil, brachiaria (a cover crop), weeds (morning glory), and coffee plants. Orthomosaic images produced from the RPA data were used to train and validate the models. The results indicated that while SVM and RF both performed well in classifying the various types of land cover, RF slightly beat SVM. The study also showed that applying targeted herbicide spraying based on the weed maps produced by the RPA and classification algorithms could result in significant cost savings (about 92.68%). The possible uses of RPA and ML for accurate weed control in coffee plantations is demonstrated by the authors' work. The study's focus on a particular weed species and coffee variety, however, places restrictions on it. The suggested method's success might differ depending on the surrounding situations or weed species. In order to confirm that the results can be generalized and investigate the possibilities of DL methods for weed visualizing and detection in coffee plantations, additional research is required.

A vision-based classification system was proposed in the research paper by (Lottes et al. 2017) for the purpose of distinguishing weeds from crops in UAV images. The system uses both RGB-only and RGB+NIR (near-infrared) images to identify common weed species in Northern Europe and the sugar beet fields. Planting detection, plant-specific feature extraction, and multi-class RF classification are the steps in the process. The idea behind the system is to take advantage of the logical arrangement of plants and, if available, to use already present knowledge about crop rows. The authors used a variety of UAVs and cameras to implement and assess their system on actual sugar beet fields. The results show that the system can successfully classify sugar beets and various weed species even in difficult situations like irregular plant arrangements and weeds growing inside of rows. Although the system performs well with RGB-only images also, the addition of NIR information was found to improve crops detection and classification performance. Their research provides a reliable and flexible method for classifying crops and weeds using UAV images, which makes an important contribution to the field of accurate farming. Evaluating the system's performance under various field conditions and with a larger variety of weed species, however, could improve the study even more. More improvements in accuracy and effectiveness might result from investigating the possibilities of DL methods for feature extraction and classification.

In the field of accuracy in agriculture, the research paper by (Peteinatos et al. 2013) provides an in-depth examination of ground-based sensor technologies that may be utilized for weed detection and the evaluation of weed infestation levels. Spectrophotometric, optoelectronic, fluorescence, and distance (LiDAR and ultrasonic) sensors are among the sensor technologies covered in the paper. The functioning values of these sensors, their possible uses in weed detection, and the difficulties in implementing them in actual agricultural environments are all covered by the authors. In order to increase the precision of weed management decisions and improve the accuracy of weed detection, the paper additionally highlights the importance of sensor combination and the integration of multiple sensor technologies. In their concluding states, the authors offer a viewpoint on the potential improvements of these sensor technologies, highlighting the importance of additional study to get previous the challenges and limitations that come with using them in real-world agricultural situations. They propose that the development of stronger and efficient weed detection systems can result from improvements in sensor hardware, algorithmic analysis, and the integration of various sensor technologies. The authors also mentioned how important it is to take into consideration factors like cost, maintenance requirements, and simplicity of use when creating commercial weed detection sensors. The ultimate objective is to create sensor technologies that can combine in effortlessly with current farming methods and give farmers the resources they need to successfully apply SSWM plans. Although the paper provides a useful overview of ground-based sensor technologies for weed detection, it does not present new study findings instead, it focuses primarily on describing the technologies and their potential applications. A deeper investigation of the relative efficacy of various sensor technologies and their applicability for different weed management scenarios would improve the paper even more. A deeper understanding of these technologies' possible effects on sustainable agriculture would also come from a discussion of the financial and environmental consequences of their usage.

In the research paper, (Razfar et al. 2022) propose a DL model-based vision-based weed detection system for weed identification in soybean farms. Three customized CNN models, ResNet50, MobileNetV2, and ResNet50 are the five DL models that the authors evaluate. Effective weed detection is the goal of the project, taking into consider the limitations of mobile installation, including low memory and processing power. The models are evaluated on a robust GPU platform (Nvidia 1080ti) and a limited in resources edge device (Raspberry Pi 4) according to their validation accuracy, speed, and duration. Based on the results, it can be realized that the particular 5-layer CNN architecture performs better than the other models. It achieves a high detection accuracy of 97.7% while using the least amount of memory and latency. The authors draw the conclusion that the soybean industry can benefit from their suggested custom DL CNN model in terms of productivity, time, and efficiency. The research done by the authors indicates that implementing DL-based weed detection systems on edge devices is possible, which makes a significant contribution to the field of precision agriculture. A more comprehensive evaluation of the models' performance in a variety of field settings and with a larger range of weed species, however, could improve the research even more. Also, investigating the possibilities of more modern DL architectures and methodologies, like GANs or attention mechanisms, may result in additional improvements in the robustness and accuracy of weed detection.

The use of remote sensing methods for weed detection in rice fields is reviewed in detail in the research paper by (Rosle et al. 2021). The difficulties with using traditional weed control techniques are covered in the paper, in addition with how remote sensing technologies might be able to help. The authors worry the importance of quickly recognizing weeds in rice fields and the range of weed species that are frequently encountered in these areas. The research then explores the various remote sensing methods such as RGB, multispectral, and hyperspectral sensors and their benefits and drawbacks for weed detection. The authors also go across how feature extraction, classification algorithms, and image processing work in weed detection. The positive impacts of remote sensing methods on weed management are highlighted in the paper's conclusion, especially in terms of the long-term financial benefits and environmental sustainability. The authors suggest that more effective and efficient weed control methods for rice fields may result from the combination of remote sensing and ML techniques. The authors give an in-depth examination of the most recent developments in remote sensing-based weed detection. A deeper investigation of the challenges and limitations related to the application of these strategies in actual agricultural settings would, however, improve the paper's current strong points. Also, an in-depth investigation of the financial and ecological advantages of implementing remote sensing-based weed detection systems could solidify the case for their broad adoption.

A region-based deep CNN is suggested in the research paper by (Sarker and Kim 2016) as a fast and accurate approach for weed identification in agricultural areas. The authors create attention to the errors of conventional weed control techniques as well as the opportunities of DL for accurate and independent weed identification. The study focuses on weed detection using a modified ResNet-101 architecture and a R-FCN. The authors use data augmentation and dropout techniques to overcome the problem of limited datasets for training deep CNNs. Using both the Faster R-CNN and R-FCN frameworks, the suggested model's performance is evaluated using MAP and compared with other CNN models, such as GoogLeNet and VGG. The results show that, when combined with dropout and data augmentation, the proposed R-FCN with ResNet-101 achieves the highest accuracy (81%) when compared to other models. The authors draw the conclusion that, especially when working with small datasets, their approach provides a straightforward but efficient solution for weed detection in farm land. The authors' work improves the field of weed detection technology by putting forward a model that tackles the problems associated with limited data and real-time processing. However, the results may not be as broadly applicable which could be due to the study's focus on a small dataset and a restricted number of weed categories. To assess the model's performance with a larger variety of weed species and in a variety of field conditions, additional research is required. In addition, exploring the possibilities of more modern DL architectures and methodologies may result in additional advancements in the robustness and accuracy of weed detection.

The YOLOv7 object detection algorithm is suggested for use in real-time weed detection in agriculture in the research paper by (Narayana and Ramana 2023). The authors create attention to the errors of conventional weed control techniques as well as the promise of DL for accurate and effective weed identification. The study focuses on using the 4weed dataset and the early crop weed dataset to identify weeds using YOLOv7, a modern real-time object detection algorithm. Metrics like precision, recall, F1-score, MAP, and precision are used to evaluate a model's performance. With MAPs of 78.53% on the 4weed dataset and 99.6% on the early crop weed dataset, the results show that YOLOv7 performs well on both datasets. The authors come to the conclusion that YOLOv7's great accuracy and efficiency make it an appropriate choice for real-time weed detection in agriculture. The authors' work highlights the potential of YOLOv7 for real-time applications and includes to the expanding body of research on DL-based weed detection. The study's dependence on a small number of datasets and weed species, however, may limit how broadly applicable the conclusions can be. To evaluate the model's performance under various field circumstances and with a larger variety of weed species, more investigation is required. In addition, investigating how to combine YOLOv7 with other technologies like robotics or UAV may result in the creation of more advanced and automated weed control systems.

An in-depth examination of precision weed management methods and weed species identification in a range of crops is provided in the research paper by (Mishra and Gautam 2021). The study highlights how important it is to use DL, specifically CNNs, for early weed detection and growth estimation. It talks about how weeds hurt crop yield and how quickly they should be identified and removed. The authors look at a range of weed classification and management techniques, such as chemical, biological, cultural, physical, and preventive methods. Using methods like GLCM and HSV for color analysis, the paper also explores the methodology of weed recognition and classification. This includes gathering images, preprocessing, segmentation, feature extraction, and classification. The authors go into more detail about the use of TL and CNNs for weed detection, highlighting the use of pre-trained models such as VGGNet, AlexNet, GoogLeNet, and ResNet. The challenges and solutions related to weed classification and detection, including closing, variable lighting, and distinct development stages, are discussed in the paper's conclusion. The authors give a useful summary of the current state of weed identification and classification, bringing insight into the various methods used in this area. To improve the paper, however, a more thorough explanation of the benefits and drawbacks of each approach as well as a performance comparison would be beneficial. Also, an additional review of the difficulties and real-world applications of these strategies in agricultural contexts would be helpful to the paper.

The use of machine vision is proposed in the research paper by (Perez et al. 2000) for the identification of broad-leaved weeds in cereal crops. The system distinguishes between weeds and crops using color and shape analysis techniques. To distinguish plants from the background, color information is used, particularly a NDI based on green and red channels. The next step is to use shape analysis techniques, such as logical features and the present characteristics, to differentiate between weed and crop plants. In order to decrease the quantity of objects who have shape analysis, the system also integrates the identification of crop row locations. By comparing results with human classification, the algorithms' performance was evaluated and they showed an acceptable success rate. According to the authors, the system is capable of accurately estimating the relative leaf area of weeds, which can be useful for aimed herbicide application and classified manual weed surveys. Due to the authors' method, which makes use of both color and shape information, weed detection in cereal crops can be successfully addressed. The study does, however, have certain drawbacks. The system's applicability to various crop and weed types or environmental conditions may be limited due to its dependence on particular color indices and shapes. Moreover, the system's performance in real-world scenarios may not be fully captured by the evaluation based on comparison with human classification. To evaluate the system's durability and flexibility in a range of agricultural settings and investigate the possibility of adding advanced ML methods for greater accuracy in weed identification, more investigation is required.

An illumination-variable, vision-based weed identification system is proposed in the research paper by (Tang et al. 2016). For image turning gray and binarization, the system uses the YCrCb color space and extracts the Cg component, which is less sensitive to changes in light. Utilizing a combination of linear scanning and vertical projection techniques, the center line of crop rows is found. To reduce computational complexity, the traditional WIR is modified. The Modified WIR within grid cells is then calculated using an improved horizontal scanning method. In the end, real-time weed spraying decisions are made using a Bayesian decision-making process under normal distribution. The accuracy of the system was 92.5%, beating both BP and SVM algorithms in the evaluation and comparison of its performance. The method used by the authors shows how accurate weed identification under various lighting conditions can be achieved by combining color space transformations and Bayesian decision-making. The study's focus on a single crop (maize) and a small number of weed species, however, may have restricted how broadly the results can be applied. To evaluate how well the system works under various field circumstances and with a larger variety of crops and weed species, more investigation is required. Also, investigating the integration of more advanced  or ML techniques may help to increase the robustness and accuracy of weed detection.

The use of DL methods for weed identification in bell pepper fields is proposed in the research paper by (Subeesh et al. 2022). Four pre-trained deep CNN models AlexNet, GoogLeNet, InceptionV3, and Xception were evaluated for sustainability by the authors. The models were evaluated using metrics like accuracy, precision, recall, and F1-score after being trained on RGB photos of bell pepper fields in various lighting conditions. With a batch size of 16 and 30 epochs, InceptionV3 beat the other models, as shown by the results. Its accuracy was 97.7%. The authors came to the conclusion that DL models, especially InceptionV3, present an option to identify weeds in bell pepper crops accurately and efficiently. This provides the way for integration with image-based herbicide applicators for accurate weed control. The analysis of various  deep CNN models and the hyperparameter optimization for weed identification in bell pepper fields are the research's strongest points. However, by evaluating the models' performance on a bigger and more varied dataset, which includes photos taken in various field settings and with various weed species, the research could be improved even more. A more thorough examination of the models' computational effectiveness and capacity for real-time application in agricultural settings would also be helpful for the research.

(Abdulsalam and Aouf 2020)  propose an innovative technique for classifying and detecting weeds in precision agriculture. This method addresses the drawbacks of current DL techniques, which frequently fail to differentiate weeds from their background. The authors present a hybrid model that combines the YOLOv2 object detector for accurate weed localization with a pre-trained ResNet-50 network for image classification. This method allows the system to classify individual weed types in addition to detecting their presence, allowing for targeted weed control strategies. Four classes of common cornfield weeds were included in the dataset of 4000 images that were used to train the model. With an overall accuracy of 99%, the performance evaluation, which was based on precision and recall metrics, showed how effective the proposed approach was. The authors point out that their approach beats a Faster R-CNN based model, especially when it comes to training loss and detection accuracy. Faster and more stable learning was shown by the fused ResNet-YOLO model's more smooth and minimal training loss. It also performed better in identifying and classifying specific weed classes than the Faster R-CNN model, indicating its usefulness for real-time weed control in accurate farming. The study by the authors provides a hybrid DL model for accurate weed detection and classification, which is an important addition to the field of precision agriculture. Specific weed control actions are made possible by the accurate localization and identification of weed species made possible by the combination of YOLOv2 for detection and ResNet-50 for classification. The model's considered to be high efficiency and accuracy provide additional support to its potential for practical uses. The study's focus on a small dataset and a particular crop (corn), although may restrict how broadly the results can be applied. To evaluate the model's performance with a larger variety of crops and weed species and in a variety of field conditions, more investigation is required. The authors also note that by using a more typical and balanced training dataset, there may be an opportunity for improvement in the identification of some weed classes, like sedge.

A thorough review of the literature and background information on the application of DCGAN for weed detection in agriculture are provided in this chapter. The suggested methodology focuses on augmenting training datasets with artificial weed images produced by a DCGAN, consisting of a discriminator and a generator. By enhancing small datasets with a variety of high-quality synthetic images, this approach aims to increase the classification accuracy of weed images while also improving the model's generalization abilities. CNNs and RNNs are two examples of the DL approaches addressed in the review, and their efficacy in weed detection is highlighted. Additionally, it talks about the difficulties caused by class imbalance in datasets and investigates how DCGANs, when paired with methods like SMOTE, might help with these problems. The use of DL and GANs in agricultural settings is illustrated through a number of studies, highlighting the advantages and disadvantages of these technologies in real-world scenarios. The scalability, expense, and environmental effects of applying such modern technologies in actual farming situations are also discussed in the document.

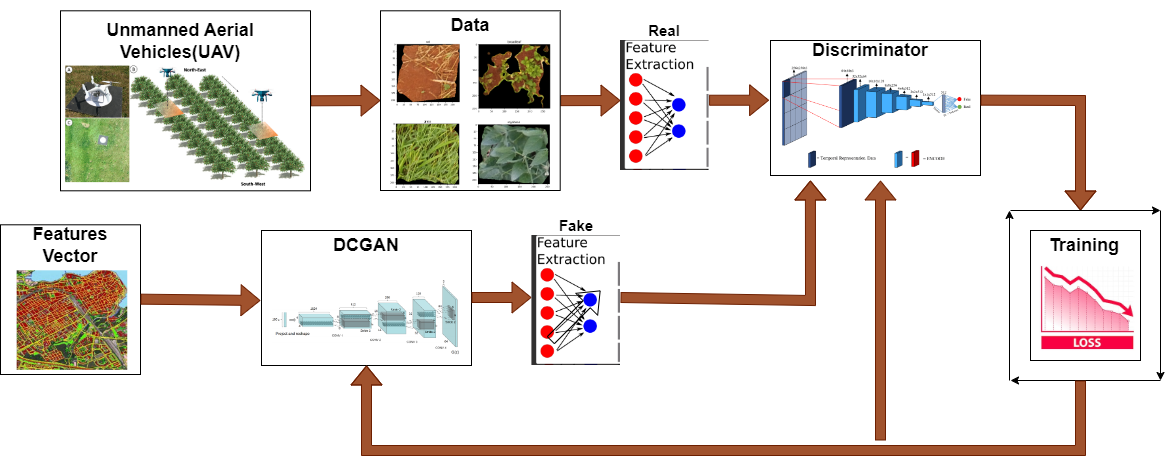
# Unsupervised Representation Learning With DCGAN for Weed Classification

## 3.1. Proposed DCGAN-based Weed Classification

In this section, we look into the use of a DCGAN to improve data augmentation for the classification of weed images. Two NNs make up the DCGAN: a generator that generates realistic weed images from random noise and a discriminator that separates real images from synthetic ones. By means of competing training, the generator's capacity to generate superior synthetic images is improved, therefore augmenting the variety and quantity of training data. This augmentation improves the model's ability to generalize by adding variances to the backgrounds, lighting, and weed appearances. We use these artificial images with actual data to enable the development of more reliable and accurate weed classification models. This section describes the framework, which includes feature extraction, DCGAN training, preprocessing, data collection, and CNN-based classification. The model's performance is significantly enhanced by the augmented dataset, which combines synthetic and real images. This eventually allows for more accurate weed detection in soybean fields and supports effective, environmentally friendly weed management techniques.

## 3.1.1. Deep Convolutional GAN

In the context of classifying weed images, we utilize a DCGAN as an efficient method for data augmentation in this work. Two NNs make up the DCGAN architecture: a discriminator that learns to distinguish between real and artificial images and a generator that learns to generate synthetic weed images from random noise as shown in **Figure 3.1.1.** The generator's ability to create realistic weed images significantly improves through an adversarial training process, in which the discriminator aims to correctly identify fakes and the generator tries to fool it. We effectively increase the diversity and volume of training data available for the next weed classification models by carefully gathering these high-quality synthetic images during training and integrating them into the original dataset. Because it creates variations in weed looks, shades of light, and backgrounds, this augmentation technique is especially useful when working with small real-world datasets. This enhances the model's capacity to generalize to unseen examples, which ultimately improves its classification accuracy and robustness in real-world scenarios.



**Figure 3.1.1:** **Framework for Weed Detection in Soybean Fields using a DCGAN and CNN**

A workflow that combines UAVs and a DCGAN for potential purposes like image evaluation and improvement in jobs like weed detection in agriculture is shown in **Figure 3.1.1**. Let's examine how each element of the graph flows:

1. **UAV**

UAVs, denoted by a drone icon, start the process by taking aerial pictures of the ground. These photos offer a bird's-eye perspective of the agricultural fields. UAVs or drones are airplanes that run without a human pilot present. They can fly independently using pre-programmed flight routes or more advanced dynamic automation systems, or they can be remotely controlled by a human operator. Precision Agriculture, aerial photography, surveillance, mapping, and communication systems are just a few of the many tasks that UAVs can carry out. UAVs are utilized in agriculture, for example, to monitor crops, analyze soil, and find weeds. By providing real-time data, these drones can help improve farming practices. Because of their adaptability and capacity to enter places that humans might find challenging or hazardous, UAVs are crucial in a variety of fields, such as transport, environmental monitoring, and the military.

1. **Data and Preprocessing**

The initial stage involves gathering information via aerial photos, which most likely provide a broad view of the soybean field (dos Santos Ferreira et al., 2017). After that, these pictures are processed probably using methods like object detection or image segmentation to extract specific weed samples. In order to guarantee accuracy for the analysis that follows, the extracted weed samples are further processed to normalize their size and format.

To make sure the CNN model receives high-quality, well-structured input, there are a few crucial steps involved in the collection and preparation of data for the CNN model's training to classify different types of weeds. First things first, a large dataset with pictures of different kinds of weeds labeled needs to be gathered. To help in the model's good generalization, these photos must show a variety of lighting scenarios, viewpoints, and backgrounds found in actual agricultural fields.

The data preparation stage starts after the dataset has been gathered. Accurate labelling of the images is required for SL, and this is what this involves. Every image has a label that identifies the kind of weed it depicts. These category labels are translated into numerical values using a mapping dictionary (label\_to\_index) in order to help in the understanding of the model. Each type of weed is given a unique index by this dictionary, which formats the labels so that they can be used for model training.

Data transformations are then used to add to and standardize the dataset. With the transforms in PyTorch. Format is applicable a number of image transformations, such as resizing photos to a standard size (in this case, 28x28 pixels), converting them to tensors, and normalizing the pixel values. Normalization guarantees that the pixel values are on a common scale, which facilitates quicker and more stable training. It is commonly performed using the mean and standard deviation of the RGB channels.

Also, the training dataset can be made more diverse by applying DCGAN-based data augmentation. This method increases the robustness of the model and makes it invariant to these kinds of transformations. In order to manage the dataset effectively, a custom dataset class called WeedDataset is finally developed. This class is compatible with PyTorch's DataLoader for batch processing during training, as it contains methods for labelling, loading images, and performing transformations.

1. **Addressing Class Imbalance**

In ML, class imbalance refers to the situation where certain classes have a lower representation in the dataset than others. This means that some weed types may have significantly fewer images than others in the context of weed classification. Predictions that are biased in favor of the majority class can result from class imbalance, which can negatively impact the classification model's performance.

1. **Effects on Classification Performance**
2. **Biased Predictions:** Despite the input image, the model may develop a bias in favor of the majority class and predict it more frequently. This occurs as a result of the majority class's loss function, which can often be affected by the frequency of each class, minimizing errors more successfully.
3. **Poor Generalization:** The model's inability to effectively learn to identify characteristics of minority classes could result in poor generalization. This is especially problematic for weed detection applications, where accurate detection of all weed species is essential for efficient agricultural management.
4. **Metric Damage:** In unbalanced datasets, performance metrics like accuracy can be misleading. Even if the minority class predictions are off, high accuracy could still be attained by just projecting the majority class.
5. **Methods for Reducing Class Inequality**
6. **Data Augmentation:** You can balance the distribution of classes by adding synthetic images produced by methods such as DCGAN to the dataset. The model can learn more accurate representations of these classes by producing realistic images of poorly represented weed types.
7. **Resampling Techniques:** To balance the dataset, utilize techniques such as oversampling the minority class and undersampling the majority class. Undersampling lowers the number of examples in the majority class, whereas oversampling increases the number of duplicates or new examples of the minority class.
8. **Class Weights:** To encourage the model to focus more on these classes, give the minority class in the loss function a higher weight. Due to this modification, it becomes more expensive to incorrectly classify minority classes during training.
9. **Synthetic Data Generation:** To increase the variety and volume of training data, additional techniques such as SMOTE (Synthetic Minority Over-sampling Technique) can be employed in addition to DCGANs to create synthetic examples for minority classes.
10. **Evaluation Metrics:** Taking into account the accuracy of minority class predictions, metrics like precision, recall, F1-score, and area under the ROC curve (AUC-ROC) can offer a more fair assessment of model performance.
11. **Real Feature Extraction**

A Real Feature Extraction block is applied to the pre-processed real data. This block takes the real images and extracts high-level features from them, most likely using a CNN. These features, which are important for training the discriminator, capture the most important characteristics of the original data.

1. **DCGAN for Data Augmentation**

A DCGAN model is fed with the processed images of weed. The DCGAN's generator component tries to create artificial weed images that closely replicate real ones by using a random noise vector as input. On the other hand, the DCGAN's discriminator component is trained to discern between actual cannabis images and the artificial ones produced by the generator. The two networks take part in an adversarial learning procedure in which the discriminator attempts to accurately classify the images as real or fake, and the generator attempts to produce images that fool the discriminator.

The generator and discriminator improve their performance iteratively during this training session. While the discriminator gets stronger at recognizing fake images, the generator gets better at creating realistic weed images. The generator eventually produces high-quality, similar-looking fake weed images as a result of this competitive process.

1. **Fake Feature Extraction**

The artificial images produced by the DCGAN are put through a "Fake Feature Extraction" block, just like the real data. This block takes features from the fake images, most likely using a CNN again.

1. **Feature Extraction and CNN for Classification**

Feature extraction is applied to both fake weed images produced by the DCGAN and real weed images. The process of feature extraction involves transforming an image's raw pixel data into a set of example features that sum up the key characteristics of the weeds. As shown in the diagram, a CNN is usually used for this.

A CNN-based classifier receives the extracted features and the corresponding labels (fake or real). To learn small variations between real and fake images, the classifier is trained on both sets of images. The objective is to create a model that can distinguish between real weed images and fake images generated by a GAN with accuracy.

In order to train the discriminator, a loss function that measures how well it can classify the images must be minimized. The discriminator's performance is improving over time, as seen by the diagram's decreasing pattern in the training loss.

1. **Discriminator**

Features from both the fake and real images are fed into the discriminator. Its goal is to differentiate between these two sets of characteristics and categorize them as "real" or "fake."

1. **Training Loop and Loss**

A loss function is calculated by comparing the discriminator's classifications with the real or fake ground truth labels. This loss represents the discriminator's inaccuracy in identifying authenticity from fraudulent images. The discriminator and generator are then updated with the loss through backpropagation, which enhances their performance in the subsequent iteration. This process of minimizing the loss over time is shown in the "Training" box with the downward-sloping graph.

This integrated framework has a lot of uses in accurate farming. The amount of training data that is available for weed detection models can be significantly boosted by the capacity to produce synthetic weed images. This is especially helpful in situations where it is costly or difficult to gather a significant amount of labeled weed data. The CNN classifier can be trained using both synthetic and real data, which will significantly improve the model's ability to extend previously unseen weed samples.

Targeted and effective weed management techniques can then be implemented due to the enhanced weed detection model's ability to produce accurate weed diagrams of agricultural fields. This may result in a smaller amount of being used, reducing agriculture's negative environmental effects while increasing crop yields.

3.1.1.1 Generator

This section explores the architecture and operation of the DCGAN generator that creates images of synthetic weed. The generator model, shown in **Figure 3.1.1.1**, uses a sequence of transposed convolutional layers to convert an input 100-dimensional noise vector into a realistic 28x28 RGB image. With the exception of the last layer, these layers slowly raise the image dimensions while decreasing the number of channels. Each transposed convolution is followed by Batch Normalization and ReLU activation. The last layer uses a Tanh activation function, which is widely used to normalize image data, to normalize the output pixel values between -1 and 1.

The generator architecture guarantees that the generated images closely resemble real-world weed images by including specific configurations for channel numbers, strides, and kernel sizes at each layer. The method for initializing the generator class is covered in more detail in this section. It includes functions for defining the forward pass, reshaping the input noise tensor, and building the network layers. With the help of this structured process, the generator can generate high-quality synthetic images iteratively, expanding the training dataset and enhancing the reliability and performance of the weed classification models that follow.



**Figure 3.1.1.1 Generator**

A generator model used for image generation likely inside a GAN is shown in **Figure 3.1.1.1.** The input of the model is a 100-dimensional noise vector that is first transformed into a 64-channel visual representation (the "Reshape" layer). Next, a sequence of transposed convolutional layers (ConvTranspose2d) processes this reshaped input. Each of these layers reduces the number of channels while increasing the image's overall dimension. With the exception of the final layer, each transposed convolution is followed by Batch Normalization (BatchNorm2d) for stable training. A ReLU activation function introduces non-linearity, which helps in the learning of complex patterns.

In order to constrain the output pixel values to be between -1 and 1, which is a standard normalization for image data, the final layer applies a Tanh activation. A 28x28 image with three channels (probably RGB) that represents a fake image that was created is the model's output.

The generator contains exact kernel sizes, strides, and number of channels used in each layer, which are also represented visually in the figure. The generator can produce realistic images that closely resemble the training data.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm 1: To Initialize Generator Class** | | | | |
|  | **Initialization parameters for Generator Class:** Image channels **im\_chan**, hidden dimension **hidden\_dim**, and noise dimension **z\_dim**. | | | |
| **1** | **Generator Function:** make\_gen\_block(self, input\_channels, output\_channels, kernel\_size=3, stride=2, final\_layer=False) | | | |
| **2** |  | **If final\_layer is False:** | | |
| **3** |  |  | **Return nn.Sequential(nn stands for Neural Network) with:** | |
| **4** |  |  |  | nn.ConvTranspose2d(input\_channels, output\_channels, kernel\_size=kernel\_size, stride=stride) |
| **5** |  |  |  | nn.BatchNorm2d with output\_channels |
| **6** |  |  |  | nn.ReLU activation. |
| **7** |  | **Else:** | | |
| **8** |  |  | **Return nn.Sequential with:** | |
| **9** |  |  |  | nn.ConvTranspose2d(input\_channels, output\_channels, kernel\_size, stride) |
| **10** |  |  |  | nn.Tanh activation. |
| **11** | **Define unsqueeze\_noise Function** // Random noise tensors are produced. | | | |
| **12** |  | **Input:** noise | | |
| **13** |  | **Output:** Reshaped noise tensor with dimensions (n\_samples, z\_dim, 1, 1). | | |
| **14** | **Define forward Function //** The reshaped noise is passed through the generator network to produce images. | | | |
| **15** |  | **Input:** noise | | |
| **16** |  | **Output:** Generated images. | | |
|  | | | | |

**Algorithm 3.1.1.1 Algorithm for the Generator**

**Algorithm 3.1.1.1** lists the steps taken by the generator to generate the fake images based on the noise. To ensure that a GAN can produce realistic images from random noise, the initialization algorithm for a Generator class entails multiple crucial steps. First, the Generator class's initialization parameters are defined. These include the number of image channels (im\_chan), the size of the hidden dimension (hidden\_dim), and the size of the noise dimension (z\_dim). The layers and functions of the generator must be configured using these parameters.

The make\_gen\_block function, which builds individual blocks of the generator network, is the generator's important component. Input\_channels, Output\_channels, kernel\_size, stride, and a boolean final\_layer flag are the parameters taken by this function. To build a sequential block of neural network layers is the aim of this function. The function returns a sequential block with a ConvTranspose2d layer that performs the transposed convolution operation, essentially upsampling the input, if the final\_layer flag is set to False. Subsequently, a BatchNorm2d layer is added to enhance training stability by normalizing the output. A ReLU activation function is then used to introduce non-linearity. Together, these layers enable the output image to acquire and produce complex features.

A sequential block with only a Tanh activation function and a ConvTranspose2d layer is returned by the function if the final\_layer flag, which indicates that this block is the last layer of the generator, is True. The output is scaled by the Tanh activation function to a range of -1 to 1, which is appropriate for image data.

The unsqueeze\_noise function is another essential function in the Generator class. The random noise tensor that is used as the generator's input must be reshaped by this function. The noise tensor is fed into it and reshaped so that its dimensions are (n\_samples, z\_dim, 1, 1), where n\_samples is the batch number. The generator network can now receive this reshaped noise tensor.

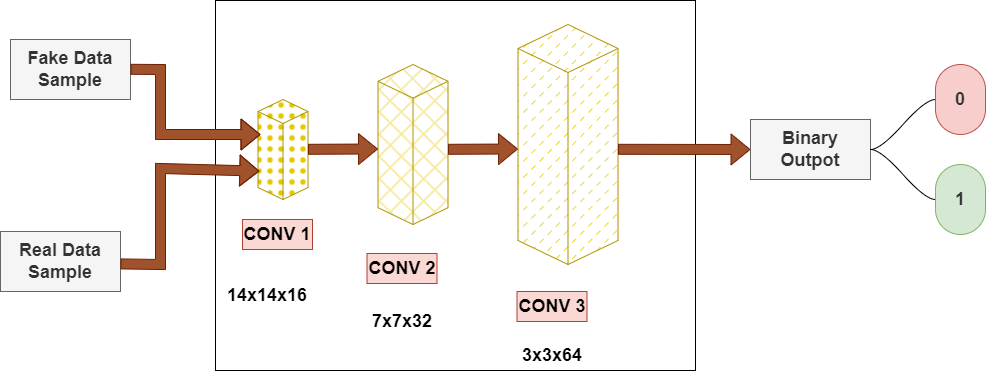
The generator network's forward pass is defined by the forward function. To create images, it feeds the reshaped noise tensor through the layers of the generator. The make\_gen\_block function generates sequential blocks, which are then applied one after the other to convert the input noise into the final output image. The generated images, which are intended to look like actual images from the training dataset, are finally produced by the forward function.

To sum up, this algorithm describes how to initialize and build a Generator class in a GAN. The Generator class is capable of producing realistic images from random noise by setting beginning parameters, building generator blocks with the make\_gen\_block function, reshaping noise input with the unsqueeze\_noise function, and implementing the forward pass in the forward function. Together, these parts form a network that can learn and produce high-quality images. This network is essential to the GAN framework because it can exceed a discriminator network in terms of output quality. With this logical approach, the generator is guaranteed to be able to learn and generate images that closely resemble the real images found in the training set.

3.1.1.2 Discriminator

This section looks at the design and operation of the Discriminator model, which is a crucial part of the GAN framework and is shown in **Figure 3.1.1.2.** The discriminator's main job is to distinguish between the false images produced by the GAN and the actual images from the dataset. Three convolutional layers form the discriminator: CONV 1, CONV 2, and CONV 3. As the input image moves through the network, each layer extracts features that are eventually more complex. After processing the input image, the first convolutional layer (CONV 1) creates a 14x14 feature map with 16 channels. This feature map is further processed by the second layer (CONV 2), which reduces its dimension to 7x7 and provides 32 channels. The last convolutional layer (CONV 3) adds 64 channels while decreasing the dimension of space even more to 3x3.

The feature map is evaluated after going through these convolutional layers to produce a binary output that represents the discriminator's evaluation of the image's authenticity real or fake. Because GANs are adversarial, both the discriminator and the generator get better over time. As the generator generates more realistic images, the discriminator gets better at telling fake images from real ones.



**Figure 3.1.1.2: Discriminator.**

The architecture of a discriminator model, an important component of GANs, is shown in **Figure 3.1.1.2.** The main function of the discriminator is to differentiate between fake images (Fake Data samples) produced by the GAN network and real images (Real Data Samples) from a dataset.

CONV 1, CONV 2, and CONV 3 are the convolutional layers that make up the discriminator. As the data moves through the network, each convolutional layer applies a set of filters to the input image, extracting features that get more and more complex. With 16 channels and a resolution of 14 x 14 pixels, the input image is processed by the first convolutional layer (CONV 1). It generates a 14x14 feature map with 16 channels. After that, this is sent to the second convolutional layer (CONV 2), which increases the number of channels to 32 while decreasing the size of the pixels to 7x7. The spatial dimensions are further reduced to 3x3 and the number of channels is increased to 64 by the third and final convolutional layer (CONV 3).

Once the feature map has passed through the CONV layers, it is processed to produce a binary output. This output shows the DISC's evaluation of the being real of an input image, which is shown by the green circle labeled "1" and the red circle labeled "0" respectively. The model is trained to identify small variations between real and fake images, maximizing its capacity to classify them correctly. GANs are adversarial in nature, meaning that both networks get better over time as the GAN tries to trick the DISC. As a result, the discriminator gets better at spotting fakes while the generator produces images that are more and more realistic.

|  |  |  |  |
| --- | --- | --- | --- |
| **Algorithm 2: To Initialize Discriminator Class** | | | |
|  | Define the **DISC class inheriting from nn.Module(nn stands for Neural Network).** | | |
| **1** |  | Initialize the class with parameters **im\_chan** (number of channels in the images) and **hidden\_dim** (inner dimension). | |
| **2** |  | Call the parent class **(nn.Module)** constructor using super(DISC, self).\_\_init\_\_(). | |
| **3** |  | Create the sequential model **self.disc** which consists of multiple **DISC** blocks. Each block is created using the **make\_disc\_block** method. | |
| **4** | Create **DISC** Function (**make\_disc\_block** method) | | |
| **5** |  | Define the **make\_disc\_block** method with parameters: **input\_channels,** **output\_channels, kernel\_size, stride, padding, and final\_layer.** | |
| **6** |  | Check if the block is the final layer using the **final\_layer** flag. | |
| **7** |  | If **not** the final layer: | |
| **8** |  |  | Create a **sequential block** with **nn.Conv2d, nn.BatchNorm2d, and nn.LeakyReLU layers.** |
| **9** |  | If **it is** the final layer: | |
| **10** |  |  | Create a sequential block with only the **nn.Conv2d** layer. |
| **11** | Create Forward Function (**forward** method) | | |
| **12** |  | Define the **forward** method with a parameter **image**. | |
| **13** |  | Pass the input image through the sequential model **self.disc** to get the DISC's prediction (**disc\_pred**). | |
| **14** |  | Reshape **disc\_pred** using a view to ensure it is a **1-dimensional tensor** per input image. | |
| **15** |  | Return the reshaped tensor **disc\_pred.** | |

**Algorithm 3.1.1.2 Algorithm for the Discriminator**

In a GAN framework, the discriminator class is an important part of the NN architecture and is initialized by **Algorithm 3.1.1.2**. Its purpose is to distinguish between real and generated images. There are multiple steps in the algorithm for initializing the Discriminator class in a neural network model. The DISC class, which derives from PyTorch's nn.Module class, is defined first. Two parameters are passed in to the initialization method (\_\_init\_\_): hidden\_dim, an internal dimension parameter that controls the number of features in the hidden layers, and im\_chan, which indicates the number of channels in the input images. Next, we use super(DISC, self) to invoke the parent class's constructor, nn.Module.\_\_init\_\_ (). This makes sure that all the required properties and methods from nn.Module are correctly initialized into the DISC class.

After initializing, we build a sequential model called self.disc that consists of multiple DISC blocks. Making a disc block method is used to construct these blocks. This method, which defines the various layers and their configurations within each block, is an essential component of the DISC class. The make\_disc\_block method requires the following parameters: padding, which specifies the amount of input padding, kernel\_size and stride, which define the size and stride of the convolution filter, and input\_channels and output\_channels, which define the number of input and output channels for the convolutional layers. A boolean flag called final\_layer is an extra parameter that indicates whether or not the block is the last layer in the sequence.

The make\_disc\_block method generates a sequential block with a nn.Conv2d layer, a nn.BatchNorm2d layer, and a nn.LeakyReLU activation function if the block is not the final layer. While the nn.BatchNorm2d layer normalizes the outputs to enhance training stability and performance, the nn.Conv2d layer carries out the convolution operation, extracting features from the input image. The model gains non-linearity from the nn.LeakyReLU activation function, which enables it to learn complex patterns. The method generates a sequential block with only the nn.Conv2d layer if the block is the final layer. Without using any additional activation or normalization functions, this layer outputs the final mapping of features.

The forward method, which controls how the input image tensor travels through the discriminator network, is implemented after the blocks have been defined. The input image tensor is represented by an image parameter that is accepted by the forward method. After passing this tensor through the sequential model self.disc, disc\_pred the DISC's prediction is obtained. Next, to guarantee that the disc\_pred tensor is a 1-D tensor for every input image, it is reshaped using the view method. The final result of the DISC is this reshaped tensor, which indicates whether or not the input image is real.

To summarize, the process of initializing the DISC class involves creating a custom class that inherits from nn.Module, initializing it with parameters for the image channel and hidden dimension, using the make\_disc\_block method to create a sequential model made up of multiple DISC blocks, and defining the model's forward pass through the forward method. By using a structured approach, the Discriminator can learn from the features extracted by its convolutional layers, making it capable of differentiating between real and fake images.

3.1.1.3 Data Extraction

In this section, we go into how to use PyTorch's Dataset and DataLoader to prepare and process image data for model training. To guarantee that the photos are consistently preprocessed and prepared for the model, the data extraction algorithm consists of multiple distinct steps. In order to meet the needs of the model, we first define a list of labels that correspond to the various classes .We also set parameters like image size and batch size. Then, in order to make handling during model training easier, we translate these labels to numerical values using a label\_to\_index dictionary. We use the transforms. Compose function to apply a number of transformations, such as resizing, converting to tensors, and normalizing, in order to prepare the images.

WeedDataset is a new dataset class that we developed by deriving from PyTorch's Dataset class. Labels, the label\_to\_index dictionary, and the transformation operations are initialized for this class. It also contains methods for retrieving and processing images along with their labels, as well as for determining the length of the dataset. The getitem function takes an image, reads it, transforms it into RGB, applies the transformations, and then returns the transformed image in tensor form with its label.

We initialize the WeedDataset class and construct a data loader using PyTorch's DataLoader. In order to ensure effective and seamless data handling during model training, this data loader handles batch processing, shuffling, and parallel data loading. We can efficiently manage big datasets, perform the required transformations, and get the data ready for training reliable and accurate weed classification models by using this structured approach.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm 3: Data Extraction Using PyTorch Dataset And Dataloader** | | | | |
|  | Define the list of **labels** and **parameters**: | | | |
| **1** |  | Create a **list** of labels containing **label names**. | | |
| **2** | Set the **image\_size** to match the input size expected by the model. | | | |
| **3** | Set **batch\_size** for the data loader. | | | |
| **4** | Convert **labels** **to numerical values:** | | | |
| **5** |  | Create a dictionary **label\_to\_index** mapping each label to a unique index number. | | |
| **6** | Use **transforms.Compose** to create a series of image transformations | | | |
| **7** |  | Convert images to **PIL format.** | | |
| **8** |  | Resize images to the specified **image\_size.** | | |
| **9** |  | Convert **images** to **tensors.** | | |
| **10** |  | Normalize the images with mean and standard deviation values (assuming RGB images). | | |
| **11** | Define a custom dataset class **WeedDataset:** | | | |
| **12** |  | In the **\_\_init\_\_** method | | |
| **13** |  |  | Initialize the dataset with **labels, label\_to\_index, and transform**. | |
| **14** |  |  | Create empty lists **image\_paths** and **image\_labels** to store image file paths and their corresponding labels. | |
| **15** |  |  | Loop through each **label** in **labels:** | |
| **16** |  |  |  | Construct the dataset path for each label. |
| **17** |  |  |  | Use **glob.glob** to get all file paths in the dataset directory. |
| **18** |  |  |  | Append each file path to **image\_paths.** |
| **19** |  |  |  | Append the corresponding label index to **image\_labels**. |
| **20** |  | Define the **\_\_len\_\_ method** to return the **length** of **image\_paths**. | | |
| **21** |  | Define the **\_\_getitem\_\_** method: | | |
| **22** |  |  | Retrieve the **image file path** and **label index** for the given idx. | |
| **23** |  |  | Read the **image** using **cv2.imread** and convert it **to RGB using cv2.cvtColor.** | |
| **24** |  |  | Apply the transformations if they exist. | |
| **25** |  |  | Convert the label to a tensor. | |
| **26** |  |  | Return the transformed image and label the tensor. | |
| **27** | Instantiate the **WeedDataset** with **labels, label\_to\_index, and transform.** | | | |
| **28** | Use **DataLoader** to create a data loader from the dataset with **specified batch\_size, shuffling enabled, and the number of worker threads.** | | | |

**Algorithm 3.1.1.3 Algorithm for the Data Extraction.**

In order to prepare and process image data for model training, a set of well-defined steps are involved in **Algorithm 3.1.1.3** which uses PyTorch's Dataset and DataLoader. To begin, we create a list of labels for the classes like various weed classes that we wish to categorize. This list facilitates the process of assigning a label to each image. In order to guarantee that every image fed into the model has a consistent size, we then adjust the image size to match the input size that our model expects. Additionally, we provide the data loader with the batch size, which defines how many samples must be processed before the internal model parameters are updated.

We construct a dictionary called label\_to\_index in order to translate the labels into numerical values. This dictionary makes it easier to convert categorical labels into the numerical format needed for model training by assigning each label to a distinct index number. After that, we apply transforms.To create a sequence of image transformations, use Compose from PyTorch. If the images are in RGB format, these transformations include converting them to PIL format, resizing them to the desired image size, converting them to tensors, and normalizing them using mean and standard deviation values. By taking these steps, you can be sure that the images have been preprocessed consistently and are ready to be fed into the model.

Next, we create a custom dataset class called WeedDataset, which is derived from the Dataset class in PyTorch. The label\_to\_index dictionary, the transform operations, and the list of labels are used to initialize the dataset in this class's \_\_init\_\_ method. To store the file paths of the images and the labels that go with them, we make two empty lists called image\_paths and image\_labels. We create the dataset path for each label in the labels list, get all file paths from the directory using glob.glob, and then append these paths to image\_paths. In order to create a mapping between image files and their labels, we additionally append the corresponding label index to image\_labels.

The length of image\_paths, which represents the total number of samples in the dataset, is the return value of the \_\_len\_\_ method. For a given index (idx), the \_\_getitem\_\_ method returns the image file path and label index. It uses cv2.imread to read the image, cv2.cvtColor to convert it to RGB format, and if any transformations are available, it applies them. Next, the label is transformed into a tensor. The transformed image and matching label tensor are returned by this method, preparing it for model training.

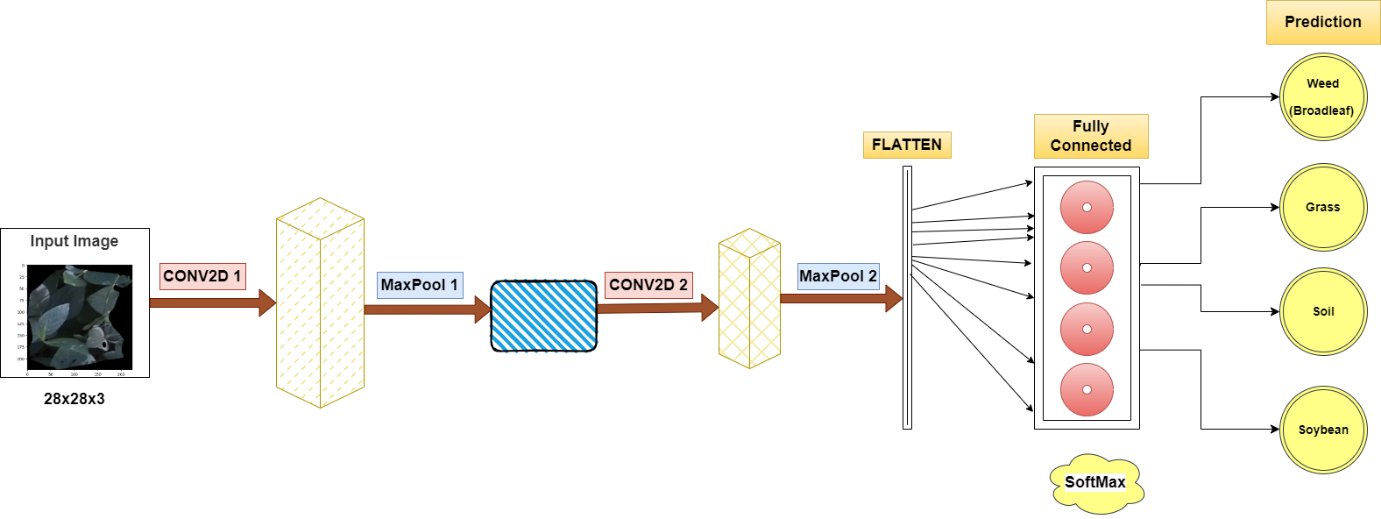
We instantiate the WeedDataset with the list of labels, the label\_to\_index dictionary, and the transform operations after defining the custom dataset class. Using this dataset, we define the number of worker threads for parallel data loading, enable shuffling to guarantee random sampling and construct a data loader with the selected batch size. During training, the data loader effectively handles the batches, rearranging, and loading of data, resulting in a smooth and effective procedure.

To sum up, this algorithm describes in detail how to use PyTorch's Dataset and DataLoader to prepare and process image data. Through the creation of a personalized dataset class and the utilization of PyTorch's strong information loading features, we can effectively manage substantial datasets, implement essential modifications, and guarantee that the data is appropriately structured for training models. The efficient and seamless training of  models on image data is made possible by this methodical approach.

3.1.1.4 Classification based on DCGAN augmented data

This section details the use of augmented data produced by a DCGAN to implement a CNN model for classifying weed in the soyabean farm . The CNN architecture, which is shown in Figure 3.1.1.4, uses three color channels (RGB) to process input images of fields that are 28 by 28 pixels in size. The max-pooling layers (MaxPool2D) in the model reduce the spatial dimensions and computational load while retaining important features. This is done after a series of convolutional layers (Conv2D) apply filters to the input image to extract features. The fully connected layers (Dense) in the model come after the convolutional layers and are responsible for learning complex patterns and connections between the weed classes and the extracted features. The final output layer creates probabilities for classifying the images into four categories: broadleaf, grass, soil, and soybean using a SoftMax activation function.

The CNN model's algorithm consists of multiple steps: using Keras' Sequential API to define the model structure; convolutional and pooling layers added for feature extraction; flattening the output for the fully connected layers; and model compilation using suitable loss and optimization functions. Transposing the channel dimensions to match the input format required by Keras and converting PyTorch tensors to NumPy arrays are two steps in the data preparation process. The augmented data is then used to train the model, which involves batch processing and several epochs to maximize accuracy and performance. By using the enhanced data produced by DCGAN, this methodical approach guarantees that the CNN model learns to classify weed images with effectiveness and accuracy.



**Figure 3.1.1.3 CNN Model.**

A CNN model created especially for classifying different types of weeds in an agricultural environment is shown in **Figure 3.1.1.3**. An image of a field that might contain weeds, measuring 28 by 28 pixels and with three colour channels (RGB), is fed into the model.

Convolutional filters, particularly Conv2D 1, are the first layer that are applied to the input image. A convolution function is carried out by each filter, which is a 3x3 matrix that moves over the image to extract local features. As a result, 32 feature maps are produced, each of which illustrates a distinct pattern or texture found in the original image. These filters are trained to recognize the distinctive qualities of different kinds of classes.

The next step involves a MaxPool 1 layer, a pooling operation that reduces the spatial dimensions of the feature maps. It does this by taking the maximum value within a 2x2 region of each feature map, effectively down sampling the image while retaining the most salient features. This not only reduces the computational burden on subsequent layers but also helps the model focus on the most relevant information for classification.

Following reduction, the feature maps are run through Conv2D 2, another convolutional layer with 64 filters. This layer, like Conv2D 1, combines the data from the feature maps of the previous layer to extract higher-level features. These higher-level characteristics might be complex patterns or shapes that are characteristic of particular classes of weeds.

The overall dimensions of the feature maps are further reduced by an additional MaxPool 2 layer. The data is then ready for the fully connected layers by the Flatten layer, which converts the multi-dimensional feature maps into a one-dimensional vector.

There are 128 neurons in the Fully Connected layer, also known as the Dense layer. This layer is in control of understanding complicated relationships between the various weed classes and the extracted features. To add non-linearity and help in the model's ability to learn complex patterns, a non-linear ReLU activation function is used.

Lastly, there are four neurons in the output layer, which is also a dense layer. These neurons stand for the four different classes of weeds: broadleaf, grass, soil, and soybean. The raw output values are transformed into probabilities using the SoftMax activation function, where each probability represents the model's confidence in assigning the input image to a specific weed class. The final prediction is made for the class with the highest probability.

To summarize, the CNN model effectively uses convolutional and pooling layers to extract relevant characteristics from the input images. Subsequently, fully connected layers are employed to classify the images into various categories of weeds. The model's capacity to generalize improves and overfitting is reduced with the addition of batch normalization and dropout layers.

|  |  |  |  |
| --- | --- | --- | --- |
| **Algorithm 4: CNN Model For Weed Classification** | | | |
| **1** | Define the **Model**: | | |
| **2** |  | Create an instance of **Sequential** from the **Keras** library. | |
| **3** | Add **Convolutional** Layers: | | |
| **4** |  | Add a **Conv2D layer** with **32 filters**, a **kernel** **size** of **3x3, a ReLU activation function,** and **an input shape (28, 28, 3).** | |
| **5** |  | Add a **MaxPooling2D** layer with a **pool size of 2x2.** | |
| **6** |  | Add another **Conv2D layer** with **64 filters**, a **kernel size of 3x3**, **and ReLU activation function.** | |
| **7** |  | Add another **MaxPooling2D** layer with a **pool size of 2x2.** | |
| **8** | **Flatten** the Output: | | |
| **9** |  | Add a **Flattened layer** to convert **the 2D matrix to a vector.** | |
| **10** | Add **Dense Layers**: | | |
| **11** |  | Add **a Dense layer** with **128 units** and **ReLU activation function.** | |
| **12** |  | Add a **Dense layer** with **4 units** and **softmax activation function** to classify into **4 categories.** | |
| **13** | Compile the **Model**: | | |
| **14** |  | Use **SparseCategoricalCrossentropy** as the **loss function.** | |
| **15** |  | Use **adam** as the optimizer. | |
| **16** |  | Track **accuracy** as the **evaluation metric.** | |
| **17** | **Prepare the Data:** | | |
| **18** |  | Convert **PyTorch Tensors** to **NumPy Arrays:** | |
| **19** |  |  | Move **X\_train** and **y\_train** tensors to the **CPU** and **convert them to NumPy arrays.** |
| **20** |  |  | Move **X\_test** and **y\_test** tensors to **CPU** **and convert them to NumPy arrays.** |
| **21** |  | **Transpose** the **channel dimension** to the **last position** for both **training and testing data.** | |
| **22** | **Train the Model:** | | |
| **23** |  | Train the model using **model.fit** with the following **parameters:** | |
| **24** |  |  | Training data (X\_train\_np and y\_train\_np) |
| **25** |  |  | Batch size of 100 |
| **26** |  |  | The number of epochs set to 150 |
| **27** |  |  | Validation data (X\_test\_np and y\_test\_np) |

**Algorithm 3.1.1.4 Algorithm for the CNN Model.**

**Algorithm 3.1.1.4** describes a step-by-step process for building and training a CNN model with the Keras library for the classification of weeds. In order to help with the simple installation of neural network layers, the process starts with defining the model by generating an instance of Sequential from Keras.

Convolutional layers are added to the model in the following stage. With 32 filters, a 3x3 kernel size, a ReLU activation function, and an input shape of (28, 28, 3), the first Conv2D layer is set up to represent a 28x28 image with three color channels (RGB). A MaxPooling2D layer with a pool size of 2x2 follows this layer. By halving the spatial dimensions of the feature maps, this layer helps in down sampling the input and lowers computational complexity. To extract more features from the input images, a second Conv2D layer is added, containing 64 filters, the same 3x3 kernel size, and the ReLU activation function. Another MaxPooling2D layer with a 2x2 pool size comes after this.

A Flatten layer is used to flatten the output after the convolutional layers. The 2D feature maps are transformed into a 1D vector in this step so that they can be fed into dense, fully connected layers for classification. With 128 units and the ReLU activation function, the first dense layer gives the model non-linearity and helps in its ability to learn complex patterns. The softmax activation function, which is appropriate for multi-class classification problems, is used in the final dense layer, which has four units. A probability distribution covering the four classes broadenleaf, grass, soil, and soybean is produced by the softmax function.

Next, the SparseCategoricalCrossentropy loss function which works well for classification tasks involving integer labels is used to compile the model. The Adam optimizer, known for its effectiveness and strong performance over a broad range of problems, is employed in training. The evaluation metric for the model is accuracy, which is used to track its performance.

The training and testing data must be converted from PyTorch tensors to NumPy arrays as part of the data preparation process. Because NumPy arrays are the expected format for input data in Keras models, this step is required. Tensors representing training data and labels, X\_train and y\_train, and testing data and labels, X\_test and y\_test, are transferred to the CPU and transformed into NumPy arrays. The shape of the images is also altered from (N, C, H, W) to (N, H, W, C), where N is the number of samples, C is the number of channels, H is the height, and W is the width. This is done by transposing the channel dimension of the images to the final position. The input requirements of the Keras model are compatible with this format.

Finally, the model is trained with the prepared training data (X\_train\_np and y\_train\_np) using the model.fit function. There are 150 training epochs and a batch size of 100. The validation data (X\_test\_np and y\_test\_np) are used to assess the model's performance during training, enabling monitoring and possible modifications to avoid overfitting and ensure strong generalization to new data.

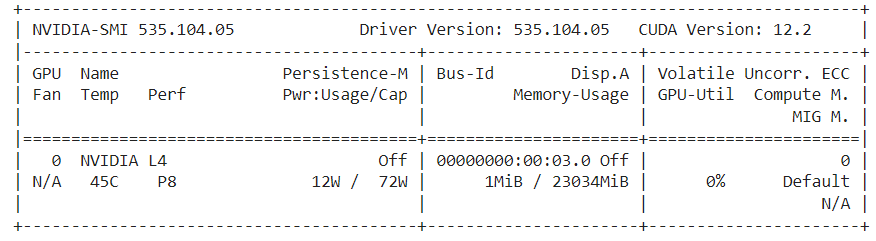
This algorithm efficiently combines the use of convolutional layers for feature extraction and dense layers for classification. It also ensures that the data is in the proper format for Keras, compiling and training the model with suitable loss and optimization functions. The systematic approach guarantees accuracy and consistency when creating a strong CNN for weed classification.

# Performance Evaluation and Analysis of Classification with DCGAN-based Data Augmentation

## 4.1. Environment Setup

### 4.1.2. Hardware

The GC Pro, a cloud-based platform with access to strong computational resources, was used to develop and evaluate the weed classification model. When compared to regular CPU-based environments, GC Pro's NVIDIA L4 GPU with 23034 MiB of memory as shown in **Fig 4.1.2** allowed for faster training and experimentation.



**Figure 4.1.1. GPU Version.**

Although there are GPU usage restrictions on the GC free tier, the Pro subscription gave users reliable access to the GPU for the duration of the project, which sped up the process of development and experimentation.

### 4.1.2. Software Requirements

I use a combination of DL frameworks and necessary Python libraries to develop the data augmentation and classification model. The main framework for building and training the CNN for classification tasks, as well as for implementing the DCGAN to augment the dataset, is PyTorch . PyTorch is the tool of choice for both CV and NLP applications because of its ability to take advantage of GPU acceleration, which significantly divides out the amount of time needed for neural network training. Furthermore, Scikit-learn is used to produce the classification report and compute evaluation metrics. Additional necessary libraries are matplotlib for plotting different graphs to visualize the performance of the model, seaborn for creating visualizations like heatmaps, and numpy for numerical operations. A strong environment for data augmentation, model training, evaluation, and visualization is offered by this combination of libraries and tools.

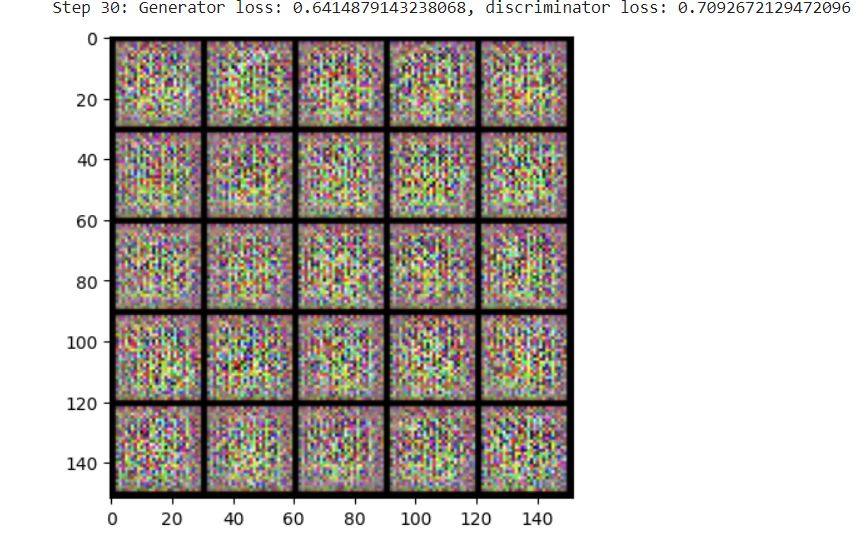
### 4.1.3. Dataset

A total of 400 photos were obtained by selecting all of the UAV's collected images that contained weeds. These images have been divided using the Pynovisao software and the SLIC algorithm. The segments were then manually analyzed with the appropriate class. The image dataset was constructed using these segments. This image dataset consists of 15336 segments: 1191 broadleaf weeds, 3249 soil, 7376 soybeans, and 3520 grass.

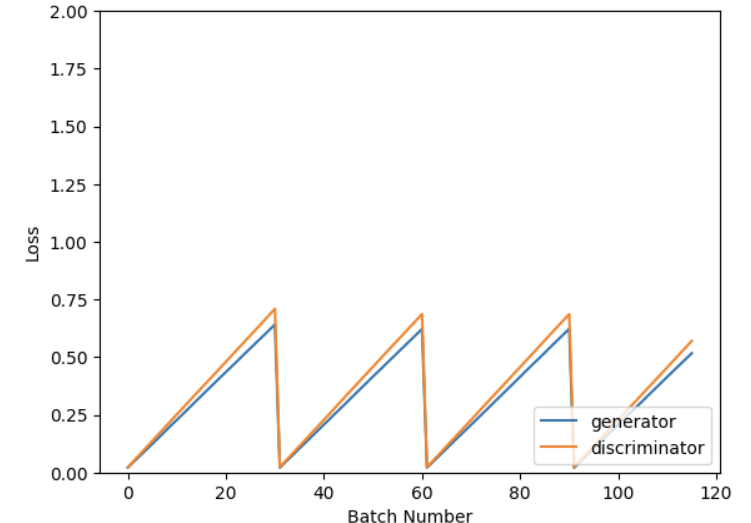
The dataset used in this study was obtained from **(dos Santos Ferreira et al., 2017).** Four classes of images are included in the dataset: broadleaf, grass, soil, and soybean. Each class's images are arranged into a different folder. 70% of the images in the dataset are in the testing set, and the remaining thirty percent are in the training set. There are different numbers of images in each class in the original dataset, which is altered. DCGAN is used to apply data augmentation techniques to rectify this imbalance and increase the size of the dataset. By producing artificial images that closely match the original ones, the augmented dataset improves the training set and strengthens the model's capacity to generalize and correctly classify various forms of weeds. To ensure an easier and reliable training process, the augmented images are added to the dataloader for the CNN model training.

### 4.1.4. Augmentation using DCGAN

The output of a DCGAN after data augmentation is shown in **Fig 4.1.4.1**. The picture displays the generator's progress and discriminator losses as a grid of generated samples following thirty training steps.

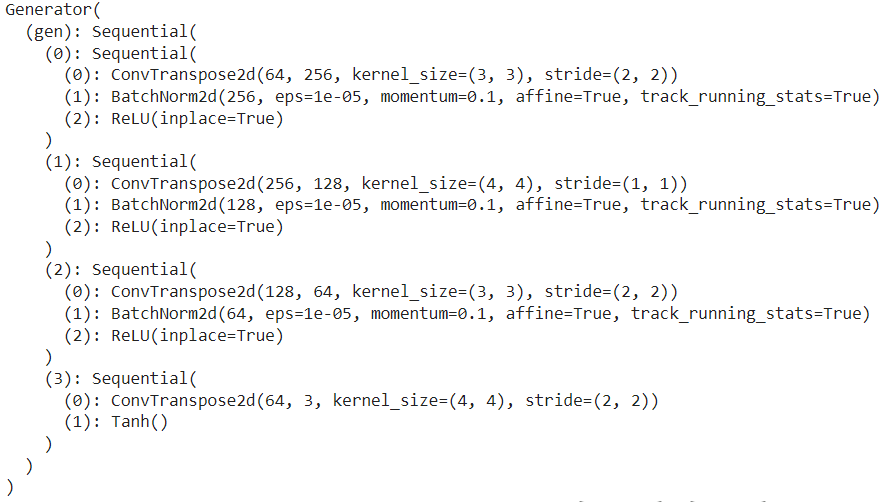


**Figure 4.1.4.1 Augmented Data.**



**Figure 4.1.4.2: Training of DCGAN.**

The dynamic evolution of a DCGAN during training is shown in **Figure 4.1.4.2**. The batch number is shown on the x-axis, and the loss values for the generator and discriminator components are shown on the y-axis. As the generator gains experience in creating more realistic images, its loss gradually goes down, but as the discriminator gets better at identifying fakes, it sometimes goes up. On the other hand, when the discriminator gets better at distinguishing between real and fake images, its loss goes down, but when the generator starts creating more realistic fakes, it goes up. The DCGAN continuously gets better as a result of this back-and-forth competition. The code for the generator is given in the **Figure 4.1.4.3**



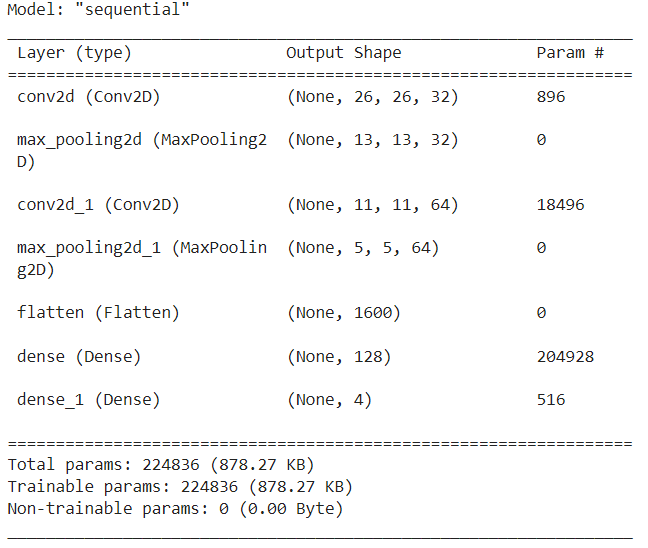
**Figure 4.1.4.3: Generator Code.**

A neural network called The Generator is used to create artificial images. It starts by progressively transforming an input vector of random noise through several layers. These layers are composed of ReLU activations for non-linearity, batch normalization (BatchNorm2d) to stabilize training and transposed convolutions (ConvTranspose2d) that up-sample the input. Transposed convolution and Tanh activation, which scales the output to the desired image range (usually -1 to 1), compose the final layer. In order to generate outputs that are identical to real images, the generator's architecture is specially designed to gradually increase the spatial dimensions and fine-tune the image's features.



**Figure 4.1.4.4: Discriminator Code.**

The CNN called the Discriminator is used to differentiate between genuine and false images. **Figure 4.1.4.4** shows the Discriminator Code. It is made up of four consecutive blocks, each of which has a convolutional layer with a 4x4 kernel size, a 2 stride, and a 1 padding. Along with LeakyReLU activation functions with a negative slope of 0.2 to introduce non-linearity and help prevent gradients from decreasing, the first three blocks also include Batch Normalization to stabilize training. A single value, close to 1, showing the discriminator's confidence that the input image is real or fake, is output by the last block. The architecture of the model gradually reduces the input image's resolution while adding more feature maps, enabling it to recognize complex patterns for exact discrimination.



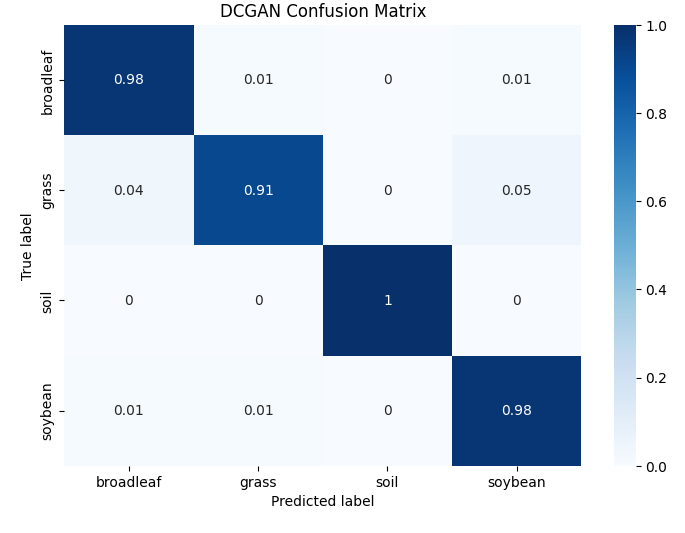
**Figure 4.1.4.5 CNN Model.**

**Fig 4.1.4.5** shows the summary of the CNN Model. Specifically, this CNN model is meant to be used with input images that have three color channels and a size of 28 by 28 pixels for image classification. First, a 3x3 convolutional layer with 32 filters is used, and then a ReLU activation function is added to introduce non-linearity. The following layer, a MaxPooling layer with a 2x2 pool size, aids in cropping the image and extracting the key characteristics. A second MaxPooling layer comes after a second convolutional layer with 64 filters and ReLU activation. After being flattened into a one-dimensional vector, the generated feature maps are then fed into a fully connected (Dense) layer that has 128 neurons and ReLU activation. The output layer comprises four neurons that display softmax activation, which collectively produce a probability distribution across the four potential output classes. This architecture is a standard method for image classification tasks and is well-suited for learning spatial structures of features.

## 4.2. Performance Metrics for Classification

It is essential to evaluate how well the classification models that differentiate between weeds and crops work. This section explores a number of metrics that offer a thorough understanding of the model's performance, including accuracy,  F1 score, and confusion matrix. While accuracy provides the overall success rate of predictions. The F1 score balances precision and recall, making it particularly useful in cases of imbalanced datasets. Additionally, the confusion matrix presents a detailed breakdown of true positives, true negatives, false positives, and false negatives, facilitating a nuanced evaluation of model performance. These metrics collectively enable a thorough assessment of the model’s ability to correctly identify and differentiate weeds from crops, ensuring robust and reliable weed detection.

### 4.2.1 Confusion Matrix



**Figure 4.2.1 DCGAN Confusion Matrix.**

A DCGAN model's classification performance for four classes broadleaf, grass, soil, and soybean is shown in **Fig. 4.2.1.** The model correctly classifies 98% of the soybean samples and 100% of the soil samples, demonstrating remarkable accuracy in identifying soil and soybean samples. 1% of the soybean samples were mistakenly identified as grass and another 1% as broadleaf. The model does not perform as well on grass, correctly classifying 91% of the grass but mislabeling 4% as broadleaf and 5% as soybean. It does, however, perform well on broadleaf, misclassifying only 1% as grass and 1% as soybean. All things considered, the DCGAN shows good classification accuracy, with soil and soybean identification showing specific ability.

### 4.2.2 Accuracy



Figure 4.2.2.1 DCGAN Test Accuracy.

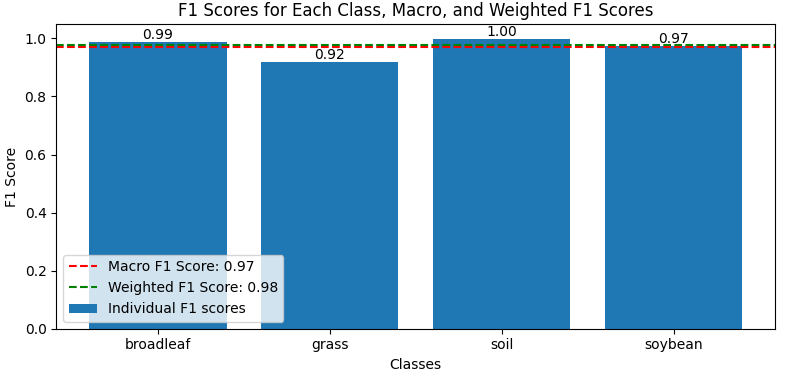


Figure 4.2.2.2 DCGAN Train Accuracy.

Outstanding results were obtained from the DCGAN model on the training and testing datasets. When the model achieves a perfect accuracy of 1.00 on the training set, it has successfully learned the features and patterns found in the training data. The model especially retained a high accuracy of 0.98 on the testing set, indicating that it is not overfitting to the training examples and that it generalizes well to new data. This performance demonstrates how well the DCGAN architecture and training approach work for tasks involving the classification of weeds.

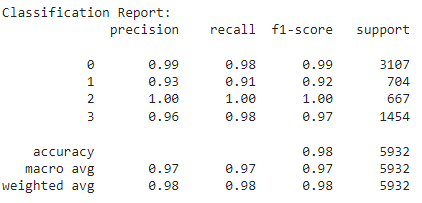
### 4.2.3 F1 Score

The F1 scores that a model received for the broadleaf, grass, soil, and soybean classes in a weed classification task are shown in **Fig. 4.2.3.1** Precision and recall are combined in a balanced metric called the F1 score, which offers a complete assessment of a model's accuracy.



**Figure 4.2.3.1 F1 Score.**

The model performs relatively poorly in correctly classifying instances of this class; however, it displays high F1 scores for broadleaf (0.99), soil (1.00), and soybean (0.97). The model's score for grass (0.92) is somewhat lower. The model's overall performance across all classes is equal to the macro F1 score of 0.97, which is an unweighted average of the F1 scores of all classes. The weighted F1 score is 0.98, indicating that the model performs slightly better when taking into account the class imbalance. The weighted F1 score accounts for the class distribution by weighting the F1 scores according to the number of samples in each class.

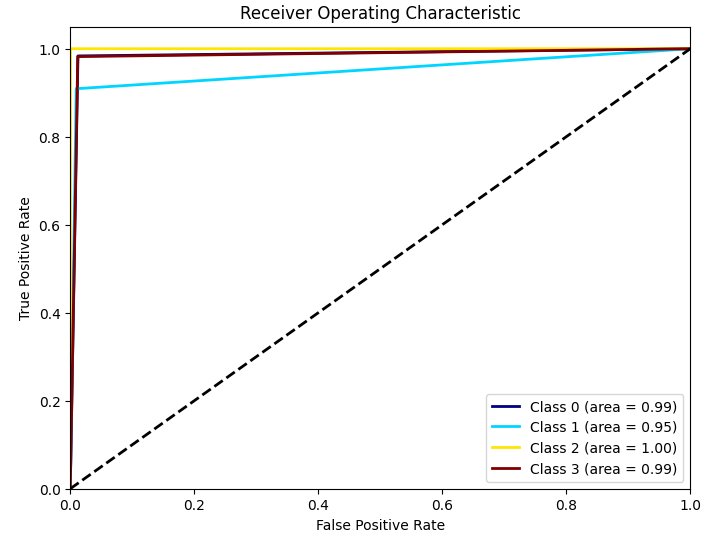


**Figure 4.2.3.2 Classification Report.**

The classification report is displayed in **Fig. 4.2.3.2** and shows how well a model performed in classifying four categories: broadleaf (0), grass (1), soil (2), and soybean (3). The model performs exceptionally well in terms of recall, F1-score of 1.00, and precision when predicting soil samples. With broadleaf and soybean, it also does well, earning F1-scores of 0.99 and 0.97, respectively. With a precision of 0.93, recall of 0.91, and F1-score of 0.92, grass exhibits the lowest performance of the model, suggesting some challenge in differentiating it from other classes. The model's macro/weighted average F1-scores of 0.97 and 0.98, respectively, indicate its high accuracy overall and its robustness in classifying the dataset.

### 4.2.4 AUC ROC

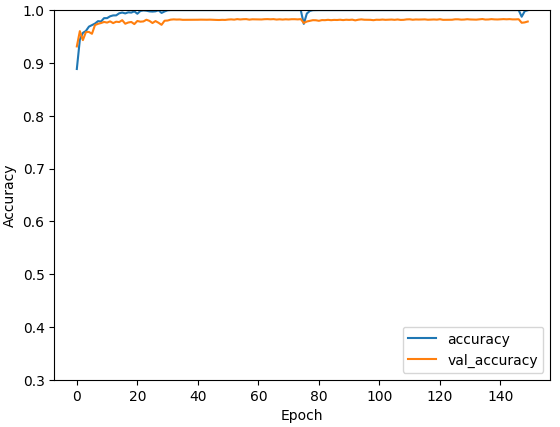
The ROC curves in **Figure 4.2.4** show how well the model can differentiate between the various types of weeds (soybean, broadleaf, grass, and soil) and the other classes. The unequal ability of the model is higher the closer a curve is to the top-left corner. The model shows perfect discrimination with an AUC of 1.00, indicating excellent performance in class 2 soil classification. With AUCs of 0.99, it also performs well in classifying soybeans (class 3) and broadleaf (class 0), indicating a very high true positive rate in comparison to the false positive rate. But for grass (class 1), the model performs slightly worse, with an AUC of 0.95, suggesting a little bit more overlap between the grass and the other classes and a slightly less accurate prediction capability than the other classes.



**Figure 4.2.4 AUC ROC Curves.**

.

### 4.2.5 CNN Results with DCGAN Augmentation



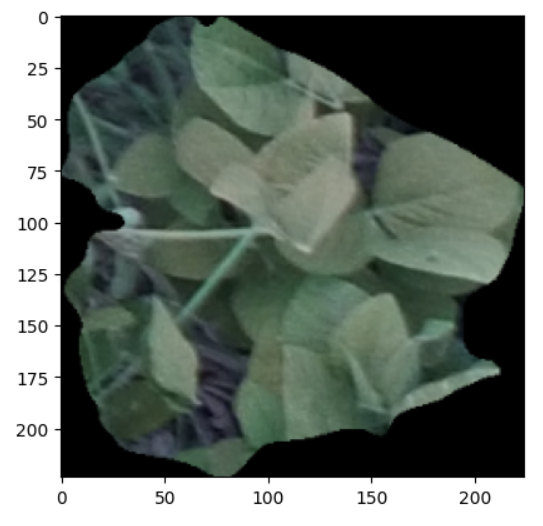
**Figure 4.2.5 CNN Result with DCGAN.**

The accuracy and validation accuracy of a model over 150 epochs are shown in **Figure 4.2.5**. The model picks up new information quickly at first, with training accuracy rising faster than validation accuracy. However, there is a noticeable drop in validation and training accuracy around 70 epochs. Following this drop, the training accuracy keeps increasing and reaches an endpoint close to 1.0, while the validation accuracy bounces back and even slightly beats the previous peak levels.

### 4.3 Performance Analysis of CNN with GAN Augmentation

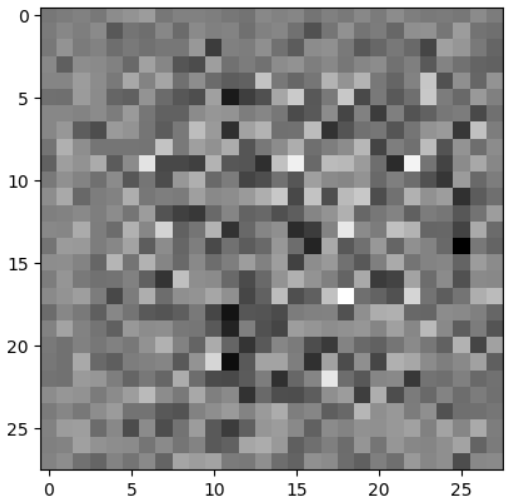
### 4.3.1 Image Augmentation using GAN

A dark background surrounds a group of leaves in **Figure 4.3.1.1**. The textures of the leaves are visible and they have a soft green shade with shades of brown and purple. The uneven edges of the picture imply that the leaves were either segmented or cropped from a larger scene. All things considered, the picture seems to be an actual photo of plant leaves.



**Figure 4.3.1.1 Before Augmentation.**

The grayscale image with 28 x 28 pixels is displayed in **Figure 4.3.1.2**. It looks like a poor picture of a plant. An apparent leaf-like shape with various levels of brightness in the center indicates the texture and shape. Still, the picture is a little unclear and lacks the fine details that one would find in a realistic image. This image is then used for CNN model classification.



**Figure 4.3.1.2 After Augmentation.**

### 4.3.2 GAN Model Design

**Figure 4.3.2.1** shows the architecture of a generator model, likely a component of a GAN. It's a sequential model that starts with a dense layer followed by batch normalization and a LeakyReLU activation. The output is then reshaped and passed through a series of Conv2DTranspose layers (transposed convolutions), each followed by batch normalization and LeakyReLU activation, except for the last layer, which has a Tanh activation. The output shape of the model is (None, 28, 28, 3), suggesting it generates 28x28 RGB images. The model has a total of 2,334,144 parameters, with most of them being trainable.

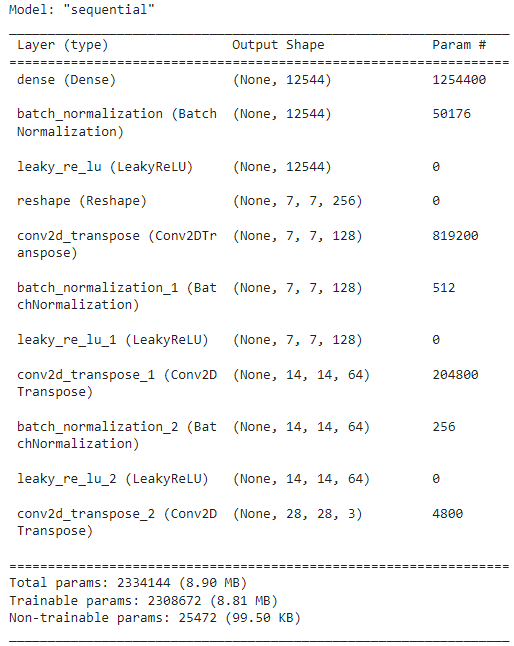
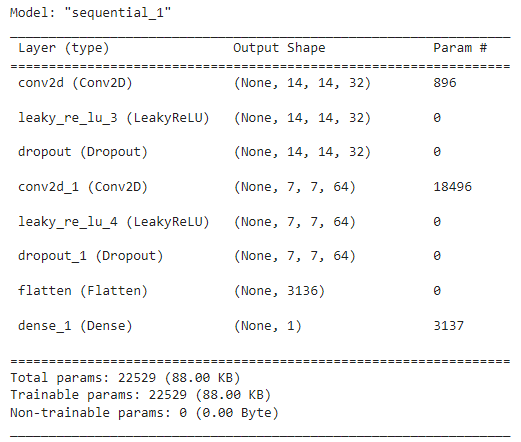


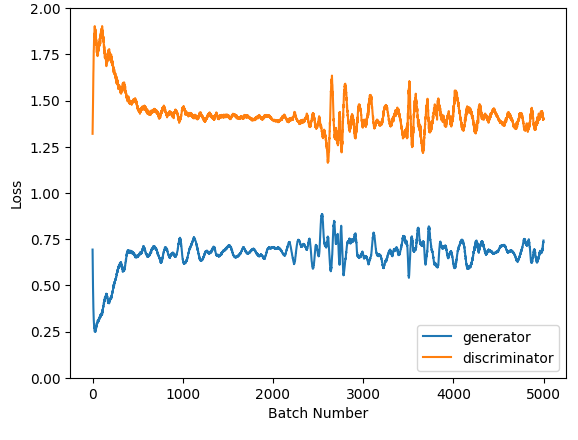
Figure 4.3.2.1 Generator Summary.

The architecture of a discriminator model, which is likely used in a GAN, is shown in **Figure 4.3.2.2.** This sequential model has two convolutional layers (Conv2D) with 32 and 64 filters, respectively, each of which is followed by a Dropout layer and a LeakyReLU activation. After that, the output is compressed and sent via a dense layer that contains a single output neuron. There are 22,529 trainable parameters in the model overall. The goal of this architecture is to learn complex patterns for differentiating between real and fake images by gradually downsampling the input image and increasing the number of feature maps.



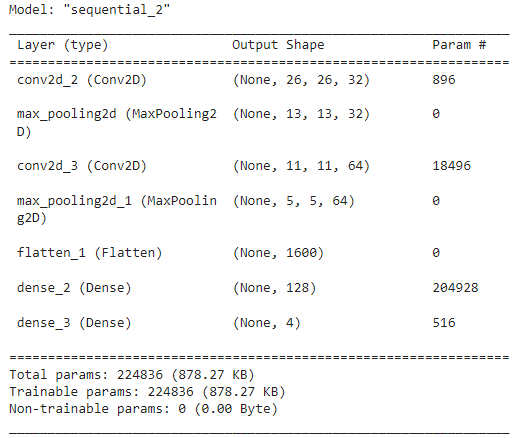
**Figure 4.3.2.2 Discriminator Summary.**

The loss of the discriminator and generator over 5000 batches while a GAN is being trained is shown in **Figure 4.3.2.3.** While the discriminator's loss increases as it becomes more difficult to differentiate between real and fake images, the generator's loss initially decreases as it learns to produce more believable images. But when the discriminator gets better at identifying fakes, the generator loses more, which makes it have to work harder at creating better images. Through an adversarial process, a dynamic balance is reached where both losses vary over time, with the discriminator growing more perceptive and the generator progressively producing fake images of higher quality.



**Figure 4.3.2.3 Generator and Discriminator Loss.**

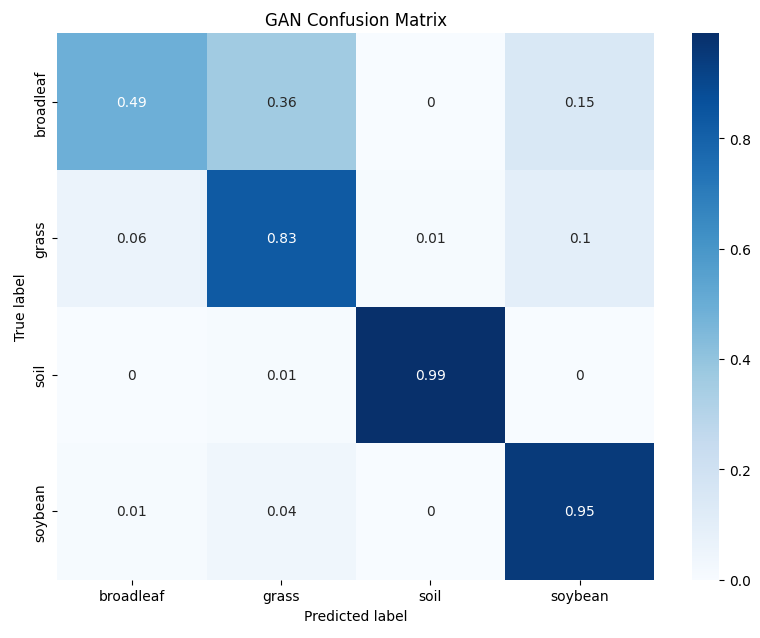
### 4.3.3 CNN based on DC-GAN Augmentations



**Figure 4.3.3 CNN Model Summary.**

The CNN model known as "sequential\_2" is intended for image classification, as seen in **Figure 4.3.3**. It begins with a convolutional layer (conv2d\_2) that applies 32 filters, each of size 3x3, to input images with a resolution of 28x28 pixels and three channels in order to extract initial features. A max-pooling layer (max\_pooling2d) is added after this to lower the dimensionality while keeping the most noticeable features. Conv2d\_3, the subsequent convolutional layer, has 64 3x3 filters before being succeeded by another max-pooling layer. Next, two dense layers (dense\_2 and dense\_3) are passed through the resultant feature maps after they have been flattened (flatten\_1). 128 neurons make up the first dense layer, and 4 neurons make up the second layer, or output layer, which corresponds to the 4 classes that need to be predicted. A total of 224,836 parameters form the model.

### 4.3.4 Result of GAN with CNN



**Figure 4.3.4.1 GAN with CNN Confusion Matrix.**

The performance of a GAN-based model in classifying four different types of classes broadleaf, grass, soil, and soybean is shown in  **Figure 4.3.4.1** With 99% of soil samples and 95% of soybean samples properly classified, the model demonstrates a high degree of accuracy in soil and soybean sample classification. When it comes to grass, the model shows a moderate level of accuracy; 83% of samples are correctly identified, but 10% are incorrectly classified as broadleaf. The class that presents the greatest challenge to the model is broadleaf, with 49% of cases correctly classified and a sizable portion incorrectly classified as soybean (15%) and grass (36%). All things considered, the GAN model does a good job of differentiating between soil and soybeans; however, it still needs to work on differentiating between grass and broadleaf.

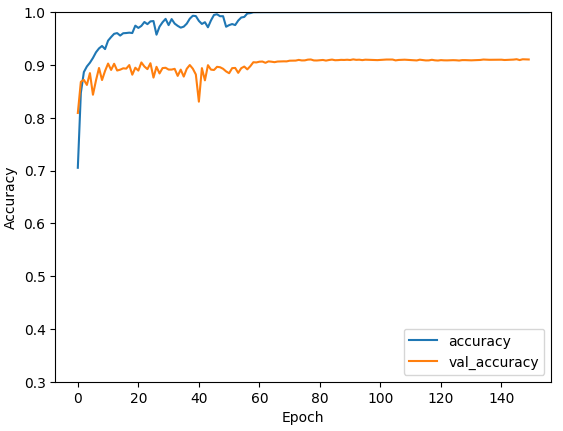
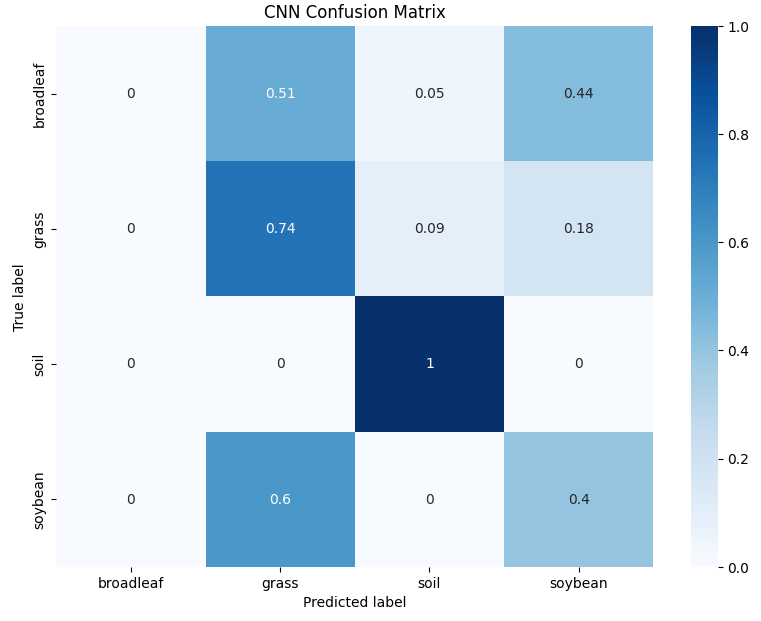


Figure 4.3.4.2 GAN with CNN Accuracy Score.

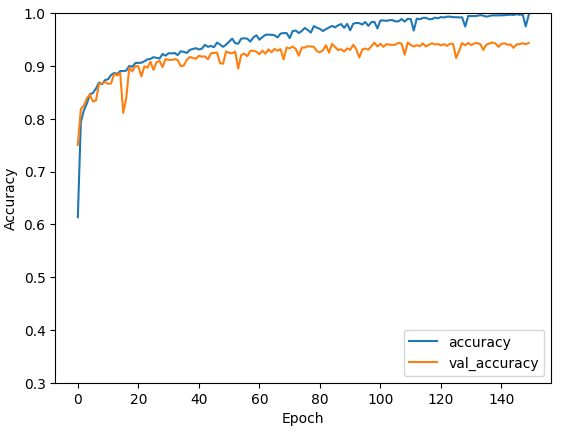
A graph showing the accuracy and validation accuracy of a GAN-based model combined with a CNN classifier over 150 epochs can be found in **Figure 4.3.4.2**. In the early epochs, the training accuracy (blue line) rises quickly and reaches a high degree of accuracy fast. The validation accuracy (orange line), on the other hand, shows a more gradual improvement that peaks at epoch 40. This shows that even though the model is doing a good job of learning the training set, it may be overfitting a little bit and finding it difficult to generalize to new data.

### 4.4 Performance Analysis without Augmentation



**Figure 4.4.1 CNN Confusion Matrix.**

**Figure 4.4.1** shows how well a CNN model classified four different types of classes: soil, soybean, grass, and broadleaf. All soil samples are correctly classified, demonstrating that the model can identify soil samples with perfect accuracy. With a 74% accuracy rate, it performs well on grass as well. However, 51% of samples classified as broadleaf and 44% of samples classified as soybean were incorrectly identified by the model, indicating that it has difficulty differentiating between broadleaf and soybean. Overall, the model needs work to distinguish between broadleaf and soybean, even though it is capable at identifying soil and has a fair degree of accuracy for grass.



**Figure 4.4.2 CNN Accuracy Score**

The accuracy and validation accuracy of a CNN model trained on real data over 150 epochs without any augmentation are shown in **Figure 4.4.2.** During the first few epochs, both training and validation accuracy increase quickly, suggesting quick learning. With a validation accuracy of roughly 0.95 and a high training accuracy of roughly 0.99, the model performs well. As shown by the relatively close alignment of training and validation accuracy, the model exhibits strong generalization to new data, even in the presence of small variations in both accuracy measures. It indicates that the underlying patterns in the data have been successfully and unmistakably discovered by the model, free from overfitting.

This chapter examines the use of GAN-based data augmentation to improve weed classification. compared with traditional CPU-based environments, the GC Pro platform's hardware environment which includes an NVIDIA L4 GPU markedly sped up the training and exploration phases for this project. With the complex structure and computational demands of DL tasks, this configuration guaranteed reliable and strong computational resources, enabling effective development cycles.

A wide range of tools were included in the software environment, such as matplotlib and seaborn for visualization, numpy for numerical operations, Scikit-learn for generating metrics and reports on classification, and PyTorch for DL. Both the DCGAN implementation for data augmentation and the CNN training for weed classification benefited greatly from PyTorch's GPU acceleration capabilities. By combining these libraries, a complete toolkit for creating, developing, and evaluating the models was made available, ensuring a smooth process from data preparation to performance analysis.

Images of four classes were included in the dataset, which was obtained from Kaggle: broadleaf, grass, soil, and soybean. A 70-30 split of this dataset was made into training and testing sets. A major obstacle was the unequal distribution of images within each class, which was resolved by employing DCGAN for data augmentation. The augmented dataset enhanced the training set and balanced the class distribution by producing artificial images that closely matched the original ones. This improved the model's capacity to classify various weed types accurately and essentially.

The most important aspect in producing more training data was the DCGAN framework. By gradually converting random noise vectors into realistic samples via a sequence of layers that included batch normalization, transposed convolutions, and ReLU activations, the generator produced artificial images. The discriminator, a CNN, had to identify between real and fake images by deriving complex patterns via a series of convolutional layers. The quality of generated images was continuously improved through the adversarial training process, which is illustrated by the dynamic interaction between discriminator and generator losses. High-quality synthetic images were eventually produced by this iterative process, significantly improving the training dataset.

The convolutional and max-pooling layers were followed by dense layers in the typical architecture of the CNN model that was used for classification. To extract features, 28x28 pixel input images were passed through pooling layers and convolutional layers with ReLU activations. The extracted features were then flattened and fed into fully connected layers. Softmax activation was used in the final output layer to generate a probability distribution among the four classes. The spatial structures of features required for accurate image classification were successfully learned by this architecture.

Several metrics were used to evaluate the performance of the model, including accuracy, F1 score, confusion matrix, and AUC-ROC curves. With perfect training accuracy and nearly perfect testing accuracy, the DCGAN-augmented CNN demonstrated remarkable accuracy on both training and testing sets, indicating strong generalization. For the majority of classes, the confusion matrix showed high precision and recall; however, the model had some difficulty classifying grass. This was reflected in the F1 scores, where grass received lower marks than other classes. The biased power of the model was further validated by the AUC-ROC curves, particularly for the classes of soil and soybeans.

The value of data augmentation was demonstrated by comparing CNN's performance with and without GAN-based augmentation. The confusion matrix and accuracy scores, in particular, demonstrated how much better the augmented model was at accuracy and generalization. The model significantly misclassified soybean and broadleaf classes when it was left unaugmented, highlighting the problem of class imbalance and the importance of synthetic data in resolving it.

Strong classification results were obtained by integrating CNN training with data generated by GANs. High accuracy and consistency were shown by the improved CNN model during both the training and validation stages, indicating that the enriched dataset was effectively discovered and generalized. Even with irregular overfitting patterns, overall performance was significantly better than with the non-augmented method.

The important advantages of utilizing GAN-based data augmentation for weed classification are clearly illustrated in Chapter 4. The CNN model was able to achieve greater accuracy and better generalization. By producing realistic synthetic data, the DCGAN framework successfully addressed class imbalance, a significant problem in many real-world datasets. The thorough setup, which includes performance analysis and environment configuration, highlights GANs' potential to enhance classification tasks in a variety of domains. This work shows a standard for the use of GANs in other difficult classification problems in addition to validating their effectiveness in data augmentation.

Using an NVIDIA L4 GPU and the GC Pro platform, the study used PyTorch for CNN training and DCGAN-based data augmentation. The images in the dataset were divided 70-30 for training and testing, and they were classified into four classes: broadleaf, grass, soil, and soybean. By addressing class imbalances, augmenting images, and improving the CNN model's generalization capacity, by using DCGAN  the dataset is improved. Performance metrics showed high accuracy, with 98% and 100% accuracy for soil and soybean classification, respectively. The weighted F1 score is slightly greater at 0.98 than the macro F1 score of 0.97, and the overall test accuracy was 98%. The model did, however, have some difficulty in correctly classifying broadleaf (99%) and grass (91%); the F1 scores for grass were comparatively lower at 0.92. In comparison, when a GAN and CNN were used together, the model performed well in classifying soil and soybeans but less well in classifying grass and broadleaf, achieving 95% accuracy for soybeans but only 83% and 49% accuracy for grass and broadleaf, respectively. The study showed that although overall classification was improved by GAN-based augmentation, more effort is required to improve the distinction between grass and broadleaf.

### 4.5 Discussion

The evaluation of DCGAN-generated synthetic UAV images and its effect on CNN model's classification accuracy for weed identification in precision agriculture were covered in this chapter. Through an analysis of the accuracy scores obtained and the resulting consequences, the research questions presented at the beginning of the study are addressed in the discussion. The appearance and statistical features of the generated images were compared with actual UAV-captured images in order to evaluate the effectiveness of DCGANs in creating synthetic UAV images. Upon visual inspection, the produced images showed a great deal of similarity to actual UAV images, as they were able to accurately capture the minute details and variances found in real agricultural fields. Quantitative data also supported this, showing that when artificial images were used in place of real images to train CNN models, the artificial images produced high accuracy scores. A number of metrics, including the structural similarity index and other important statistical measures, were used to evaluate the statistical similarity between real and synthetic images. The outcomes showed that the artificial images generated by DCGANs were statistically comparable to real images, suggesting that the generated data kept the key elements required for efficient CNN training. This resemblance is essential to guaranteeing the true nature of the augmented data and its ability to boost CNN model's durability.

It was predicted that adding DCGAN-augmented synthetic UAV imagery would increase CNN model's classification accuracy. This hypothesis was strongly supported by the outcomes of the experiment. Compared to models trained on just real images, there was an apparent rise in classification accuracy when the CNN models were trained on a dataset that contained both real and synthetic images. The model was able to learn from the synthetic images and improve its ability to generalize and make predictions in real-world scenarios, as shown by the accuracy scores. The improvement can be assigned to the synthetic image's larger and more varied training data set, which enhanced the CNN model's capacity to identify weeds by better capturing the variation in their appearances. The outcomes are consistent with a greater amount of research on data augmentation, which has demonstrated the benefits of synthetic data in improving model performance, especially when acquiring large quantities of labeled real-world data is difficult. Enhancing model performance through artificial data augmentation has significant applications in precision agriculture, where prompt and precise weed identification is essential.

# Conclusion and Future Work

### 5.1. Conclusion

This thesis used DCGAN for data augmentation to investigate the potentials and challenges of enhancing CNN models for weed identification in precision agriculture. The main goal of this research has been to address significant challenges that may seriously limit the effectiveness of ML models in practical agricultural applications, such as class imbalance and a lack of training data.

Establishing an assumption for CNN performance in the absence of any data augmentation is an important part of this study because it gives an essential point of comparison to evaluate the effectiveness of techniques such as GAN-based augmentation. Across several classes, the CNN model produced inconsistent results when trained exclusively on the original dataset. For soil samples, it showed a perfect identification accuracy of 100%, proving the model's effectiveness in handling extremely distinct classes. However, the model's performance drastically decreased when it tried to categorize samples of grass, broadleaf, and soybeans. Although the accuracy of the grass sample classification was not as high as that of the soil classification, it was still an acceptable 74%. The broadleaf and soybean classes presented the biggest challenge, with the model misclassifying 51% and 44% of the samples, respectively, indicating significant difficulty. These findings create attention to two crucial problems: the impact of class imbalance on model performance and the fundamental difficulty of differentiating between broadleaf and soybean-based on the data in hand.

The difficulty the CNN had differentiating between soybean and broadleaf without augmentation highlights the complexity these specific classes are. Since colour, texture, and shape are shared by both broadleaf and soybean, it is difficult for the model to make generalizations from the training set alone. This problem is made worse by the dataset's imbalance, which makes some classes unrepresented and makes the model less capable of identifying those classes. In ML, class imbalance is a major problem, especially in domains like precision agriculture where certain crop or weed types are more common than others. Due to this imbalance, models may become biased and perform poorly for minority classes while being excessively confident in their ability to predict majority classes. The DCGAN technique, which is designed to artificially increase the number of instances of the underrepresented classes in the dataset, is used in the thesis to address this issue through data augmentation. Synthetic images that closely resemble real-world UAV-captured images were produced using the DCGAN model. Subsequently, these artificial images were incorporated into the initial dataset, augmenting the variety and quantity of training data accessible to the CNN model. This augmentation assisted in reducing the overfitting issue in addition to offering additional examples for the minority classes. When a model performs remarkably well on training data but is unable to generalize to new, unseen data, it is said to be overfitted. This is a common problem when the training dataset is small and lacking in variety.

The DCGAN successfully increased the training dataset by producing a large number of synthetic images, which helped CNN understand the basic characteristics of each class. For the minority classes such as soybean and broadleaf which in the past had high rates of misclassification, this is especially crucial. The training process produced a more uniform distribution of class representation because the DCGAN-generated images contributed to the input data's balancing. Because of this, the model was able to learn across all classes more successfully, which decreased the possibility of misclassification and increased accuracy overall.

The results of this research offer strong proof of the usefulness of DCGAN-based data augmentation in raising the CNN model's weed identification performance. A near-perfect testing accuracy of 0.98 and a perfect training accuracy of 1.00 were attained by the model trained with DCGAN augmentation. With a testing accuracy of 0.95 and a training accuracy of 0.99, the model trained without augmentation did not perform as well as the augmented model, but these results still indicate a significant improvement over that model. Because it shows how well the model can now generalize to new data, the increase in testing accuracy is especially significant. In real-world applications, where the model must correctly identify weeds in a variety of settings and conditions that may differ from the training data, this is an essential component. The augmented model's greater accuracy on the testing set indicates that the DCGAN-based augmentation captured the variability required for successful generalization, in addition to providing more relevant data. When working with small and uneven datasets, overfitting is a common issue that can be directly linked to this improvement in generalization. The DCGAN improved the model's performance on unseen data by introducing artificial images that closely resemble real-world images and helping the model learn more robust features. This discovery highlights the potential of GAN-based augmentation as a successful way of getting around the drawbacks of conventional data augmentation methods, especially in fields like precision agriculture where acquiring significant, labelled datasets can be difficult.

The results of this thesis have a number of significant consequences for the field of precision agriculture as well as the more general use of ML in real-world situations. The potential of GAN-based data augmentation as a solution for the common issues of class imbalance and limited data availability is demonstrated by the effective use of DCGANs to enhance CNN performance. This strategy could be applied to other agricultural sectors in addition to where comparable difficulties exist. Even though the outcomes are encouraging, the thesis also notes that the present strategy has its drawbacks. While DCGAN-generated artificial images are effective in enhancing model performance, the issue of class imbalance might not be fully resolved. These artificial images might fail to capture all the subtleties of real-world data or they might introduce new biases. Therefore, more investigation is required to improve DCGAN-based augmentation methods and investigate their scalability and generalizability in various datasets and applications.

### 5.2. Future Work

The effective use of GAN-based data augmentation in this thesis provides a platform for future studies and advancements in agricultural AI. In order to optimize the creation of synthetic data for particular agricultural tasks, future work could investigate various GAN architectures and methodologies. Furthermore, to enhance model performance even more, it is possible to look into how GANs can be used in combination with other ML strategies like TF or AL.

Future research should also take into account the useful application of GAN-augmented models in actual agricultural systems. Although this thesis's results show that using GANs to enhance weed detection is feasible, more work needs to be done to resolve issues with scalability, computational cost, and data collection before farmers can begin to use these models extensively. Creating lighter, more effective GAN architectures that can be used on edge devices like tractors or drones could contribute to the field's deployment of these innovations.

In conclusion, GAN-based data augmentation has proven to be an effective method for enhancing the DL models' generalization, robustness, and accuracy for weed detection. Through the fixing of significant issues like overfitting and class imbalance, GANs improve the development of more dependable and expandable AI-based precision agricultural solutions. GAN-augmented models have the ability to completely change how farmers handle weeds and other crop-related issues as research in this field progresses, leading to more effective and environmentally friendly agricultural methods.

**References**

dos Santos Ferreira, A., Pistori, H., Matte Freitas, D. and Gonçalves da Silva, G., 2017. Data for: Weed Detection in Soybean Crops Using ConvNets. Mendeley Data, V2. Available at: https://doi.org/10.17632/3fmjm7ncc6.2 [Accessed 07 June. 2024].

Hasan, A. S. M. M., Sohel, F., Diepeveen, D., Laga, H., & Jones, M. G. K. (2021). A survey of deep learning techniques for weed detection from images. ArXiv Preprint, arXiv:2103.01415. Available at: <https://arxiv.org/abs/2103.01415> [Accessed 05 July. 2024].

Fawakherji, M., Potena, C., Prevedello, I., Pretto, A., Bloisi, D. D., & Nardi, D. (2020). Data Augmentation Using GANs for Crop/Weed Segmentation in Precision Farming. IEEE Conference on Control Technology and Applications (CCTA), Montréal, Canada. Available at: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9206297>[Accessed 05 July. 2024].

Pai, D. G., Kamath, R., & Balachandra, M. (2024). Deep Learning Techniques for Weed Detection in Agricultural Environments: A Comprehensive Review. Available at: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=10570187>[Accessed 07 July. 2024].

Kerdegari, H., Razaak, M., Argyriou, V., & Remagnino, P. (2023). Semi-supervised GAN for Classification of Multispectral Imagery Acquired by UAVs. Precision Agriculture. Available at: <https://arxiv.org/pdf/1905.10920>[Accessed 12 July. 2024].

Xiaojun J,Junchee, & Yong C (2021). Weed Identification Using Deep Learning and Image Processing in Vegetable Plantation. Available at: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9317797>[Accessed 12 July. 2024].

Lopez-Granados, F. (2011) .Weed detection for site-specific weed management: mapping and real-time approaches. Available at: <https://onlinelibrary.wiley.com/doi/epdf/10.1111/j.1365-3180.2010.00829.x?saml_referrer>[Accessed 12 July. 2024].

Dankhara, F., Patel, K., & Doshi, N. (2019). Analysis of robust weed detection techniques based on the Internet of Things (IoT). Available at: <https://www.sciencedirect.com/science/article/pii/S1877050919317259>[Accessed 13 July. 2024].

Kulkarni, S., & Angadi, S. A. (2019). IOT-Based Weed Detection Using Image Processing and CNN. Available at: <http://ijeast.com/papers/606-609,Tesma403,IJEAST.pdf>[Accessed 13 July. 2024].

Murad, N.Y., Mahmood, T., Forkan, A.R.M., Morshed, A., Jayaraman, P.P., & Siddiqui, M.S. (2023). Weed Detection Using Deep Learning: A Systematic Literature Review. Available at: <https://www.mdpi.com/1424-8220/23/7/3670>[Accessed 13 July. 2024].

Islam, N., Rashid, M.M., Wibowo, S., Xu, C.-Y., Morshed, A., Wasimi, S., Moore, S., & Rahman, S.M. (2021). Early Weed Detection Using Image Processing and Machine Learning Techniques in an Australian Chilli Farm. Available at: <https://www.mdpi.com/2077-0472/11/5/387>[Accessed 15 July. 2024].

Umamaheswari, S., Arjun, R., & Meganathan, D. (2018). Weed Detection in Farm Crops using Parallel Image Processing. 2018. Available at: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8722369>[Accessed 15 July. 2024].

Bento, N.L., Ferraz, G.A.e.S., Amorim, J.d.S., Santana, L.S., Barata, R.A.P., Soares, D.V., & Ferraz, P.F.P. (2023). Weed Detection and Mapping of a Coffee Farm by a Remotely Piloted Aircraft System. Available at: <https://www.mdpi.com/2073-4395/13/3/830>[Accessed 15 July. 2024].

Lottes, P., Khanna, R., Pfeifer, J., Siegwart, R., & Stachniss, C. (2017). UAV-Based Crop and Weed Classification for Smart Farming. Available at: <https://ieeexplore.ieee.org/abstract/document/7989347>[Accessed 15 July. 2024].

Peteinatos, G.G., Weis, M., Andujar, D., Rueda Ayala, V., & Gerhards, R. (2013). Potential use of ground-based sensor technologies for weed detection. Available at: <https://scijournals.onlinelibrary.wiley.com/doi/full/10.1002/ps.3677>[Accessed 15 July. 2024].

Razfar, N., True, J., Bassiouny, R., Venkatesh, V., & Kashef, R. (2022). Weed detection in soybean crops using custom lightweight deep learning models. Available at: <https://www.sciencedirect.com/science/article/pii/S2666154322000412>[Accessed 15 July. 2024].

Rosle, R., Che’Ya, N.N., Ang, Y., Rahmat, F., Wayayok, A., Berahim, Z., Fazlil Ilahi, W.F., Ismail, M.R., & Omar, M.H. (2021). Weed Detection in Rice Fields Using Remote Sensing Technique: A Review. Available at: <https://www.mdpi.com/2076-3417/11/22/10701>[Accessed 15 July. 2024].

Sarker, M. I., & Kim, H. (2016). Farmland weed detection with region-based deep convolutional neural networks. Available at: <https://arxiv.org/abs/1906.01885>[Accessed 15 July. 2024].

Narayana, C. L., & Ramana, K. V. (2023). An Efficient Real-Time Weed Detection Technique using YOLOv7. Available at: <https://www.semanticscholar.org/paper/An-Efficient-Real-Time-Weed-Detection-Technique-Narayana-Ramana/a4a6e4fe1855ebce9d86cf425a76d50af9c83aa7?p2df>[Accessed 16 July. 2024].

Mishra, A. M., & Gautam, V. (2021). Weed Species Identification in Different Crops using Precision Weed Management: A Review. Available at: <https://www.researchgate.net/profile/Anand-Mishra-21/publication/349463320_Weed_Species_Identification_in_Different_Crops_using_Precision_Weed_Management_A_Review/links/6064a5e645851534986e3148/Weed-Species-Identification-in-Different-Crops-using-Precision-Weed-Management-A-Review.pdf\>[Accessed 16 July. 2024].

Perez, A.J., Lopez, F., Benlloch, J.V., & Christensen, S. (2000). Colour and shape analysis techniques for weed detection in cereal fields. Available at: https://www.sciencedirect.com/science/article/pii/S016816999900068X[Accessed 16 July. 2024].

Tang, J.-L., Chen, X.-Q., Miao, R.-H., & Wang, D. (2016). Weed detection using image processing under different illumination for site-specific areas spraying. Available at: https://www.sciencedirect.com/science/article/pii/S0168169915003981[Accessed 16 July. 2024].

Subeesh, A., Bhole, S., Singh, K., Chandel, N.S., Rajwade, Y.A., Rao, K.V.R., Kumar, S.P., & Jat, D. (2022). Deep convolutional neural network models for weed detection in polyhouse grown bell peppers. Available at: https://www.sciencedirect.com/science/article/pii/S2589721722000034[Accessed 16 July. 2024].

Abdulsalam, M., & Aouf, N. (2020). Deep Weed Detector/Classifier Network for Precision Agriculture. Available at: <https://ieeexplore.ieee.org/abstract/document/9183325>[Accessed 16 July. 2024].

**Appendix**



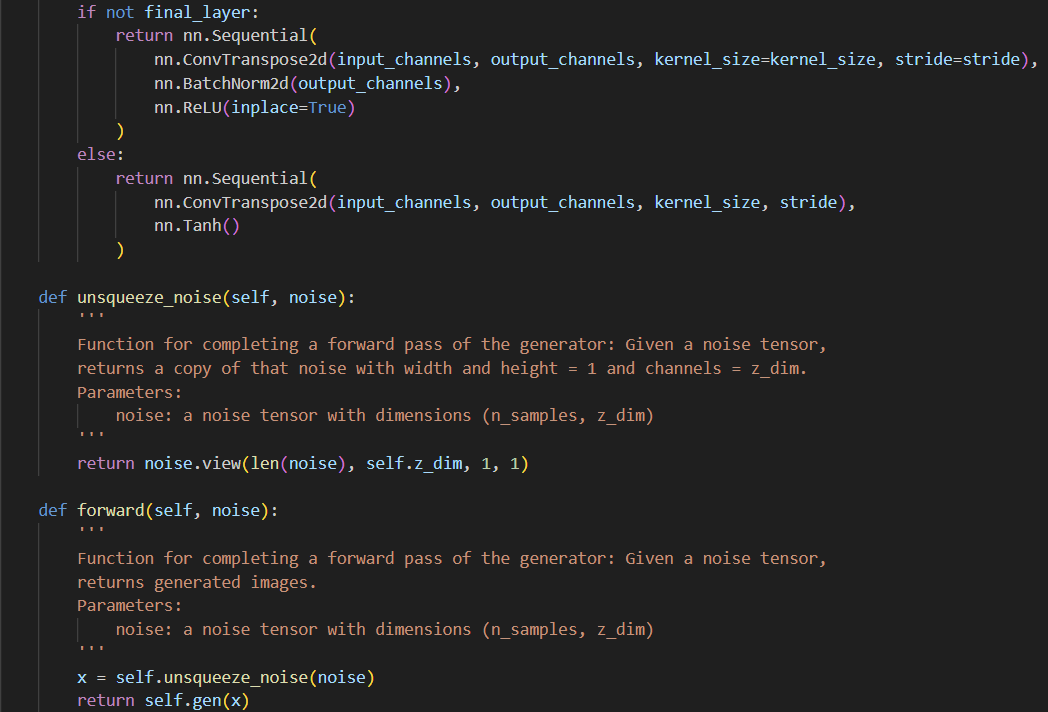


Figure 5.1 Generator Model Code.



Figure 5.2 Initialize Generator Model Code.

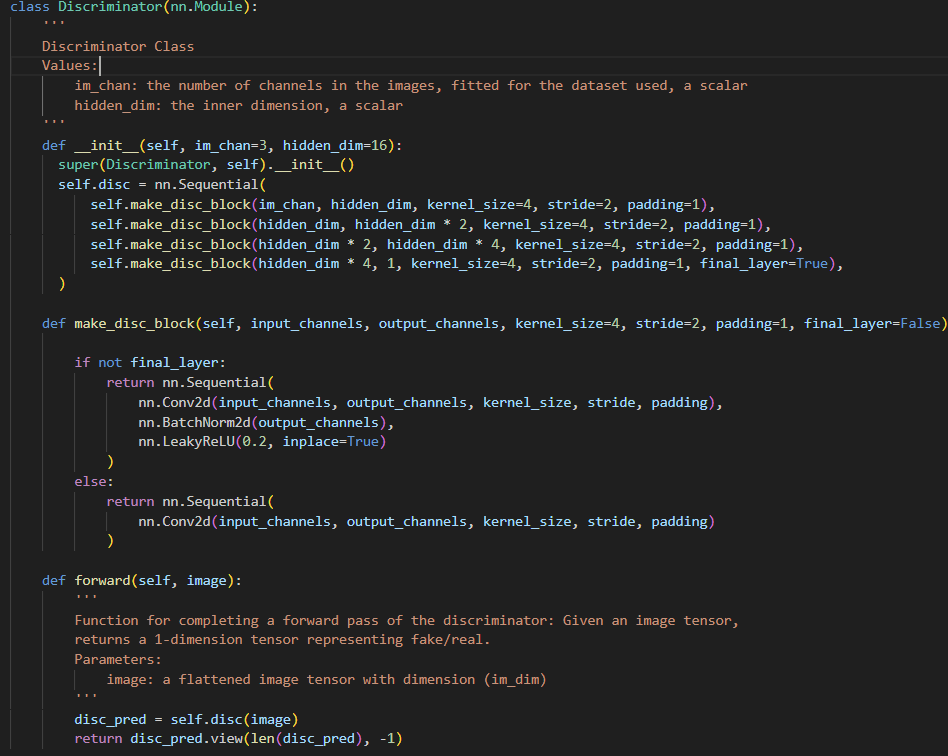


Figure 5.3 Discriminator Model Code.



Figure 5.4 Initialize Discriminator Model Code.

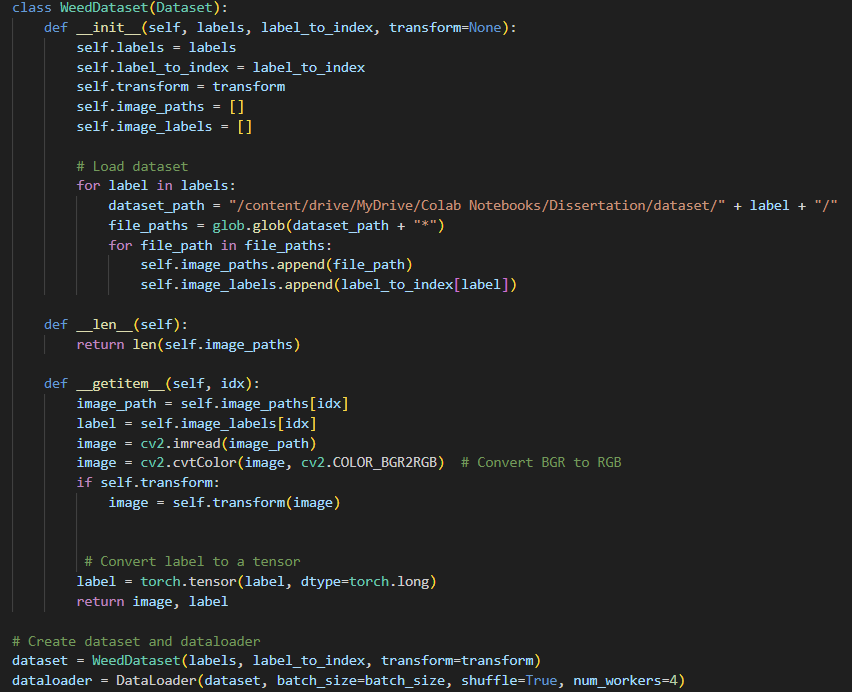


Figure 5.5 Dataset Extraction Code.

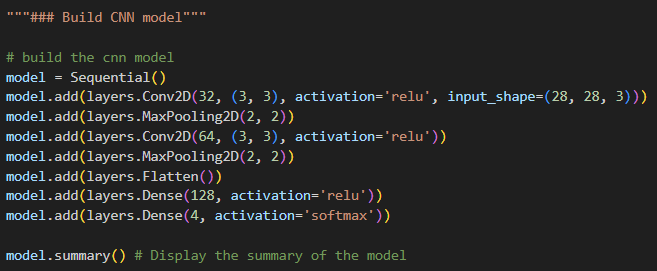


Figure 5.6 CNN Model Code.