Multi-Object Detection with CNNs

Report of the first project of Deep Learning Course

Tam Gabriele 153262

Deep Learning Course
University of Udine

Contents

List of Figures

Introduction

Multi-object Detection

The problem of multiple object detection is a key challenge in the field of computer vision, where the objective is to detect and classify multiple objects within an image. In the specific field of the project, the attention is focused on images containing plants, addressing the need to predict not only the presence of plants, but also their location and type.

Convolutional Neural Networks (CNNs)

Convolutional neural networks (CNNs) represent an advanced approach to tackle computer vision problems, particularly effective in pattern recognition in images. Unlike traditional neural networks, CNNs utilise a convolution layer to automatically extract relevant features from images. This allows the networks to efficiently learn hierarchies, patterns, facilitating the detection of objects and their classification.

The Dataset

The dataset called "CropOrWeed" consists of a set of labelled images, each containing plants, both crops and weeds. Each image is accompanied by detailed information on the location of the plants and their category, distinguishing between crops and weed.

Sliding Window

The sliding window technique is an effective strategy to simplify the complexity of detection on image. This method involves the subdivision of the image into smaller images that are subsequently supplied to a neural network: during the design phase, both the images of the training and the testing sets are divided into smaller regions, forming sub-images. The main challenge is to determine the optimal size of these regions, and different sizes, such as 100x100, 200x200 and 500x500, have been explored.

The sliding window approach was also chosen considering what was learnt during the theoretical lectures. However, it is important to emphasise that an alternative approach was also considered: give the entire image as input to the neural network. This approach, although interesting, encountered practical obstacles during its implementation: even though the neural network was constructed in a relatively simple structure, with only a few layers, it was difficult to train due to the large number of parameters involved because of the vast input provided. This problem manifested itself with the exhaustion of RAM memory during the training phase, both in the Google Colab environment and local environment.

The solutions proposed below will therefore focus on the sliding window approach, considered more practical and manageable.

CNN Approach using Image Classification and Sliding Window

My first approach was to simplify the complex problem of locating plants in an image into a more manageable problem of image classification. The objective was to make it possible that, given an image fragment, the system could recognise whether represented an area with crops ("type0"), weed ("type1"), or whether it contained no plants ("dirt").

Creation of the Dataset

I started by creating a dataset from the original images and their associated CSV files. These images were divided into fragments of the size of the tested sliding window, and each fragment was associated with a specific label. The label was assigned according to the following criteria:

- If there was no bounding box intersecting the image, the label was set to "dirt".
- If there was at least one bounding box intersecting the image, the label of the box covering the largest possible area was chosen.

Subsequently, the images were organized into folders, where each folder represented the name of the label. To improve the model's accuracy, a data augmentation technique was implemented by generating new images through random transformations such as horizontal/vertical flipping, contrast and brightness adjustment, and rotation. To address the dataset imbalance, some samples from the "dirt" class were randomly removed.

Building the model

I designed a model based on a sequence of convolutional layers and Max pooling layers, followed by a series of fully connected layers. The model is completed with a Softmax layer with three different values, representing the probability that the image belongs to one of the previously mentioned classes.

During the training phase, the model appeared to perform well, with the loss decreasing and the accuracy increasing for the classes.

Evaluation of the model

However, the subsequent challenge was the actual reconstruction of bounding box regions. For this purpose, we implemented a system of binary masks indicating the presence or absence of a specific class relative to the entire image. Each image had two masks: one for "type0" and the other for "type1". These masks were compared with the ground truth using the Intersection over Union (IOU) as a metric.

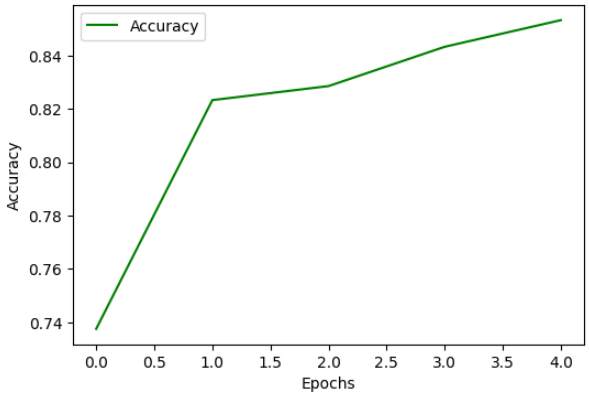
During the comparison of predicted masks with the ground truth in the test set, I observed the following results:

- The model is capable of recognizing areas where plants are present.
- However, the bounding box is not precise because its area is a multiple of the sliding window: this is a limit of this approach.
- However, the bounding box is not precise because its area is a multiple of the sliding window: this is a limit of this approach.
- Another limitation of this approach is that it can only recognise one label per sub-image (so if it contains both weed and crop at the same time, the system can only recognise one).
- The larger the size of the sliding window, the higher the accuracy of the predicted class.
- Using smaller sliding windows improves the precision of the box but reduces the accuracy of the label.
- The best results, as well as the right compromise, were obtained using sub-images of 100x100 pixels.

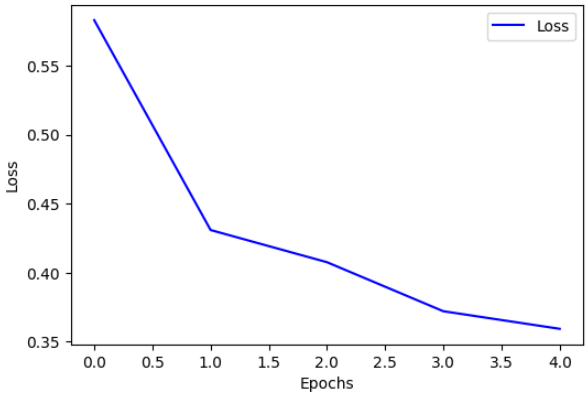
An example of a predicted test set image is Figure 2, while accuracy and loss during model training are presented in Figure 1.

Conclusion

In conclusion, the approach has demonstrated success in recognizing the presence of plants, but there are still challenges to address in the precision of bounding box localization.

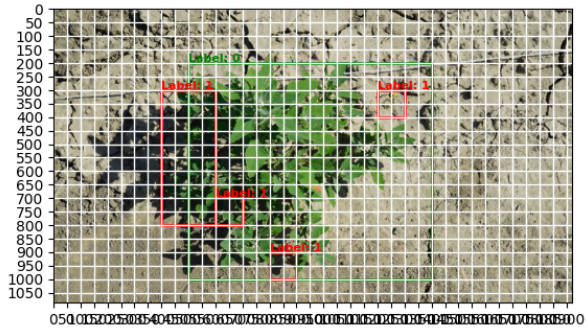


(a) Accuracy during training of classification model

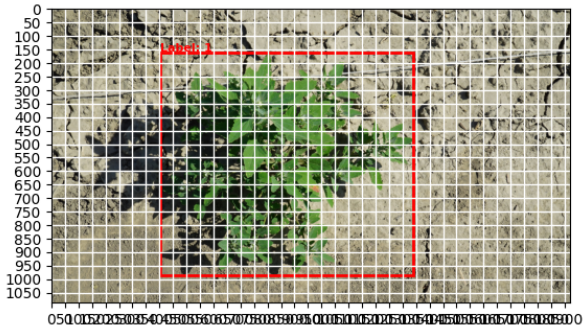


(b) Loss during training of classification model

Figure 1: Accuracy and Loss during training of classification model



(a) Real image



(b) Prediction

Figure 2: Real image and Prediction using classification model

CNN Approach using Image Classification with localisation and Sliding window

Afterwards, another approach, aimed at improving the accuracy of bounding boxes, was adopted. The idea is to associate, in addition to the corresponding label, a bounding box, represented by four values, with each image. For the selection of these values, two alternatives were explored:

- For each sub-image, the intersection with the bounding boxes is evaluated, and if present, the one with a larger area is considered. The four values represent the points of the box inside the sub-image.
- For each sub-image, it is checked whether the center of the bounding box is present in the image. If affirmative, the four points represent the center of the bounding box with respect to the image, its height, and width.

Regarding the labels, two alternatives were also considered:

- Reusing the three labels presented earlier.
- Introducing another value called "presence". This binary value represents the absence (with a value of 0) or presence (with a value of 1) of a bounding box associated with the sub-image. The label will therefore consist of only two classes and will be taken into account only if the "presence" value is 1.

The best performance was obtained by using the co-ordinates of the centre and the height and width for the box and using two labels with the presence value in combination.

Architectures

Four different architectures were designed to build the network. The diagrams of the architectures are also presented with pictures at the end of the report.

Deep with final split

It consists of 5 groups of convolutional layers, batch normalization layers, max pooling layers with a kernel size of 2, and a ReLU activation function. Subsequently, a split into 3 distinct branches occurs, each passing through 2 fully connected layers. Each branch produces a specific result: one for presence (with the application of the sigmoid function), one for class (with softmax activation function), and the last one for the four bounding box values.

This approach showed lower performance compared to others tested, with the network exhibiting poor specialization for different values to predict. In essence, the network seemed to be trying to handle everything together, specializing only in the final part.

Split-in-the-middle

Following the lessons learned from the first architecture, a second architecture was created. After an initial shared phase where convolutional layers (this time with size 5) and max pool layers are alternated, it splits into three distinct branches, each dedicated to predicting a specific value. On each branch, 3 fully connected layers are applied.

This model showed significantly better performance than the previous one, suggesting that increased depth, more parameters, and an early split can improve network performance.

Bifurcation

The idea was to predict the desired values at different times: the goal was to mimic human reasoning, first understanding if the plant is present or not in the image, then understanding the type of plant, and finally defining its position. This architecture was designed with 3 groups of convolutional layers and max pooling. After each group, a separate branch was created, to which 2 fully connected layers were applied, producing the predicted value.

1x1 filter

In the last architecture, the use of 1x1 convolutional filters was explored, completely replacing fully connected layers. This approach led to a significant improvement in performance in terms of model training time. This architecture,

however, was not considered in the tests because it did not achieve the desired performance or was worse than its predecessors.

Considerations

Some general considerations:

- The decision was made to exclusively examine filters of sizes 5x5, 7x7, and 3x3, as even-sized filters, such as those of sizes 2x2 and 4x4, led to the loss of some pixels.
- Among filters of sizes 3x3, 5x5, and 7x7, an interesting pattern emerged: the larger the filter size, the larger the area of involved pixels. Following various tests, including the constant setting of all filters to values like 5x5, 7x7, or 3x3, as well as experimenting with combinations of filters (e.g., using a 7x7 filter in the early stages and a 3x3 filter in the later stages, and vice versa), it was observed that optimal performance was achieved with filters of sizes 3x3 and 5x5.
- Noteworthy is the effectiveness of the 1x1 filter as a valid alternative to fully connected layers. This filter significantly reduces the total number of network parameters, thus contributing to a more efficient resource management.
- It is important to highlight the use of "flatten" layers in the proposed CNNs: these were used to create a 1D vector, which was then used as input for the fully connected layers.
- During training, of all the architectures, the sum of the losses with respect to presence, label and box was considered as loss. Approaches giving different weights to the losses were also tried but appeared very similar to considering them all with the same weight.

Results

The three models were trained on of the whole trainset, first doing a phase in which I increased the number of images labelled "type0" (which were less numerous than those labelled "type1") by generating new images with transformations (such as horizontal and vertical flips, adding a blur, random resizing changing contrast and brightness) and randomly eliminating samples with "dirt" type to make the dataset balanced. Figures 3,4 and 5 show the training phase of the three models: all of the three values sought, decreases epoch after epoch in all cases. Figure 9 shows some images taken from the test set and analysed with the three models: one can see how the second model is able to predict many more areas than the others, which fail to detect plants. Particularly noticeable is how the first and third models fail to identify small plants well, the first trying to give as small a bounding box as possible. The third one fail to detect the second type of plant. The second model, on the other hand, detects many more bounding boxes and this makes it better.

However, comparing these three models trained with the test set, none of the three gives exorbitant results (figure 10).

In conclusion, the second model seems to be the best, and therefore the architecture that exploits a first shared part and then has a last part where the branches split where "specialisation" takes place seems to be the best idea. Furthermore, again considering the best results obtained, I think that using 3x3 or 5x5 filters on the convolutional layers is the best idea. Also, the deeper the network, the better it is able to understand the image: the first part where you have a group of convolutional layers and a maxpool helps the network to understand and get an idea of the image, afterwards the branches have a better understanding in relation to their task.

This project was a way to develop my skills and put into practice what I learnt during the theory lessons.

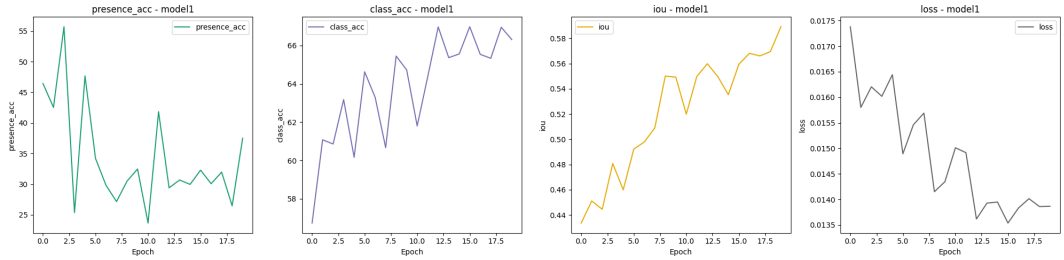


Figure 3: Accuracy of “presence”,“label”, IOU and loss during the train of “Deep model with final split”

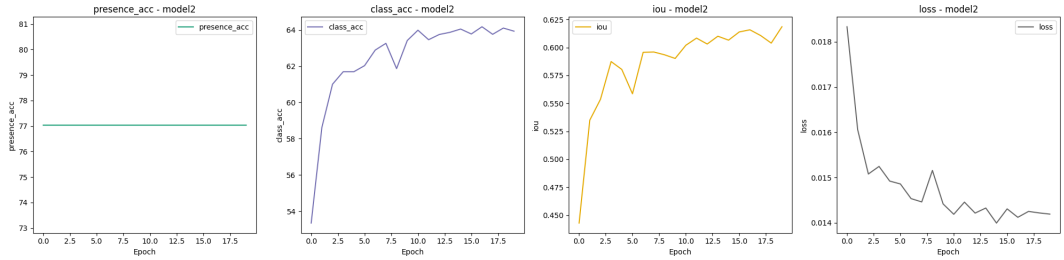


Figure 4: Accuracy of “presence”,“label”, IOU and loss during the train of “Model with split in the middle”

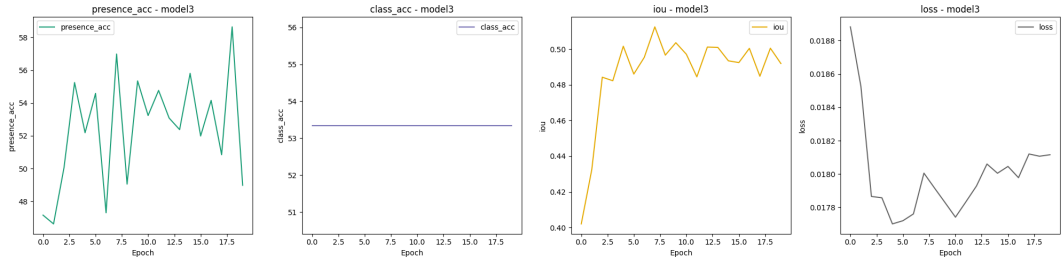


Figure 5: Accuracy of “presence”,“label”, IOU and loss during the train of “Model with bifurcations”

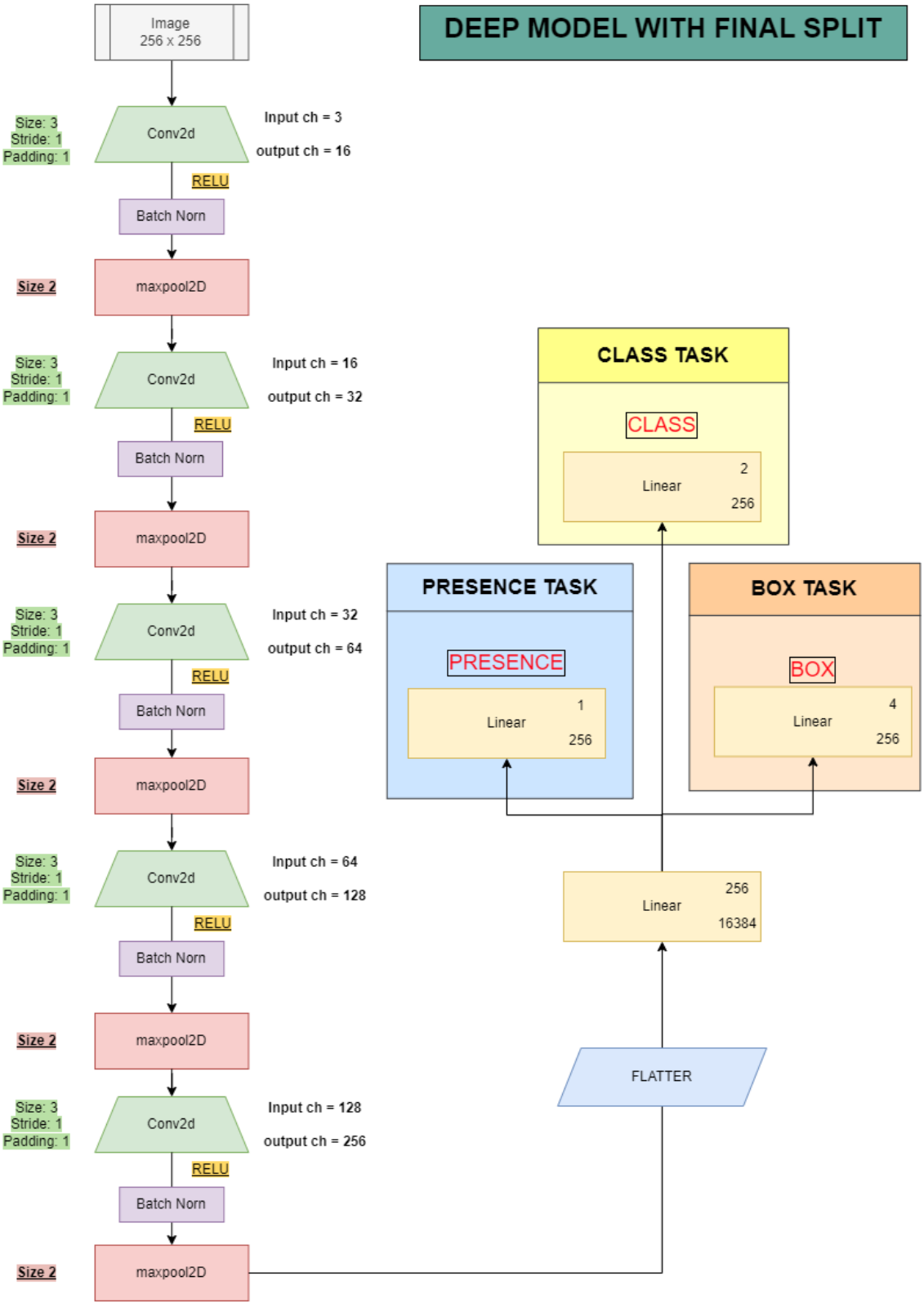


Figure 6: Model architecture diagram of "Deep model with final split"

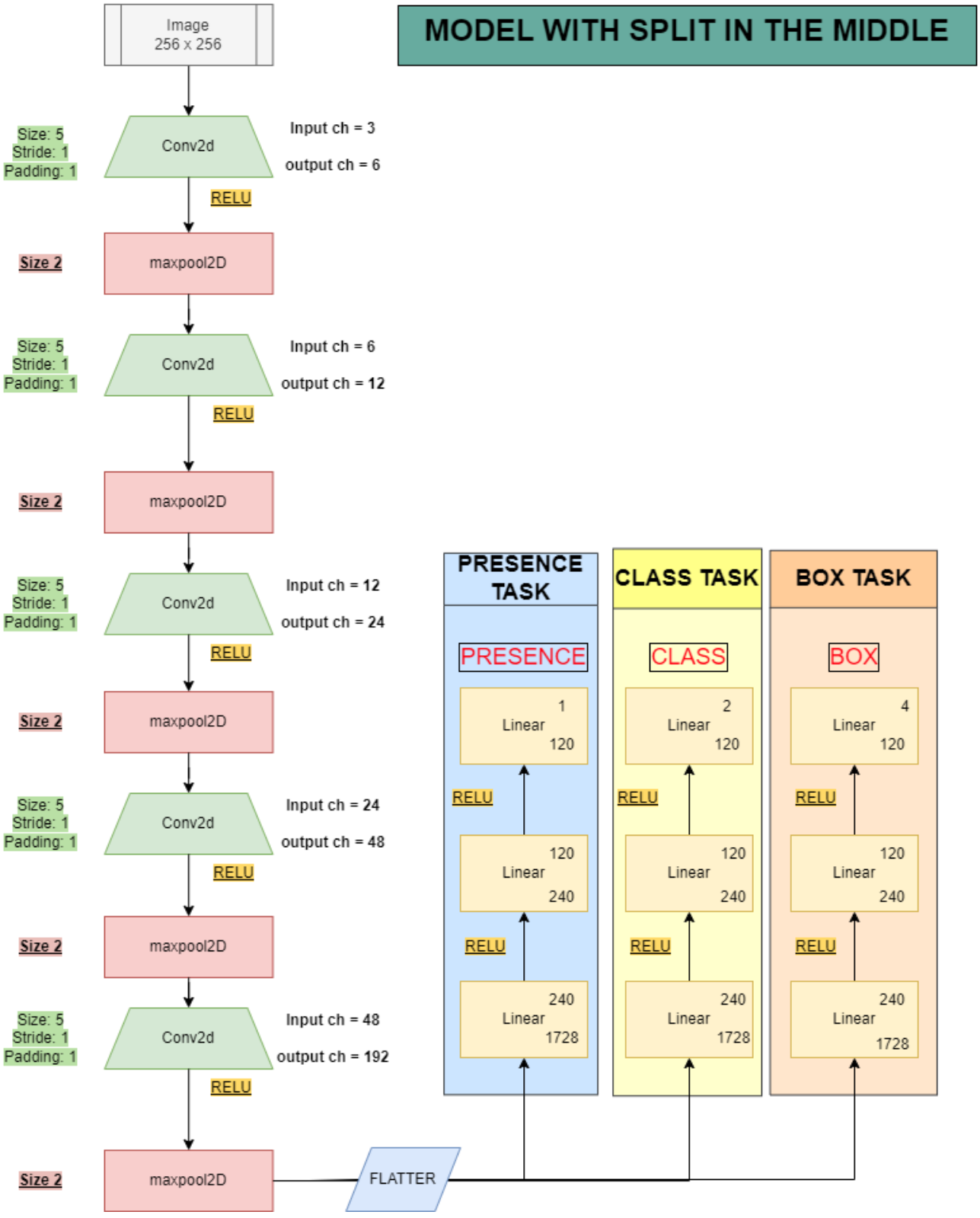


Figure 7: Model architecture diagram of "Model with split in the middle"

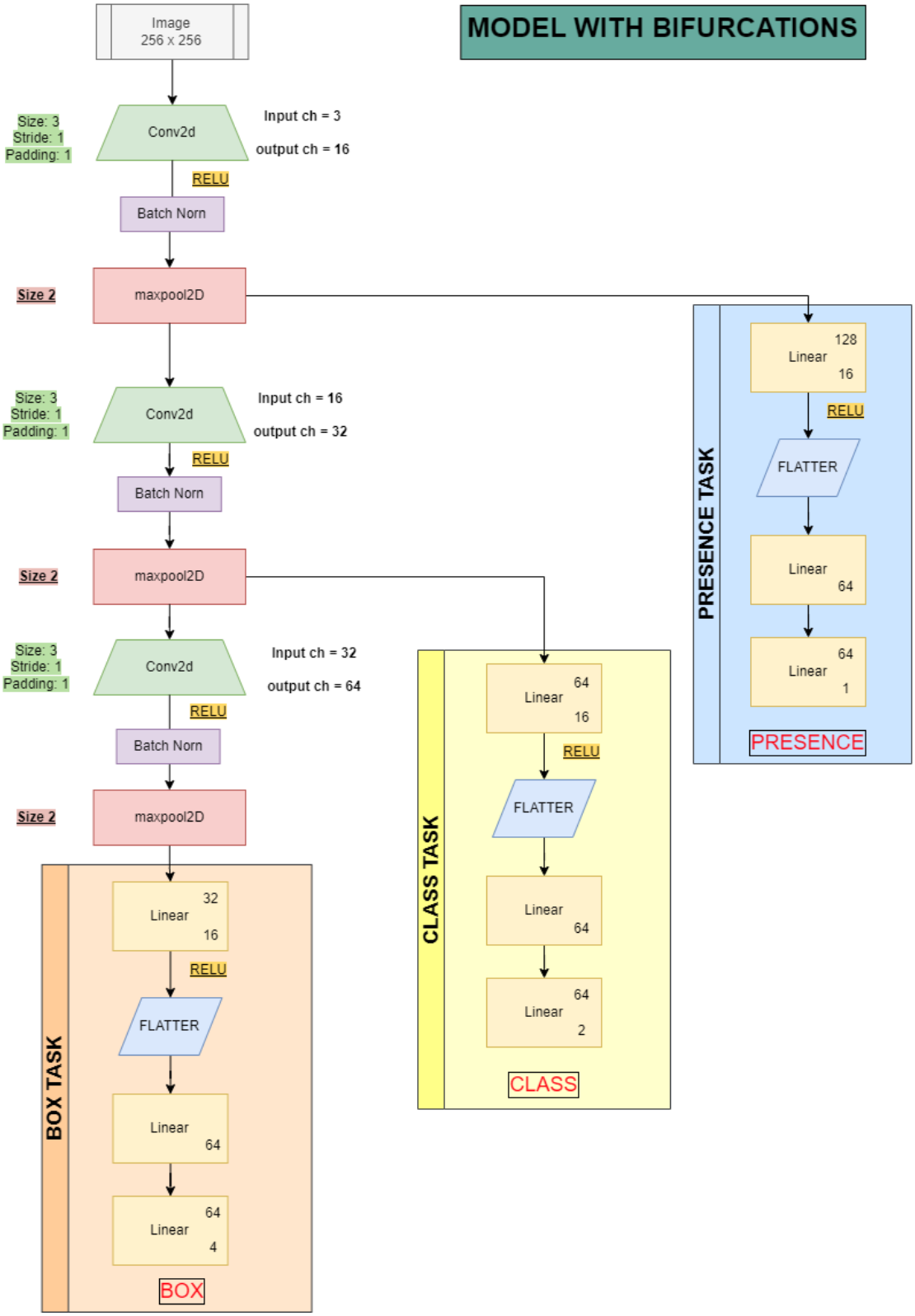


Figure 8: Model architecture diagram of "Model with bifurcations"

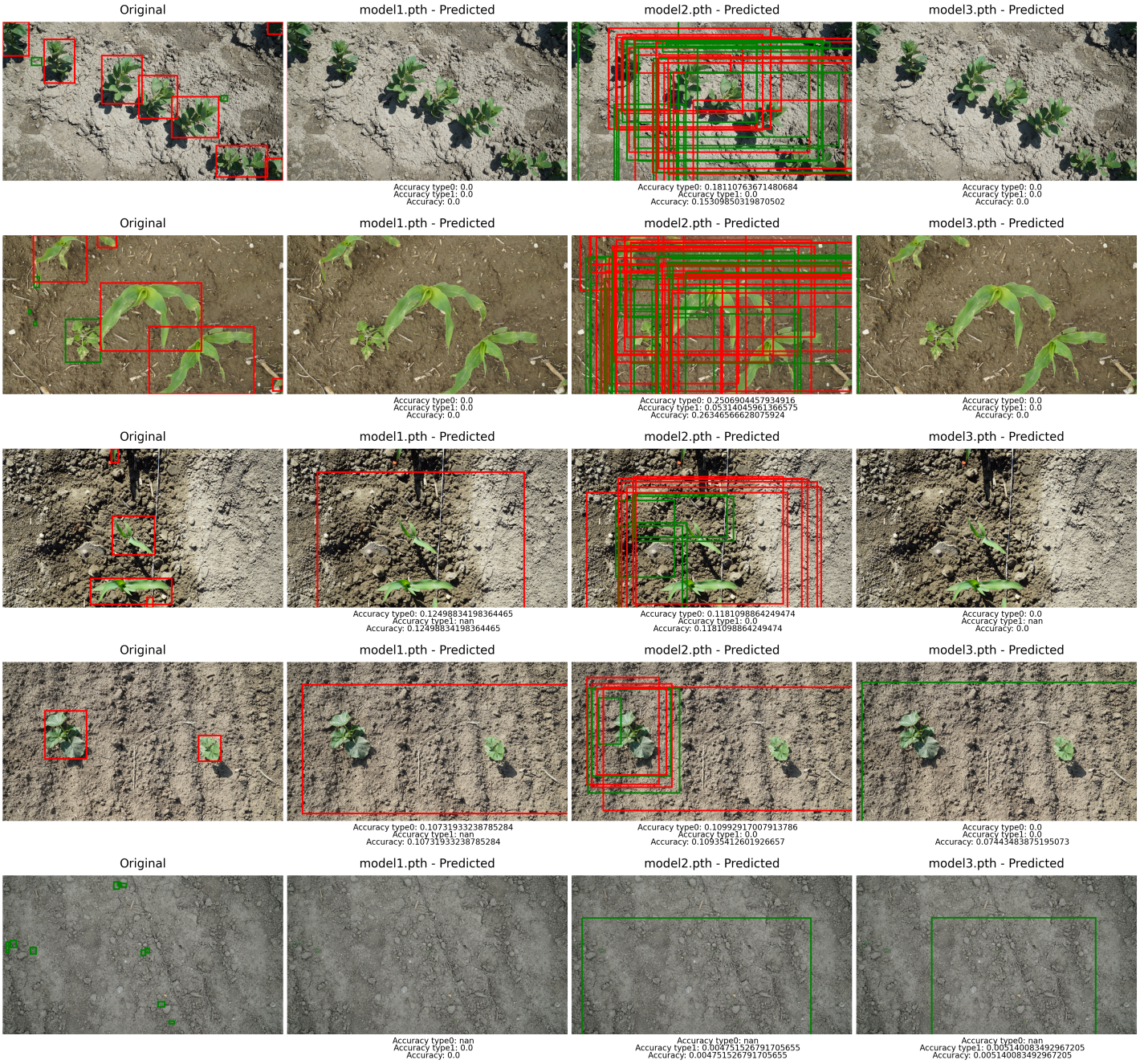


Figure 9: Some examples of image prediction taken from the test set. From left to right: original image, prediction of the first model, prediction of the second model, prediction of the third model.

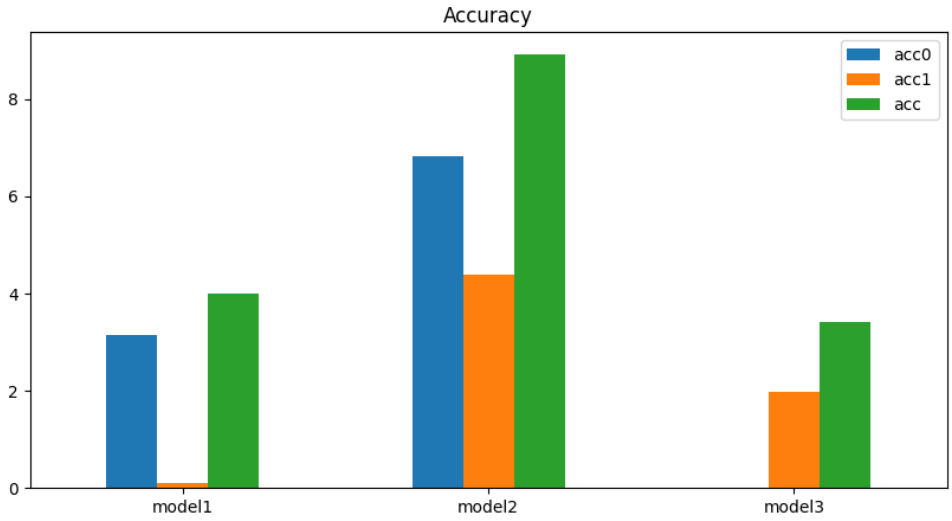


Figure 10: Architecture 1, architecture 2 and architecture 3 with the test set. “Acc0” reppresent the IOU of “type0” boxes, “Acc1” reppresent the IOU of “type1” boxes, “Acc” reppresent the IOU of general boxes without consider a fixed label.