8-1-2018

Apple Flower Detection Using Deep Convolutional Networks

Philipe A. Dias
Marquette University

Amy Tabb
Marquette University

Henry P. Medeiros
Marquette University, henry.medeiros@marquette.edu

Accepted version. *Computers in Industry,* Vol. 99 (August 2018): 17-28. DOI.

Apple Flower Detection Using Deep Convolutional Networks

Philipe A. Dias
Marquette University, Department of Electrical and Computer Engineering, Milwaukee, WI
Amy Tabb
U.S. Department of Agriculture (USDA), Kearneysville, WV
Henry Medeiros
Marquette University, Department of Electrical and Computer Engineering, Milwaukee, WI

Abstract

To optimize fruit production, a portion of the flowers and fruitlets of apple trees must be removed early in the growing season. The proportion to be removed is determined by the bloom intensity, i.e., the number of flowers present in the orchard. Several automated computer vision systems have been proposed to estimate bloom intensity, but their overall performance is still far from satisfactory even in relatively controlled environments. With the goal of devising a technique for flower identification which is robust to clutter and to changes in illumination, this paper presents a method in which a pre-trained convolutional neural network is fine-tuned to become specially sensitive to flowers. Experimental results on a challenging dataset demonstrate that our method significantly outperforms three approaches that represent the state of the art in flower detection, with recall and precision rates higher than 90%. Moreover, a performance assessment on three additional datasets

previously unseen by the network, which consist of different flower species and were acquired under different conditions, reveals that the proposed method highly surpasses baseline approaches in terms of generalization capability.

Keywords

Bloom intensity estimation, Apple flower detection, Deep learning, Convolutional neural networks, Precision agriculture

1. Introduction

Various studies have established the relationships between bloom intensity, fruit load and fruit quality [[1], [2]]. Together with factors such as climate, bloom intensity is especially important to guide thinning, which consists of removing some flowers and fruitlets in the early growing season. Proper thinning directly impacts fruit market value, since it affects fruit size, coloration, taste and firmness.

Despite its importance, there has been relatively limited progress so far in automating bloom intensity estimation. Currently, this activity is typically carried out manually with the assistance of rudimentary tools. More specifically, it is generally done by inspecting a random sample of trees within the orchard and then extrapolating the estimates obtained from individual trees to the remainder of the orchard [3]. As the example in Fig. 1 illustrates, obstacles that hamper this process are: (1) manual tree inspection is time-consuming and labor-intensive, which contributes to making labor responsible for more than 50% of apple production costs [4]; (2) estimation by visual inspection is characterized by large uncertainties and is prone to errors; (3) extrapolation of the results from the level of the inspected trees to the row or parcel level relies heavily on the grower's experience; and (4) inspection of a small number of trees does not provide information about the spatial variability which exists in the orchard, making it difficult to develop and adopt site-specific crop load management strategies that could lead to optimal fruit quality and yield.

Fig. 1. Example of image from a flower detection dataset used in this paper.

With the goal of introducing more accurate and less labor intensive techniques for the estimation of bloom intensity, machine vision systems using different types of sensors and image processingtechniques have been proposed [5]. Most existing methods, which are mainly based on simple color thresholding, have their applicability hindered especially by variable lighting conditions and occlusion by leaves, stems or other flowers [6].

Inspired by successful works using convolutional neural networks (CNNs) in multiple computer visiontasks, we propose a novel method for apple flower detection based on features extracted using a CNN. In our approach,

an existing CNN trained for saliency detection is fine-tuned to become particularly sensitive to flowers. This network is then used to extract features from portraits generated by means of superpixel segmentation. After dimensionality reduction, these features are fed into a pre-trained classifier that ultimately determines whether each image region contains flowers or not. The proposed method significantly outperformed state-of-the-art approaches on four datasets composed of images acquired under different conditions.

Our main contributions are:

(1) a novel CNN-based flower detection algorithm;
(2) an extensive evaluation on a challenging dataset acquired under realistic and uncontrolled conditions;
(3) an analysis of the generalization capability of the proposed approach on additional datasets previously unseen by the evaluated models.

The remainder of paper is organized as follows. Section 2 discusses the most relevant existing approaches for automated flower and fruit detection. Our proposed approach is described in Section 3, which also includes a description of three baseline comparison methods as well as implementation details. Experiments performed to evaluate the impact of specific design choices are described in Section 4, followed by an extensive comparison of our optimal model against the baseline methods on four different datasets. Our concluding remarks are presented in Section 5.

2. Related work

While existing techniques employed for flower detection are based only on color information, methods available for fruit quantification exploit more modern computer vision techniques. For this reason, in this section we first review the most relevant works on automated flower detection, followed by a discussion of the relevant literature on fruit quantification. Moreover, to make this article self-contained and therefore accessible to a wider audience, we also provide a brief introduction to the fundamentals of CNNs.

2.1. Flower and fruits quantification

Aggelopoulou and colleagues presented in [7] one of the first works using computer vision techniques to detect flowers. That method is based on color thresholding and requires image acquisition at specific daylight times, with the presence of a black cloth screen behind the trees. Thus, although its reported error in predicted yield is relatively low (18%), such approach is applicable only for that controlled scenario.

Similar to the work of Thorp and Dierig [8] for identification of *Lesquerella* flowers, the technique described by Hočevar et al. [9] does not require a background screen, but it is still not robust to changes in the environment. The image analysis procedure is based on hard thresholding according to color (in the HSL color space) and size features, such that parameters have to be adjusted whenever changes in illumination (daylight/night), in flowering density (high/low concentration) or in camera position (far/near trees) occur.

Horton and his team described in [10] a system for peach bloom intensity estimation that uses a different imaging approach. Based on the premise that the photosynthetic activity of this species increases during bloom period, the system relies on multispectral aerial images of the orchard, yielding an average detection rate of 84.3% for 20 test images. Similarly to the aforementioned methods, the applicability of this method also has the intrinsic limitation of considering only color/spectral information (thresholding near-infrared and blue bands), such that its performance is sensitive to changes in illumination conditions.

More advanced computer vision techniques have been employed for fruit quantification [5]. A multi-class image segmentation for agrovision is proposed by Hung et al. [11], classifying image pixels into leaves, almonds, trunk,

ground and sky. Their method combines sparse autoencoders for feature extraction, logistic regression for label associations and conditional random fields to model correlations between pixels. Some other methods are based on support vector machine (SVM) classifiers that use information obtained from different shape descriptors and color spaces as input [[12], [13]]. Compared to existing methods for flower detection, these methods are more robust since morphological characteristics are taken into account. As many other shape-based and spectral-based approaches [[14], [15], [16], [17]], these techniques are, however, still limited by background clutter and variable lighting conditions in orchards [3].

Recent works on fruit quantification include the use of metadata information. Bargoti and colleagues [18] built on [11] to propose an approach that considers pixel positions, orchard row numbers and the position of the sun relative to the camera. Similarly, Cheng et al. [19] proposed the use of information such as fruit number, fruit area, area of apple clusters and foliage area to improve accuracy of early yield prediction, especially in scenarios with significant occlusion. However, the inclusion of metadata is highly prone to overfitting, particularly when limited training data is available and the variability of the training set is hence low [18].

2.2. Deep learning

Following the success of Krizhevsky's model [20] in the ImageNet 2012 Challenge, deep learning methods based on CNNs became the dominant approach in many computer vision tasks. The architecture of traditional CNNs consists of a fixed-size input, multiple convolutional layers, pooling (downsampling) layers and fully connected layers [21]. Winner of the ImageNet 2013 Classification task, the Clarifai model is one such network [22]. Illustrated on the right side of Fig. 2, it takes input imageportraits of size 227 × 227 pixels, which traverse a composition of 5 convolutional layers (C1–C5) and 3 fully connected layers (FC6–FC7 and the softmax FC8). Each type of layer plays a different role within the CNN architecture: while convolutional layers allow feature extraction, the latter fully connected layers act on this information to perform classification.

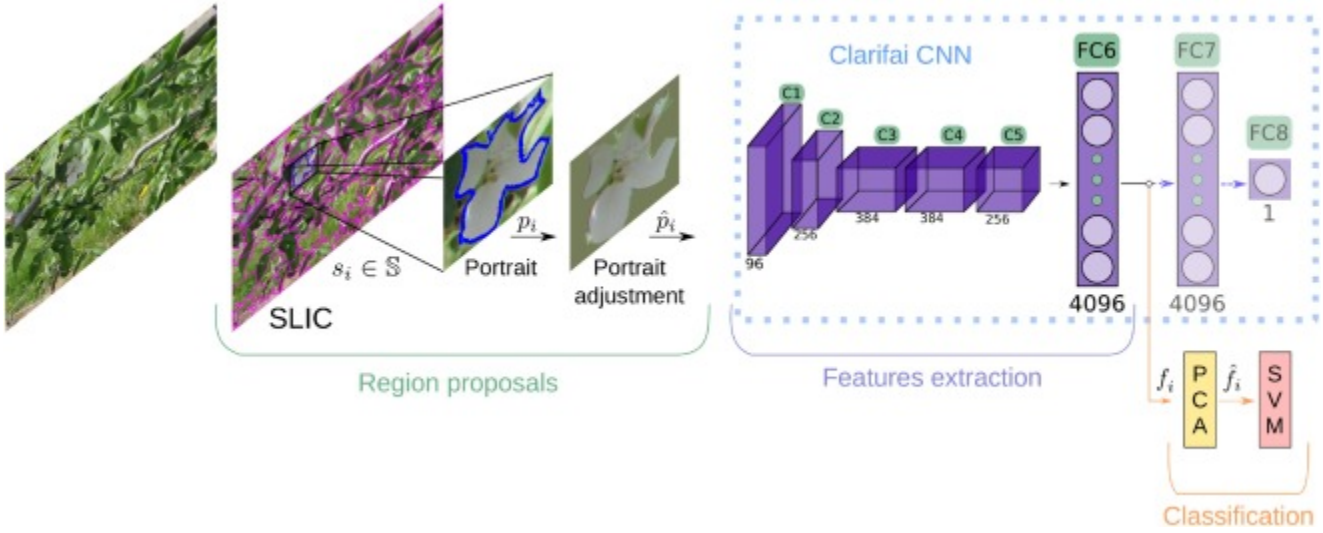

Fig. 2. Diagram illustrating the sequence of image analysis tasks performed by the proposed model for flower identification. Layers FC7–FC8 are used only during fine-tuning (training). For final prediction, features are collected from the output of layer FC6. Each task and its corresponding output (shown above the arrows) are described in Algorithm 1.

In computer vision and image processing, a *feature* corresponds to information that is meaningful for describing an image and its regions of interest for further processing. Feature extraction is therefore crucial in image analysis, since it represents the transition from pictorial (qualitative) to nonpictorial (quantitative) data representation [23]. Rather than relying on hand-engineered features (e.g., HOG [24]), deep CNNs combine

multiple convolutional layers and downsampling techniques to learn hierarchical features, which are a key factor for the success of these models [25]. As described in [22], the convolutional layers C1–C2 learn to identify low-level features such as corners and other edge/color combinations. The following layers C3–C5 combine this low-level information into more complex structures, such as motifs, object parts and finally entire objects.

Traditional deep CNNs are composed of millions of learned parameters (over 60 million in AlexNet [20]), such that large amounts of labeled data are required for network training. Deep learning modelsbecame feasible relatively recently, after the introduction of large publicly available datasets, of graphics processing units (GPUs), and of training algorithms that exploit GPUs to efficiently handle large amounts of data [25]. Nevertheless, gathering domain specific training data is an expensive task. One alternative to reduce the required amount of labeled data is data augmentation, a technique proven to benefit the training of multiple machine learning models [26]. It is typically performed by applying transformations such as translation, rotation and color space shifts to pre-labeled data.

In addition, transfer learning approaches such as fine-tuning have been investigated. Earlier layers of a deep network tend to contain more generic information (low-level features), which are then combined by the latter layers into task specific objects of interest. Thus, a network that can recognize different objects present in a large dataset must contain a set of low-level descriptors robust enough to characterize a wide range of patterns. Under this premise, fine-tuning procedures typically aim at adjusting the higher-level part of a network pre-trained on a large generic dataset, rather than training the full network from scratch. This greatly reduces the need for task-specific data, since only a smaller set of parameters has to be refined for the particular application [27].

At the classification side, most CNN architectures employ fully connected layers for final categorization. They determine which features are mostly correlated to each specific class employing a logistic regression classifier. For scenarios in which the output is binary, consistent albeit small improvements on popular datasets have been demonstrated by replacing the final CNN layer by a SVM classifier [28]. SVM models tend to generalize better than logistic regression, since they target a solution that not only minimizes the training error, but also maximizes the margin distance between classes.

Following the success of CNNs on image classification tasks, the work of Girschick et al. [29] introduced the concept of region-based CNNs (R-CNN), outperforming by a large margin previous hand-engineered methods for object detection. In that work, a CNN is first pre-trained on a large auxiliary dataset (ImageNet) and then fine-tuned using a smaller but more specific dataset (PASCAL dataset for object detection). The Faster R-CNN proposed in [30] improved this model by replacing selective search [31] with the concept of Region Proposal Network (RPN), which shares convolutional layers with the classification network. Both modules compose a single, unified network for object detection.

Recent works adapt the Faster R-CNN for fruit detection. Bargoti and Underwood [32] present a Faster R-CNN trained for detection of mangoes, almonds and apples fruits on trees. Stein et al. [33] extended this model for tracking and localization of mangoes, combining it with a monocular multi-view tracking module that relies on a GPS system. Sa et al. [34] applied the Faster R-CNN to RGB and near-infrared multi-modal images. Each modality was fine-tuned independently, with optimal results obtained using a late fusion approach. Still in the context of agricultural applications, CNNs have been also successfully used for plant identification from leaf vein patterns [35].

In summary, existing methods for flower identification are based on hand-engineered image processing techniques that work only under specific conditions. Color and size thresholding parameters composing these algorithms have to be readjusted in case of variations of lightning conditions, camera position with respect to

the orchard (distance and angle), or expected bloom intensity. Recent techniques employed for fruit quantification exploit additional features and machine learning strategies, providing insights to further develop strategies for flower detection. Aiming at a technique for flower identification that is robust to clutter, changes in illumination and applicable for different flower species, we therefore propose a novel method in which an existing CNN trained for saliency detection is fine-tuned to become particularly sensitive to flowers.

3. Proposed approach

In this section, we first describe the prediction steps performed by our method, i.e., the sequence of operations applied to an image in order to detect the presence of flowers. Subsequently, we describe the fine-tuning procedure carried out to obtain the core component of our model: a CNN highly sensitive to flowers. We conclude with a discussion of alternative flower detection approaches against which we evaluate our proposed method and a brief mention of relevant details regarding the implementation of our methods.

In the discussion that follows, we will refer to our proposed approach for flower detection as the *CNN + SVM* method. As illustrated in Fig. 2, our CNN + SVM method consists of three main steps: (i) computation of region proposals; (ii) feature extraction using our fine-tuned CNN, which follows the Clarifai architecture [22]; and (iii) final classification of each region according to the presence of flowers. The operations that comprise these steps are described in detail below. In our description, we make reference to Algorithm 1, which lists the operations performed by our method on each input image. The sensitivity of the method to specific design choices is detailed in Section 4.1.

(1) Step 1 – Region proposals: The first step in the proposed method consists of generating region proposals by grouping similar nearby pixels into superpixels, which are perceptually meaningful clusters of variable size and shape (Line 1 of Algorithm 1). To this end, we use the *simple linear iterative clustering* (SLIC) superpixel algorithm. Currently one of the most widely-used algorithms for superpixel segmentation, it adapts *k*-means clustering to group pixels according to a weighted distance measure that considers both color and spatial proximity [36]. For additional information on superpixel approaches, we refer the reader to the review provided in [37]. The second leftmost image in Fig. 2illustrates the superpixels si∈S generated by the SLIC algorithm when applied to a typical image obtained in an orchard.

Although other approaches such as the Faster R-CNN [30] provide a unified architecture in which both region proposal and classification modules can be fine-tuned for a specific task, they have more parameters that need to be learned in a supervised manner. Since in most cases flowers are salient with respect to its surrounding background, an unsupervised, local-context based approach such as superpixel segmentation should be sufficient to obtain region proposals suitable for flower detection.

Algorithm 1 Proposed approach for flower detection

Input: Image *I*.	
Output: Regions in *I* containing flowers.	
1:	Segment *I* into set of superpixels S using SLIC.
2:	for each superpixel si∈S do
3:	Crop smallest squared portrait p_i enclosing s_i.
4:	Generate piˆ by mean-padding the background surrounding s_i in p_i.
5:	Extract features f_i from the mean-centered piˆ using the fine-tuned CNN.
6:	Obtain fˆi by performing PCA analysis on f_i.
7:	Classify s_i by applying pre-trained SVM on fˆi.
8:	end for

Once the image is segmented into superpixels, as Algorithm 1 indicates, we iterate over each superpixel in the image. Since the input size required by the Clarifai CNN model is 227 × 227, we first extract the smallest square portrait enclosing the superpixel under analysis (Line 3), which we denote p_i. The output of this step is illustrated in the third leftmost image of Fig. 2 for one superpixel. The background surrounding the superpixel of interest within a portrait is then padded with the training set mean, i.e., the average RGB color of all images composing the dataset (greenish color). Finally, the portrait is resized to 227 × 227 (Line 4). The resulting region proposal, pˆi, is illustrated in the fourth image of Fig. 2.

(2) Step 2 – Feature extraction: In the feature extraction step (Line 5), each of the portraits generated above is mean-centered and then evaluated individually by our CNN. The mean-centering step consists of subtracting from the portrait the same average training set RGB mean used for padding its background. This procedure is commonly employed to facilitate training convergence of deep learning models, since it ensures similarly ranged features within the network. For each input portrait, we collect as features the output of the rectified linear unit (ReLU) associated with the first fully connected layer (FC6). With a dimensionality of $N = 4096$, the feature vector $fi \in \mathbb{R}N$ collected at this stage of the network encapsulates the hierarchical features extracted by layers $C1$–$C5$, which contain the key information required for accurate classification.

(3) Step 3 – Classification: To classify each proposed region as containing a flower or not, we first perform principal component analysis (PCA) to reduce the feature dimensionality to a value $k < N$ such that the new feature vector $fˆi \in \mathbb{R}k$ (Line 6). As demonstrated in our experimental evaluation in Section 4.1.1 a value of $k = 69$, which corresponds to approximately 94% of the original variance of the data, provides performance levels virtually identical to those of the original features. Finally, based on these features a pre-trained SVM model binary classifies superpixels according to the presence of flowers (Line 7). Details on SVM training are provided in the next section.

3.1. Network fine-tuning and SVM training

Based on the techniques introduced by Girshick et al. [29] and Zhao et al. [38] for object and saliency detection, in our model an existing CNN architecture is made particularly sensitive to flowers by means of fine-tuning. In the work of Zhao et al. [38], the Clarifai model [22] was adopted as the starting point and fine-tuned for saliency detection. We further tuned Zhao et al.'s model for flower identification using labeled portraits from our training set, which we describe below.

The generation of training samples for network tuning takes place similarly to prediction. For each labeled image composing the training set, we computed region proposals according to Step 1 described above. Using these training examples, 10,000 backpropagation training iterations were performed in order to minimize the network classification error. After fine-tuning, we computed the CNN features of the training examples, reduced their dimensionality to $k = 69$, and used them to train the SVM classifier.

Image dataset. Images of apple trees were collected using a camera model Canon EOS 60D under natural daylight illumination (i.e., uncontrolled environment). This dataset, which we refer to as *AppleA*, is composed of a total of 147 images with resolution of 5184 × 3456 pixels acquired under multiple angles and distances of capture. Fig. 3 shows some images that comprise this dataset. For performance evaluation and learning purposes, the entire dataset was labeled using a MATLAB GUI in which the user selected only superpixels that contain parts of flowers in approximately half of its total area. As summarized in Table 1, the labeled images were randomly split into training and validation sets composed of 100 and 47 images, respectively. This corresponds to a total of 91,488 training portraits (i.e., superpixels) and 42,430 validation ones. The training

examples were used to fine-tune the network and train the SVM as described above. The validation examples were used in the performance evaluation discussed in Sections 4.1 and 4.2.

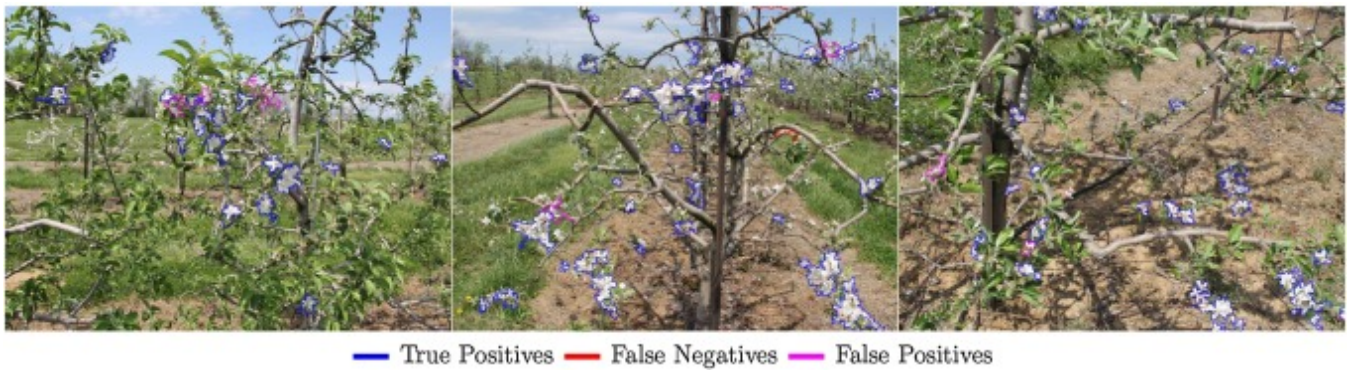

Fig. 3. Examples of images composing the *AppleA* dataset, with the corresponding detection provided by the proposed algorithm (more examples are available in the supplementary material).

Table 1. Statistics of the training and validation dataset (*AppleA*).

		Portraits (i.e., superpixels)		
	Images	**Positives**	**Negatives**	**Total**
Training	100	3691 (4%)	87,797 (96%)	91,488
Validation	47	1719 (4%)	40,711 (96%)	42,430
Total	147	5410 (4%)	128,508 (96%)	133,918

Data augmentation. According to our labeling, only 4% of the samples contain flowers (positives). Imbalanced datasets represent a problem for supervised machine learning approaches, since overall accuracy measures become biased towards recognizing mostly the majority class [39]. In our case, that means the learner would present a bias towards classifying the portraits as negatives. To overcome this situation and increase the amount of training data, we quadrupled the number of positive samples using data augmentation. As illustrated in Fig. 4, this was accomplished by mirroring each positive sample with respect to: (i) the vertical axis, (ii) the horizontal axis, and (iii) both axes.

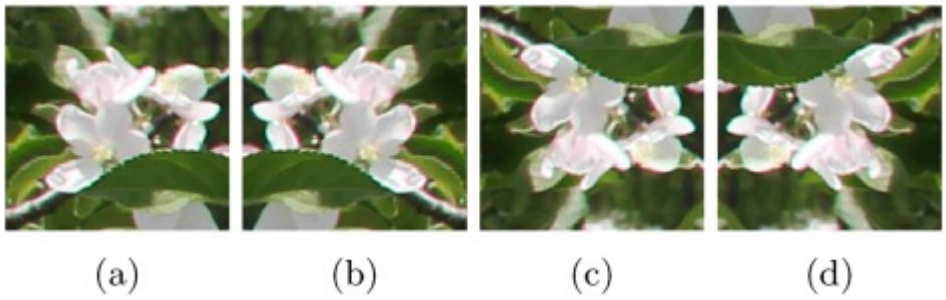

Fig. 4. Example of data augmentation. (a) Original portrait, (b) portrait mirrored with respect to the vertical axis, (c) the horizontal axis, and (d) both axes.

Parameters' optimization. Support vector machines are supervised learning models that search for a hyperplane that maximizes the margin distance to each class. This characteristic allows SVM models to generalize better than classifiers such as the ones based on logistic regression. For non-linearly separable data, kernel functions such as the popular radial basis function (RBF, or *Gaussian*) are used. We refer to [[40], [41]] for further details on the formulation of SVMs.

Two main parameters control the performance of SVMs with a Gaussian kernel function, the regularization cost *C* and the width of the Gaussian kernel *γ*. By regulating the penalty applied to misclassifications, the parameter *C* controls the trade-off between maximizing the margin with which two classes are separated and the complexity of the separating hyperplane. The parameter *γ* regulates the flexibility of the classifier's hyperplane. For both parameters, excessively large values can lead to overfitting.

The optimization of *C* and *γ* is a problem without straightforward numerical solution. Therefore, it is typically solved using grid search strategies [[40], [41]] in which multiple parameter combinations are evaluated according to a performance metric. We adopt this strategy in this work.

3.2. Comparison approaches

As has been noted in Section 2, current algorithms for automated identification of flowers are mostly based on binarization by thresholding information from different color-spaces (typically RGB or HSV) [[7], [8]], occasionally combined with size filtering [9]. We implement three alternative baseline approaches which reflect the state of the art in fruit/flower detection. The first implementation, which we call the *HSV* method, replicates the algorithm described by Hocevar and his team in [9]. Images are binarized at pixel-level based on HSV color information, followed by filtering according to minimum and maximum cluster sizes.

We refer to the second baseline implementation as *HSV + Bh*. Similar to our proposed approach, the starting point for this method is the generation of superpixels using the SLIC algorithm. We then compute a 100-bin histogram of each superpixel in the HSV color space, which has the advantage of dissociating brightness from chromaticity and saturation. Studies on human vision and color-based image retrieval have demonstrated that most of the color information is contained in the hue channel, with the saturation playing a significant role in applications where identifying white (or black) objects is important [[42], [43]]. In our experiments, we construct a single 1-D histogram consisting of 100 bins, which corresponds to the concatenation of a 50-bin hue channel histogram, a 40-bin saturation histogram and a 10-bin value histogram. Afterwards, we use the Bhattacharyya distance [44] to compare each superpixel histogram against the histograms of all positive samples composing the training set. We compute the Bhattacharyya distance using a Gaussian kernel function, as formulated in [[45], [46]]. The average Bhattacharyya distance is taken as the likelihood that the superpixel includes a flower, and superpixels with distance lower than an optimal threshold are classified as flowers.

Since the technique described above is based on the average Bhattacharyya distance in the HSV color space, it makes no distinction between poorly and highly informative training sample features. Its ability to make accurate classification decisions is therefore limited in such complex feature spaces. Inspired by works on fruit quantification [[12], [13]], we extend this method by combining the same HSV histograms with an SVM classifier for apple flower detection. That is, rather than determining whether a superpixel contains a flower based on the Bhattacharyya distance, we train an SVM classifier that uses the HSV histograms as inputs. We call this approach the *HSV + SVM* method.

3.3. Implementation details

Most image analysis tasks were performed using MATLAB R2016b. Additionally, we used the open source Caffe framework [47] for fine-tuning and extracting features from the CNN. To reduce computation times by exploiting GPUs, we used the cuSVM software package for SVM training and prediction [48].

4. Experiments and results

Experiments were performed with three main goals. Our optimal CNN + SVM model extracts features from the CNN's first fully connected layer (*FC*6), reduces feature dimensionality to 69, and performs final classification using SVM. Thus, we first evaluated the impact of these specific design choices on the final performance of our

method. We then compared it against the three baseline methods (HSV, HSV + Bh and HSV + SVM). Finally, we evaluated the performance of the proposed approach on previously unseen datasets to determine its generalization capability.

As described in Section 3.1, our datasets are severely imbalanced. In such scenarios, evaluations of performance using only accuracy measurements and ROC curves may be misleading, since they are insensitive to changes in the rate of class distribution. We therefore perform our analysis in terms of precision-recall curves (PR) and the corresponding F_1 score [49]. Precision is normalized by the number of positives rather than the number of true negatives, so that false positive detections have the same relative weight as true positives. While the maximum F_1 score indicates the optimal performance of a classifier, the area under the respective PR curve (AUC-PR) corresponds to its expected performance across a range of decision thresholds, such that a model with higher AUC-PR is more likely to generalize better.

4.1. Analysis of design choices

In order to validate our design choices, we performed experiments to evaluate how the final performance of the classifier is affected by: (i) the dimensionality of the feature space; (ii) the point where features are collected from the CNN; and (iii) the type of input portrait.

4.1.1. Dimensionality analysis

PCA is one of the most widespread techniques for dimensionality reduction. It consists of projecting N-dimensional input data onto a k-dimensional subspace in such a way that this projection minimizes the reconstruction error (i.e., L_2 norm between original and projected data) [50]. PCA can be performed by computing the eigenvectors and eigenvalues of the covariance matrix and ranking principal components according to the obtained eigenvalues [51].

In this application, the original dimensionality corresponds to the number of elements in the CNN vectors extracted from a fully connected layer, i.e., $N = 4096$ as represented for the two last layers in Fig. 2. The first two columns of Table 2 show the reduced dimensionality k of the feature vector and the corresponding ratio of the total variance of the N-dimensional dataset that is retained at that dimensionality for layer *FC*6. As the table indicates, the first most significant dimension alone already covers almost half of the total variance, and 23 dimensions are sufficient to cover nearly 90% of it.

Table 2. Classification performance according to the number of principal components (dimensions) selected after applying PCA to the extracted features. Best results in terms of F_1 and AUC-PR are shown in boldface.

No. of dimensions	Variance ratio	F_1	Recall	Precision	AUC-PR
1	48.3%	90.4%	92.2%	88.6%	96.5%
2	63.7%	91.4%	92.7%	90.2%	94.0%
5	79.9%	91.9%	92.3%	91.5%	96.9%
15	87.4%	91.5%	92.6%	90.4%	94.3%
23	89.6%	**92.1 %**	92.9%	91.2%	95.2%
69	93.8%	91.9%	92.6%	91.2%	**97.2**%
150	95.8%	91.3%	92.7%	90.0%	97.1%
300	97.2%	91.6%	91.6%	91.7%	97.2%
500	98.0%	91.8%	91.8%	91.8%	95.0%
1080	99.0%	91.7%	91.5%	91.8%	94.9%

We investigated then how samples are mapped into the lower dimensional feature space. Fig. 5 shows the projections in dimensions 1 and 2 as well as dimensions 1 and 3. These plots illustrate how the convolutional network maps the samples into a space where it is possible to differentiate between multiple clusters. With *dim.* as an abbreviation for *dimension*, let ↓ denote low dimensionality values and ↑ high values, respectively. The following clusters can be identified: flowers (↓ dim.1, ↑ dim.2); grass/floor (↑ dim.1, ↑ dim.2); branches/leaves (↓ dim.2); sky (↑ dim.3). This indicates that positive and negative samples are almost linearly separable even for 2*D* projections of the original feature space.

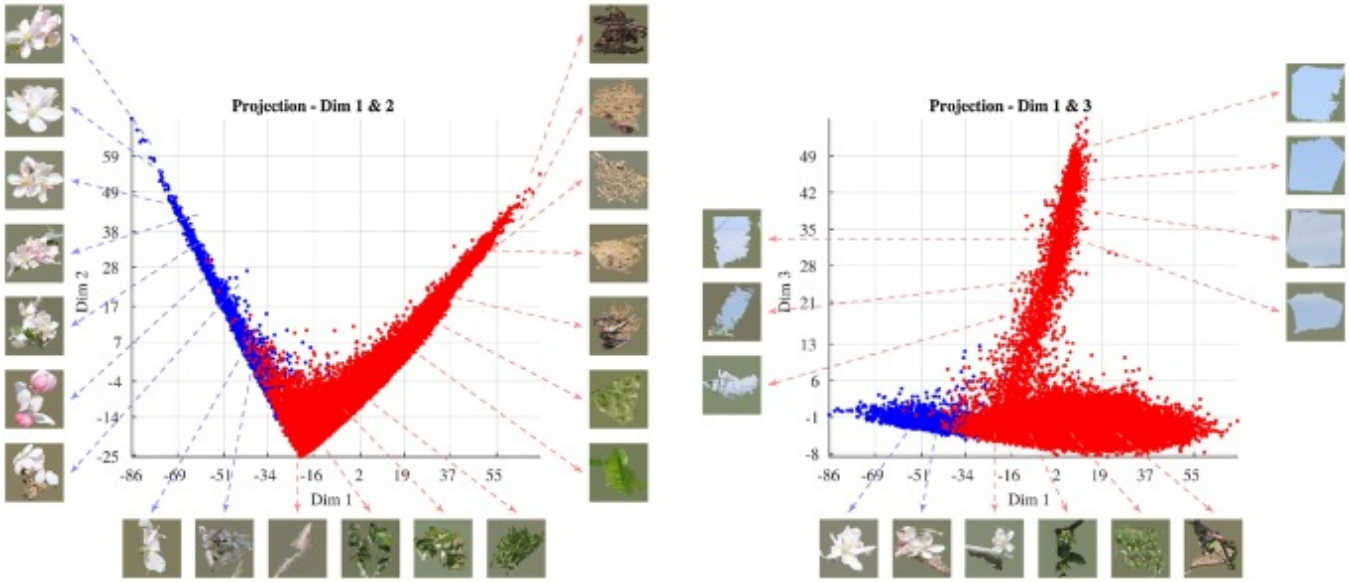

Fig. 5. Projections of samples on 2D feature spaces, with positive samples in blue and negatives ones in red. *Left*: sample distribution on the plane corresponding to dimensions 1 and 2 according to PCA. *Right*: sample distribution on the plane corresponding to dimensions 1 and 3. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

To quantitatively assess how the classification performance is affected by the dimensionality of the feature space, we trained SVM classifiers for different numbers of dimensions. For each dimensionality, Table 2 presents the optimal performance metrics and corresponding AUC-PR. As expected, these results demonstrate that the impact of dimensionality on the optimal performance of our method is rather low. A very good performance is already obtained using a 2D feature space, with both F_1 score and AUC-PR only around 0.7% and 3.2% lower than the highest obtained values, respectively. In terms of optimal recall and precision, this is equivalent to missing extra 4 positive samples out of 1719, while including more 19 false-positives out of 40,711. Moreover, the table shows that a dimensionality of 69 is nearly optimal: the performance in terms of optimal F_1 score is only 0.2% lower than the highest obtained value (23 dimensions) and it is optimal in terms of AUC-PR.

Although in the discussion above we present results obtained using SVMs, such a high separabilityeven for low dimensionalities indicates that the final prediction accuracy of our model is almost independent of the type of classifier employed. This conjecture is validated in the next subsection, where we demonstrate that the performance of our system does not change significantly by either including an additional fully connected layer to our CNN or by carrying out classification using the using network's softmax layer directly.

4.1.2. Feature analysis

As explained in Section 3, after fine-tuning the model, we use it to extract features that allow the classification of superpixels according to the presence of flowers within them. Three combinations of features and

classification mechanisms were investigated: (A) predict using solely the neural network, by means of its softmax output layer; (B) train a SVM classifier on features collected after the last fully connected layer (FC7); (C) train a SVM classifier on features collected after the first fully connected layer (FC6). Fig. 6 shows the points where features are collected and how classification scores are computed using these features. Following the notation used in Fig. 2, C1–C5 correspond to the convolutional layers of the fine-tuned Clarifai network, FC6–FC7 are the fully connected layers, and FC8 is the softmax layer.

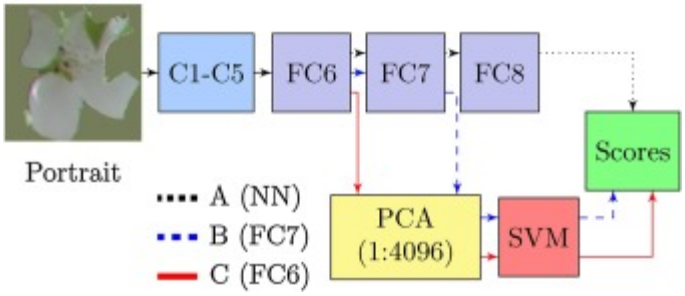

Fig. 6. Diagram illustrating how classification scores are computed using the extracted features.

For approaches B and C, features are collected from the output of the rectified linear units (ReLUs) located right after the respective fully connected layers. The same sequence of operations is performed for both methods B and C, i.e., the framework is the same regardless of whether the features are collected from the last (FC7) or first fully connected layer (FC6). Based on the results obtained in the previous section, for both cases 69 dimensions are kept after PCA analysis.

Results obtained for classification on the validation set are summarized in Table 3 and Fig. 7. As Fig. 7indicates, all three approaches show very similar performance. A closer inspection of Table 3 reveals that the SVM-based approaches slightly outperform the direct use of the neural network softmax layer both in terms of optimal F_1 score and AUC-PR. The performances obtained with methods B and C are very similar for both metrics. We therefore opted for method C, which uses features extracted from the earlier layer FC6 and provides slight increases in both optimal F_1 score and AUC-PR.

Table 3. Classification performance according to the CNN layer at which features are collected – methods A, B, C.

	AUC-PR	**F_1**	**Recall**	**Precision**
A (NN)	96.9%	90.6%	91.7%	89.6%
B (FC7)	97.2%	91.6%	91.8%	91.4%
C (FC6)	97.3%	91.9%	92.6%	91.2%

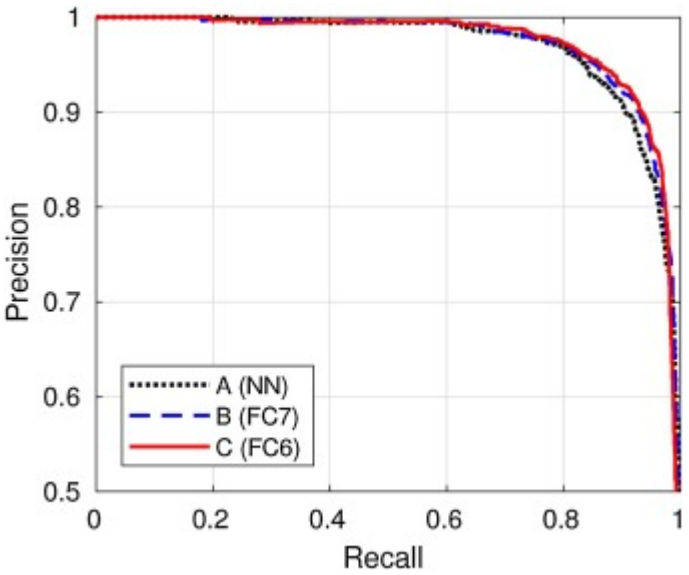

Fig. 7. PR curves illustrating the performance on the validation set according to the CNN layer at which features are collected. *NN* stands for prediction using solely the network softmax output layer, while *FC6* and *FC7* correspond to SVM classifiers trained on features collected at the first and second fully connected layers, respectively.

4.1.3. Different types of portraits

Using superpixels for region proposal computation and subsequent generation of portraits implies that our goal is to evaluate whether the superpixel itself is composed of flowers or not. In order to assess the influence of the local context surrounding the superpixel on the classification results, in addition to the approach based on replacing the region around the superpixel with the mean RGB value, two alternative approaches for portrait generation were considered. The first consists of retaining the unmodified image area surrounding the superpixel, whereas the second corresponds to blurring the background surrounding the superpixel with a low-pass filter. For all three cases, the portrait is mean-centered before being fed into the neural network. The three types of evaluated portraits are illustrated in Fig. 8.

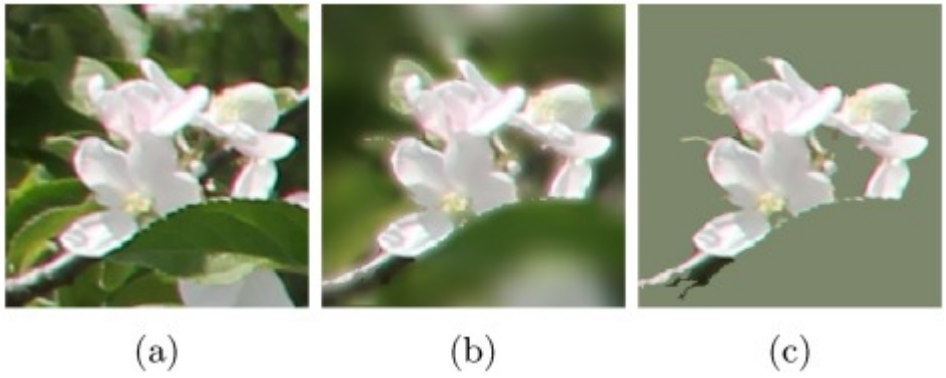

Fig. 8. Example of the three types of portrait evaluated. (a) *Original*; (b) blurred background (*Blur*); (c) mean padded background.

Fig. 9 shows the PR curves obtained for each portrait type. The best performance is obtained with mean-padded portraits, a behavior explained by the existence of cases such as the ones illustrated in Fig. 10. The superpixels highlighted in the images on the top row do not contain flowers in more than 50% of their area and should therefore not be classified as flowers. However, these superpixels are surrounded by flowers, as depicted in the

corresponding figures in the bottom row, and hence the approach of simply cropping a square region around the superpixel leads to cases in which the portrait contains a well-defined flower. As a consequence, features extracted from the CNN for the entire portrait will indicate the presence of flowers and therefore lead to high confidence false positives, which explain the non-maximal precision ratios in the upper-left part of the respective PR curve. This problem is eliminated by mean-padding the background.

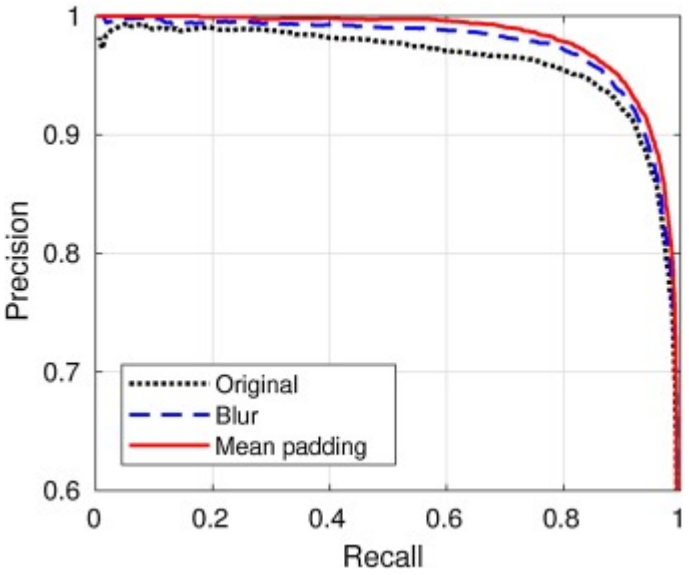

Fig. 9. Classification performance according to the portrait adjustment strategy. *Original* stands for portraits evaluated without any further adjustment, *Blur* corresponds to portraits where the background is blurred, and *mean padding* denotes the strategy of padding the background with the training set mean.

Fig. 10. Examples of superpixels incorrectly classified for *Original* and *Blur* portraits. The superpixels are shown in the top row and the bottom row shows the entire portraits enclosing the superpixels.

4.2. Comparison against baseline methods

The analysis in Section 4.1 above validates the design choices of our optimal CNN + SVM model described in Section 3. That is, our optimal model uses mean-padded portraits and 69-dimensional features obtained from

the FC6 layer of the CNN. In this section, we compare this optimal CNN + SVM model against the three baseline methods described in Section 3.2.

The parameters of all four methods were optimized using a grid search, as described in Section 3.3. Optimization of the SVM hyperparameters based on F_1 score resulted in the following values for regularization factor (C) and RBF kernel bandwidth (γ): HSV + SVM (C = 180;γ = 10); CNN + SVM (C = 30; $\gamma = 10^{-4}$). For the HSV + Bh method, we performed an analogous grid search to optimize the standard deviation associated with the Gaussian kernel function, obtaining an optimal parameter of σ = 5. For the HSV method, we performed an extensive grid search on our training dataset to selected an optimal set of threshold values. This procedure indicated that pixels composing flowers are distributed over the entire H range of [0, 255], with optimal ranges of S within [0, 32], V within [139, 255], minimum size of 1200 pixels and maximum size of 45,000 pixels.

Once the optimal parameters for all the classification models were determined, we evaluated the overall performance of each method using 10-fold cross-validation. All the 133,918 samples composing the full *AppleA* dataset were combined and divided into 10 folds containing 13,391 samples each. A total of 10 iterations was performed, in which each subsample was used exactly once as validation data.

The final PR curves associated with each method are shown in Fig. 11. Table 4 provides the AUC-PR for each method along with the metrics obtained for the optimal models as determined by the F_1 score.

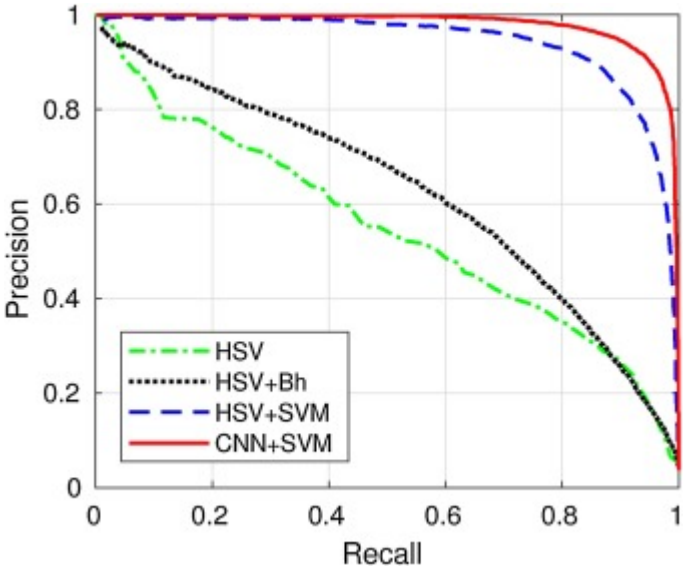

Fig. 11. Precision-recall (PR) curve illustrating the performance of our proposed approach (CNN + SVM) in comparison with the three baseline methods (HSV, HSV + Bh, and HSV + SVM).

Table 4. Summary of results obtained for our approach (CNN + SVM) and the three baseline methods (HSV, HSV + Bh and HSV + SVM). Best results in terms of F_1 and AUC-PR are shown in boldface.

	AUC-PR	**F_1**	**Recall**	**Precision**
HSV	54.9%	54.1%	58.3%	50.4%
HSV + Bh	61.6%	64.6%	56.9%	60.5%
HSV + SVM	92.9%	87.1%	88.4%	87.8%
CNN + SVM	**97.7 %**	**93.4 %**	92.0%	92.7%

The HSV method, which closely replicates existing approaches for flower detection, performs poorly in terms of both recall and precision. Such low performance is expected for methods that rely solely on color information. Since these techniques do not consider morphology or higher-level context to characterize flowers, they are very sensitive to changes in illumination and to clutter. Small performance improvements are obtained using the HSV + Bh method, which replaces pixel-wise hard thresholding by HSV histogram analysis at superpixel level, thereby incorporating a limited amount of context information into its classification decisions.

As illustrated by the results obtained with the HSV + SVM method, the use of an SVM classifier on the same HSV color features leads to dramatic improvements in both F_1 and AUC-PR ratios (around 20% and 30%, respectively). Rather than giving the same importance to all histogram regions, the SVM classifier is capable of distinguishing between poorly and highly informative features. However, as depicted in Fig. 12, the precision of this method is still compromised by gross errors such as classifying parts of tree branches as flowers, since it does not take into account any morphological information.

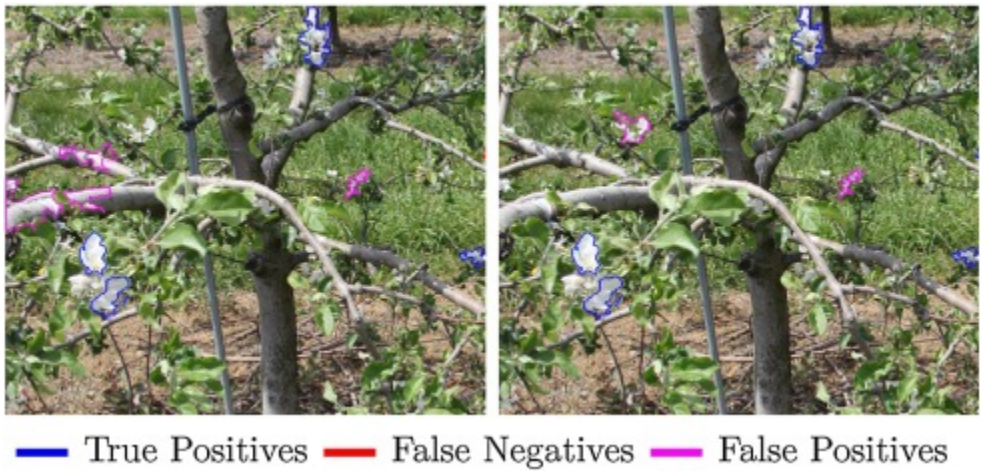

Fig. 12. Example of classification results obtained using (left) the baseline HSV + SVM method and (right) our proposed CNN + SVM method. Some examples of false positives generated by the HSV + VM method that our approach correctly classifies can be seen on the branches near the left border of the image.

The proposed approach (CNN + SVM) outperforms both baseline methods by extracting features using a convolutional neural network. Differently from the previous methods, the hierarchical features evaluated within the CNN take into account not only color but also morphological/spatial characteristics from each superpixel. Our results demonstrate the effectiveness of this approach, with significant improvements in both recall and precision ratios that culminate in an optimal F_1 score higher than 92% and AUC-PR above 97% for the evaluated dataset. Fig. 3 shows examples of the final classification yielded by this method.

4.3. Performance on additional datasets

To evaluate the generalization capability of our method, we assessed its performance on three additional datasets, composed of 20 images each and illustrated in Fig. 13. We compare the results of our method with the performance of the best performing baseline approach (HSV + SVM). No dataset-specific adjustment of parameters is performed for our method nor for the baseline (HSV + SVM), i.e., both methods are assessed with the same optimal configuration obtained for the *AppleA* dataset.

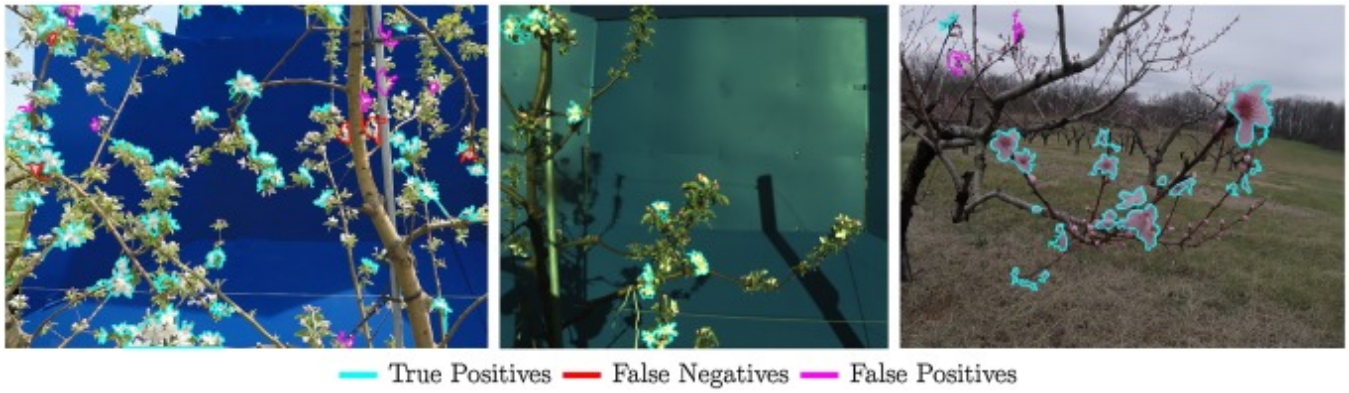

Fig. 13. Examples of images composing the additional datasets *AppleB* (left), *AppleC* (middle) and *Peach* (right), overlaid with the corresponding detections obtained by our method. (For interpretation of the references to colour in this figure citation, the reader is referred to the web version of this article.)

Two of the additional datasets also correspond to apple trees, but with a blue background panel positioned behind the trees to visually separate them from other rows of the orchard, a common practice in agricultural vision systems. We denote the first dataset *AppleB*, which is composed of images with resolution 2704 × 1520 acquired using a camera model GoPro HERO5. In this dataset there is a substantial number of occlusions between branches, leaves and flowers.

The second dataset, which we call *AppleC*, is composed of images with resolution 2456 × 2058 acquired with a camera model JAI BB-500GE. In this dataset occlusions are less frequent but the saturation color component of the images is concentrated in a much narrower range of the spectrum than in the original *AppleA* dataset. The contrast between objects such as flowers and leaves is therefore significantly lower.

The third additional dataset contains images of peach flowers (we therefore call it *Peach*) with resolution 2704 × 1520 acquired using a camera model GoPro HERO5. Peach blossoms show a noticeable pink hue in comparison to the mostly white apple flowers composing the training dataset. Additionally, images were acquired during an overcast day, such that in comparison to the training set (*AppleA*) the illumination is lower and the sky composing the background is gray instead of blue. Although the main scope of this work is on apple flower detection, we ultimately aim at a highly generalizable system that can be applied by fruit growers of different crops without the need for species-specific adjustments. In fruit orchards, each species of tree is typically constrained to specific areas. Hence, rather than differentiating between flower species, it is preferable to have a system that can distinguish between flowers and non-flower elements (e.g., leaves, branches, sky) regardless of species. Thus, this dataset represents a good evaluation of detection robustness.

Transfer learning steps. For all three additional datasets, both feature extraction and final classification were performed using the same parameters obtained by training with the *AppleA* dataset, without any dataset specific fine-tuning. Our transfer learning strategy relies solely on generic pre-processingoperations that approximate the characteristics of the previously unseen images to those of the training samples.

Our first pre-processing step consists of removing the different backgrounds of the additional datasets. Whether the background is composed of a blue panel (*AppleB* and *AppleC*) or a gray sky (*Peach*), background identification for subsequent subtraction can be performed by means of texture analysis. For each image we compute the corresponding local entropy, which is then binarized using Otsu's threshold [52] to identify low texture clusters. We then apply morphological size filtering to the binarized image and model the background as a multimodal distribution.

To model the background, we compute the RGB-mean of the *n* largest (in terms of number of pixels) low texture clusters to build a *n*-modal reference set. The likelihood that remaining low texture clusters belong to the background is estimated as the Euclidean distance between their means and the nearest reference in the RGB space. This metric allows differentiating between low texture components composing the background from the ones composing flowers, without any dataset specific color thresholding. For the *AppleB* and *Peach* datasets we adopted a bimodal distribution, where the modes correspond to the blue panel/gray sky and trunk/branches. Since the blue panel in the background of images composing the *AppleC* dataset is reflective, shadows are visible and therefore we included a third mode to automatically filter these undesired elements out. Automatically determining the number of background components is part of our future work.

Afterwards, histogram equalization and histogram matching are performed on the saturation channel of each image. While equalization aims at spreading the histogram components, histogram matching consists in approximating its distribution to the characteristic form of the training set channel distribution [23]. Finally, to mitigate the effects of illumination discrepancies, we subtract the difference between the mean of the value channel components in the input image and in the training set.

Fig. 14 shows the PR curves summarizing the performance on these datasets of our method (CNN + SVM) in comparison with the best performing baseline approach (HSV + SVM). The proposed method provides AUC-PR above 85% for all datasets, significantly outperforming the baseline method. Since the HSV + SVM method relies solely on color information, its results are acceptable only for the *AppleB* dataset, the one that most closely resembles the training dataset. Its performance is notably poor for the *Peach* set, as this species differs to a great extent from apple flowers in terms of color. A large performance difference is also evident for the *AppleC* dataset, in which flowers and leaves share more similar color components than in the training set. Table 5 shows that the proposed approach also outperforms the baseline by a large margin in terms of optimal F_1 score and the corresponding precision and recall values.

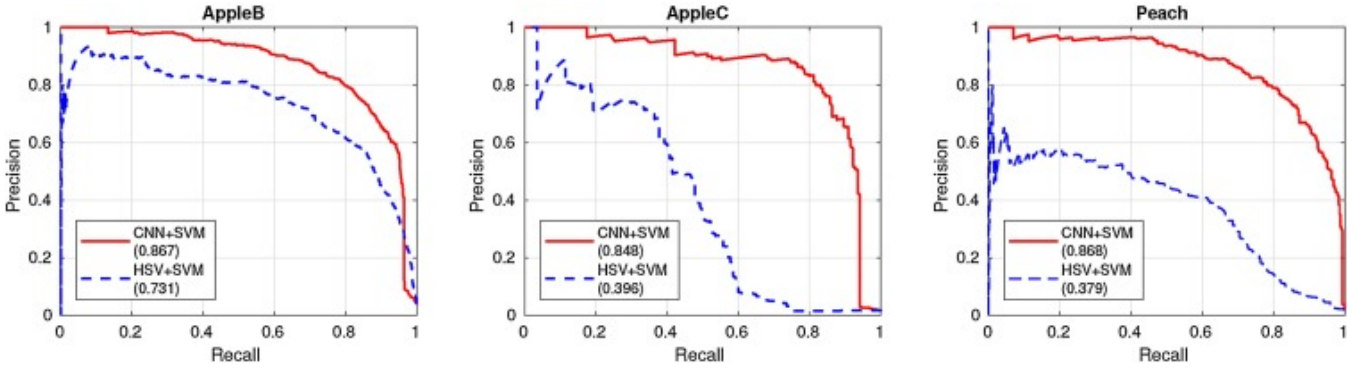

Fig. 14. PR curves expressing the performance of our method (CNN + SVM) and the optimal baseline (HSV + SVM) approach on the three additional datasets. The AUC-PR values associated with each curve are presented within parentheses.

Table 5. Summary of results obtained for our approach (CNN + SVM) and the best baseline method (HSV + SVM) for the three additional datasets. Best results in terms of F_1 are shown in boldface.

		F_1	Recall	Precision
AppleB	HSV + SVM	70.7%	69.8%	71.6%
	CNN + SVM	**80.2 %**	81.9%	78.5%
AppleC	HSV + SVM	48.6%	37.9%	68.0%
	CNN + SVM	**82.2 %**	81.2%	83.3%
Peach	HSV + SVM	49.0%	61.3%	40.8%

	CNN + SVM	**79.9 %**	81.5%	78.3%

Additionally, it is noteworthy that a large number of superpixels classified as false positives by our proposed approach (CNN + SVM) correspond to regions where flowers are indeed present, but compose less than 50% of the corresponding superpixel total area. This is illustrated in Fig. 15, which contains examples for the three additional datasets. In other words, the sensitivity of the feature extractor to the presence of flowers is very high and the final performance would be improved if the region proposals were more accurate.

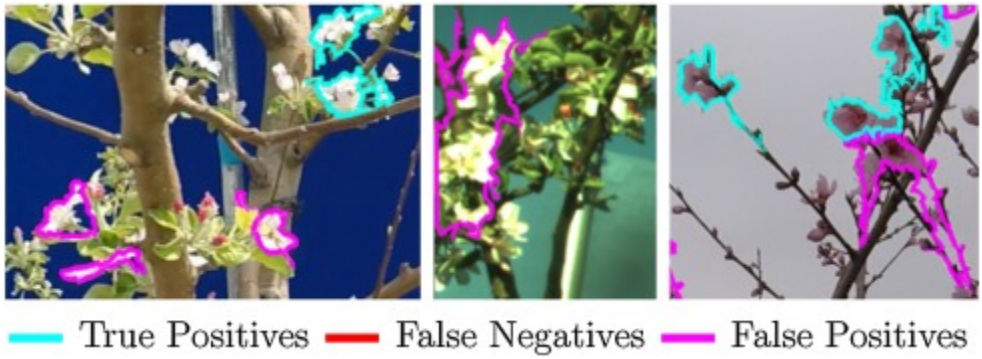

Fig. 15. Example of false positives caused by poor superpixel segmentation.

5. Conclusion

In this work, we introduced a novel approach for apple flower detection, which is based on deep learning techniques that represent the state of the art for computer vision applications. In comparison with existing methods, which are mainly based solely on color analysis and have limited applicabilityin scenarios involving changes in illumination or occlusion levels, the hierarchical features extracted by our CNN effectively combine both color and morphological information, leading to significantly better performances for all the cases under consideration. Experiments performed on four different datasets demonstrated that the proposed CNN-based model allows accurate flower identification even in scenarios of different flower species and illumination conditions, with optimal recall and precision rates near 80% even for datasets significantly dissimilar from the training sequences.

As part of our future work, we intend to explore existing datasets and state-of-the-art models for semantic image segmentation. Particularly successful strategies consist of end-to-end architectures that, without external computation of region proposals, generate pixel dense prediction maps for inputs with arbitrary size [[53], [54], [55]].

Moreover, similar to the approach proposed in [33] for fruits, we intend to extend our module for flower tracking and localization based on probabilistic approaches that use the estimated motion between frames (e.g., particle filtering [56]) to predict the location of flowers. To extend the applicability of our model to the detection of fruitlets as well as other flower species, we will consider additional transfer learning approaches such as data augmentation by affine transformations and the use of external datasets.

Conflict of interest

None declared.

Appendix A. Supplementary data

The following are the supplementary data to this article:Download video (7MB)Help with mp4 filesDownload video (7MB)Help with mp4 filesDownload video (7MB)Help with mp4 filesDownload video (3MB)Help with mp4 filesDownload video (3MB)Help with mp4 filesDownload video (3MB)Help with mp4 files

References

[1] C.G. Forshey Chemical Fruit Thinning of Apples, vol. 116, New York State Agricultural Experiment Station, Geneva, NY (1986)

[2] H. Link **Significance of flower and fruit thinning on fruit quality** Plant Growth Regul., 31 (1) (2000), pp. 17-26

[3] A. Gongal, A. Silwal, S. Amatya, M. Karkee, Q. Zhang, K. Lewis **Apple crop-load estimation with over-the-row machine vision system** Comput. Electron. Agric., 120 (2016), pp. 26-35, 10.1016/j.compag.2015.10.022

[4]S. Singh, M. Bergerman, J. Cannons, B. Grocholsky, B. Hamner, G. Holguin, L. Hull, V.Jones, G. Kantor, H. Kosel ka, G. Li, J. Owen, J. Park, W. Shi, J. Teza **Comprehensive automation for specialty crops: year 1 results and lessons learned** Intell. Serv. Robot., 3 (4) (2010), pp. 245-262, 10.1007/s11370-010-0074-3

[5] K. Kapach, E. Barnea, R. Mairon, Y. Edan, O.B. Shahar **Computer vision for fruit harvesting robots – state of the art and challenges ahead** Int. J. Comput. Vision Robot., 3 (1/2) (2012), p. 4, 10.1504/IJCVR.2012.046419

[6] A. Gongal, S. Amatya, M. Karkee, Q. Zhang, K. Lewis **Sensors and systems for fruit detection and localization: a review** Comput. Electron. Agric., 116 (2015), pp. 8-19

[7] A.D. Aggelopoulou, D. Bochtis, S. Fountas, K.C. Swain, T.A. Gemtos, G.D. Nanos **Yield prediction in apple orchards based on image processing** Precis. Agric., 12 (3) (2011), pp. 448-456, 10.1007/s11119-010-9187-0

[8] K.R. Thorp, D.A. Dierig **Color image segmentation approach to monitor flowering in lesquerella** Ind. Crops Prod., 34 (1) (2011), pp. 1150-1159, 10.1016/j.indcrop.2011.04.002

[9] M. Hočevar, B. Širok, T. Godeša, M. Stopar **Flowering estimation in apple orchards by image analysis** Precis. Agric., 15 (4) (2014), pp. 466-478, 10.1007/s11119-013-9341-6

[10] R. Horton, E. Cano, D. Bulanon, E. Fallahi **Peach flower monitoring using aerial multispectral imaging** 2016 ASABE International Meeting (2017), 10.13031/aim.20162461520

[11] C. Hung, J. Nieto, Z. Taylor, J. Underwood, S. Sukkarieh **Orchard fruit segmentation using multi-spectral feature learning** IEEE International Conference on Intelligent Robots and Systems (2013), pp. 5314-5320, 10.1109/IROS.2013.6697125

[12] J. Das, G. Cross, C. Qu, A. Makineni, P. Tokekar, Y. Mulgaonkar, V. Kumar **Devices, systems, and methods for automated monitoring enabling precision agriculture** IEEE International Conference on Automation Science and Engineering (CASE), IEEE(2015), pp. 462-469

[13] W. Ji, D. Zhao, F. Cheng, B. Xu, Y. Zhang, J. Wang **Automatic recognition vision system guided for apple harvesting robot** Comput. Electr. Eng., 38 (5) (2012), pp. 1186-1195, 10.1016/j.compeleceng.2011.11.005

[14] R. Linker, O. Cohen, A. Naor **Determination of the number of green apples in RGB images recorded in orchards** Comput. Electron. Agric., 81 (2012), pp. 45-57, 10.1016/j.compag.2011.11.007

[15] J.P. Wachs, H.I. Stern, T. Burks, V. Alchanatis **Low and high-level visual feature-based apple detection from multi-modal images** Precis. Agric., 11 (6) (2010), pp. 717-735, 10.1007/s11119-010-9198-x

[16] Q. Wang, S. Nuske, M. Bergerman, S. Singh **Automated crop yield estimation for apple orchards** Proceedings of International Symposium of Experimental Robotics (ISER) (2012), 10.1007/978-3-319-00065-7

[17] U.-O. Dorj, M. Lee, S.-s. Yun **An yield estimation in citrus orchards via fruit detection and counting using image processing** Comput. Electron. Agric., 140 (2017), pp. 103-112, 10.1016/j.compag.2017.05.019

[18] S. Bargoti, J. Underwood **Image segmentation for fruit detection and yield estimation in apple orchards** J. Field Robot., 00 (0) (2016), pp. 1-22, 10.1002/rob.21699

[19] H. Cheng, L. Damerow, Y. Sun, M. Blanke **Early yield prediction using image analysis of apple fruit and tree canopy features with neural networks** J. Imaging, 3 (1) (2017), p. 6, 10.3390/jimaging3010006

[20] G.E. Hinton **Learning multiple layers of representation** Trends Cogn. Sci., 11 (10) (2007), pp. 428-434, 10.1016/j.tics.2007.09.004

[21] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, M.S. Lew **Deep learning for visual understanding: a review** Neurocomputing, 187 (2016), pp. 27-48

[22] M.D. Zeiler, R. Fergus **Visualizing and understanding convolutional networks** European Conference on Computer Vision (ECCV), Springer (2014), pp. 818-833

[23] O. Marques **Practical Image and Video Processing using MATLAB** John Wiley & Sons (2011)

[24] N. Dalal, B. Triggs **Histograms of oriented gradients for human detection** IEEE Conference on Computer Vision and Pattern Recognition (CVPR), vol. 1 (2005), pp. 886-893, 10.1109/CVPR.2005.177

[25] Y. LeCun, Y. Bengio, G. Hinton **Deep learning** Nature, 521 (7553) (2015), pp. 436-444

[26] S.C. Wong, A. Gatt, V. Stamatescu, M.D. McDonnell **Understanding data augmentation for classification: when to warp?** 2016 International Conference on Digital Image Computing: Techniques and Applications (DICTA), IEEE (2016), pp. 1-6

[27] J. Yosinski, J. Clune, Y. Bengio, H. Lipson **How transferable are features in deep neural networks?** Advances in Neural Information Processing Systems (2014), pp. 3320-3328

[28]
Y. Tang**Deep learning using linear support vector machines**
International Conference on Machine Learning: Challenges in Representation Learning Workshop (2013)

[29] R. Girshick, J. Donahue, T. Darrell, J. Malik **Rich feature hierarchies for accurate object detection and semantic segmentation** Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (2014), pp. 580-587, 10.1109/CVPR.2014.81

[30] S. Ren, K. He, R. Girshick, J. Sun **Faster R-CNN: towards real-time object detection with region proposal networks** Nips (2015), pp. 1-10, 10.1016/j.nima.2015.05.028

[31] J.R.R. Uijlings, K.E.A. Van De Sande, T. Gevers, A.W.M. Smeulders **Selective search for object recognition** Int. J. Comput. Vision, 104 (2) (2013), pp. 154-171, 10.1007/s11263-013-0620-5

[32] S. Bargoti, J. Underwood **Deep Fruit Detection in Orchards** Australian Centre for Field Robotics (2016), pp. 1-8

[33] M. Stein, S. Bargoti, J. Underwood **Image based mango fruit detection, localisation and yield estimation using multiple view geometry** Sensors, 16 (11) (2016), p. 1915, 10.3390/s16111915

[34] I. Sa, Z. Ge, F. Dayoub, B. Upcroft, T. Perez, C. McCool **DeepFruits: a fruit detection system using deep neural networks** Sensors, 16 (8) (2016), p. 1222, 10.3390/s16081222

[35] G.L. Grinblat, L.C. Uzal, M.G. Larese, P.M. Granitto **Deep learning for plant identification using vein morphological patterns** Comput. Electron. Agric., 127 (2016), pp. 418-424, 10.1016/j.compag.2016.07.003

[36] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, S. Süsstrunk **SLIC superpixels compared to state-of-the-art superpixel methods** IEEE Trans. Pattern Anal. Mach. Intell., 34 (11) (2012), pp. 2274-2281, 10.1109/TPAMI.2012.120

[37] D. Stutz, A. Hermans, B. Leibe **Superpixels: an evaluation of the state-of-the-art** Comput. Vision Image Understand., 166 (2018), pp. 1-27

[38] R. Zhao, W. Ouyang, H. Li, X. Wang **Saliency detection by multi-context deep learning** Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 7–12 June (2015), pp. 1265-1274, 10.1109/CVPR.2015.7298731

[39] S. Visa **Issues in mining imbalanced data sets - a review paper** Proceedings of the Sixteen Midwest Artificial Intelligence and Cognitive Science Conference 2005 (2005), pp. 67-73 doi:10.1.1.87.6193
[40] A. Ben-Hur, J. Weston **A user's guide to support vector machines** Methods Mol. Biol. (Clifton, N. J.), 609 (2010), pp. 223-239, 10.1007/978-1-60327-241-4_13
[41] C.-W. Hsu, C.-C. Chang, C.-J. Lin **A practical guide to support vector classification** BJU Int., 101 (1) (2008), pp. 1396-1400, 10.1177/02632760022050997
[42] R.C. Gonzalez, R.E. Woods **Digital Image Processing** (3rd ed.), Prentice-Hall, Inc., Upper Saddle River, NJ, USA (2006)
[43] M.A. Stricker, M. Orengo **Similarity of color images** Storage and Retrieval for Image and Video Databases III, vol. 2420, International Society for Optics and Photonics (1995), pp. 381-393
[44] A. Bhattacharyya **On a measure of divergence between two statistical populations defined by their probability distributions** Bull. Calcutta Math. Soc., 35 (1943), pp. 99-109
[45] A. Hoak, H. Medeiros, R.J. Povinelli **Image-based multi-target tracking through multi-Bernoulli filtering with interactive likelihoods** Sensors, 17 (3) (2017), p. 501
[46] R. Hoseinnezhad, B.N. Vo, B.T. Vo, D. Suter **Visual tracking of numerous targets via multi-Bernoulli filtering of image data** Pattern Recogn., 45 (10) (2012), pp. 3625-3635, 10.1016/j.patcog.2012.04.004
[47] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T.Darrell **Caffe: Convolutional Architecture for Fast Feature Embedding** (2014)
[48] A. Carpenter **CuSVM: a CUDA implementation of support vector classification and regression** Update, 15 (10) (2009), pp. 1-9
[49] T. Fawcett **An introduction to ROC analysis** Pattern Recogn. Lett., 27 (8) (2006), pp. 861-874, 10.1016/j.patrec.2005.10.010
[50] M. Mohri **Foundations of Machine Learning – Book** The MIT Press (2012) ISBN 978-0-262-01825-8
[51] L.I. Smith **A tutorial on principal components analysis introduction** Statistics, 51 (2002), p. 52, 10.1080/03610928808829796
[52] N. Otsu **A threshold selection method from gray-level histograms** IEEE Trans. Syst. Man Cybern., 9 (1) (1979), pp. 62-66
[53] E. Shelhamer, J. Long, T. Darrell **Fully Convolutional Networks for Semantic Segmentation** (2016) CoRR abs/1605.06211
[54] L.C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A.L. Yuille **Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs** IEEE Trans. Pattern Anal. Mach. Intell., PP (99) (2017), p. 1, 10.1109/TPAMI.2017.2699184
[55] G. Lin, A. Milan, C. Shen, I. Reid **RefineNet: multi-path refinement networks for high-resolution semantic segmentation** IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
[56] R.J. Mozhdehi, H. Medeiros **Deep convolutional particle filter for visual tracking** International Conference on Image Processing (ICIP) (2017)

[1] Mention of trade names or commercial products in this publication is solely for the purpose of providing specific information and does not imply recommendation or endorsement by the U.S. Department of Agriculture. USDA is an equal opportunity provider and employer.