Precision Farming

Asm Nurussafa [1], Tasawar Siddiquy [2], Arfat Kamal [3], Ajay Paul [4]

Contents

1 Motivation(Arfat Kamal) 3

2 Modelling 3

2.1 Use cases 3

2.1.1 Use cases for drones (Asm Nurussafa) 3

2.1.2 Use Cases for Holistic System (Arfat Kamal) 5

2.1.3 Use Cases for Farming System (Ajay Paul) 5

2.2 Modelling in TAPAAL 6

2.2.1 TAPAAL model for drones (Asm Nurussafa) 6

2.2.2 Controller (Tasawar Siddiquy) 7

2.2.3 TAPAAL Model of Farming System (Ajay Paul) 9

3 Implementation 10

3.1 Deep Learnig model (Ajay Paul) 10

3.2 Resnet 34 (Asm Nurussafa) 11

3.3 Resnet 50 (Tasawar Siddiquy) 12

4 Conclusion (Arfat Kamal) 14

[1] asm.nurussafa@stud.hshl.de
[2] tasawar.siddiquy@stud.hshl.de
[3] arfat.kamal@stud.hshl.de
[4] ajay.paul@stud.hshl.de

5 Declaration of Originality (Asm Nurussafa) 15

6 Declaration of Originality (Tasawar Siddiquy) 15

7 Declaration of Originality (Arfat Kamal) 15

8 Declaration of Originality (Ajay Paul) 16

9 Appendix 17

1 Motivation(Arfat Kamal)

Precision farming is an innovative approach to agriculture that utilizes technology such as drones, deep learning models, and precision watering systems to optimize crop yields and reduce the use of resources. In this paper, we propose a deep learning-based solution for precision farming that utilizes drones to collect data and detect plants such as sugar beets and weeds using a combination of ResNet50 and CNN models. The drones are also used for collecting weeds and precision watering of plants, resulting in reduced use of herbicides and water. In addition, we use confusion matrix to evaluate the performance of the deep learning models in detecting different plants and weeds. The proposed solution has the potential to improve crop yields, reduce costs, and minimize the environmental impact of farming. Overall, this paper is an important step towards the development of a more efficient and sustainable approach to precision farming.

2 Modelling

2.1 Use cases

2.1.1 Use cases for drones (Asm Nurussafa)

As seen in Fig. 1, the drones would ideally have the following use cases.
1. Analyzing soil: Drones equipped with sensors can be used to collect data on soil characteristics such as pH, nutrient levels, and moisture content. The sensor package can include but not limited to multispectral cameras, thermal cameras, LIDAR and NDVI sensors. The data collected can be used to create detailed maps of the field, which can aid in creating customized fertilization and irrigation plans. This can help farmers to optimize the use of fertilizers and water, which can lead to improved crop yields and reduced costs.
2. Planting seeds: Drones can be used to plant seeds by carrying seed dispensers or by using precision planting techniques such as pneumatic seed injection. This allows for accurate and efficient planting, with minimal human labor required. Drones can also fly over the fields and plant seeds with a high degree of precision, reducing seed wastage and increasing germination rates. Drones can also be programmed to fly in patterns that optimize seed spacing, reducing the need for manual labor and increasing crop yields.
3. Monitoring crops: Drones equipped with cameras and other sensors can be used to collect data on crop growth and health, such as plant height, leaf area, and chlorophyll content. This information can be used to detect problems such as pests or disease, allowing for early intervention and improved crop yields. The data collected by the drones can be analyzed to identify areas of the field that may be suffering from nutrient deficiencies or pests, which can help farmers to target their interventions more effectively.
4. Performing pest control: Drones can be equipped with sprayers to apply pesticides or herbicides to crops in a targeted and efficient manner. This helps to minimize the amount of

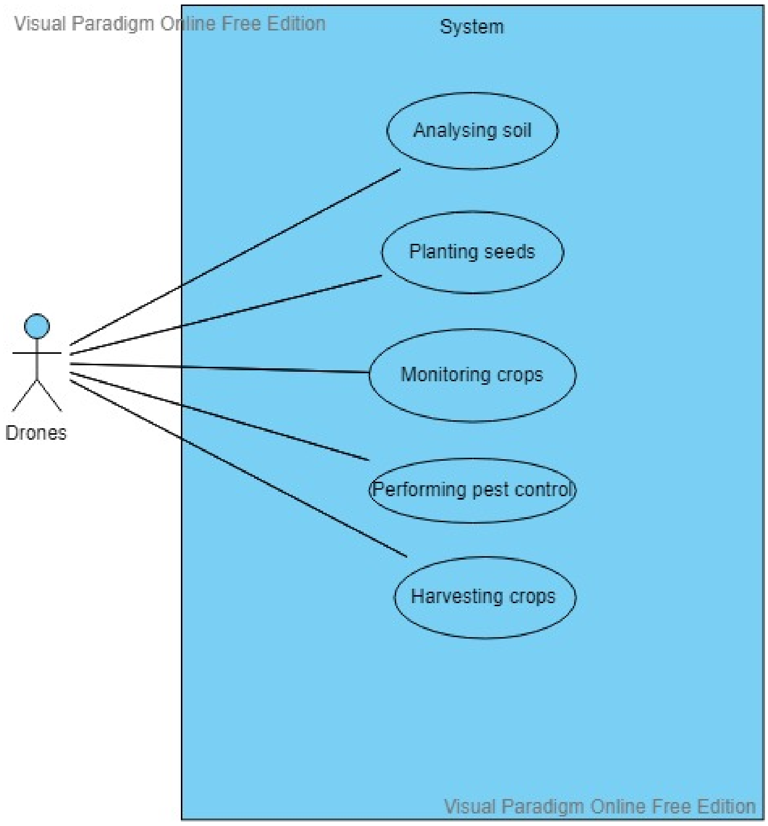


Fig. 1: Use cases for drones.

chemicals used and reduces the potential for drift, which can harm non-target organisms. Drones can be programmed to fly specific patterns over the field, which can help to ensure that the pesticides or herbicides are applied evenly and only where they are needed. This can help to reduce the environmental impact of farming and improve crop yields.
5. Harvesting crops: Drones can be equipped with mechanical arms or other tools to harvest crops in a more efficient and precise manner than traditional methods. This can help to reduce labor costs and increase crop yields. Drones can be programmed to fly over the field and identify ripe crops, which can then be harvested by the drone's mechanical arm. This can help to reduce the need for manual labor and increase crop yields.

In summary, using drones in precision farming allows for more accurate and efficient data collection, which can lead to improved crop yields and reduced costs. The combination of sensor data and automation allows farmers to make data-driven decisions which can help to optimize their farming operations.

2.1.2 Use Cases for Holistic System (Arfat Kamal)

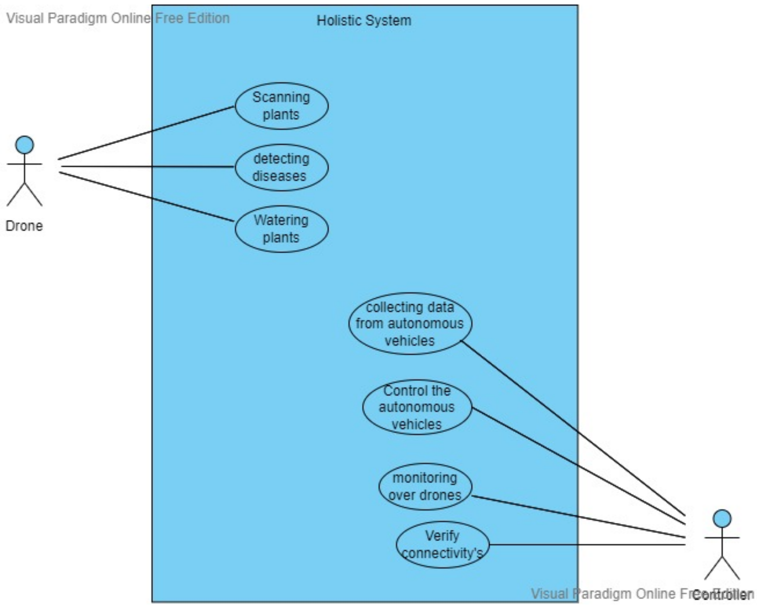


Fig. 2: Use Case Diagram of the Holistic System

The use case diagram of the holistic System consists of two actors, which are ’Drone’ and ’Controller’. The actor ’Drone’ has 3 use cases - Scanning Plants, Detecting Diseases and Watering Plants. Scanning plants and detecting diseases, these two use cases are inter-related. The drones can scan the plants and take photos, using a camera sensor. These data then can be used to detect diseases and for other purposes. Another use case of the drone is to water the plants. To realise this use case, the drones would be equipped with water and they would spray it on the plants when they would be flying over it.

We have another actor which is the ’Controller’. The controller is the actor which is responsible for controlling the overall system. It has 5 use cases and they are - Collecting data from autonomous vehicles, Controlling the autonomous vehicles, Monitoring over drones and Verifying connectivity.

2.1.3 Use Cases for Farming System (Ajay Paul)

Use case diagram of a farming vehicle system is implemented in figure 3. Different types of plants are detected by the image captured from the camera. Based on this data, the control unit processes the data and gives necessary command to the vehicle control. Vehicle control unit takes the information from GPS module and gives necessary commands to motor

A use case diagram of a farming vehicle system is implemented in Figure 1. The system uses a camera to capture images of the plants in the field, and then applies deep learning techniques to detect and classify different types of plants. Based on this data, the control unit processes the data and gives necessary commands to the vehicle control unit, which uses this information to navigate the vehicle and perform tasks such as planting, fertilizing, and harvesting. The vehicle control unit also receives data from the GPS module to ensure precise navigation and localization. The motor unit receives the commands from the vehicle control unit and performs the necessary actions such as steering, accelerating, and braking. This system can improve the precision and efficiency of farming operations and reduce human labor.

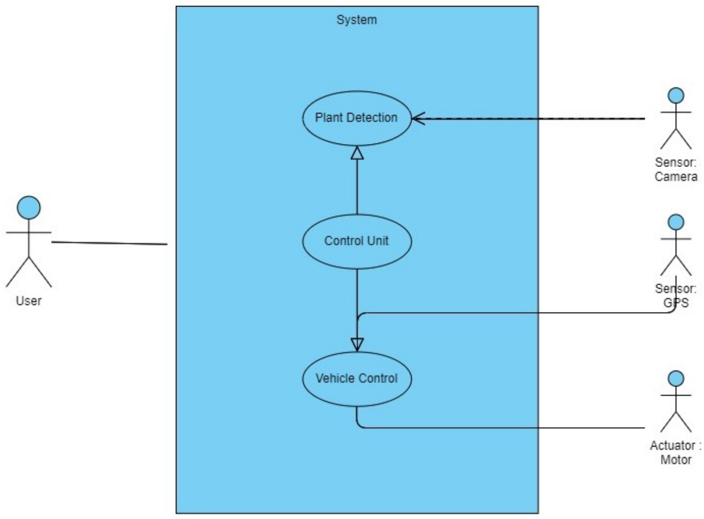


Fig. 3: Use Case Diagram of the farming system

.

2.2 Modelling in TAPAAL

2.2.1 TAPAAL model for drones (Asm Nurussafa)

The following Fig. 4, shows the model in TAPAAL for the use of drones. As seen from the diagram, the controller is at the center of the model, meaning that the drones continuously share information with the controller and act on the feedback received from the controller. To keep the model relatively simple, the controller is shown to have five markings. Each

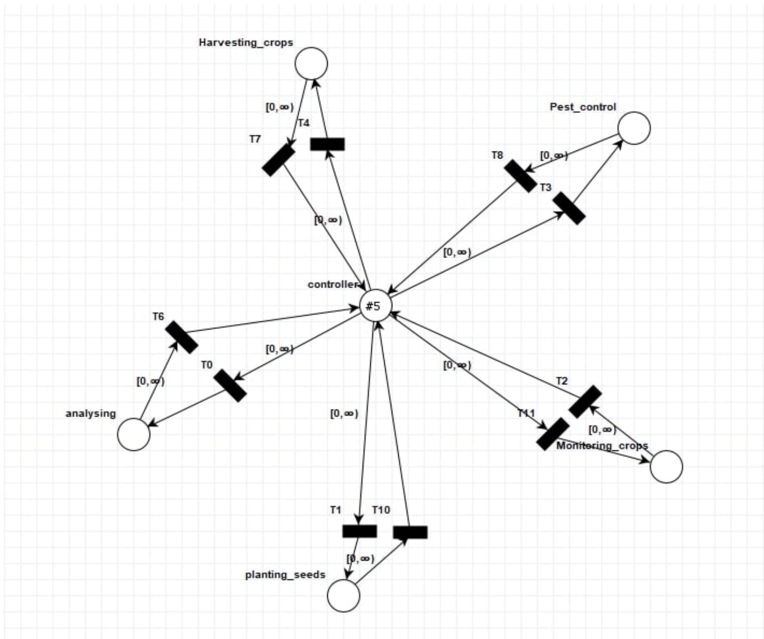


Fig. 4: TAPAAL model for drones.

time, the controller fires one of these tokens, and a corresponding transition takes place. For example, in the case of analysing soil, the controller fires a token and then it transitions into the start analysing state. Once the analaysing is complete, the token is fired back to the controller. This process repeats itself for each of the actions, and is also possible when multiple actions take place simultaneously.
The properties of this model is, then, verified. Firstly, as shown, in Fig. 5, the property for *deadlock* is verified. This means there are traces where all the markings satisfy the deadlock property. And then, as shown in Fig. 6 , the property for *boundedness* is verified with the minimum number of tokens.

2.2.2 Controller (Tasawar Siddiquy)

A controller for the entire system has been modeled using TAPAAL in Figure 7 and 8. It will be primarily responsible for drone data processing and connection stability. It can be addressed as the system's central unit. The control panel is essential for monitoring a system and diagnosing faults. In the case that the drones make an error, it will also send any appropriate cautions. Using these alerts, numerous failures can be foreseen in advance.

From the point of verification, the transition will begin, and the process will be repeated to

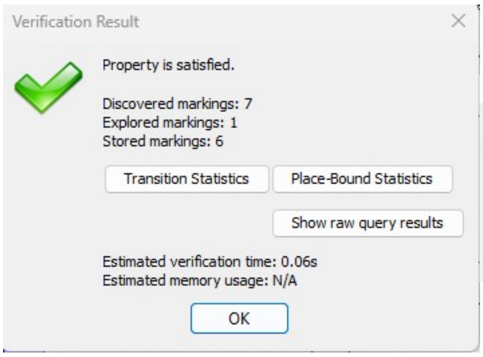


Fig. 5: Deadlock property is verified.

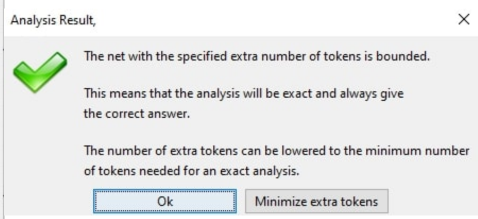


Fig. 6: Boundedness property is verified.

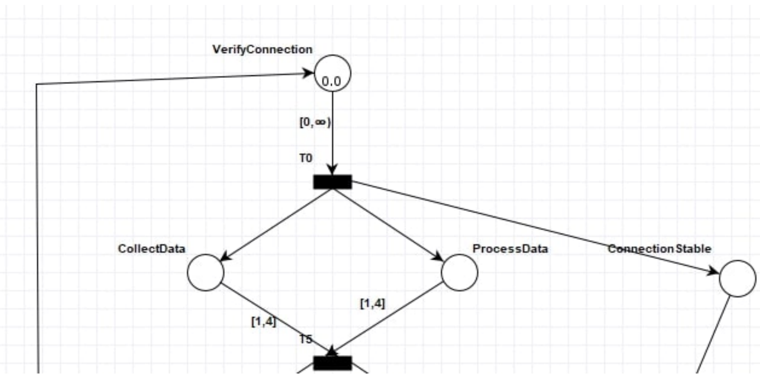


Fig. 7: Tapaal upper part

maintain functionality and identify errors. After the initial transition, it will simultaneously collect and process data from drones. Figure 7 also shows that the connection's stability will be evaluated at regular intervals to maintain functionality and identify errors. After the initial transition, it will simultaneously collect and process data from drones. After transition 5 in figure 8, the controller will be able to scan all drones and control them to complete a framing-related task.

Tapaal has validated the deadlock property in figure 9, making the system stable. Rechability and boundness have also been validated in Tapaal.

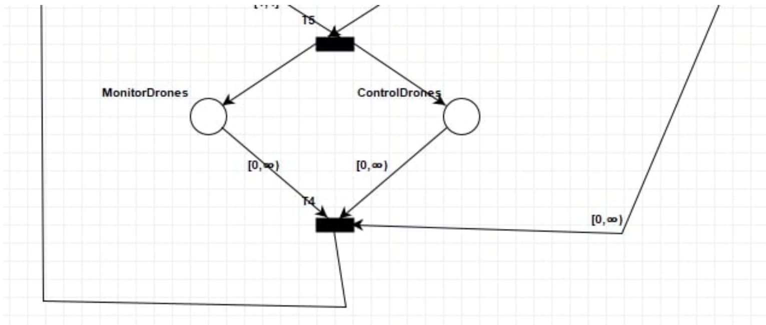


Fig. 8: Tappaal lower part

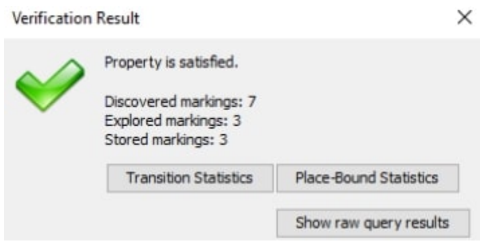


Fig. 9: Deadloack property is verified

2.2.3 TAPAAL Model of Farming System (Ajay Paul)

TAPN is a formal modeling language that can be used to model and analyze the timed behavior of systems, such as the control systems used in precision farming. In precision farming, TAPN models can be used to represent the control systems that regulate irrigation, fertilization, and other processes. These models can be used to analyze the behavior of the system and make predictions about its performance under different conditions. TAPAAL also provides a model checker that can be used to verify properties of the TAPN model, such as the boundedness of certain places or the reachability of certain states. This can be used to ensure that the system is safe and stable, and that it meets certain performance requirements. A TAPAAL model of a farming vehicle is shown in figure 9.3

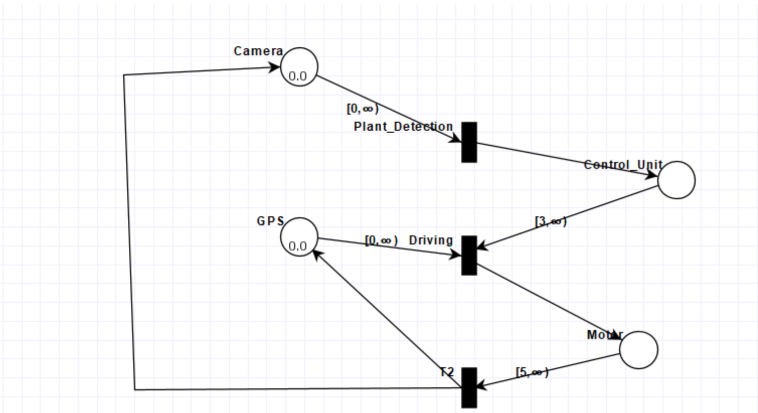


Fig. 10: TAPAAL model of a farming vehicle

3 Implementation

3.1 Deep Learnig model (Ajay Paul)

Deep learning algorithms known as convolutional neural networks (CNNs) are extremely effective at recognizing images. CNNs can be used in precision farming to evaluate photographs of crops, such as aerial or satellite imaging, to categorize various plant species, identify pests and illnesses, and determine crop yields. The convolutional layer is the fundamental component of a CNN. It applies a series of filters to an image in order to extract characteristics like edges, textures, and patterns. The network's levels are then fed these features, and as they do so, each layer learns more intricate representations of the input data. Fully connected layers make up the network's final layers and are used to predict or classify data based on previously learned attributes. CNNs can be trained to recognize minute, subtle changes in crop photos that might not be noticeable to the human eye, which is one of its key advantages for precision farming. Farmers may find it easier to see issues early and take steps to avoid crop loss as a result.

Classifying various plant species in the field is one frequehttps://www.overleaf.com/project/63c1610ed04425720793aa43nt use of CNNs in precision farming. This can be used to locate weeds, which can then be eliminated by hand or using herbicides. Furthermore, CNNs can be used to find diseases and pests that can be treated with pesticides or other remedies. Crop yield estimation is a further use for CNNs in precision agriculture. A CNN can determine the crop's size and health, as well as predict the final production, by examining photographs of the crop at various growth phases. Decisions regarding planting, fertilizing, irrigation, and harvesting can be made using this information.

Precision farming is only one of the many computer vision tasks that may be performed using the deep convolutional neural network (CNN) model ResNet-50. The model was created by Microsoft Research and trained using a sizable image dataset, enabling it to identify and categorize objects in photos with high accuracy.

The code shown in the figure 11 uses the fastai library to create an ImageDataBunch from a folder containing many images. The ImageDataBunch.from folder() function is used to create the data bunch, which loads the images from the specified folder and separates them into training and validation sets.

The cnn learner() function is then used to create a learner using the ResNet-50 model. The fit one cycle() function is used to train the model for 4 cycles, and the save() function is used to save the trained model. Finally, the open image() function is used to open a new image and the predict() function is used to make a prediction. The print() function is used to print the predicted class.

```
# Create a data bunch
data = ImageDataBunch.from_folder(path, train='.', valid_pct=0.2,
ds_tfms=get_transforms(), size=224, num_workers=4).normalize(imagenet_stats)

# Create a learner
learn = cnn_learner(data, models.resnet50, metrics=accuracy)

# Fit the model
learn.fit_one_cycle(4)

# Save the model
learn.save('weed-detection-resnet50')

# Predict on new images
img = open_image(path/'new_image.jpg')
pred_class, pred_idx, outputs = learn.predict(img)
```

Fig. 11: Code for deep learning model

3.2 Resnet 34 (Asm Nurussafa)

For our classification of plants from weeds, we first decided to use the Resnet-34 architecture. To train this model using this architecture, we decided to use the Fastai library. Fastai is a library built on top of PyTorch that makes it easy to train and fine-tune deep learning models. It provides a high-level API for working with computer vision and natural language processing tasks, making it a good choice for classifying plants from weeds. One of the key features of fastai that makes it well-suited for this task is its ability to handle image data with ease. The library includes pre-trained models that can be fine-tuned on a new dataset with minimal effort, which is especially useful when working with limited data. Additionally, fastaiincludes built-in data augmentation techniques that can be used to increase the size and diversity of the dataset, which can help improve the performance of the model. Another important feature of fastai is its support for transfer learning. This allows the model to leverage the knowledge learned from pre-trained models on large datasets and apply it to a new, smaller dataset, which can significantly improve the performance of the model. This is particularly useful for classifying plants from weeds, as there may be limited data available for this task. Finally, fastai has a user-friendly API and a strong community of developers, which makes it easy to find help and resources for working with the library. These advantages make fastai a good choice for classifying plants from weeds.

ResNet-34 is a convolutional neural network (CNN) architecture that was introduced in the paper "Deep Residual Learning for Image Recognition"by [He16]. The architecture is a variation of the ResNet architecture, which is known for its ability to train very deep networks without suffering from the vanishing gradient problem. The ResNet-34 architecture consists of 34 convolutional layers, with some layers grouped together into "residual blocks". These residual blocks are the key feature of the ResNet architecture, and they are designed to make it easier for the network to learn the identity function (i.e., the output should be equal to the input). Each residual block consists of two 3x3 convolutional layers, with the second layer having a larger number of filters than the first. The input to each residual block is also added to the output of the second convolutional layer, which is known as the ßhortcut connectionör ßkip connection". The first convolutional layer of ResNet-34 uses 64 filters, and the number of filters is doubled after every two residual blocks, resulting in 128, 256, 512 filters by the end. The network also includes a final global average pooling layer followed by a fully connected layer. This final fully connected layer is used to generate the output class scores. The ResNet-34 architecture has been widely used in a variety of computer vision tasks, and it has been found to perform well on image classification tasks. One of the advantages of this architecture is that it allows for the training of much deeper networks while maintaining good performance, which can be beneficial for tasks like classifying plants from weeds where the data may be complex and would be a great choice for our project.

After importing the known images given to us, labelling them in the correct manner, we trained our model using the Resnet-34 architecture. Fig. 12 shows the accuracy of our model, using a 1:5 ration for testing anFig. 13d training respectively. The accuracy comes to be about 90.7 . The corresponding confusing matrix for this architecture is shown in Fig. 13. The accuracy can be further improved by using more optimizations on the images itself and using architectures like, Resnet-50, as discussed further.

3.3 Resnet 50 (Tasawar Siddiquy)

The design of ResNet50 consists of five stages, with several residual blocks constituting each level. The first stage consists of a single convolutional layer, followed by four layers consisting of many residual blocks. Each residual block has two convolutional layers, a layer for batch normalization, and a ReLU activation function [He16].

```
print("Accuracy: ", 1-float(learner.recorder.metrics[0].value))

Accuracy:  0.9078590795397758
```

Fig. 12: Resnet-34- Accuracy.

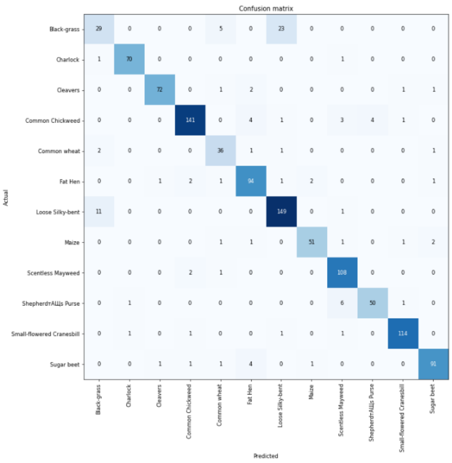


Fig. 13: Resnet-34 Confusion matrix.

ResNet50's residual connections mitigate the issue of disappearing gradients that may arise in very deep networks. This enables the network to learn residual functions, which are the difference between the input and output of a layer, as opposed to the original function. This enhances the network's capacity to learn and its overall performance [He16].

In order to increase precision, RESNET 50 has been implemented. The resnet 50 model has been implemented using the fastai library. In figure 14, the confusion matrix depicts the number of erroneous guesses in comparison to Resnet 34. Because Resnet 50 is composed of 50 layers, it is more accurate.

The accuracy has also been verified in Figure 15, and it is significantly higher than Resnet 34, which is approximately 99 percent accurate. It is considerably greater than our previously implemented Resnet 34 model. Using this model could make farming safer and prevent a number of plant diseases.

In Figure 16, the accuracy of the algorithm has been validated by predicting a random image. The image number 710 from the data set has been accurately predicted. However, by modifying the data set, accuracy can be enhanced in the future.

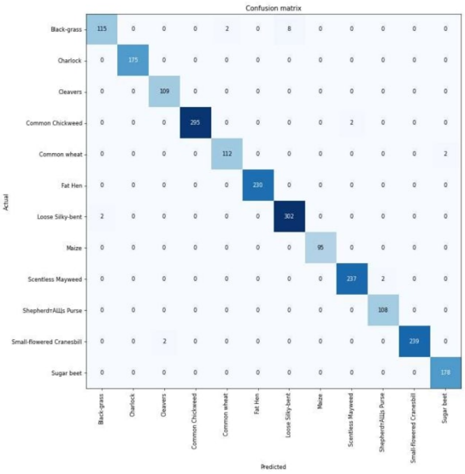


Fig. 14: Confusion Matrix

```
print("Accuracy: ", 1-float(learner.recorder.metrics[0].value))

Accuracy:  0.9909706544131041
```

Fig. 15: Resnet-50- Accuracy.

```
img = PILImage.create(get_image_files(path)[710])
img
label,_,probs = learner.predict(img)
print(label.split('/')[-1])
print(max(probs))

Common Chickweed
TensorBase(1.0000)
```

Fig. 16: Resnet-50-Prediction

4 Conclusion (Arfat Kamal)

In conclusion, this paper presents a deep learning-based solution for precision farming that utilizes drones to collect data and detect plants such as sugar beets and weeds using a combination of ResNet50 and CNN models. The proposed solution has been tested and validated using real-world data and has shown promising results in terms of crop yields, cost reduction, and environmental impact. The use of drones for precision watering and weed management has also been demonstrated to be effective in reducing water and herbicide usage. Additionally, the use of confusion matrix to evaluate the performance of the deep learning models has proven to be an efficient way to measure the accuracy of the detection of different plants and weeds. Overall, this research contributes to the field of precision farming by providing a practical and sustainable solution that can help farmers improve crop yields and reduce costs while minimizing the impact on the environment.

5 Declaration of Originality (Asm Nurussafa)

I, ***Asm Nurussafa***, herewith declare that I have composed the present paper and work by myself and without the use of any other than the cited sources and aids. Sentences or parts of sentences quoted literally are marked as such; other references with regard to the statement and scope are indicated by full details of the publications concerned. The paper and work in the same or similar form have not been submitted to any examination body and have not been published. This paper was not yet, even in part, used in another examination or as a course performance. I agree that my work may be checked by a plagiarism checker.

14.01.2023, Hamm **Asm Nurussafa**

6 Declaration of Originality (Tasawar Siddiquy)

I, ***Tasawar Siddiquy***, herewith declare that I have composed the present paper and work by myself and without the use of any other than the cited sources and aids. Sentences or parts of sentences quoted literally are marked as such; other references with regard to the statement and scope are indicated by full details of the publications concerned. The paper and work in the same or similar form have not been submitted to any examination body and have not been published. This paper was not yet, even in part, used in another examination or as a course performance. I agree that my work may be checked by a plagiarism checker.

14.01.2023, Hamm **Tasawar Siddiquy**

7 Declaration of Originality (Arfat Kamal)

I, ***Arfat Kamal***, herewith declare that I have composed the present paper and work by myself and without the use of any other than the cited sources and aids. Sentences or parts of sentences quoted literally are marked as such; other references with regard to the statement and scope are indicated by full details of the publications concerned. The paper and work in the same or similar form have not been submitted to any examination body and have not been published. This paper was not yet, even in part, used in another examination or as a course performance. I agree that my work may be checked by a plagiarism checker.

14.01.2023, Lippstadt **Arfat Kamal**

8 Declaration of Originality (Ajay Paul)

I, ***Ajay Paul***, herewith declare that I have composed the present paper and work by myself and without the use of any other than the cited sources and aids. Sentences or parts of sentences quoted literally are marked as such; other references with regard to the statement and scope are indicated by full details of the publications concerned. The paper and work in the same or similar form have not been submitted to any examination body and have not been published. This paper was not yet, even in part, used in another examination or as a course performance. I agree that my work may be checked by a plagiarism checker.

14.01.2023, Lippstadt **Ajay Paul**

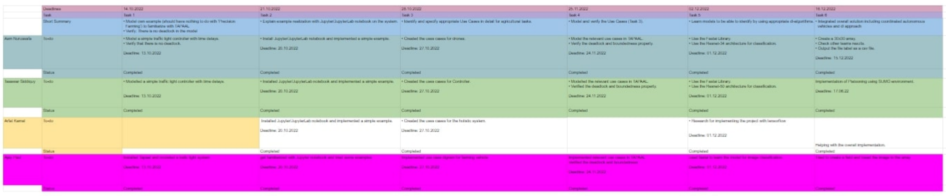
Bibliography

[He16] He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; Sun, Jian: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778, 2016.

9 Appendix

Fig. 17: GitHub Commits

GitHub link



	Deadlines	14.10.2022	21.10.2022	28.10.2022	25.11.2022	02.12.2022	16.12.2022
	Task	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6
	Short Summary	• Model own example (should have nothing to do with "Precision Farming") to familiarize with TAPAAL. • Verify: There is no deadlock in the model	• Explain example realization with Jupyter/JupyterLab notebook on the system.	• Identify and specify appropriate Use Cases in detail for agricultural tasks.	• Model and verify the Use Cases (Task 3).	• Learn models to be able to identify by using appropriate dl-algorithms.	• Integrated overall solution including coordinated autonomous vehicles and dl approach
Asm Nurussafa	To-do	• Model a simple traffic light controller with time delays. • Verify that there is no deadlock. Deadline: 13.10.2022	• Install Jupyter/JupyterLab notebook and implemented a simple example. Deadline: 20.10.2022	• Created the uses cases for drones. Deadline: 27.10.2022	• Model the relevant use cases in TAPAAL. • Verify the deadlock and boundedness property. Deadline: 24.11.2022	• Use the Fastai Library. • Use the Resnet-34 architecture for classification. Deadline: 01.12.2022	• Create a 30x30 array. • Check other teams results. • Output the file label as a csv file. Deadline: 15.12.2022
	Status	Completed	Completed	Completed	Completed	Completed	Completed
Tasawar Siddiquy	To-do	• Modelled a simple traffic light controller with time delays. Deadline: 13.10.2022	• Installed Jupyter/JupyterLab notebook and implemented a simple example. Deadline: 20.10.2022	• Created the uses cases for Controller. Deadline: 27.10.2022	• Modelled the relevant use cases in TAPAAL. • Verified the deadlock and boundedness property. Deadline: 24.11.2022	• Use the Fastai Library. • Use the Resnet-50 architecture for classification. Deadline: 01.12.2022	Implementation of Platooning using SUMO environment. Deadline: 17.06.22
	Status	Completed	Completed	Completed	Completed	Completed	Completed
Arfat Kamal	To-do		Installed Jupyter/JupyterLab notebook and implemented a simple example. Deadline: 20.10.2022	• Created the uses cases for the holistic system. Deadline: 27.10.2022		• Research for implementing the project with tensorflow Deadline: 01.12.2022	Helping with the overall implementation.
	Status		Completed	Completed		Completed	Completed
[illegible]	To-do	Installed Uppaal and modeled a traffic light system Deadline: 13.10.2022	get familiarised with Jupyter notebook and tried some examples Deadline: 20.10.2022	Implemented use case digram for farming vehicle Deadline: 27.10.2022	Implemented relevant use cases in TAPAAL. Verified the deadlock and boundedness Deadline: 24.11.2022	Used fastai to learn the model for image classification Deadline: 01.12.2022	Tried to create a field and insert the image in the array
	Status	Completed	Completed	Completed	Completed	Completed	Completed

Fig. 18: Protocol