Feature Based Plant Seedlings Classification

Dmitri Jakovlev, Julia Kamaletdinova, and Georgy Shevlyakov

Peter the Great Saint-Petersburg Polytechnic University, Department of Applied Mathematics, Russia

Abstract. The abstract should briefly summarize the contents of the paper in 150–250 words.

Keywords: Image processing · Feature extraction · Computer vision.

1 Introduction

The demand for agricultural products is increasing day by day, as the population of the Earth is growing. Even though people are working on plant classification algorithms, approaches are still not as robust as desired. A significant part of work has still been done by people. The question arises of the efficiency with which human resources are used. We will use exhaustible natural resources wisely and increase harvests if we automatize quality assurance, which objectives are to detect and distinguish weeds among the variety of crop seedlings.

All this naturally leads to the idea of automation of the classification process with the help of machine learning algorithms. From recent experience, neural networks are well suited for image processing, but we have to pay for it with computational costs. On the other hand, we could use less costly algorithms, but they require finer tuning to achieve a comparable result.

A feature-based method is a popular approach in the image classification task, there are many advantages and results with a good performance in many related pieces of research, as in [6], [8]. The main idea of the method is to process the only relevant information on an image, and to reduce the feauture space dimension as much as possible. Likewise, dimensionality reduction is the way of solving the curse of dimensionality problem, introduced by Richard Bellman in 1957 [1]. The implementation of th suggested method is described in

The goal is to implement segmentation and classification of a specific type of data set for low time and computational complexity. In this paper, we will research binary classifiers' capabilities on the dataset [4] consisting of images of 12 species and containing the most common weed species in Danish agriculture.

2 Materials and methods

2.1 Data

The dataset is a part of the database that has been recorded at Aarhus University Flakkebjerg Research station in a collaboration between the University of Southern Denmark and Aarhus University. Images are available to researches at https://vision.eng.au.dk/plant-seedlings-dataset/. The specific of the dataset is that recorded plants are in different growth stages since detecting weed in its early stage is the thing makes the task problematic.

The dataset contains 960 unique plant images of 12 species. The sizes of plant classes are not balanced among themselves - they range from 221 to 654 labeled samples for each class. Original images are cropped by plant boundaries, but their resolutions vary from 50x50px to 2000x2000px. Also, images have a different background - some of them on the ground, other on the marked paper.

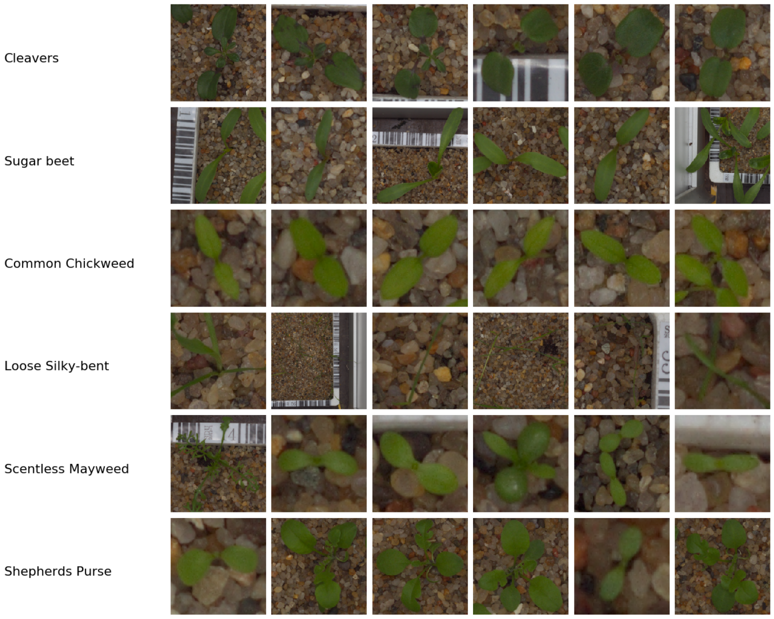


Fig. 1. Data overview

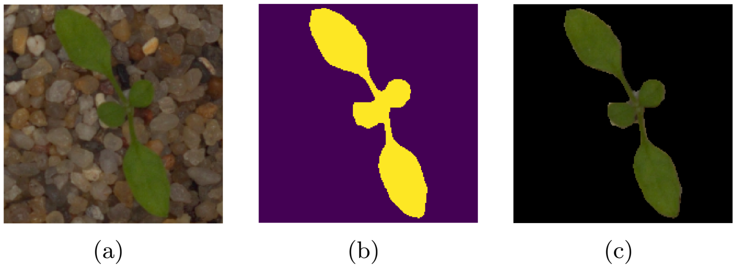
2.2 Data preprocessing

Resolution reducing

We reduce the resolution of all images to the same resolution 200x200px using bilinear interpolation. The main idea of bilinear interpolation that the new image pixel is the weighted sum of neighboring pixels of the original image. It helps to decrease computational complexity and build normalized features.

Segmentation

The objects of our study are plants, and they are painted green. Therefore, we can create a mask that filters the range of green channel and ignores the other pixels. For these purposes, the HSV (Hue Saturation Value) color model is a suitable representation. In the BGR format, the value of each component depends on the amount of light hitting the object. HSV allows us to distinguish between image color and brightness. Hue, saturation, and value let us set the lower and upper borders of the shades of a certain color, in this case — green. Then we merely mark the pixels in the green range and get a color mask (Fig. 2(b)). Now we apply the operation of logical multiplication to the original image, assign the value of the background pixels to a black color value, and get a segmented plant.


(a)

(b)

(c)

Fig. 2. (a) Source image; (b) Mask; (c) Segmented image

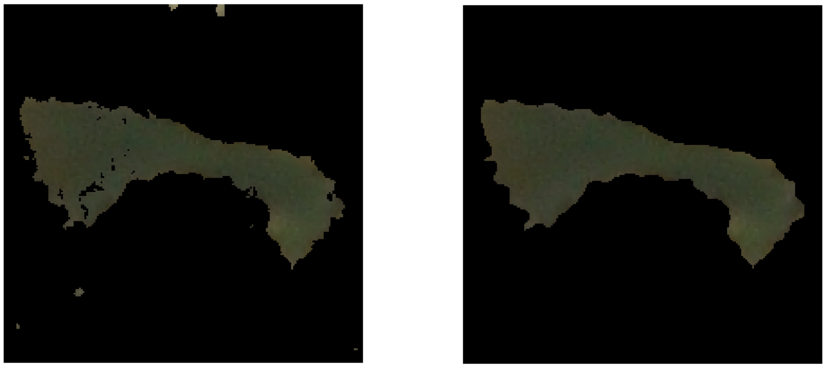
Noise removal

Segmentation does not always work well (Fig. 2(c)). Small areas of the background may fall into the range of green values which distorts the binary mask (Fig. 3(a)).

Such drawbacks can be eliminated with the morphological operations — nonlinear transformations associated with the shape and structure of image. The morphology is used to study the interaction of an image with a specific structural element — the kernel. The kernel iterates over the entire image and compares with the neighborhood of the pixels after which we apply morphological operations [3].

To improve segmentation, we use the operation of morphological closure — a combination of dilatation and erosion operations.

Then we apply the morphological closure operation to the image (Fig. 4(a)) by selecting an elliptical core of 6x6px size and delete the remaining objects with an area of less than 160px. The plant on the image (Fig. 4(b)) has no cavities, and the background is cleared of non-plant elements. But morphological closure does not always improve the segmentation result. Estimate the result of working on an image of the class Loose silky-bent. It is noticeable (Fig. 4) that the cavities corresponding to the background were restored, which does not correspond to

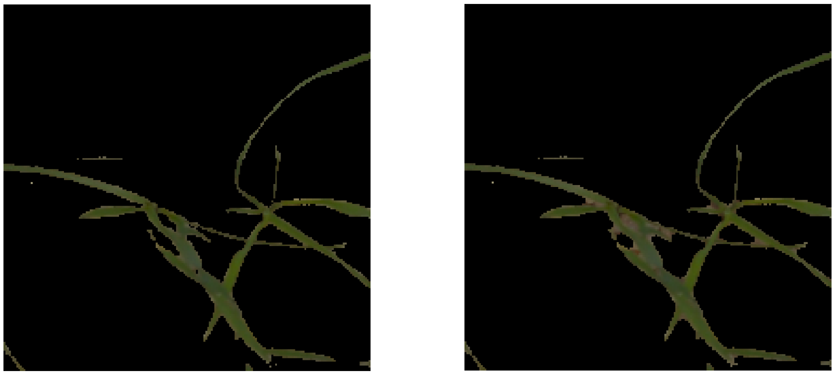


(a) Before morphological closure

(b) After morphological closure

Fig. 3. Segmentation improvement example

the desired result. We restrict ourselves to the removal of objects whose contours limit a small area.



(a) Before morphological closure

(b) After morphological closure

Fig. 4. Segmentation degradation example

2.3 Feature selection

Features of the images define their content. We recognize the information images provide us in consideration of a great number of features. Then we answer, what do we see exactly. The same process can be projected on the image classification task: image features let the classifier propose the output decision.

Another advantage of the approach is that it reduces feature space for a machine learning algorithm. We often need only a piece of the information image provides, hence we do not need to process and evaluate all the pixels, which can cause additional computational expenses.

Selecting features is a complicated and convoluted research area itself, the statement is supported by the variety of feature types, and the need of presenting essential properties on an equal basis with the previous assertion.

As discussed before, we need to define the set of features describing the dataset in the best way. Supposed features must satisfy the following criterion:

- The feature space should be low-dimensional
- The features should correlate or correlate as little as possible
- Selected features should represent the content of an image as fully as possible

We are going to group selected features and define them.

2.4 Color features

Overviewing the dataset, we can notice that all the plant species are mostly green. Additionally, images were recorded under specific conditions. We will use RGB color model, which stands for red, green, and blue colors, and calculate features described below.

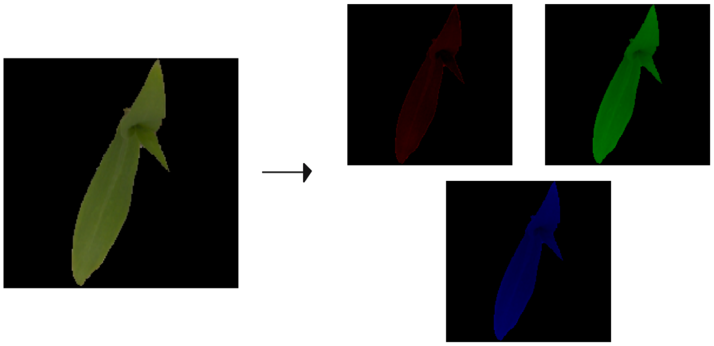


Fig. 5. RGB transformation

Let $\{x^{(k)}\}_{i=1}^{N}$, where $k = 1, 2, 3$ — an index of a channel in RGB color space respectively, N — total number of the image pixels, $x_i^{(k)}$ — i-th pixel of the k-th channel. We will compute sample mean and standard deviation for each channel:

$$\overline{x^{(k)}} = \frac{1}{N} \sum_{i=1}^{N} x_i^{(k)} \tag{1}$$

$$s^{(k)} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i^{(k)} - \overline{x^{(k)}})^2} \tag{2}$$

2.5 Shape features

The one widely-used approach to retrieve shape features is to detect and analyze bounding contours. We will use the boundary tracing algorithm for the boundary extraction. The designated algorithm [7] was implemented in the OpenCV [2] library for the Python programming language. The studies did not take into account contours bounding areas below a certain threshold, which was empirically chosen.

Let K be the number of detected bounding contours above the threshold in the further contour-related characteristics.

Total perimeter. In this feature, we will count the sum of perimeters of all the areas bounded by contours:

$$P = \sum_{i=1}^{K} p_i, \tag{3}$$

where p_i — i-th perimeter

Entire area. It includes all the areas bounded by contours:

$$S = \sum_{i=1}^{K} s_i, \tag{4}$$

where s_i — i-th area

Maximal contour area. We will be analyzing the contours bounding maximal areas:

$$S_m = \max s_i,\ i = 1, \ldots, K, \tag{5}$$

where s_i — i-th area

Rectangularity. One of the methods to estimate rectangularity is to plot minimum bounding rectangle. Rectangularity is the ratio of the entire obect area to the minimum bounding rectangle area. This featere represent how rectangular an object is:

$$f_{rect} = \frac{S}{S_{MBR}}, \tag{6}$$

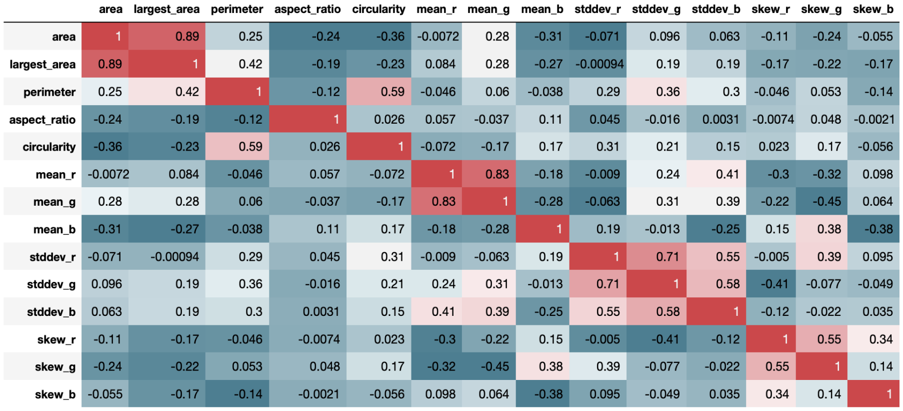
where S — entire area, S_{MBR} — minimum bounding rectangle area

Circularity. Another title of this shape factor is the isoperimetric quotient, and it shows how much area pe perimeter is bounding:

$$f_{circ} = \frac{4\pi A}{P^2}, \tag{7}$$

where P — entire perimeter, A — entire area of all detected elements of a plant

Consider the correlation matrix of the described features:



	area	largest_area	perimeter	aspect_ratio	circularity	mean_r	mean_g	mean_b	stddev_r	stddev_g	stddev_b	skew_r	skew_g	skew_b
area	1	0.89	0.25	-0.24	-0.36	-0.0072	0.28	-0.31	-0.071	0.096	0.063	-0.11	-0.24	-0.055
largest_area	0.89	1	0.42	-0.19	-0.23	0.084	0.28	-0.27	-0.00094	0.19	0.19	-0.17	-0.22	-0.17
perimeter	0.25	0.42	1	-0.12	0.59	-0.046	0.06	-0.038	0.29	0.36	0.3	-0.046	0.053	-0.14
aspect_ratio	-0.24	-0.19	-0.12	1	0.026	0.057	-0.037	0.11	0.045	-0.016	0.0031	-0.0074	0.048	-0.0021
circularity	-0.36	-0.23	0.59	0.026	1	-0.072	-0.17	0.17	0.31	0.21	0.15	0.023	0.17	-0.056
mean_r	-0.0072	0.084	-0.046	0.057	-0.072	1	0.83	-0.18	-0.009	0.24	0.41	-0.3	-0.32	0.098
mean_g	0.28	0.28	0.06	-0.037	-0.17	0.83	1	-0.28	-0.063	0.31	0.39	-0.22	-0.45	0.064
mean_b	-0.31	-0.27	-0.038	0.11	0.17	-0.18	-0.28	1	0.19	-0.013	-0.25	0.15	0.38	-0.38
stddev_r	-0.071	-0.00094	0.29	0.045	0.31	-0.009	-0.063	0.19	1	0.71	0.55	-0.005	0.39	0.095
stddev_g	0.096	0.19	0.36	-0.016	0.21	0.24	0.31	-0.013	0.71	1	0.58	-0.41	-0.077	-0.049
stddev_b	0.063	0.19	0.3	0.0031	0.15	0.41	0.39	-0.25	0.55	0.58	1	-0.12	-0.022	0.035
skew_r	-0.11	-0.17	-0.046	-0.0074	0.023	-0.3	-0.22	0.15	-0.005	-0.41	-0.12	1	0.55	0.34
skew_g	-0.24	-0.22	0.053	0.048	0.17	-0.32	-0.45	0.38	0.39	-0.077	-0.022	0.55	1	0.14
skew_b	-0.055	-0.17	-0.14	-0.0021	-0.056	0.098	0.064	-0.38	0.095	-0.049	0.035	0.34	0.14	1

Fig. 6. Feature correlation matrix

Based on the data in the figure 6, we conclude that the most linearly dependent features are the entire area and the largest area, but this is not true for all classes due to the predominance of plants, bounded by the only one contour, therefore the consideration of the maximum area feature is not rejected.

2.6 Classification

The main method for this task is Support Vector Machine (SVM) — a binary classification algorithm based on building a separating hyperplane. The algorithm is implemented in the Scikit-learn ([5]) library for the Python programming language.

We will use the Radial Basis Function (RBF) as the kernel function for SVM. The choice was made based on the following advantages:

- RBF usage allows building a hyperplane when the data is not linearly separable
- Only one parameter that affects the resulting solution needs to be tuned
- The values of the RBF function are in $(0, 1]$, which does not include zero or infinity

The SVM algorithm is sensitive to non-normalized data, especially when using the RBF kernel, which is the Euclidian distance itself. In the case, when the feature values are at different intervals, a slight difference in one of them can lead to going out of range in second feature values. The solution is to map all the values into one segment. In this task, we will choose $[0, 1]$.

3 Results

3.1 Metric

Results of classification are evaluated on a micro-averaged F-score. Given the positive and negative rates for each class, the resulting score is computed the following way:

$$Precision_{micro} = \frac{\sum_{k \in C} TP_k}{\sum_{k \in C} TP_k + FP_k}, \quad Recall_{micro} = \frac{\sum_{k \in C} TP_k}{\sum_{k \in C} TP_k + FN_k} \tag{8}$$

$$F_{micro} = \frac{2 * Precision_{micro} * Recall_{micro}}{Precision_{micro} + Recall_{micro}}, \tag{9}$$

where C — set of the plants classes

The choice of such a metric is supported by the fact that classes are imbalanced. In this case, the influence of classes decreases due to averaging by classification characteristics, not by F-scores.

The classificator result is shown in the table 1.

3.2 Models comparison

The choice of the SVM is justified by better results in comparison with other classical methods of machine learning. Table 2 shows micro-averaged F-scores for Naive Bayes, k-nearest neighbours and Decision tree classifiers. The methods are implemented in Scikit-learn ([5]) library. All the experiments were conducted 100 times each, on shuffled data, the metrics were obtained by testing on a validation sample and averaged.

Table 1. Detailed metrics for SVM classificator

Type	Precision	Recall	F-score
Sugar beet	0.901	0.936	0.918
Fat Hen	0.877	0.909	0.893
Scentless Mayweed	0.851	0.905	0.878
Charlock	0.947	0.934	0.940
Small-flowered Cranesbill	0.963	0.991	0.977
Maize	0.953	0.891	0.921
Shepherds Purse	0.833	0.714	0.769
Common Wheat	0.800	0.889	0.842
Common Chickweed	0.962	0.962	0.962
Cleavers	0.885	0.852	0.868
Loose Silky-bent	0.828	0.888	0.857
Black-grass	0.757	0.528	0.622
Micro-averaged F-score		0.885	

Table 2. Metrics for used classificators

Method	Micro-averaged F-score
naiveBayes	0.72
kNN	0.84
decisionTree	0.73
SVM	**0.885**

4 Conclusion

References

1. Bellman, R.: Dynamic Programming. Princeton University Press, Princeton, NJ, USA, 1 edn. (1957)
2. Bradski, G.: The opencv library. Dr. Dobb's Journal of Software Tools (2000)
3. Friel, J.J.: Practical Guide to Image Analysis. ASM International (2000)
4. Giselsson, T.M., Jørgensen, R.N., Jensen, P.K., Dyrmann, M., Midtiby, H.S.: A public image database for benchmark of plant seedling classification algorithms (2017)
5. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in python. J. Mach. Learn. Res. (2011)
6. Shi, L., Wan, Y., Gao, X., Wang, M.: Feature selection for object-based classification of high-resolution remote sensing images based on the combination of a genetic algorithm and tabu search. Computational Intelligence and Neuroscience (2018). https://doi.org/10.1155/2018/6595792

7. Suzuki, S., Abe, K.: Topological structural analysis of digitized binary images by border following. Computer Vision, Graphics, and Image Processing (1985). https://doi.org/10.1016/0734-189X(85)90016-7, https://doi.org/10.1016/0734-189X(85)90016-7
8. Yang, M., Kpalma, K., Ronsin, J.: A survey of shape feature extraction techniques (2008)