Weakly Supervised Learning Guided by Activation Mapping Applied to a Novel Citrus Pest Benchmark

Edson Bollis[1] Helio Pedrini[2] Sandra Avila[1]
[1]REasoning for COmplex Data Lab. (RECOD) [2] Visual Informatics Lab. (LIV)
Institute of Computing (IC), University of Campinas (UNICAMP)
Campinas, SP, Brazil, 13083-852

Abstract

Pests and diseases are relevant factors for production losses in agriculture and, therefore, promote a huge investment in the prevention and detection of its causative agents. In many countries, Integrated Pest Management is the most widely used process to prevent and mitigate the damages caused by pests and diseases in citrus crops. However, its results are credited by humans who visually inspect the orchards in order to identify the disease symptoms, insects and mite pests. In this context, we design a weakly supervised learning process guided by saliency maps to automatically select regions of interest in the images, significantly reducing the annotation task. In addition, we create a large citrus pest benchmark composed of positive samples (six classes of mite species) and negative samples. Experiments conducted on two large datasets demonstrate that our results are very promising for the problem of pest and disease classification in the agriculture field.

1. Introduction

Pests and diseases in orchards are dangerous to the world of agriculture and have caused significant losses. Particularly, the Greening (*Diaphorina citri*), also called Huanglongbing (HLB), — the actual most destructive disease in citrus agriculture [9] — cost $13.2 billion to Florida State between 2005 and 2016 [27]. The real losses are more significant when we consider other pests and diseases that infect the country's production, such as Citrus Variegated Chlorosis (*Xylella fastidiosa*), Citrus Canker (*Xanthomonas axonopodis*), and Citrus Leprosis (*Citrus leprosis virus*).

One way to detect and prevent these threats is the use of Integrated Pest Management (IPM). It describes how to avoid the problems and what are the rules to apply inputs before the problem occurs [24]. Usually, human inspectors walk along the orchards streets collecting samples to analyze them and reporting the results in paper sheets or mobile tools for data acquisition [24]. The inspectors examine stalks, leaves, and fruits for hours, trying to find mites and insects to quantify them. Depending on the level of the infection, when the number of dispersers (mites or insects) past from a safety limit, the IPM describes the rules to apply inputs, cuts parts of the plant, removes the whole plant or eliminates the plant and its neighborhood. The IPM is a mechanical process that can be done by machines to help small farmers to enforce its rules. In addition, as expected, when humans handle the job, the IPM process is prone to errors due to the inability or fatigue of the handlers.

It is common to see mobile technology in the field to perform a wide range of tasks, such as data acquisition, employee communication, and production management. In this scenario, employing mobile devices to detect pests and diseases would not be an additional hurdle. In fact, the use of Convolutional Neural Networks (CNNs) in mobile devices, such as MobileNets [29], NasNet-A Mobile [46], and EfficientNet [36], can greatly help inspectors in doing their work more efficiently and effectively.

As a consequence of the lack of other image collections, we created a novel dataset called Citrus Pest Benchmark (CPB). It contains images collected with mobile devices of mites in citrus plants, which is unseen in the literature. Our dataset supports the evaluation of our classification method. Unlike the IP102 [41] database for insect pest recognition, for instance, our benchmark is composed of very tiny regions of interest (mites) compared to the remaining area of the image. In this sense, the straightforward use of CNNs in our citrus pest classification problem would not be efficient. Inspired by recent approaches to cancer classification and object detection [21, 26, 32, 39], we develop a weakly supervised learning method that computes saliency maps to automatically locate patches of interest in the original images.

The main contributions of our work are: (i) creation of a new benchmark for the citrus pest recognition problem, where tiny regions of interest containing different types of mites are present in the original images; (ii) development

of a weakly supervised multiple instance learning method guided by saliency maps to automatically identify patches in the images and reduce the task of image labeling; (iii) implementation of a weighted evaluation strategy for properly generating a final probability for every extracted image patch; and (iv) achievement of promising classification results on two large pest benchmarks in the agriculture field.

This text is organized as follows. In Section 2, we briefly review some relevant concepts and approaches related to disease and pest classification and multiple instance learning. In Section 3, we describe our Citrus Pest Benchmark. In Section 4, we present our weakly supervised multiple instance learning method. We report and discuss the experimental results achieved on two datasets in Section 5. Finally, some concluding remarks and directions for future work are presented in Section 6.

2. Related Work

In this section, we first overview the literature of disease and pest classification, in particular we focus on CNN-based approaches. Then, we describe relevant aspects related to multiple instance learning and weakly supervised approaches.

2.1. Disease and Pest Classification

In the era of Convolutional Neural Networks (CNNs), the first works on disease and pest classifiers have the primary goal of improving the classification metrics on a given database. As CNNs require a large amount of training data, many approaches have focused their efforts on creating image databases for classifying pests and diseases in the field (for instance, [1, 12, 41]).

Concerning disease classification, Hughes and Salathé [12] created an image database called PlantVillage, which consists of 55,000 images (captured in laboratories) from disease symptoms in leaves. Mohanty et al. [23] used the Inception [34] and AlexNet [13] networks to train their models on the PlantVillage. Ferentinos [7] introduced a new version of the PlantVillage with 87,848 images (not publicly available) to evaluate CNNs for plant disease detection and diagnosis. They also proposed its use in mobile applications, but they did not present any experiments. The PlantVillage database was the first large-public database on disease detection area, and many works evaluated well-known CNNs with little or no modification [3, 18, 20, 25, 37].

With respect to pest classification, before 2018 few works explored CNNs. Liu et al. [19] used saliency maps constructed by a histogram. They used the color variation between the pests and backgrounds to extract paddy pests and created a database of 5,136 images. Alfarisy et al. [1] collected from Internet 4,511 images of paddy pests and classified them with a CNN.

Lee and Xing [15] created a pest tangerine database of 10 macro-insects, including the Psyllid (*Diaphorina Citri*, the greening vector) and they evaluated several CNNs on their data. Similar to ours, Li et al. [17] proposed a database in which the insects are very tiny concerning to the entire image. They applied a two-stage object detector to find groups of insects in the images and then extracted these regions to detect each insect. In contrast to our approach, we do not have object annotations, so we benefit from a weakly supervised method to classify the pests. Chen et al. [5] used the Google image search engine to collect 700 images from four pests, including Spider mites (*Tetranychidae*). They used CNNs to classify the images captured from sensors in the field, but they did not show any results related to these types of images.

The largest database for insect pest classification was introduced by Wu et al. [41]. The IP102 database consists of 102 classes and 75,222 images. The authors applied different CNNs (AlexNet, GoogleNet, VGGNet, and ResNet) to report their results. Ren et al. [28] improved the classification performance on IP102 by modifying ResNet blocks. Xu and Wang [42] used the IP102 dataset to demonstrate the use of XCloud, a cloud platform proposed to facilitate the use of AI.

In brief, agricultural works on the Machine Learning area lack of proposition on new methods. Usually, the works only apply the well-known CNNs in its databases. To the best of our knowledge, no work uses mite images collected with mobile cameras using a strict protocol directly in the field.

2.2. Multiple Instance Learning-based Approaches

Multiple instance learning (MIL) is a weakly supervised category of problems where its training data is arranged in *bag sets* and sets of patches from the bags, called *instances*. The labels are provided only for the bags and the instances inherit from the bags creating a weakly supervised environment [4].

The standard MIL assumption, in a binary problem, states that negative bags contain only negative instances and positive bags contain at least one positive instance. This assumption can be relaxed to use the evaluation of the interaction of several positive instances, as we use in this work [8].

Sun et al. [33] proposed a weakly supervised CNN framework, called Multiple Instance Learning Convolutional Neural Networks (MILCNN), which fuses residual network and multiple instances learning loss layer. The architecture received the number of instances from a bag, inferring the instances as separated images, and used a function to mix the probabilities to calculate a final probability for the entire bag in the last layer.

Choukroun et al. [6] introduced an MIL method for mammogram classification using a VGGNet followed by

a refining fully connected neural network modified to the MIL paradigm.

He et al. [11] created a Multiple Instance Deep Convolutional Network for image classification (MIDCN) based on a feature extractor from CaffeNet. They calculated the differences between feature vectors from instances and precalculated features, called prototypes, and predicted the classes using these differences. Li et al. [16] developed an attention-based CNN model for MIL, which used an adaptive attention mechanism into a CNN to detect significant instances for histopathology images.

The previously mentioned works used all instances of one bag at the same time in the training phase, as a batch of instances. For this, the researchers must adapt the original CNNs changing the first layers and the loss functions. In our proposal, we use the CNN architecture in its original version.

Out of the MIL methods, some works used the same idea of patches, however, in supervised ways as [26, 39]. It was not different for disease and pest classification, as Li et al. [17]. Unlike the other, Liu et al. [19] used a weakly supervised method based on saliency maps to cut the pest from the original images to create their dataset.

According to Zhou et al. [45], we can classify our transfer learning technique as a pseudo-label for CNNs. However, most of the pseudo-label works came from the inference of the unlabeled part of the databases for models already trained with the labeled part, for instance, the approach developed by Lee [14]. In the case of our work, we use pseudo-labels from the original bags. Tao et al. [38] and Zhang and Zeng [44] also used pseudo-labels with multiple instances in their projects, but not as our work does.

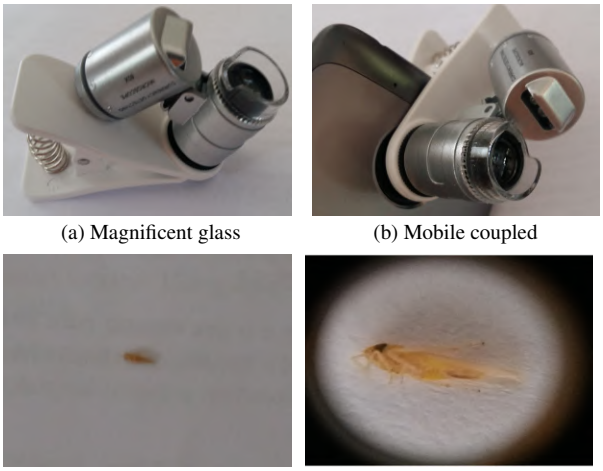
To the best of our knowledge, we have found neither MIL methods applied to disease and pest classification tasks nor MIL architectures for mobile devices, which encourages our investigation into these research topics.

3. Our Citrus Pest Benchmark

As an additional contribution of this work, we created a benchmark[1] containing images divided into seven classes (six mite species and a negative class). The images were collected via a mobile device coupled with a lens magnifier, as shown in Figure 1. In the acquisition process, we employ a Samsung Galaxy A5 with a 13 MP camera coupled with a 60× magnifier, equipped with a white LED lighting and ultraviolet LED.

The sizes of the mite species are very small in proportion to the entire image size, as illustrated in Figure 2. Due to the hard glass surface present in the device, a significant part of the images is blurred, as can be seen in Figure 3.

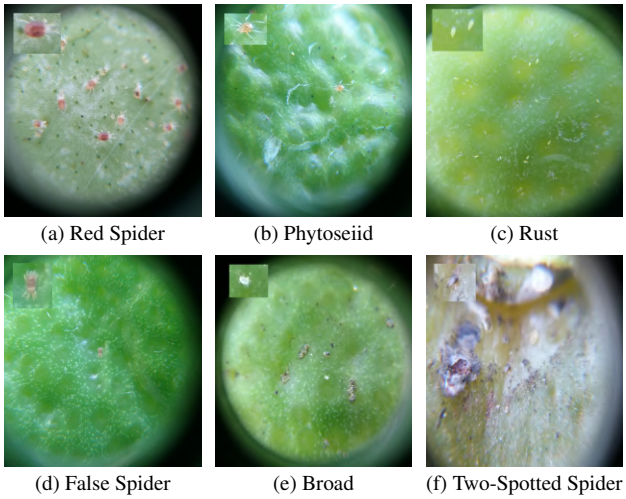
[1] https://github.com/edsonbollis/Citrus-Pest-Benchmark.



(a) Magnificent glass (b) Mobile coupled

(c) Insect with normal size (d) Insect after zoom (60×)

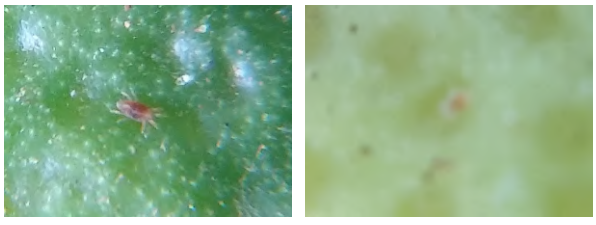
Figure 1: (a-b) Devices used to collect the citrus pest images; (c-d) insect image before and after magnification.



(a) Red Spider (b) Phytoseiid (c) Rust

(d) False Spider (e) Broad (f) Two-Spotted Spider

Figure 2: Mites captured through optical magnification of 60×. The mites are highlighted on the upper-left side of the images.



(a) Sharp image (b) Blurred image

Figure 3: Samples of False Spider mites from our Citrus Pest Benchmark.

To generate our citrus pest database, the mite images were captured at São José Farm, located in the city of Rio Claro, São Paulo State, in Brazil. The data acquisition pe-

riod was from March 2018 to January 2019. Guided by MIP inspectors, we carried out scheduled inspections in the production unit areas, which contain up to 1000 citrus trees divided into groups arranged in lines. The inspectors chose samples from the crop lines, not near the border, to analyze the fruits, new germinations and stem. Then, they moved on to the next thirtieth plant individuals. We used the samples examined by the inspectors to obtain the mite images. After completing a crop sector line, every three planting lines were examined.

Our database consists of 10,816 multi-class images categorized into seven classes: (i) 1,902 images with Red Spider mites (*Panonychus citri*, *Eutetranychus banksi*, *Tetranychus mexicanus*), the largest of all other species which produces a yellowish symptom on the leaves and fruits (Figure 2a); (ii) 1,426 images with Phytoseiid mites (*Euseius citrifolius*, *Iphiseiodes zuluagai*), the predator mite that helps control other mites (Figure 2b); (iii) 1,386 images with Rust mites (*Phyllocoptruta oleivora*), responsible for the rust symptom and significant crop losses (Figure 2c); (iv) 1,750 images with False Spider mites (*Brevipalpus phoenicis*), a vector of the Leprosis virus (Figure 2d); (v) 806 images with Broad mites (*Polyphagotarsonemus latus*), responsible for causing a white cap on the fruits (Figure 2e); (vi) 696 images with Two-Spotted Spider mites, which do not bring significant crop losses, however, they are clearly visible in the field (Figure 2f); and (vii) 3,455 negative images.

We partitioned the image collection into three groups, referred to as training, validation and test, containing approximately 60%, 20%, and 20% of the mites from each class totaling 6380, 2239 and 2197 images, respectively.

Some of the classes are very similar to each other for untrained eyes. In addition, the differences in luminosity and zoom make the database very challenging. The multi-class problem turns the tasks more interesting once we have 5% (599) of images with up to three classes simultaneously.

Although we collected the images with the aid of human inspectors, the errors inter-classes are significant due to the size of the mites. The inspectors are currently revising the multi-class labels and, for this reason, we are publishing images of 1,200×1,200 pixels for the negative and positive classes, more precisely, 7,966 mite images and 3,455 negative images.

In Table 1, we compare our benchmark to various existing databases related to the pest and disease recognition task and cited in our work.

4. Methodology

In this section, we introduce our weakly supervised approach, which is guided by saliency maps. In Section 4.1, we describe our problem within the framework of multiple instance learning (MIL). Next, in Section 4.2, we detail the proposed Patch-SaliMap, a multi-patch selection strategy based on saliency maps. Finally, in Section 4.3, we explain how to evaluate an image considering the generated patches. We depict the main stages of our pipeline in Figure 4.

4.1. Multiple Instance Learning Framework

In brief, our method consists of four steps: (1) we train a CNN (initially trained on the ImageNet) on the Citrus Pest Benchmark, (2) we automatically generate multiple patches regarding saliency maps, (3) we fine-tune our CNN model (trained on the target task) according to a multiple instance learning approach, and (4) we apply a weighted evaluation scheme to predict the image class.

As mentioned before, multiple instance learning (MIL) is a form of weakly supervised learning where training data is a set of labeled *bags* $X = \{x_i, i = 1, ..., n\}$, and each bag contains several *instances* $\overline{X} = \{x_{ij}, j = 1, ..., k\}$, where x_{ij} is part of x_i, n is the number of images, k is the number of instances, and j is the number of images from $\overline{X}$. In this context, in Step 1, the CNN model (trained on a set of labeled bags) is our Bag Model.

In Step 2, we generate patches from the bags, as detailed in Section 4.2. Our algorithm uses the saliency of the maps to identify the regions on the images where mites are highly likely to be located. In other words, we apply the algorithm in each $x_i \in X$ to generate $\{x_{ij}, j = 1, ..., k\}$ patches of x_i, with $k = 5$. Thus, we create a new instance database $\overline{X} = \{x_{ij}, i = 1, ..., n, j = 1, ..., k\}$ for MIL.

In Step 3, we assume the class label of an instance is the same of its bag (in MIL the labels are only assigned to bags). That is, if $y_i = f(x_i)$ is the label of $x_i \in X$ then $f(x_{ij}) = y_i,\ x_{ij} \in \overline{X}$. Next, we finetune the same Bag Model on the $\overline{X}$, exploring a transfer learning scheme to MIL. Since we have more mites than negative images, we use five instances of each negative bag and two instances of positive bags to balance the data and decrease the probability to miss mites on positive images.

We highlight that it is possible to use the same model with no changes for images with different sizes because there is a global pooling after the last convolutional layer for every CNN. The pooling transforms a feature map of dimension $w \times h \times c$ in a feature map of size $1 \times 1 \times c$. Therefore, we are able to reuse bag models and instances regardless of the image sizes.

In Step 4, all the models trained in $\overline{X}$ are evaluated on its subsets that contain patches of X, producing the evaluation for the bags, as described in Section 4.3. The best model evaluated in $\overline{X}$ is referred to as Instance Model and it provides a final probability for every instance and, applying the proposed Weighted Evaluation Method, for every bag.

Author	Database Name	Size	Type	Year
Hughes and Salathé [12]	PlantVillage	55,000	symptoms of diseases	2015
Barbedo et al. [2]	N/A	1,335	symptoms of diseases	2016
Nachtigall et al. [25]	N/A	2,539	symptoms of diseases	2016
Tan et al. [37]	N/A	4,000	symptoms of diseases	2016
Liu et al. [19]	Pests ID	5.136	pests	2016
Bhandari et al. [3]	N/A	N/A	symptoms of diseases	2017
Liu et al. [18]	N/A	13.689	symptoms of diseases	2018
Alfarisy et al. [1]	Paddy Pest Image	4,511	pests	2018
Lee and Xing [15]	Pest Tangerine	5,247	pests	2018
Wu et al. [41]	IP102	75,222	pests	2019
Li et al. [17]	Aphid Images	2,200	pests	2019
Chen et al. [5]	N/A	700	pests	2020
Our work	CPB	10,816	pests	2020

Table 1: Pest and disease databases. N/A means that the value was not available from the original paper.

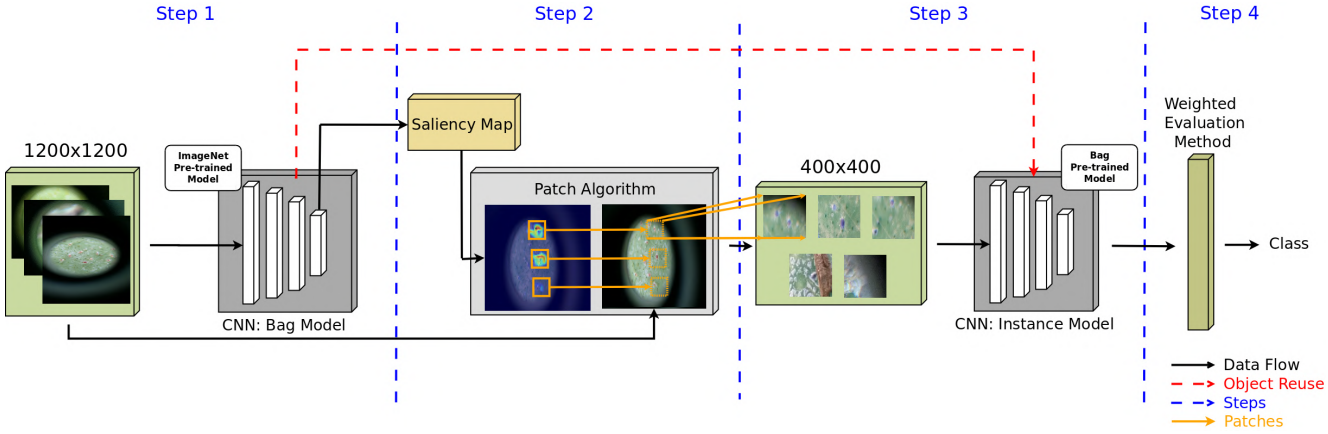


Figure 4: Our method consists of four steps. In Step 1, we train a CNN (initially trained on the ImageNet) on the Citrus Pest Benchmark. In Step 2, we automatically generate multiple patches regarding saliency maps. In Step 3, we fine-tune our CNN model (trained on the target task) according to a multiple instance learning approach. In Step 4, we apply a weighted evaluation scheme to predict the image class.

4.2. Multi-patch Selection Strategy Based on Saliency Maps

Our aim here is to learn fine-grained details since most citrus mites are not readily visible to the naked eye. We propose to select significant image patches according to the saliency map, called the *Patch-SaliMap* algorithm. In Algorithm 1, we formally describe our proposal.

Let $x_i \in X \subset \mathbb{R}^{h \times w \times 3}$ be a tensor of an image, where $h, w \in \mathbb{N}^+$ are the height and width of x_i. Let $S : \mathbb{R}^{h \times w \times 3} \rightarrow \mathbb{R}^{h \times w}$ be a saliency map function, where $S(x_i)$ is the saliency map of x_i. The *Patch-SaliMap* takes as input $x_i, S(x_i), k, l$ and produces $\{x_{ij} \in \overline{X} \subset \mathbb{R}^{l \times l \times 3}, j = 1, ..., k\}$, where $k \in \mathbb{N}^+$ is the total number of instances, $j \in \mathbb{N}^+$ is the index of instances, and $l \in \mathbb{N}^+$ is the height of a square patch.

The algorithm uses the prior knowledge that the mites are small enough to fit in images with a size smaller than the patch size, $l \times l$ pixels. As a consequence, the algorithm achieves a higher probability of obtaining instances with mites in the first patches.

Using the maximum of the saliency map matrix is an excellent choice at the inference time. However, when we are training the Instance Model, the regions of the maximum gradient for negative instances usually bring features easy to learn, which makes the algorithm addicted to find these features only. Thus, to fix it for the training set of $\overline{X}$, we produce random patches x_{ij}, where x_i has negative labels to force the Instance Model to learn more robust features.

Algorithm 1 Patch-SaliMap

Input: $x_i, S(x_i), k, l$
Output: instances

function PATCH_SALIMAP:
  $l = l/2$
  for $i := 1 : k$ **do**
    $a, b :=$ *get maximum indices from values of* $S(x_i)$
    if $a \pm l, b \pm l$ *is out of* x_i *border* **then**
      $a, b :=$ *fix a,b using* l
    # get a new patch around the indices
    new patch $:= x_i[a-l : a+l, b-l : b+l, :]$
    min $:=$ *get minimum value of* $S(x_i)$
    # occlude using saliences
    $S(x_i)[a-l : a+l, b-l : b+l, :] :=$ *min*
    instances$[i] :=$ *new patch*
return *instances*

4.3. Weighted Evaluation Method

To predict the class of bag images, we propose the Weighted Evaluation method. It uses static weights to calculate a weighted average and reports the final probabilities. Thus, given $x_i \in X, i = 1, ..., n$, its $x_{ij} \in \overline{X}, j = \{1, ..., k\}$, and the probabilities $p(.)$ from the Instance Model, the final probability $P(.)$ for each bag is expressed in Equation 1.

$$P(x_i) = \frac{\sum_{j=1}^{k}(k-j+1)p(x_{ij})}{\sum_{j=1}^{k}(k-j+1)}. \quad (1)$$

The Weighted Evaluation Method assigns a higher weight k to the first instance x_{i1}, that intuitively comes from the first saliency obtained from the Patch-SaliMap algorithm. Since the Patch-SaliMap in the first iteration achieves the highest value for the regions of the saliency map, this region has the major probability. The next saliency values are smaller than the first, so the algorithm assigns decreasing costs until the last saliency receives the weight equal to 1.

5. Results

In this section, after describing our experimental setup (Section 5.1), we report and discuss our empirical results on IP102 [41], a database for insect pest classification, and our Citrus Pest Benchmark (introduced in Section 3). In Section 5.2, we evaluate different CNNs on IP102 database. Next, in Section 5.3, we explore our proposal method on our benchmark, considering the best CNN evaluated on IP102.

5.1. Experimental Setup

We evaluated our experiments on five CNNs that are widely used in computer vision problems: Inception-v4 [35], ResNet-50 [10], NasNet-A Mobile [46], MobileNet-v2 [29], and EfficientNet-B0 [36]. We chose these networks because they cover different common features (and number of weights) presented in today's CNNs.

We trained each CNN with Stochastic Gradient Descent with AdaDelta optimizer [43], batch size of up to 128, a learning rate of 0.1, weight decay of 0.0005, and cross-entropy function on top of the softmax output as a loss function. All CNNs are pre-trained on the ImageNet [13] and then fine-tuned on the target database. We normalized the images, subtracting from the mean and dividing by the standard deviation, based on the ImageNet. For the experiments conducted on IP102, we resized all images to 224×224 pixels.

We applied in training time an automatic data augmentation to our images. All of our experiments used a zoom range between 0.6 and 1.4×, a rotation range between 0 and 360 degrees with values multiple of 15 degrees, vertical and horizontal reflection, and translation from 0 to 4 pixels along both axes.

To reduce overfitting, for IP102 database, we used dropout [31] between each of the EfficientNet-B0 modules (20%), and after every depth-wise convolution (30%). For Citrus Pest Benchmark, we also used dropout between each of the EfficientNet-B0 modules (20%), after every depth-wise convolution (40%) and before the final layer (30%).

We used the Gradient-weighted Class Activation Mapping (Grad-CAM) method [30] to extract the saliency maps.

Our models are trained on an NVIDIA RTX 5000 and an RTX 2080 Ti. We conducted all experiments using Keras/TensorFlow. Auxiliary code was developed using NumPy, Pandas and Scikit-Learn libraries. For the Grad-CAM[2] and all CNNs (except for the EfficientNet[3]), we ran the experiments using the Keras implementation.

For every setup, we used five separate training sets to reduce the effects of randomness. The code and data are available at our Github repository[4].

5.2. Results for IP102

The IP102 [41] database contains 102 classes and 75,222 images split into 45,095 training, 7,508 validation, and 22,619 test images for insect pest classification task. In addition, the database has a hierarchical structure and each sub-class is assigned with a super-class according to the type of damaged crops: field (e.g., rice, corn, wheat, beet, and alfalfa) and economic crop (e.g., mango, citrus, and vitis). All images were collected from the Internet.

We used this database to compare different CNN architectures to classify insect pests. The classification per-

[2] https://github.com/jacobgil/keras-grad-cam
[3] https://github.com/qubvel/efficientnet/blob/master/efficientnet
[4] https://github.com/edsonbollis/Weakly-Supervised-Learning-Citrus-Pest-Benchmark

formance is evaluated using the standard metrics for this database, accuracy and F1-score.

Our results for IP102 are reported in Table 2. Not surprisingly, the EfficientNet (the state of the art in CNNs) reached the best classification performance (59.8% of accuracy). However, we used its smallest version B0. That might indicate that the number reported does not represent the limit of classification performance achievable by the EfficientNet.

Regarding the number of weights (taking into account a mobile scenario), the MobileNet-v2, the smallest CNN in our experiments, reported 53.0% of accuracy, an absolute difference of 6.8% when compared to the EfficientNet performance.

CNNs	Accuracy (%)	Weights (M)
Inception-v4	48.2	41.2
ResNet-50	54.2	23.6
NasNet-A Mob.	53.4	4.4
EfficientNet-B0	**59.8**	4.1
MobileNet-v2	53.0	**2.3**

Table 2: Classification accuracy (in %) results of different CNNs on the IP102 validation set. Here, we opted for evaluating on the validation set to *not* optimize hyperparameters on the test set. Weights (in M) mean the number of weights in millions of each CNN and the highlights in bold correspond to the best results.

For reference purposes, we show in Table 3 the best results reported to date on the IP102 test set. The ResNet-50 [41] result is the best outcome achieved by the dataset creators. They also reported statistics for the benchmark, which demonstrates that it is strongly unbalanced compared to other databases. The FR-ResNet [28] approach changed the residual blocks internally, adding convolutions and reusing the initial features, since they hypothesized that the reuse of features from previous blocks improved the performance. They compared different types of convolutions in the blocks to the same number of parameters, since it is time consuming to test with many images in benchmarks as IP102. The DenseNet-121 [42] approach did not bring any information about how the authors reached the accuracy reported, neither how many times they trained the network nor if the value reported followed the database protocol. In addition, other metrics were not reported in their study, for instance, F1-score.

5.3. Results for Citrus Pest Benchmark

In this section, we evaluate our method using EfficientNet-B0. As we show in Table 4, we split the results into three parts, namely:

CNNs	Accuracy (%)	F1-Score (%)	Weights (M)
ResNet-50 [41]	49.4	40.1	23.6
FR-ResNet [28]	55.2	54.8	30.8
DenseNet-121 [42]	**61.1**	N/A	7.1
EfficientNet-B0	60.7	**59.6**	**4.1**

Table 3: Classification performance of different CNNs on the IP102 test set. Weights (in M) mean the number of weights in millions of each CNN. N/A means that the value was not available from the original paper, The highlights in bold correspond to the best results.

- **Typical**: Since EfficientNet-B0 models require input images of 224×224 pixels, we resize all images, distorting the aspect ratio to fit when needed. To highlight the mites in the convolutions, we also feed the network with the original image size of 1200×1200 pixels.
- **Baseline**: We first resize all images from 1200×1200 to 897×897 pixels. Next, we crop patches of size 299×299 pixels and we manually select the ones with mites as positive samples.
- **Our Method** (detailed in Section 4): To make the comparisons fair, we extract patches of size 400×400 from images of 1200×1200 pixels, keeping the same ratio of the number of patches per image of baseline $\left(\frac{1200 \times 1200}{400 \times 400} = \frac{897 \times 897}{299 \times 299} = 9\right)$.

EfficientNet-B0	Accuracy (%)
Typical	
No patches, 224×224 pixels	75.9
No patches, 1200×1200 pixels	81.2
Baseline	
Manually-annotated patches, 299×299 pixels	86.0
Our Method	
Automatically-generated patches, 400×400 pixels	**91.8**

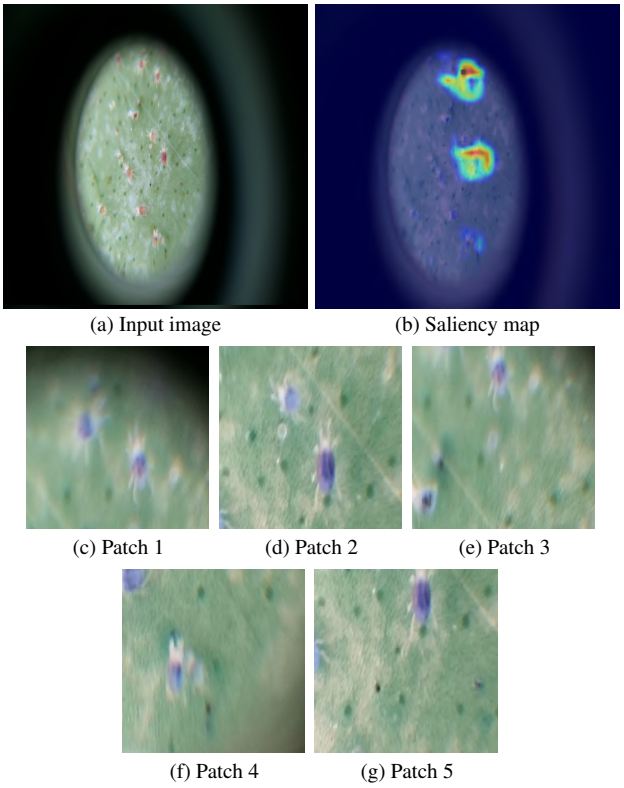
Table 4: Classification accuracy (in %) results on the Citrus Pest Benchmark validation set. Here, we opted for evaluating on the validation set to *not* optimize hyperparameters on the test set. We split the results into three parts, namely: Typical, Baseline, and Our Method. The value highlighted in bold corresponds to the best result.

The "overall picture" from Table 4 can be summarized as follows. Our Method surpasses the classification performance over all schemes. The comparison between Typical results shows that — as usually observed for image classification [22, 40] — high-resolution images lead to better performance. Baseline scenario (manually-annotated patches)

shows promising results, however, annotating patches is a tedious task, time-consuming, and error-prone. In comparison to Typical result (no patches, 1200×1200 pixels), Baseline — even using patches of size 299×299 pixels — significantly increases the classification performance, indicating that the model can benefit from patch representations. Comparing Our Method to Baseline (automatically-generated patches vs. manually-annotated patches), we observe an increase from 86.0% to 91.8%, an absolute improvement of 5.8%.

Our best model in the test set (we restricted ourselves to perform experiments in validation set) achieved an accuracy of 92.1%.

For illustration, we show in Figure 5 the automatically-generated patches guided by the saliency map. The patches are ranked according to the highest activation (from Figure 5c to 5g. The generated patches highlight the positive impact of our method.



(a) Input image (b) Saliency map

(c) Patch 1 (d) Patch 2 (e) Patch 3

(f) Patch 4 (g) Patch 5

Figure 5: Automatically-generated patches guided by the saliency map.

6. Conclusions and Future Work

In this work, we presented a new weakly supervised Multi-Instance Learning (MIL) process to classify tiny regions of interest (ROIs), a Selection Strategy Based on Saliency Maps (Patch-SaliMap), a Weighted Evaluation Method, as well as a novel database for agriculture called Citrus Pest Benchmark (CPB).

The CPB is the first database containing images acquired via mobile devices from citrus plants for pest recognition. A number of different mite species, typically invisible to the naked eye, may affect citrus leaves and fruits. The benchmark is a valuable resource for the automation of Integrated Pest Management (IPM) tasks in agriculture and for the evaluation of new classification algorithms.

From our experiments, we observed that our classification method was able to achieve superior results when compared to other approaches on the IP102 dataset. In addition, we discussed the effectiveness of our method on the CPB dataset, surpassing two different experimental scenarios. The weakly supervised multi-instance learning approach demonstrated to be effective in identifying patches of interest. The strategy for selecting the multiple patches reduced the probability of losing relevant regions, consequently improving our classification results. Overall, we believe that our method has great potential to help inspectors to classify pests and diseases through magnifying glasses and mobile devices directly in the field.

As directions for future work, we plan to further analyze the EfficientNet attention modules, so they can better operate in small areas of the images. This could reduce those patches without the occurrence of mites produced by the Patch-SaliMap. Moreover, we will investigate how small differences among the mite species would affect the multi-class task. Finally, we intend to deploy our CNN-based learning process on mobile devices.

Acknowledgments

E. Bollis is partially funded by CAPES (88882.329130/2019-01). H. Pedrini is partially funded by FAPESP (2014/12236-1, 2017/12646-3) and CNPq (309330/2018-1). S. Avila is partially funded by FAPESP (2013/08293-7, 2017/16246-0) and Google Research Awards for Latin America 2019. RECOD Lab. is partially supported by diverse projects and grants from FAPESP, CNPq, and CAPES. We gratefully acknowledge the donation of GPUs by NVIDIA Corporation.

References

[1] A. A. Alfarisy, Q. Chen, and M. Guo. Deep learning based classification for paddy pests & diseases recognition. In *International Conference on Mathematics and Artificial Intelligence*, pages 21–25, 2018. 2, 5

[2] J. G. A. Barbedo, L. V. Koenigkan, and T. T. Santos. Identifying multiple plant diseases using digital image processing. *Biosystems Engineering*, 147:104–116, 2016. 5

[3] S. Bhandari, A. Raheja, R. L. Green, and D. Do. Towards collaboration between unmanned aerial and ground vehicles

for precision agriculture. In *Autonomous Air and Ground Sensing Systems for Agricultural Optimization and Phenotyping II*, volume 10218, 2017. 2, 5

[4] M.-A. Carbonneau, V. Cheplygina, E. Granger, and G. Gagnon. Multiple instance learning: A survey of problem characteristics and applications. *Pattern Recognition*, 77:329–353, 2018. 2

[5] C. J. Chen, J. S. Wu, C. Y. Chang, and Y.-M. Huang. Agricultural pests damage detection using deep learning. In *International Conference on Network-Based Information Systems*, pages 545–554, 2020. 2, 5

[6] Y. Choukroun, R. Bakalo, R. Ben-Ari, A. Akselrod-Ballin, E. Barkan, and P. Kisilev. Mammogram classification and abnormality detection from nonlocal labels using deep multiple instance neural network. In *Eurographics Workshop on Visual Computing for Biology and Medicine*, pages 11–19, 2017. 2

[7] K. P. Ferentinos. Deep learning models for plant disease detection and diagnosis. *Computers and Electronics in Agriculture*, 145:311–318, 2018. 2

[8] J. Foulds and E. Frank. A review of multi-instance learning assumptions. *The Knowledge Engineering Review*, 25(1):1–25, 2010. 2

[9] E. Grafton-Cardwell. Huanglongbing (HLB or Citrus Greening). http://cisr.ucr.edu/citrus_greening.html, 2018. Accessed: 25-06-2020. 1

[10] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 6

[11] K. He, J. Huo, Y. Shi, Y. Gao, and D. Shen. Midcn: A multiple instance deep convolutional network for image classification. In *Pacific Rim International Conference on Artificial Intelligence*, pages 230–243. Springer, 2019. 3

[12] D. Hughes and M. Salathé. An open access repository of images on plant health to enable the development of mobile disease diagnostics. *arXiv:1511.08060*, pages 1–13, 2015. 2, 5

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012. 2, 6

[14] D.-H. Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on Challenges in Representation Learning*, volume 3, page 2, 2013. 3

[15] M. Lee and S. Xing. A study of tangerine pest recognition using advanced deep learning methods, 2018. URL https://doi.org/10.20944/preprints201811.0161.v1. 2, 5

[16] M. Li, L. Wu, A. Wiliem, K. Zhao, T. g, and B. Lovell. Deep instance-level hard negative mining model for histopathology images. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 514–522, 2019. 3

[17] R. Li, R. Wang, C. Xie, L. Liu, J. Zhang, F. Wang, and W. Liu. A coarse-to-fine network for aphid recognition and detection in the field. *Biosystems Engineering*, 187:39–52, 2019. 2, 3, 5

[18] B. Liu, Y. Zhang, D. He, and Y. Li. Identification of apple leaf diseases based on deep convolutional neural networks. *Symmetry*, 10(1):11, 2018. 2, 5

[19] Z. Liu, J. Gao, G. Yang, H. Zhang, and Y. He. Localization and classification of paddy field pests using a saliency map and deep convolutional neural network. *Scientific Reports*, 6:20410, 2016. 2, 3, 5

[20] J. Ma, K. Du, F. Zheng, L. Zhang, Z. Gong, and Z. Sun. A recognition method for cucumber diseases using leaf symptom images based on deep convolutional neural network. *Computers and Electronics in Agriculture*, 154:18–24, 2018. 2

[21] G. Maicas, G. Snaauw, A. P. Bradley, I. Reid, and G. Carneiro. Model agnostic saliency for weakly supervised lesion detection from breast DCE-MRI. In *International Symposium on Biomedical Imaging*, pages 1057–1060, 2019. 1

[22] J. Mendoza and H. Pedrini. Detection and classification of lung nodules in chest X-ray images using deep convolutional neural networks. *Computational Intelligence*, 36(2):370–401, May 2020. 7

[23] S. P. Mohanty, D. P. Hughes, and M. Salathé. Using deep learning for image-based plant disease detection. *Frontiers in Plant Science*, 7:1419, 2016. 2

[24] K. Morgan, U. Albrecht, F. Alferez, O. Batuman, et al. Florida citrus production guide. Technical report, Institute of Food and Agricultural Sciences, University of Florida, 2020. 1

[25] L. G. Nachtigall, R. M. Araujo, and G. R. Nachtigall. Classification of apple tree disorders using convolutional neural networks. In *IEEE International Conference on Tools with Artificial Intelligence*, pages 472–476, 2016. 2, 5

[26] J. Pang, C. Li, J. Shi, Z. Xu, and H. Feng. R2-CNN: Fast Tiny Object Detection in Large-Scale Remote Sensing Images. *IEEE Transactions on Geoscience and Remote Sensing*, 57(8):5512–5524, 2019. 1, 3

[27] M. Rahmani. Economic contributions of the Florida citrus industry in 2015-16, 2017. https://www.floridacitrus.org. 1

[28] F. Ren, W. Liu, and G. Wu. Feature reuse residual networks for insect pest recognition. *IEEE Access*, 7:122758–122768, 2019. 2, 7

[29] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen. MobileNetV2: Inverted Residuals and Linear Bottlenecks. *arXiv:1801.04381*, pages 1–11, 2018. 1, 6

[30] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *IEEE International Conference on Computer Vision*, pages 618–626, 2017. 6

[31] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014. 6

[32] P. Sudharshan, C. Petitjean, F. Spanhol, L. E. Oliveira, L. Heutte, and P. Honeine. Multiple instance learning for histopathological breast cancer image classification. *Expert Systems with Applications*, 117:103–111, 2019. 1

[33] M. Sun, T. X. Han, M.-C. Liu, and A. Khodayari-Rostamabad. Multiple instance learning convolutional neural networks for object recognition. In *International Conference on Pattern Recognition*, pages 3270–3275, 2016. 2

[34] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, et al. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015. 2

[35] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi. Inception-v4, Inception-ResNet and the impact of residual connections on learning. In *Thirty-First AAAI Conference on Artificial Intelligence*, pages 4278–4284, 2017. 6

[36] M. Tan and Q. V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv preprint arXiv:1905.11946*, 2019. 1, 6

[37] W. Tan, C. Zhao, and H. Wu. Intelligent alerting for fruit-melon lesion image based on momentum deep learning. *Multimedia Tools and Applications*, 75(24):16741–16761, 2016. 2, 5

[38] Q. Tao, H. Yang, and J. Cai. Zero-annotation object detection with web knowledge transfer. In *European Conference on Computer Vision*, pages 369–384, 2018. 3

[39] F. O. Unel, B. Ozkalayci, and C. Cigla. The power of tiling for small object detection. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–10, 2019. 1, 3

[40] E. Valle, M. Fornaciali, A. Menegola, J. Tavares, F. V. Bittencourt, L. T. Li, and S. Avila. Data, depth, and design: Learning reliable models for skin lesion analysis. *Neurocomputing*, 383:303–313, 2020. 7

[41] X. Wu, C. Zhan, Y.-K. Lai, M.-M. Cheng, and J. Yang. IP102: A Large-Scale Benchmark Dataset for Insect Pest Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 8787–8796, 2019. 1, 2, 5, 6, 7

[42] L. Xu and Y. Wang. XCloud: Design and Implementation of AI Cloud Platform with RESTful API Service. *arXiv:1912.10344*, 2019. 2, 7

[43] M. D. Zeiler. Adadelta: An adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, pages 1–6, 2012. 6

[44] M. Zhang and B. Zeng. A progressive learning framework based on single-instance annotation for weakly supervised object detection. *Computer Vision and Image Understanding*, page 102903, 2020. 3

[45] H.-Y. Zhou, A. Oliver, J. Wu, and Y. Zheng. When semi-supervised learning meets transfer learning: Training strategies, models and datasets. *arXiv:1812.05313*, pages 1–11, 2018. 3

[46] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le. Learning transferable architectures for scalable image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 8697–8710, 2018. 1, 6