# SDNet: Unconstrained Object Structure Detector Network for In-Field Real-Time Crop Part Location And Phenotyping

Louis Lac[1,2]
louis.lac@ims-bordeaux.fr
Jean-Pierre Da Costa[1,2]
jean-pierre.dacosta@ims-bordeaux.fr
Marc Donias[1,2]
marc.donias@ims-bordeaux.fr
Barna Keresztes[1,2]
barna.keresztes@ims-bordeaux.fr
Marine Louargant[3]
marine.louargant@ctifl.fr

[1] University of Bordeaux
IMS UMR 5218, F-33405
Talence, France
[2] CNRS
IMS UMR 5218, F-33405
Talence, France
[3] CTIFL
28 Route des Nebouts
Prigonrieux, France

**Abstract**

Most modern multi-instance pose estimation neural networks –either bottom-up or top-down variants– are built around a highly specialized and constrained architecture. They rely on the detection of a fixed set of keypoints specific to the object in order to regress the pose in images. While efficient in various contexts including human pose estimation, those architectures are not very flexible and cannot be applied to objects with a less stable structure such as plants. In this paper, we propose a neural network called SDNet which is suitable for the real-time detection of object poses with an unconstrained number of keypoints. To demonstrate its capability as well as its potential application for precision agriculture we evaluate it on a custom crop structure dataset, and we compare its performance to the state-of-the-art neural network for real-time object detection Tiny YOLOv4 on two tasks where both of them can compete: (i) multi-instance crop detection and leaf counting –which can be applied to in-field phenotyping– and (ii) stem and leaf keypoints detection and location –which can be used for real-time precision hoeing. We show that SDNet achieves a good performance on both tasks while still providing additional information via its unique structure detection ability.

## 1 Introduction

In recent years deep learning has been successfully applied to the detection of pose and keypoints of various objects in images, including human [1], hand [10] and vehicle [16] pose or facial landmark detection. One class of deep neural network usually employed for this task is the top-down architecture where whole objects are first detected then the pose is regressed

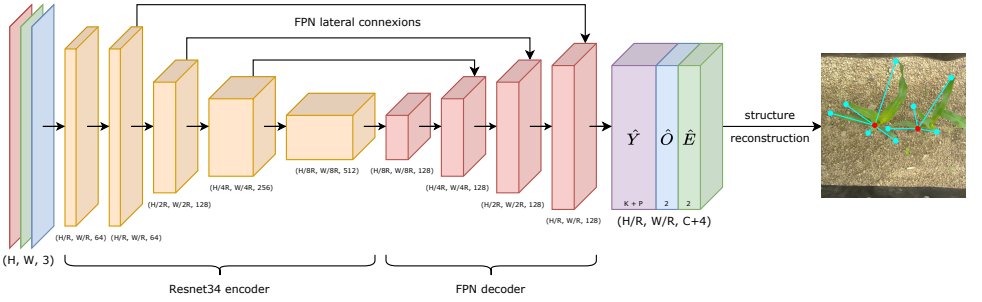


Figure 1: Scheme of the architecture of the structure detection network. The encoder is a Resnet34 [7] and the decoder a Feature Pyramid Network (FPN) [14] of three upsampling blocks of depth 128. The post-processing step is detailed in figure 3 in section 2.2

for each of them. Two-stage neural networks such as Mask R-CNN [8] are part of this category. However, to yield a higher inference speed single-stage networks are usually preferred [9, 11] because they generally achieve a better speed-accuracy trade-off. Such networks use the bottom-up approach: individual objects components (keypoints and joints) and grouping cues (embeddings) are first regressed and then a reconstruction algorithm groups the individual poses. Various designs were proposed such as Stacked Hourglass [19] (single-human pose), Part Affinity Fields [3] or CenterNet [23].

However, while human-related tasks such as human pose are well investigated by deep learning techniques, other fields such as precision agriculture are less covered [21] and lack long-term support [15]. Nonetheless, precision agriculture tasks are challenging: (i) plants and crops can have a complex and weakly defined structure, (ii) phenotype, breeds, health and soil conditions are broad and (iii) algorithms should work in real-time for an in-field application. Semantic segmentation is often chosen for such tasks [17, 18], but it is usually not real-time on embedded systems and requires a low and costly annotation process. Other works such as [5, 6, 20] propose deep learning models for plant phenotyping, but they are neither real-time nor applicable outdoor because they operate in laboratory conditions.

In this paper, we propose a bottom-up pose estimation network able to regress objects having an arbitrary number of keypoints per instance, thus applicable to weakly-defined structures such as crops. This network runs at 80 fps which meets real-time constraints of in-field precision agriculture tasks. We show that this architecture can be used to output various indicators useful for precision agriculture tasks such as phenotyping of multiple crops and precise location of crop structures. Finally, we evaluate our work against a state-of-the-art object detector on a subset of tasks both of them can handle. We first present the neural network architecture in section 2, then we introduce our database, protocols and experiments in section 3 before drawing some conclusions and perspectives in section 4. The source code of the developed architecture is available at https://github.com/laclouis5/StructureDetector.

## 2 SDNet Architecture

The developed network is a single-stage Fully Convolutional Network we call Structure Detection Network (SDNet). As illustrated in figure 1 it is based on an encoder-decoder archi-

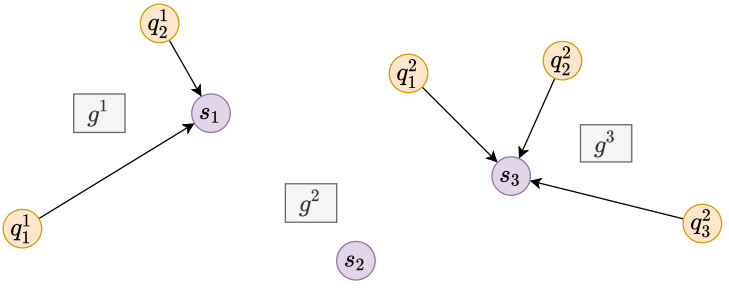


Figure 2: Illustration of 3 star-graphs for one image. The 3 graphs have all 1 anchor (center node) in purple and respectively 2, 0 and 3 parts (leaf nodes) in yellow.

tecture followed by a Structure Reconstruction Algorithm (SRA) which regresses the object structure.

The key novelty of our approach is that object keypoints are regressed independently of the object detection. Contrary to standard bottom-up pose estimators such as [23], object keypoints are not regressed as properties of an object but as independent entities that are then linked to their corresponding object via the concomitant regression of embeddings, which has the effect of removing the fixed number of keypoints constraint.

For this purpose, we choose to describe an object by a unique and object-specific keypoint that we call an *anchor* and by an unconstrained number of associated keypoints that we call *parts*. All anchors have a type, *e.g.* maize or bean stem for our application, and part keypoints as well (leaf tips for our application).

Formally, the addressed problem can be seen as the regression of a set of $N$ star graphs $G = \left\{g^i\right\}_{i=1\ldots N}$, $g^i \in \mathcal{G}_{k_i}$ from one image, where $N$ is the number of graphs in that image and $\mathcal{G}_{k_i}$ is the family of star graphs with $k_i$ leaf nodes. A star graph $g^i$ is composed of a center node (the anchor) $s^i$ and zero or more leaf nodes (the parts) $Q^i = \{q_1^i, \cdots, q_{k_i}^i\}$. We call $S = \{s^1, \cdots, s^N\}$ the set of anchors and $Q = \bigcup_{i=1}^{N} Q^i$ the set of parts in the following. Figure 2 illustrates these notations.

## 2.1 Encoder-Decoder

SDNet takes as input an image $I \in \mathbb{R}^{W \times H \times 3}$ where $W$ and $H$ are the image width and height and produces a keypoint heatmap $\hat{Y} \in [0,1]^{\frac{W}{R} \times \frac{H}{R} \times C}$ where $C$ represents the number of heatmaps (one per keypoint type) that can be regressed and $R$ is the downscaling ratio of the network, which has an impact on the speed-accuracy trade-off. The $C$ keypoint types are composed of $K$ anchor keypoint types and $P$ part keypoint types. Heatmaps are illustrated in figure 3b and 3c where opacity is proportional to the confidence in the detection of a keypoint. The network also predicts a local offset vector field $\hat{O} \in \mathbb{R}^{\frac{W}{R} \times \frac{H}{R} \times 2}$ to retrieve the discretization error caused by the downscaling ratio, and embeddings $\hat{E} \in \mathbb{R}^{\frac{W}{R} \times \frac{H}{R} \times 2}$ which act as a grouping cue used by the SRA to compute the object pose.

We chose a lightweight architecture matching the real-time constraints and our relatively small dataset. The encoder is a Resnet34 [7] backbone pre-trained on ImageNet [4] and the decoder is a Feature Pyramid Network (FPN) [14] made of three upsampling blocks of depth 128. For this specific encoder-decoder architecture $R = 4$. The head of the network is a one-by-one convolution with $C+4$ filters. The architecture is detailed in figure 1.

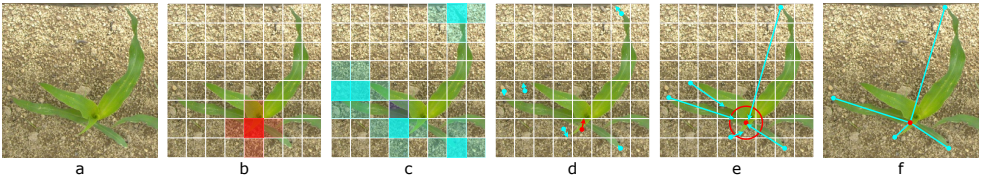


Figure 3: Simplified illustration of the structure reconstruction process. The original image is displayed in *a*. *b* and *c* are respectively the maize stem anchor point heatmap and the part keypoint heatmap. *d* presents the discretization recovery with the offsets (arrows). *e* illustrates the part association with the embeddings (blue arrows) and *f* shows the final star graph regressed. $t_d$ is represented by the red circle around the stem anchor point.

## 2.2 Structure Reconstruction Algorithm

The first step consists of a spatial filtering of the keypoint heatmap $\hat{Y}$ with a 3x3 max-pooling operation to extract the local peak values. Then, all detections with a confidence lower than a given threshold $t_c$ are discarded. This results in a set $\hat{S}$ of $n_s$ predicted center nodes (the anchors) and a set $\hat{Q}$ of $n_q$ predicted leaf nodes (the parts). The filtering process is illustrated in figures 3b and 3c.

At this point, coordinates are integers and need to be refined to recover the lost spatial precision. This is performed by adding the offset regressed at the node location: $\hat{S}' = \left\{\hat{s}^i + \hat{O}_{\hat{s}^i},\ \hat{s}^i \in \hat{S}\right\}$ and $\hat{Q}' = \left\{\hat{q}^j + \hat{O}_{\hat{s}^j},\ \hat{q}^j \in \hat{Q}\right\}$. This process is illustrated in figure 3d.

The second step consists of the association of parts and anchors. Embeddings are vectors learned at leaf node locations which act as the grouping cue for the association. An embedding is a prediction of the offset from the part location to the anchor location (blue arrows in figure 3e). The embeddings are used to compute the estimated anchor positions from the parts: $\hat{S}^* = \{\hat{s}^*_j\}_{j=1\ldots n_q} = \left\{\hat{q}_j + \hat{E}_{\hat{q}_j},\ \hat{q}_j \in \hat{Q}'\right\}$. Then, for each anchor $s^i \in \hat{S}'$ the associated part positions (*i.e.* the leaf nodes of the star graph with the center node $s^i$) are calculated as:

$$\hat{Q}^i = \left\{ q_j \in \hat{Q}' \ / \ \underset{\hat{s}^k \in \hat{S}'}{\operatorname{argmin}} \left\| \hat{s}^k - \hat{s}^*_j \right\|_2 = \hat{s}^i \right\} \tag{1}$$

where $\|\cdot\|_2$ is the L2 norm. In other words, predicted part keypoints are associated with the closest anchor point according to their embedding value. Finally, part keypoints with an estimated anchor location not close enough to the associated anchor are filtered out according to a distance threshold $t_d$, and the star graph is built from the aggregation of the anchor and the filtered parts:

$$\hat{g}^i = s^i \cup \left\{ \hat{q}_j \in \hat{Q}^i \ / \ \left\| \hat{s}_j - \hat{s}^*_j \right\|_2 < t_d \right\} \tag{2}$$

This process results in the predicted set of graphs $\hat{G}$. This final step is illustrated in figures 3e and 3f.

## 2.3 Loss Function

The encoder-decoder architecture is trained with the loss function described in equation 3, where $\lambda_o$ and $\lambda_e$ are two loss scaling weights.

$$L = L_{hm} + \lambda_o L_{off} + \lambda_e L_{emb} \tag{3}$$

$L_{hm}$ and $L_{off}$ are respectively the heatmap and offset loss functions. Their computation, identical to [23], requires the ground truth heatmap $Y$ and offsets $O$. $Y$ is deduced from the annotated images by applying a Gaussian filter with standard deviation $\sigma$ and kernel size $3\sigma$. $O$ is calculated by discretization of the annotations.

$L_{hm}$ is a L2 loss while $L_{off}$ is a L1 loss similar to the embedding loss. The embedding loss function $L_{emb}$ is a L1 loss computed at the ground truth part locations only and is detailed in equation 4 where $S$ is the set of ground truth anchors, $P_s$ the set of parts associated with anchor $s$, $\hat{E}_{\tilde{p}}$ the predicted embedding at the discretized part location $\tilde{p} = \left\lfloor \frac{p}{R} \right\rfloor$ and $M'_p$ the total number of ground truth parts.

$$L_{emb} = \frac{1}{M'_p} \sum_{s \in S} \sum_{p \in P_s} \left| (p - s) - \hat{E}_{\tilde{p}} \right| \tag{4}$$

# 3 Experimental Setup and Results

While evaluating the performance of standard pose estimation algorithms is straightforward, this is not the case for unconstrained pose estimation as there is no one-to-one correspondence between detected and ground-truth keypoints. Moreover, to our knowledge there is no established database, challenge or competing solution to benchmark our algorithms against. As a consequence we choose to evaluate our work on a subset of tasks both our network and other work can compete. We selected a recent detector named Tiny YOLOv4 [2] because it is both state-of-the-art on object detection performance and real-time (we consider 30 fps to be the real-time limit). Moreover, it is suitable for embedded systems [22], which is a requirement for precision agriculture applications.

We evaluated the two networks on two tasks modelling real use-cases of precision agriculture: the classification task and the keypoint detection task. In the first one, we evaluate the ability to detect multiple crops in an image and to regress the correct crop bread (maize or bean) and number of leaves. This ability can be used as an indicator for crop assessment in fields and phenotyping. In the second one, we evaluate the ability to detect the precise spatial location of crop organs (maize stem, bean stem and leaf tip). This ability can be used for precision hoeing tools operating in the intra-row, for instance tools that hoe every ground location excepted the stem locations, as this would destroy the crops.

In the following we first describe the dataset, the training settings and the evaluation metrics used, then we detail grid-searches on main SDNet hyperparameters, and finally we comment the results on both tasks.

## 3.1 Dataset and Annotations

We gathered a dataset[1] of images taken in vegetable fields in the same conditions as if the acquisition system was used for real-time automatic hoeing, *i.e.* (i) crops may not be spaced predictably and many can appear on the same image, (ii) self-occlusions and weed occlusions can occur, (iii) neither soil conditions nor weed infestation are controlled and (iv) crops are at early but variable stages where weeding is crucial but can be at different growth stages. We chose two crops at an early stage of development (two to five weeks) for the study: maize and bean.

[1] The publication of this dataset will be the subject of a future article.

It contains a total of 874 images on which we annotated every crop using a custom annotation tool. Each crop annotation is composed of a rectangular bounding box and several keypoints: the stem entry point in the ground and the leaf tips. The stem annotation corresponds to the anchor point and the leaf tip to the part keypoint, thus for our network $K = 2$ (bean stem and maize stem) and $P = 1$ (leaf tip which we choose to be class-agnostic). We annotated 1302 bean crops and stems and 2834 associated leaf tips as well as 1067 maize crops and stems and 2747 leaf tips. All images are taken vertically at the same distance from the ground, thus the image height is constant and corresponds to 38 cm on the ground.

## 3.2 Training and Inference

The dataset presented in section 3.1 is used for training and validation of both networks. 80 % is used for training and the remaining 20 % is used for validation. The images are normalized in the same way as ImageNet [4] and resized to a 512 by 512 resolution. Training is performed on a computer running Ubuntu 18 LTS equipped with a GTX 2080 SUPER 8 GB and an Intel Core i7-7700 4 cores at 3.6 GHz 32 GB.

For SDNet, the data augmentation consists of random color jitters (brightness and contrast $\pm 25$ %, saturation $\pm 15$ %, hue $\pm 5$ %), horizontal and vertical flips with a probability of 50 % as well as a random rescale of a factor in [0.75, 1.25]. The loss weights are empirically set to $\lambda_o = \lambda_e = 0.001$ in order to scale the three losses in the same magnitude range during training. The learning rate is set to 0.001 and a scheduler divides its value by 10 every 33 epochs for a total training time of 100 epochs. The batch size is set to 8 and the Adam optimizer [12] is used. Two experiments reported in section 3.4 allowed to tune the Gaussian kernel size to $3\sigma = 5\%$ and to find the best values for the hyperparameters $t_{conf}$ = 40 % and $t_{dist}$ = 10 % (around 4 cm). Percentages are expressed regarding the shortest image side length. Training takes around 1 hour and inference runs at 80 fps, including image pre-processing and the structure reconstruction post-processing.

TY4 training differs depending on the evaluated task and the exact settings are presented in section 3.3. We used the default training settings of the TY4 framework, and we chose the same image size as for SDNet training. Training takes one hour and inference runs at 300 fps with hardware optimizations turned on (CUDNN, Tensor Cores, half precision, etc.). Note that SDNet was not optimized to take advantage of these optimizations.

## 3.3 Evaluation Metrics

We chose to evaluate the performance of the classification and the keypoints detection with the Recall, Precision and F1-score. Both networks are multi-instance detectors, so we chose to use the COCO [13] assignment algorithm to map detections to ground-truths. The Recall, Precision and F1-score are computed at the best confidence threshold found via a grid-search for both networks (see section 3.4 for details on grid-searches of SDNet hyperparameters).

We also provide two metrics to quantify the errors for each evaluated task: $MAE_{count}$ which is the Mean Absolute Error (MAE) of leaf counts and $MAE_{loc}$ which is the MAE of keypoint localizations.

Concerning the classification task, we trained TY4 to detect the bounding box of crops and a label depicting the crop bread (maize or bean) and the number of leaves from 0 to 8 (the maximum number of leaves in our dataset). A detection is a True Positive (TP) if the label is correct and the bounding box Intersection over Union is greater than 50 %. For SDNet, a detection is a TP if the star graph anchor label is correct (maize or bean), if its location is


Figure 4: *a*: Predictions of TY4 for the leaf counting task, *b*: predictions of SDNet for the keypoints detection task, *c* predictions of the crop structure by SDNet, and *d* common prediction errors of SDNet.

in a 2 cm[2] radius of the ground truth anchor and if the number of leaves is the same as the ground truth. Some samples of predictions are presented in figure 4a.

Concerning the keypoint detection tasks, TY4 is trained to detect the bounding box of crop keypoints (maize stem, bean stem and leaf end). The bounding box is centered on the crop keypoint, and is of square shape with a side length found by grid search equal to 3 cm (in the field ground referential). A prediction is a TP if the box center is in a 2 cm radius of the ground truth and the label is identical to the ground truth. For SDNet, all anchor et part predictions are used and are considered TP under the same rules as for the classification task. Some samples of predictions are presented in figure 4b.

Our network has the additional ability to detect the star graph of crops, some samples are presented in figure 4c.

## 3.4 Hyperparameters Search

We conducted two experiments on three hyperparameters that SDNet depends on: the confidence threshold $t_c$, the decoder distance threshold $t_d$, and the Gaussian kernel size used to compute the ground truth keypoint heatmaps. We also performed three other grid searches for TY4 not presented here which allowed finding the best confidence threshold $t_c = 50$ % for the classification task, $t_c = 25$ % for the keypoint detection task and a bounding box side length of 3 cm for training it on the classification task.

The first experiment presented in figure 5 is a grid-search on $t_c$ and $t_d$. Only a limited number of values are reported here for a matter of concision and $t_d$ is expressed as a fraction of the smaller image side length. The classification F1-score presented in section 3.5 is used to compare the accuracy of each couple of values. It is shown that the influence of $t_c$ is high compared to the one of $t_d$ and that its optimal value is attained around 40 % for all $t_d$ thresholds. Below 15 % and above 65 % the F1-score drops rapidly toward 0 % (not shown on the graph). This finding is expected as a low threshold let pass a lot of false alarms while a high threshold yields many missed detections.

[2]This value is suitable for the precision hoeing task targeted.

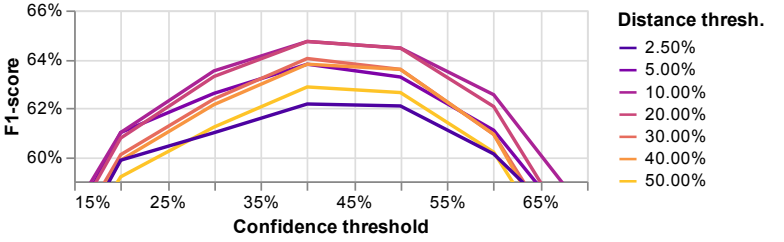


Figure 5: Grid-search on the $t_c$ and $t_d$ hyperparameters using the classification F1-score as the comparison metric.

| Kernel size | 3 % | 5 % | 10 % | 40 % |
|---|---|---|---|---|
| F1-score | 31.68 % | 64.73 % | 63.75 % | 34.89 % |

Table 1: Influence of the Gaussian kernel size used for ground truth keypoint heatmaps computation on the classification F1-score.

In contrast, $t_d$ has a lower impact in the F1-score. The figure shows that its optimal value is attained between 10 % and 20 %. Bellow this value the F1-score slightly decreases for all $t_c$ values, and only begins to drop significantly bellow 2.5 %. Above $t_d = 20$ % the F1-score also starts to decrease slightly, but even very high values such as 50 % do not have a high negative impact on the metric which stays around 63 % for the best $t_c$ threshold. The optimal couple of values is attained for $t_c = 40$ % and $t_d = 10$ % (4 cm).

The second experiment focuses on the influence of the Gaussian kernel size used to create the keypoint training heatmap. We defined the kernel size as its $3\sigma$ width, expressed as a fraction of the shortest image side length. We only tested four values spanning a large range as it requires retraining the network each time. As for the grid-search, the comparison metric we used is the classification F1-score, computed at the optimal $t_c$ and $t_d$ values.

Table 1 shows that good F1-scores are obtained for kernel sizes of 5 % and 10 %. However, the performance is significantly lower when using a small value of 3 % and a high value of 40 %. This experiment roughly indicates the range of kernel sizes which are suitable for a correct training. We chose a value of 5 % (2 cm) because it gives the best performance and because it matches the required precision for the precision hoeing task targeted.

## 3.5 Classification Task

Table 2 shows that a better F1-score is obtained for SDNet, reaching 64.73 %, 2.18 % better than TY4. However, it can be noted that TY4 has a better precision than SDNet and a lower recall, thus, TY4 outputs fewer false positives but misses more detections. The $MAE_{count}$ is

| Network | Recall | Precision | F1-score | $MAE_{count}$ |
|---|---|---|---|---|
| Tiny YOLO v4 | 56.24 % | 70.45 % | 62.55 % | 0.33 ± 0.03 leaves |
| SDNet (ours) | 63.04 % | 66.51 % | 64.73 % | 0.37 ± 0.03 leaves |

Table 2: Precision, Recall and F1-score of the two networks for the classification task. The Mean Absolute Error in count ($MAE_{count}$) with its standard error is additionally shown.

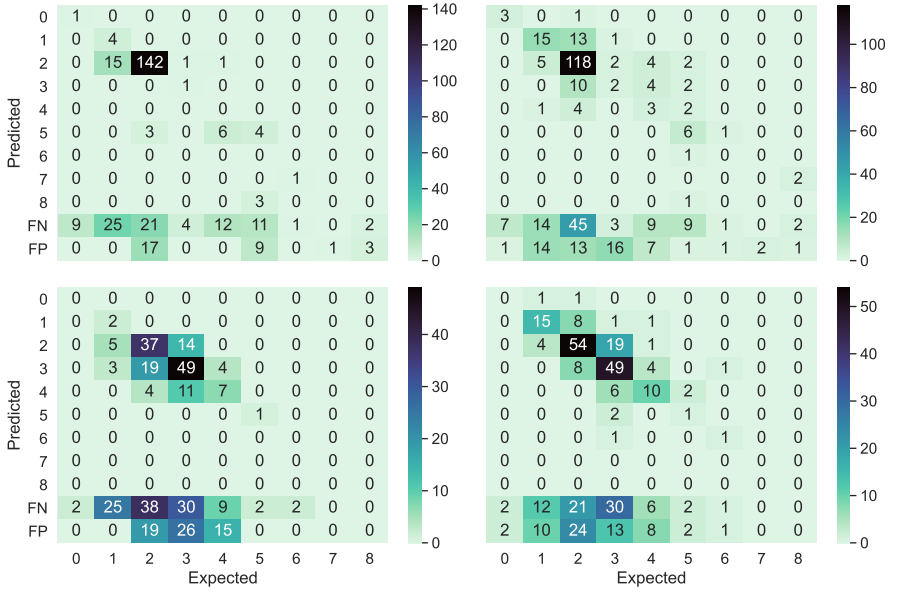


Figure 6: Confusion matrix for the leaf counting task. TY4 (left) and SDNet (right), bean (top) and maize crops (bottom). Axes refer to the leaf number, the FN row refers to false negatives and the FP row refers to false positives.

0.33 leaves for TY4 and 0.37 leaves for SDNet. Those errors are both small, with a slight advantage for TY4 though not significant considering the associated standard errors. Figure 6 investigates in more details the repartition or errors among crop types and leaf counts.

The confusion matrix presented in figure 6 shows that SDNet outputs more false positives (FP) than TY4, more specifically for the bean crop (+26 FP). However, SDNet yields fewer false negatives (FN): -34 FN for maize crops and -29 FN for bean crops. Concerning the classification, errors are more frequent for TY4 on the maize. For instance, it classifies 25 crops has having 3 or 4 leaves instead of 2 while SDNet only misclassified 16 of them incorrectly. On bean crops SDNet struggles on beans with 2 leaves, misclassifying 27 compared to 3 for TY4, but the opposite is true for beans with 1 leaf where TY4 misclassified 15 crops compared to 6 for SDNet.

These results show that both networks achieve a comparable classification accuracy, and that the better overall performance of SDNet highlighted in table 2 is mainly due to its lower false negative rate.

## 3.6 Keypoint Detection and Location Task

Table 3 shows that SDNet achieves better F1-scores for bean stems (+3.31 %), maize stems (+1.52 %) and leaf ends (+2.63 %), and in total for all classes SDNet achieves +2.54 % better F1-score than TY4. The localization errors $MAE_{loc}$ are comparable (3.22 mm for TY4 vs. 3.26 mm for SDNet), and the standard error shows that the difference in $MAE_{loc}$ is not significant. This level of error is sufficiently good for precision agriculture tasks.

However, bean stems are more difficult to detect than maize stems for both networks

| Network | Keypoint | Recall | Precision | F1-score | $MAE_{loc}$ |
|---|---|---|---|---|---|
| Tiny YOLO v4 | Bean | 83.54 % | 87.61 % | 85.53 % | 4.71 ± 0.26 mm |
| | Maize | 94.61 % | 91.47 % | 93.01 % | 3.61 ± 0.19 mm |
| | Leaf | 83.82 % | 88.69 % | 86.19 % | 2.80 ± 0.07 mm |
| | Total | 85.28 % | 88.94 % | 87.07 % | 3.22 ± 0.07 mm |
| SDNet (ours) | Bean | 85.65 % | 92.27 % | 88.84 % | 5.07 ± 0.28 mm |
| | Maize | 93.14 % | 95.96 % | 94.53 % | 3.67 ± 0.19 mm |
| | Leaf | 86.08 % | 91.75 % | 88.82 % | 2.75 ± 0.08 mm |
| | Total | 87.00 % | 92.44 % | 89.63 % | 3.26 ± 0.08 mm |

Table 3: Precision, Recall, F1-score and Mean Absolute Error ($MAE_{loc}$) in localization with its standard error for SDNet and TY4 for the stem and leaf detection and location task.

(-5.45 % for SDNet and -7.48 % for TY4) and $MAE_{loc}$ is greater (respectively +1.40 mm and +1.10 mm). We believe that the higher density and overlap of bean crops compared to maize crops is responsible for this lower accuracy location error. Leaf ends are also harder to detect for both networks but the $MAE_{loc}$ is the lowest one of the three keypoints. This better location accuracy is probably due to the fact that stems are almost never fully visible due to occlusions by leaves, which is not the case of leaves ends.

Finally, some common SDNet failures are presented in figure 4d. They include missed part keypoints and wrong part associations. The first and second images show examples of wrong part associations on bean and maize. The high crop proximity and overlap causes some anchors to be masked or difficult to see, which seem to be the cause of some leaves being associated with the wrong anchor point. The last image presents an example where leaf parts are not detected correctly. The unusual orientation of crops and the dusty leaves may be the source of confusion in this case.

We did not make experiments on more mature crops as our database does not contain such samples, but we hypothesize that this would produce more crop overlap, thus leading to more wrong part associations than usual. However, hoeing is performed at an early development stage when the weed competition is high, thus in practice performance on more mature crops is less crucial.

# 4 Conclusions

In this paper, we proposed an unconstrained object structure detector which improves the flexibility of pose estimation networks in contexts where object pose is not fixed and objects contain an unbounded number of keypoints, such as plants. We designed a real-time detector called SDNet and demonstrated its effectiveness in precision agriculture tasks such as in-field crop phenotyping and mechanical precision hoeing. We compared its performance with the state-of-the-art real-time object detector Tiny YOLOv4 on two tasks where both of them can be compared: (i) crop detection and leaf counting and (ii) stem and leaf keypoint location. We show that SDNet introduces some performance gains, achieving respectively +2.18 % and +2.54 % for the F1-score compared to Tiny YOLO v4.

Future work will investigate the behavior on objects with a more complex structure such as more mature crops. A special attention will also be given to the mutual analysis of the network location accuracy and of the ground truth uncertainty due to annotation variability.

# References

[1] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2D Human Pose Estimation: New Benchmark and State of the Art Analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3686–3693, 2014.

[2] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv:2004.10934 [cs, eess]*, April 2020.

[3] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime Multi-person 2D Pose Estimation Using Part Affinity Fields. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1302–1310, Honolulu, HI, July 2017. IEEE. ISBN 978-1-5386-0457-1. doi: 10.1109/CVPR.2017.143.

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, Miami, FL, June 2009. IEEE. ISBN 978-1-4244-3992-8. doi: 10.1109/CVPR.2009.5206848.

[5] Andrei Dobrescu, Mario Valerio Giuffrida, and Sotirios A. Tsaftaris. Doing More With Less: A Multitask Deep Learning Approach in Plant Phenotyping. *Frontiers in Plant Science*, 11:141, February 2020. ISSN 1664-462X. doi: 10.3389/fpls.2020.00141.

[6] Mario Valerio Giuffrida, Massimo Minervini, and Sotirios Tsaftaris. Learning to Count Leaves in Rosette Plants. In *Procedings of the Proceedings of the Computer Vision Problems in Plant Phenotyping Workshop 2015*, pages 1.1–1.13, Swansea, 2015. British Machine Vision Association. ISBN 978-1-901725-55-1. doi: 10.5244/C.29.CVPPP.1.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, Las Vegas, NV, USA, June 2016. IEEE. ISBN 978-1-4673-8851-1. doi: 10.1109/CVPR.2016.90.

[8] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, October 2017. doi: 10.1109/ICCV.2017.322.

[9] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, and Kevin Murphy. Speed/Accuracy Trade-Offs for Modern Convolutional Object Detectors. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3296–3297, July 2017. doi: 10.1109/CVPR.2017.351.

[10] Weiting Huang, Pengfei Ren, Jingyu Wang, Qi Qi, and Haifeng Sun. AWR: Adaptive Weighting Regression for 3D Hand Pose Estimation. *arXiv:2007.09590 [cs, eess]*, July 2020.

[11] Licheng Jiao, Fan Zhang, Fang Liu, Shuyuan Yang, Lingling Li, Zhixi Feng, and Rong Qu. A Survey of Deep Learning-Based Object Detection. *IEEE Access*, 7:128837–128868, 2019. ISSN 2169-3536. doi: 10.1109/ACCESS.2019.2939201.

[12] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*, January 2017.

[13] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, volume 8693, pages 740–755. Springer International Publishing, Cham, 2014. ISBN 978-3-319-10601-4 978-3-319-10602-1. doi: 10.1007/978-3-319-10602-1_48.

[14] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature Pyramid Networks for Object Detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 936–944, July 2017. doi: 10.1109/CVPR.2017.106.

[15] Guillaume Lobet. Image Analysis in Plant Sciences: Publish Then Perish. *Trends in Plant Science*, 22(7):559–566, July 2017. ISSN 13601385. doi: 10.1016/j.tplants.2017.05.002.

[16] Javier Garcia Lopez, Antonio Agudo, and Francesc Moreno-Noguer. Vehicle pose estimation via regression of semantic points of interest. In *2019 11th International Symposium on Image and Signal Processing and Analysis (ISPA)*, pages 209–214, Dubrovnik, Croatia, September 2019. IEEE. ISBN 978-1-72813-140-5. doi: 10.1109/ISPA.2019.8868508.

[17] Philipp Lottes, Jens Behley, Andres Milioto, and Cyrill Stachniss. Fully Convolutional Networks With Sequential Information for Robust Crop and Weed Detection in Precision Farming. *IEEE Robotics and Automation Letters*, 3(4):2870–2877, October 2018. ISSN 2377-3766. doi: 10.1109/LRA.2018.2846289.

[18] Andres Milioto, Philipp Lottes, and Cyrill Stachniss. Real-Time Semantic Segmentation of Crop and Weed for Precision Agriculture Robots Leveraging Background Knowledge in CNNs. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2229–2235, May 2018. doi: 10.1109/ICRA.2018.8460962.

[19] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked Hourglass Networks for Human Pose Estimation. *arXiv:1603.06937 [cs]*, March 2016.

[20] M. P. Pound, J. A. Atkinson, D. M. Wells, T. P. Pridmore, and A. P. French. Deep Learning for Multi-task Plant Phenotyping. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 2055–2063, October 2017. doi: 10.1109/ICCVW.2017.241.

[21] Sotirios A. Tsaftaris, Massimo Minervini, and Hanno Scharr. Machine Learning for Plant Phenotyping Needs Image Processing. *Trends in Plant Science*, 21(12):989–991, December 2016. ISSN 13601385. doi: 10.1016/j.tplants.2016.10.002.

[22] M. Verucchi, G. Brilli, D. Sapienza, M. Verasani, M. Arena, F. Gatti, A. Capotondi, R. Cavicchioli, M. Bertogna, and M. Solieri. A Systematic Assessment of Embedded Neural Networks for Object Detection. In *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, volume 1, pages 937–944, September 2020. doi: 10.1109/ETFA46521.2020.9212130.

[23] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as Points. *arXiv:1904.07850 [cs]*, April 2019.