Unsupervised adversarial deep domain adaptation method for potato defects classification

Sofia Marino, Pierre Beauseroy, André Smolarz

Unsupervised Adversarial Deep Domain Adaptation Method for Potato Defects Classification

Sofia Marino, Pierre Beauseroy and André Smolarz

Institut Charles Delaunay/M2S, FRE 2019, University of Technology of Troyes, 12, rue Marie Curie CS 42060 - 10004, Troyes Cedex, France

Abstract

In the last few years, deep learning methods have been proposed to automate the quality control of various agricultural products. Despite the excellent results obtained, one of their main drawbacks is the need for a large annotated dataset to obtain satisfactory performance. Building this dataset is time-consuming and tedious. Moreover, in some real-world applications, vision systems can be modified over time. Such changes may generate a drop in network performance trained with the initial database (source images). To avoid the creation of a new labeled database every time a change in data distribution occurs (target images), several domain adaptation methods have been proposed. In this article, we introduce an unsupervised deep domain adaptation method based on adversarial training. A large dataset is used, including six classes of potatoes: healthy, damaged, greening, black dot, common scab, and black scurf. Two different scenarios of domain adaptation problem are considered. Firstly, a simply modification of the image acquisition system is simulated by artificially increasing the brightness of some white potatoes images (target images). Secondly, a significantly different dataset including red potatoes is introduced. In this setting, white potatoes are used as source images and red tubers as target images. We propose to train a target classifier using a pseudo-label loss, due to the unavailability of target annotations. Experimental results show that a domain adaptation method is mandatory, going from an average F1-score of 0.46 without adaptation, to 0.84 by applying our method. Finally, a comparative analysis is achieved showing that adversarial-based unsupervised domain adaptation methods outperform discrepancy-based approaches.

Keywords: Unsupervised domain adaptation, convolutional neural networks, adversarial training, disease detection, agricultural applications

1. Introduction

Potato is a well-known tuberous crop with a world production exceeding 374 million tons [1]. The surface of this tuber is often affected by a variety of defects that impact its quality and selling value. Precise quality control is essential to define not only the correct sale price but also the market to which the crop will be oriented. In most industries, quality

control is still performed manually, where one or more operators observe the potatoes to classify them according to defects. This manual task has several disadvantages: it is laborious, subjective and time-consuming, which leads to sorting errors that could be avoided. Therefore, several methods to automate the quality control of agricultural products have been proposed. The first works were focused on computer vision systems, which were mainly based on hand-crafted features [2, 3, 4, 5, 6, 7]. Despite the good results found, these methods require human expertise to define the features to extract. Normally, these features are adapted to each particular problem and they lack of generalization.

In recent years, methods based on deep neural networks have been proposed for quality control in agriculture [8, 9, 10, 11, 12]. Most of these methods have proven to be more accurate than traditional methods based on hand-crafted features. However, they have a major drawback: they need a large labeled database to obtain satisfactory results. Also, even if the labeled database is built, the model will perform well only with samples generated from the same distribution of training samples. In real-world applications, it may be necessary to make changes to the system initially used for collecting the training dataset, e.g. changes in the illumination, image quality and/or pose. These variations, known as a domain shift problem [13], would produce a drop in performance if the model trained with the initial images (*source* images) is used to predict the classes of the new images (*target* images). The construction of a labeled database with the target images is very laborious and sometimes impossible. Thus, different deep domain adaptation methods have been proposed. The main idea is to leverage the available annotated source images to be able to construct a new model with little or no labeled target data. In the latter case, where fully unlabeled target data is available, unsupervised domain adaptation algorithms (UDA) are applied.

There are three main approaches to solve the problem of domain variation through deep UDA methods [14]: (a) the discrepancy-based approaches, which normally seek to align the shift of statistical distribution between the source and target domains using different techniques. Some of the most known techniques are maximum mean discrepancy (MMD) [15, 16, 17, 18], and correlation alignment (Deep CORAL) [19]. (b) The adversarial-based approaches, which train a discriminator network to differentiate samples coming from source or target domains, and a generator network that tries to fool the discriminator. The main objective of the adversarial training is to learn source and target features that are indistinguishable. Generative models [20, 21, 22] and discriminative models [23, 24, 25, 26, 27] are included in these approaches. (c) The reconstruction-based approaches, which use the reconstruction of source or target samples to create domain-invariant features [28, 29, 30].

The UDA methods found in the literature are normally applied to specific datasets, such as office-31 [31] and digits [32, 33]. In this paper, we propose to address the domain shift problem applied to potato quality control. This problem can occur when modifications are made to image acquisition systems, or when a new product variety has to be analyzed, which is common in real-life applications. The main contributions are as follows:

- A large labeled data set is used with potato images, including six classes: healthy, damaged, greening, black dot, common scab, and black scurf. We evaluate two domain change scenarios. In the first case, we simulate the target domain by artificially increase the brightness of some white potato images. In the second case, we use white and red tubers as source and target samples respectively.

- An adversarial unsupervised domain adaptation method is proposed to tackle the problem of domain shift. A pseudo-label loss function is proposed to improve the performance of the target classifier.

- A comparison of the proposed method with discrepancy-based and adversarial-based methods is carried out.

The paper is structured as follows: Section 2 presents a summary of the related work. In Section 3 a detailed explanation of the proposed method is given. Discussion and results are exposed in Section 4. Finally, the paper is concluded in Section 5.

2. Related Works

Unsupervised domain adaptation has been intensively studied in the last few years. Authors in [15] trained a convolutional neural network (CNN) combining a classification loss on source images and a maximum mean discrepancy metric (MMD) to minimize the distance between source and target domains. The output of a fully-connected layer, called "bottleneck adaptation layer" was used to obtain the source and target representations. The empirical approximation of the MMD metric is normally calculated as follows:

$$\hat{\mathrm{MMD}}^2(X_s, X_t) = \left\| \frac{1}{n_s} \sum_{i=1}^{n_s} \phi(x_s^{(i)}) - \frac{1}{n_t} \sum_{j=1}^{n_t} \phi(x_t^{(j)}) \right\|^2 \tag{1}$$

where $\|\cdot\|$ is the L_2 norm, n_s the number of source samples, n_t the number of target samples, $\phi(\cdot)$ is a mapping function, $x_s \in X_S$ and $x_t \in X_T$ the source and target data points from distributions $D_s(X_s)$ and $D_t(X_t)$ respectively. From Eq. 1 we can notice that if distributions of both domains are similar, MMD will be small, and large if distributions are different. Instead of applying MMD to a single layer, authors in [16] proposed a Deep Adaptation Network (DAN) architecture, where they applied a multi-layer adaptation based on a multiple kernel variant of MMD, aka MK-MMD. A loss of transferability normally occurs in the deepest layers of CNN. That is why the authors proposed minimizing the MK-MDD in the last three fully-connected layers. While great results were found, a shared-classifier assumption was made, in which the source classifier was applied directly to the target images. This strong assumption may not be true in several real applications. To overcome this issue, authors in [17] and in [18] proposed to take into account possible changes in the conditional distributions. In the first work, they used the MMD metric to match distributions

and performed a target classifier adaptation by a residual transfer module. In the second work, they proposed a joint adaptation network (JAN) to reduce the shift in joint distributions of multiple task-specific layers. The adaptation was achieved by a joint maximum mean discrepancy (JMMD) penalty that computes the discrepancy between source and target kernel embedding of empirical joint distributions.

Previous methods were mainly based on the MMD metric to bridge the distribution difference between domains. In contrast, authors in [19] applied a nonlinear transformation to align the second-order statistics of source and target distributions (Deep CORAL). Combining a source classification loss and a CORAL loss, the CNN was trained end-to-end, where the final learned features were supposed to perform well on both source and target domains.

Despite the great results that were found with the mentioned discrepancy-based approaches, several methods started to apply the principle of the generative adversarial network (GAN) [34] to unsupervised domain adaptation. A coupled generative adversarial network (CoGAN) was proposed by authors in [20]. They used two GANs to generate images from source and target domains. The networks were trained with tied weights in the first and last few layers with the aim of learning a domain-invariant feature space, see Figure 1. In the specific application of Unsupervised Domain Adaptation (UDA), a softmax layer was attached to the last hidden layer of the source discriminator network. The CoGAN was trained to jointly solve the classification problem using source labeled samples, and the generative problem using source and target samples. By doing this, the authors showed that the networks were capable of learning the joint distribution of images without supervision. Instead of generating images, authors in [23] introduced a domain

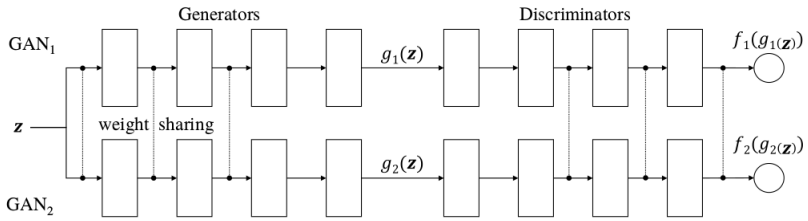


Figure 1: CoGAN architecture [20].

adversarial neural network (DANN) that was formed by a feature extractor, a label classifier, and a domain classifier. The adversarial training was achieved by training the feature extractor to fool the domain classifier and to minimize the classification loss of the label classifier. Simultaneously, the domain classifier was trained to minimize the domain classification loss. Therefore, the feature extractor was able to learn discriminative and domain-invariant features. Instead of using the same feature extractor for source and target samples, authors in [24] proposed an adversarial discriminative domain adaptation (ADDA) method. The model was composed of a fixed pre-trained source CNN, a target CNN, and a discriminator network. The target CNN was initialized by the pre-trained source CNN. Then, the discriminator was trained to correctly classify the domain of samples, while the target CNN's objective was to

maximize the loss of domain classification. The fact of using untying parameters of source and target CNNs allowed the learning of domain-specific features. However, they assumed that a pre-trained source classifier could be applied directly to target images, which is a strong assumption in some cases [35].

3. Materials and Methods

In this Section, we first present our datasets. Then, a detailed explanation of the proposed method is given.

3.1. Datasets

Two datasets were created and used. The first set (*DB1*) aimed to evaluate the relevance of the unsupervised domain adaptation approach in the context of a limited change simulated by a modification in image brightness. The second set, called *DB2*, was used to demonstrate the relevance of the proposed method in a situation where a significantly different new variety of potatoes needs to be analyzed (red tubers).

DB1. A vision system was developed to take images from potatoes automatically. A led panel and a digital camera were used. A dataset of 9688 images was created, including white potatoes from different varieties and divided into six classes: healthy, damaged, greening, black dot, common scab, and black scurf. Subsequently, the dataset was divided into two sets: 70% of images were used to create the training set, and the rest was used to test models. The training set was divided into source images and target images. As for the test set, all images were from the target domain. To obtain the target domain, we artificially increased the brightness of raw images as follows:

$$I_{new} = I_b \times (1 - \alpha_b) + I \times \alpha_b \tag{2}$$

where I is the original image, I_b is a black image of the same size as I, I_{new} is the modified image and α_b is a factor that was set to 1.5.

In Table 1 we show the number of images per class. An example of the dataset images is presented in Figure 2, where the same image is shown as source and target to observe the difference between domains. When training sets were generated, the same image could merely belong to one domain.

DB2. The vision system described above was used to take 8016 images from red tubers from the six different classes: healthy, damaged, greening, black dot, common scab, and black scurf. 80% of images were used to form the training target set and the rest was used to create the test target set. White potatoes were used as the source labeled samples.

Table 1: Number of images in the dataset *DB1* (white potatoes).

Class	No. of images	No. of training source images	No. of training target images	No. of testing target images
Healthy	5325	2989	747	1586
Damaged	984	535	134	315
Greening	1263	700	175	391
Black dot	597	344	86	167
Common scab	1276	733	183	360
Black scurf	243	139	35	69
Total	9688	5440	1360	2888

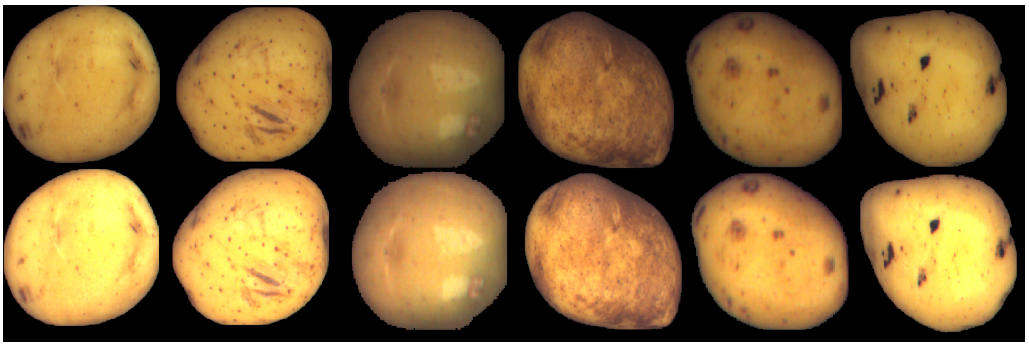


Figure 2: Images from *DB1* dataset. Domain by rows: source and target. Classes by columns: healthy, damaged, greening, black dot, common scab, and black scurf.

Table 2 show the number of images per class and per domain. Images belonging to each domain can be seen in the Figure 3. As we can observe, white and red varieties are significantly different.

Table 2: Number of images in the dataset *DB2* (white and red potatoes).

Class	No. of images	No. of training source images	No. of training target images	No. of testing target images
Healthy	7850	3736	3274	840
Damaged	1420	669	612	139
Greening	2030	875	931	224
Black dot	958	430	427	101
Common scab	2182	918	1024	240
Black scurf	376	172	156	48
Total	14816	6800	6424	1592

3.2. Unsupervised domain adaptation

The main objective of domain adaptation methods is to alleviate domain change when transferring knowledge from a source domain to a target domain. When we have access to source labels, but no label information is available for the target, we are in a particular case called unsupervised domain adaptation. We are given n_s labeled source data belonging to K classes defined by $D_s = \{(x_s^{(i)}, y_s^{(i)})\}_{i=1}^{n_s}$, where $\mathbf{x}_s \in \mathbf{X}_s$ and $\mathbf{y}_s \in \mathbf{Y}_s = \{1, 2, \ldots, K\}$. Also, n_t unlabeled target data defined by $D_t = \{x_t^{(j)}\}_{j=1}^{n_t}$, where $\mathbf{x}_t \in \mathbf{X}_t$. In a covariate shift scenario [36], we assume that $\mathbf{X}_s$ and $\mathbf{X}_t$

Figure 3: Images from *DB2* dataset. Domain by rows: source and target. Classes by columns: healthy, damaged, greening, black dot, common scab, and black scurf.

are related but different, and target task $\mathbf{Y}_t$ is assumed to be the same as source task $\mathbf{Y}_s$. Our main objective is to adapt the deep neural network used to classify source samples so that it can label the new target samples. From an operational point of view and to guarantee a similar performance, it is important to keep the model architecture that works correctly on the source database unchanged. In this way, when a shift that modifies the source images occurs (e.g. the lighting conditions), the adaptation of the network is straightforward, and putting it into production can be done quickly.

3.3. Proposed method

The overall scheme of the proposed method is presented in Figure 4. Firstly, following the work of authors in [24], a source fully convolutional network (FCN_s) and a source classifier (C_s) are trained on labeled source images. Secondly, adversarial training is applied to align the target distribution to that of the source. A pseudo-label loss is also applied to train the target classifier (C_t). In this phase, parameters of the FCN_s are frozen. Finally, the target fully convolutional network (FCN_t) and the target classifier (C_t) are used to perform the inference on the target dataset. A detailed explanation of each step is given in the following sections.

3.4. Training on source labeled images

In the first stage, we use source images $\mathbf{X}_s$ and their labels $\mathbf{Y}_s$ to train the FCN_s and the C_s in a supervised manner. A pre-trained GoogLeNet [37] is used as the FCN_s in which we remove the last fully-connected layer. The network is initially trained on ImageNet [38] to classify images between 1000 classes. The C_s consists of a fully-connected layer of K units, where K is the number of categories, and a softmax activation function. The whole network is trained by minimizing the cross-entropy loss of Equation 3. In the case of adopting the trained model to the target images, a reduction in performance will be observed due to the discrepancy between the source and the target. Therefore, it is mandatory to adapt the model to minimize the distance between the two domains. The adapted model must correctly classify the new target images without using their labels during training.

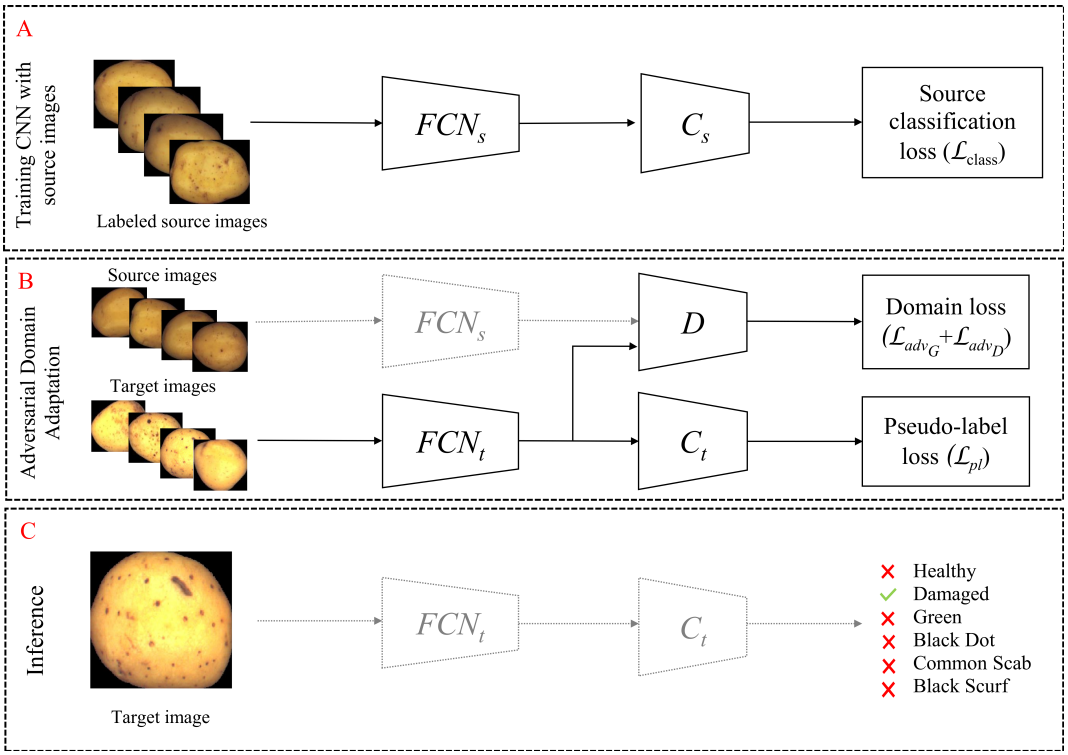


Figure 4: Scheme of the proposed method. Dashed grey lines indicate fixed parameters.

$$\mathcal{L}_{class} = -\mathbb{E}_{(\mathbf{x}_s,\mathbf{y}_s)\sim(\mathbf{X}_s,\mathbf{Y}_s)} \sum_{k=1}^{K} \mathbb{1}_{(k=\mathbf{y}_s)} \log(C_s(FCN_s(\mathbf{x}_s))) \tag{3}$$

3.5. Adversarial training

This phase aims to align the target distribution to that of the source. After training the FCN_s, the learned parameters are frozen, which maintains the performance of the source data unchanged. Parameters of the target network, FCN_t and C_t, are initialized with the trained source network. Comparable to GANs [34], the output of the FCN_s can be seen as the real images and the FCN_t can be seen as the generator of fake images. We train the FCN_t in an adversarial manner by using a domain classifier D. Source and target images are passed to the FCN_s and FCN_t respectively. Features extracted of both networks are used as input of D, which is trained to indicate if features come from source or target. On the other hand, FCN_t is trained to fool D, resulting in a minimax two-player game as follows:

$$\begin{aligned} \max_{FCN_t} \min_{D} \mathcal{L}_{adv_D} = &-\mathbb{E}_{\mathbf{x}_s\sim\mathbf{X}_s}[\log(D(FCN_s(\mathbf{x}_s)))] \\ &-\mathbb{E}_{\mathbf{x}_t\sim\mathbf{X}_t}[\log(1 - D(FCN_t(\mathbf{x}_t)))] \end{aligned} \tag{4}$$

In practice, when training the FCN_t as shown in Eq. 4, $-\log(1 - D(FCN_t(\mathbf{x}_t)))$ saturates at the beginning of training because the discriminator can easily recognize the different domains. Following the work of authors in [34], we avoid the saturation problem by training the FCN_t to minimize $-\log(D(FCN_t(\mathbf{x}_t)))$, instead of maximizing $-\log(1 - D(FCN_t(\mathbf{x}_t)))$. By doing this, stronger gradients are provided and the optimization can be divided into two objectives, where the discriminator D is trained by minimizing $\mathcal{L}_{adv_D}$, and the FCN_t by minimizing the following loss function:

$$\mathcal{L}_{adv_G} = -\mathbb{E}_{\mathbf{x}_t \sim \mathbf{X}_t}[\log(D(FCN_t(\mathbf{x}_t)))] \tag{5}$$

In this phase, we use untied weights between the FCN_s and FCN_t, which allows the networks to learn features that are specific to each domain.

Until now, the target classifier is not trained. Authors in [35] demonstrated that even if the source and target domains are aligned, it is possible that the source classifier does not work properly in the target domain. Because target labels are unavailable, it is not straightforward to train C_t. This is why we propose to assign pseudo-labels to the target samples and train the target classifier by minimizing the following pseudo-label loss:

$$\mathcal{L}_{pl} = -\mathbb{E}_{(\mathbf{x}_t, \hat{\mathbf{y}}_t) \sim (\mathbf{X}_t, \hat{\mathbf{Y}}_t)} \sum_{k=1}^{K} \mathbb{1}_{(k=\hat{\mathbf{y}}_t)} \log(C_t(FCN_t(\mathbf{x}_t))) \tag{6}$$

The target network is used to compute the pseudo-labels $\hat{\mathbf{y}}_t$ of target samples $\mathbf{x}_t$. However, only samples with a pseudo-label probability exceeding a confidence threshold are taken into account for computing the pseudo-label loss. In this way, we assume that if the network is uncertain of the assigned class, the prediction may not be correct. While training, the number of target samples that have a pseudo-label probability higher than the confidence threshold (set at 0.9 in experiences) will increase. This is because the target fully convolutional network (FCN_t) learns a more appropriate representation.

In summary, this stage is based on iteratively minimizing the following loss function:

$$\mathcal{L}_{total} = \mathcal{L}_{adv_D} + \mathcal{L}_{adv_G} + \lambda \mathcal{L}_{pl} \tag{7}$$

where λ is the hyper-parameter that balances the pseudo-label loss. Once training is accomplished, we use FCN_t and C_t to perform inference on the target samples.

3.6. Evaluation Metrics

When working with an unbalanced database, the accuracy of the classification is not an adequate evaluation measure [39]. That is why the following metrics were chosen to evaluate and compare different unsupervised domain adaptation methods:

- $\text{Precision}_\text{k}$:

$$\text{P}_\text{k} = \frac{\text{TP}_\text{k}}{\text{TP}_\text{k} + \text{FP}_\text{k}} \tag{8}$$

- Recall_k:

$$\text{R}_\text{k} = \frac{\text{TP}_\text{k}}{\text{TP}_\text{k} + \text{FN}_\text{k}} \tag{9}$$

- F1-score_k:

$$\text{F1-score}_\text{k} = 2 \times \frac{\text{P}_\text{k} \times \text{R}_\text{k}}{\text{P}_\text{k} + \text{R}_\text{k}} \tag{10}$$

  where TP_k is the true positives of class k, FP_k is the false positives of class k and FN_k is the false negatives of class k.

- Confusion Matrix: compare the output classes (columns) with the ground-truth expert label (rows). This matrix allows us to identify the types of confusion that occur between the different classes.

Comparison methods. We compared our method with three unsupervised domain adaptation approaches: Deep Correlation Alignment (Deep CORAL) [19], Joint Adaptation Networks (JAN) [18] and Adversarial Discriminative Domain Adaptation (ADDA) [24]. All these methods satisfy the requirement that the architecture of the network trained with source images should not change when applying the adaptation phase. In this way, we can implement the new model to classify the target samples quickly, which is essential in real-world applications.

For the Deep CORAL method, the following loss function is added to the baseline CNN:

$$\mathcal{L}_{CORAL} = \frac{1}{4d^2} \|C_s - C_t\|_F^2 \tag{11}$$

where $\|\cdot\|_F^2$ indicates the squared matrix Frobenius norm, C_s and C_t are the covariance matrices of the d-dimensional features of source and target respectively. It is necessary to emphasize that in this method both networks (source and target) are trained with tied weights. The classification loss ($\mathcal{L}_{class}$) of Eq. 3 is added at the top of the source classifier C_s. Finally, the training is achieved by minimizing the following loss function:

$$\mathcal{L}_{totalCORAL} = \mathcal{L}_{class} + \lambda_C \mathcal{L}_{CORAL} \tag{12}$$

where the hyper-parameter λ_C compensates the classification loss and the adaptation.

On the other hand, JAN adds a joint maximum mean discrepancy (JMMD) penalty to train the CNN. The activations generated by a network with $\mathcal{M}$ layers are $\{(\mathrm{z}_{s1}^{(i)}, \cdots, \mathrm{z}_{s\mathcal{M}}^{(i)})\}_{i=1}^{n_s}$ and $\{(\mathrm{z}_{t1}^{(i)}, \cdots, \mathrm{z}_{t\mathcal{M}}^{(i)})\}_{i=1}^{n_t}$, for source and target samples respectively. By applying the kernel trick to compute the MMD in multiple layers, the following empirical estimate of JMMD can be computed:

$$\begin{aligned}\mathrm{J\hat{M}MD}(D_s, D_t) = & \frac{1}{n_s^2}\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}\prod_{m\in\mathcal{M}} k_m(\mathrm{z}_{sm}^{(i)}, \mathrm{z}_{sm}^{(j)}) \\ & + \frac{1}{n_t^2}\sum_{i=1}^{n_t}\sum_{j=1}^{n_t}\prod_{m\in\mathcal{M}} k_m(\mathrm{z}_{tm}^{(i)}, \mathrm{z}_{tm}^{(j)}) \\ & - \frac{2}{n_s n_t}\sum_{i=1}^{n_s}\sum_{j=1}^{n_t}\prod_{m\in\mathcal{M}} k_m(\mathrm{z}_{sm}^{(i)}, \mathrm{z}_{tm}^{(j)})\end{aligned} \tag{13}$$

where $k_m(\cdot,\cdot)$ is the kernel function of layer m, and n_s and n_t are the number of source and target samples respectively. By embedding the JM$\hat{\mathrm{M}}$D into the CNN, the discrepancy in the joint distributions of multiple task-specific layers is reduced. To avoid the quadratic complexity of Eq. 13 a linear-time estimate of the JMMD is used for training:

$$\begin{aligned}&\mathrm{J\hat{M}MD}(D_s, D_t) = \\ & \frac{2}{n_s}\sum_{i=1}^{n_s/2}\left(\prod_{m\in\mathcal{M}} k_m(\mathrm{z}_{sm}^{(2i-1)}, \mathrm{z}_{sm}^{(2i)}) + \prod_{m\in\mathcal{M}} k_m(\mathrm{z}_{tm}^{(2i-1)}, \mathrm{z}_{tm}^{(2i)})\right) \\ & - \frac{2}{n_s}\sum_{i=1}^{n_s/2}\left(\prod_{m\in\mathcal{M}} k_m(\mathrm{z}_{sm}^{(2i-1)}, \mathrm{z}_{tm}^{(2i)}) + \prod_{m\in\mathcal{M}} k_m(\mathrm{z}_{tm}^{(2i-1)}, \mathrm{z}_{sm}^{(2i)})\right)\end{aligned} \tag{14}$$

Finally, the JMMD penalty is combined with the classification loss to train the network:

$$\mathcal{L}_{totalJAN} = \mathcal{L}_{class} + \lambda_J \mathrm{J\hat{M}MD}(D_s, D_t) \tag{15}$$

ADDA is a two-step method. At first, a source CNN (FCN_s) and a classifier (C_s) is trained with source images using the classification loss ($\mathcal{L}_{class}$) of Eq. 3. Then, a target CNN (FCN_t) is initialized with the pre-trained FCN_s. While keeping parameters of the FCN_s frozen, adversarial training is adopted by using a discriminator to classify the domain label of samples. An iterative training is performed by minimizing the following loss function:

$$\mathcal{L}_{total} = \mathcal{L}_{adv_D} + \mathcal{L}_{adv_G} \tag{16}$$

When adversarial training is finished, the C_s initially trained with source images is used on the top of FCN_t to predict the label of the target sample.

4. Results and discussion

In this section, we show the results obtained by the proposed method. Furthermore, we apply discrepancy-based [19, 18] and adversarial-based [24] methods for comparison. The results obtained show that we can apply our method to successfully classify defects in potatoes coming from a target domain when only using labels from a source domain. Implementations were developed with Pytorch [40]. All experiments were done using a GPU NVIDIA GeForce®GTX 1070 Ti (8 GB memory).

4.1. Setup

Datasets. We evaluated the proposed method with our created datasets, as explained in Section 3.1. In *DB1*, raw white potato images were used as coming from the source domain. To create the target domain, images were modified by increasing their brightness. In this way, we tried simulating a change in the lighting conditions of the acquisition system. In a second scenario (*DB2*), white potatoes were used as coming from the source domain and red potatoes were used to create the target domain.

Implementation details. For all methods, the pre-trained GoogLeNet without the last fully-connected layer was used as fully convolutional network (FCN_s and FCN_t). The source and target classifiers (C_s and C_t) consisted of a fully-connected layer of 6 neurons, where 6 is the number of classes of our dataset. Input images were resized to 224×224 according to GoogLeNet architecture. For our method, the discriminator (D) consisted of 3 fully-connected layers: the first layer with 1024 units, the second with 500 units and the third was a one-unit layer to classify the domain. All fully-connected layers, except the last one, used ReLU activation functions. Data augmentation techniques such as rotation, and horizontal and vertical flipping, were randomly applied when training the network. Labeled source samples and unlabeled target samples were used for training to follow the unsupervised domain adaptation setting.

In the first step of our proposed method (Section 3.4), the FCN_s and C_s were trained with stochastic gradient descent (SGD) with batch size of 32, momentum of 0.9 and weight decay of 5×10^{-4}. The learning rate was set to 1×10^{-4}, except for the last fully connected layer, which was set to 10 times the baseline learning rate. In the second step (Section 3.5), the fully-convolutional network (FCN_t) and classifier (C_t) were trained using SGD with a batch size of 256 (128 images from each domain) and learning rate of 1×10^{-5}. The discriminator D was trained using Adam optimizer with β_1 of 0.5, β_2 of 0.999 and learning rate of 1×10^{-4}. The number of epochs was set to 200. The hyper-parameter λ that balance the pseudo-label loss was progressively modified from 0 to 1 by $\lambda = \dfrac{n_e - 1}{t_e - 1}$, where n_e is the current epoch number and t_e is the maximum number of epochs. This strategy was adopted to suppress noisy pseudo-label information at the beginning of training.

In the case of the Deep CORAL method, features obtained by the last convolutional layers of FCN_s and FCN_t were used to compute the CORAL loss ($\mathcal{L}_{CORAL}$). Stochastic gradient descent (SGD) with a batch size of 32, momentum of 0.9 and weight decay of 5×10^{-4} was used to train the network. The learning rate was set to 1×10^{-4}, except for the fully connected layer of the C_s, which was set to 10 times the initial learning rate. The hyper-parameter λ_C was set to 5 to compensate for classification loss and the adaptation.

For JAN we used a Gaussian kernel with bandwidth set to median pairwise squared distances on the training data, as proposed by authors in [41]. Two layers were used to compute the JMMD penalty of Eq. 14: the output of the FCN_s network and the output of the C_s. As in Deep CORAL, source and target networks used tied weights, so $FCN_s = FCN_t$ and $C_s = C_t$. The SGD with the same hyper-parameters used to train Deep CORAL was applied. We gradually changed the weight parameter λ_J from 0 to 1.

To obtain ADDA results, we used the optimizers and hyper-parameters as our proposed method. Since ADDA uses the same classifier for source and target data, no pseudo label loss had been applied.

Following the work of authors in [42], we computed the test performance for all methods at the epoch where the maximum number of target samples surpassed the confidence threshold (used to calculate the pseudo-label loss).

4.2. Experimental results

The precision, recall and F1-score of five random experiments on *DB1* dataset are shown in Table 3. A lower bound and two upper bound performance were computed to compare methods. Source only corresponds to the lower bound performance, where no adaptation is carried out. Target only and Source+Target represent the upper bound results. In the first case, the network is trained on the target dataset using the labels. In the second case, both source and target datasets with labels are used to train the CNN.

We can observe that applying a domain adaptation method is mandatory to prevent a significant decrease in performance on the target domain. The class with the lowest performance was the black dot, which is the most difficult to detect even with a target only setting. Furthermore, we note that adversarial-based methods (ADDA and ours) outperformed discrepancy-based methods (Deep CORAL and JAN). If we compare ADDA with our method, we can see that adding the pseudo-label loss help the classifier to increase the performance on the target dataset. Furthermore, our method is comparable to the target only setting, which is trained in a fully-supervised manner.

In Table 4 the results obtained of five random experiments on *DB2* can be observed. In this scenario, white potatoes were used as source images and red potatoes as target images. From the results obtained, we can confirm that the network trained with source images cannot be directly used to classify the new target images (mean F1-score of 0.12). Therefore, applying a domain adaptation method is mandatory to improve network performance on target

Table 3: Precision, recall and F1-score on testing target *DB1* dataset. Classes are: H=Healthy, D=Damaged, G=Greening, BD=Black Dot, CS=Common Scab and BS=Black Scurf.

Metric	Classes	Source only	Deep CORAL	JAN	ADDA	Ours	Target only	Source+Target
Precision	H	0.77 ± 0.06	0.85 ± 0.01	0.89 ± 0.00	0.89 ± 0.00	0.90 ± 0.01	0.89 ± 0.00	**0.99 ± 0.00**
	D	0.50 ± 0.14	0.84 ± 0.02	0.83 ± 0.03	0.84 ± 0.03	0.85 ± 0.01	0.87 ± 0.01	**0.90 ± 0.01**
	G	**1.00 ± 0.00**	0.91 ± 0.03	0.92 ± 0.02	0.99 ± 0.01	0.98 ± 0.01	0.95 ± 0.01	0.98 ± 0.01
	BD	0.00 ± 0.00	0.56 ± 0.03	0.71 ± 0.01	0.74 ± 0.05	0.77 ± 0.02	0.78 ± 0.02	**0.81 ± 0.01**
	CS	0.54 ± 0.09	0.77 ± 0.02	0.83 ± 0.03	0.85 ± 0.03	0.83 ± 0.02	0.86 ± 0.02	**0.89 ± 0.01**
	BS	0.66 ± 0.18	**0.91 ± 0.01**	0.84 ± 0.04	0.85 ± 0.05	0.88 ± 0.03	0.87 ± 0.01	**0.85 ± 0.03**
	Average	0.58 ± 0.08	0.81 ± 0.02	0.84 ± 0.02	0.86 ± 0.03	0.87 ± 0.02	0.87 ± 0.01	**0.89 ± 0.01**
Recall	H	0.82 ± 0.08	0.92 ± 0.01	0.93 ± 0.01	0.95 ± 0.01	0.95 ± 0.01	0.95 ± 0.00	**0.96 ± 0.00**
	D	0.66 ± 0.09	0.75 ± 0.01	0.83 ± 0.02	0.85 ± 0.01	0.86 ± 0.01	0.83 ± 0.01	**0.87 ± 0.01**
	G	0.20 ± 0.08	0.71 ± 0.02	0.82 ± 0.03	0.78 ± 0.01	0.79 ± 0.01	0.81 ± 0.02	**0.89 ± 0.01**
	BD	0.00 ± 0.00	0.57 ± 0.05	0.70 ± 0.02	0.73 ± 0.03	0.73 ± 0.04	0.72 ± 0.02	**0.78 ± 0.01**
	CS	0.74 ± 0.14	0.78 ± 0.02	0.79 ± 0.05	0.78 ± 0.03	0.82 ± 0.03	0.79 ± 0.02	**0.83 ± 0.01**
	BS	0.50 ± 0.17	0.55 ± 0.05	0.82 ± 0.04	0.79 ± 0.03	0.79 ± 0.03	0.85 ± 0.02	**0.91 ± 0.01**
	Average	0.49 ± 0.09	0.72 ± 0.02	0.81 ± 0.03	0.81 ± 0.02	0.82 ± 0.02	0.82 ± 0.01	**0.87 ± 0.01**
F1-score	H	0.79 ± 0.02	0.88 ± 0.00	0.91 ± 0.00	0.92 ± 0.00	0.92 ± 0.00	0.92 ± 0.00	**0.94 ± 0.00**
	D	0.54 ± 0.07	0.79 ± 0.01	0.83 ± 0.01	0.85 ± 0.01	0.85 ± 0.00	0.85 ± 0.00	**0.88 ± 0.00**
	G	0.32 ± 0.11	0.80 ± 0.01	0.87 ± 0.01	0.87 ± 0.01	0.87 ± 0.01	0.87 ± 0.01	**0.93 ± 0.01**
	BD	0.00 ± 0.00	0.57 ± 0.02	0.70 ± 0.01	0.73 ± 0.02	0.75 ± 0.02	0.74 ± 0.01	**0.80 ± 0.01**
	CS	0.59 ± 0.06	0.77 ± 0.01	0.81 ± 0.02	0.81 ± 0.01	0.83 ± 0.01	0.82 ± 0.00	**0.86 ± 0.01**
	BS	0.51 ± 0.09	0.69 ± 0.03	0.83 ± 0.02	0.81 ± 0.03	0.83 ± 0.03	0.86 ± 0.01	**0.88 ± 0.02**
	Average	0.46 ± 0.06	0.75 ± 0.01	0.82 ± 0.01	0.83 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	**0.88 ± 0.01**

images. Similar to results obtained on the *DB1* dataset, adversarial-based methods (ADDA and ours) outperformed discrepancy-based methods (Deep CORAL and JAN). The most affected classes were the black dot and common scab, which can be explained by the fact that these diseases present different visual characteristics according to the variety of potato in which it occurs (white or red). Contrary to the previous scenario (*DB1*), the best results were obtained when training the network with target labeled images (Target only), and not with the source and target labeled images together (Source+Target). This can be explained by the fact that both domains are quite different.

Confusion matrices are computed to understand the types of mistakes occurred on both datasets. Figures 5 and 6 show confusion matrices obtained by applying different methods on *DB1* and *DB2* respectively. The following conclusions can be obtained from the results: (1) the class that was most affected by the change of brightness in images was the black dot (BD). Without performing a domain adaptation method, this class was no longer detected (Figure 5a), and it was usually confused with the healthy class. (2) Another class that was greatly affected by the change in brightness of images was the green (G) class, going from 78.6% of detection using our method to 19.6% without adaptation. This can be explained by the importance of image colors to correctly detect this class. (3) The confusion between similar classes, for example, common scab (CS) and black scurf (BS), increased when a domain adaptation method was not applied. (4) The biggest difference between ADDA and our method on the *BD1* was the

Table 4: Precision, recall and F1-score on testing target *DB2* dataset. Classes are: H=Healthy, D=Damaged, G=Greening, BD=Black Dot, CS=Common Scab and BS=Black Scurf.

Metric	Classes	Source only	Deep CORAL	JAN	ADDA	Ours	Target only	Source+Target
Precision	H	0.59 ± 0.23	0.68 ± 0.01	0.76 ± 0.01	0.76 ± 0.00	0.75 ± 0.00	**0.89 ± 0.01**	0.88 ± 0.01
	D	0.21 ± 0.03	0.54 ± 0.01	0.66 ± 0.04	0.71 ± 0.02	0.72 ± 0.01	**0.88 ± 0.01**	**0.88 ± 0.02**
	G	0.57 ± 0.22	0.81 ± 0.02	0.85 ± 0.06	0.90 ± 0.01	0.90 ± 0.01	**0.93 ± 0.01**	0.91 ± 0.02
	BD	0.07 ± 0.00	0.18 ± 0.03	0.16 ± 0.07	0.36 ± 0.07	0.39 ± 0.10	**0.81 ± 0.05**	**0.81 ± 0.04**
	CS	0.16 ± 0.09	0.5 ± 0.01	0.48 ± 0.05	0.66 ± 0.02	0.68 ± 0.04	0.81 ± 0.01	**0.82 ± 0.01**
	BS	0.00 ± 0.00	0.86 ± 0.06	0.88 ± 0.03	0.82 ± 0.04	0.85 ± 0.05	**0.93 ± 0.02**	0.84 ± 0.04
	Average	0.27 ± 0.08	0.59 ± 0.02	0.63 ± 0.04	0.70 ± 0.03	0.71 ± 0.04	**0.87 ± 0.02**	0.86 ± 0.02
Recall	H	0.03 ± 0.03	0.90 ± 0.01	0.85 ± 0.02	0.92 ± 0.01	**0.93 ± 0.00**	0.92 ± 0.01	0.91 ± 0.01
	D	0.38 ± 0.13	0.48 ± 0.03	0.63 ± 0.06	0.63 ± 0.04	0.61 ± 0.00	0.85 ± 0.03	**0.86 ± 0.01**
	G	0.05 ± 0.03	0.43 ± 0.07	0.80 ± 0.03	0.90 ± 0.02	0.90 ± 0.01	0.92 ± 0.00	**0.93 ± 0.01**
	BD	0.09 ± 0.05	0.17 ± 0.02	0.12 ± 0.05	0.17 ± 0.06	0.15 ± 0.08	**0.74 ± 0.02**	0.70 ± 0.05
	CS	0.75 ± 0.12	0.23 ± 0.01	0.40 ± 0.05	0.43 ± 0.02	0.40 ± 0.01	**0.76 ± 0.02**	0.75 ± 0.03
	BS	0.00 ± 0.00	0.45 ± 0.04	0.52 ± 0.05	0.59 ± 0.08	0.66 ± 0.06	**0.87 ± 0.02**	0.84 ± 0.03
	Average	0.22 ± 0.06	0.44 ± 0.03	0.55 ± 0.04	0.61 ± 0.04	0.61 ± 0.03	**0.84 ± 0.02**	0.83 ± 0.02
F1-score	H	0.05 ± 0.06	0.78 ± 0.00	0.80 ± 0.00	0.83 ± 0.00	0.83 ± 0.00	**0.90 ± 0.00**	**0.90 ± 0.00**
	D	0.25 ± 0.07	0.51 ± 0.01	0.64 ± 0.02	0.66 ± 0.02	0.66 ± 0.01	**0.87 ± 0.01**	**0.87 ± 0.01**
	G	0.09 ± 0.06	0.56 ± 0.06	0.82 ± 0.03	0.90 ± 0.01	0.90 ± 0.00	**0.93 ± 0.00**	0.92 ± 0.01
	BD	0.07 ± 0.02	0.17 ± 0.03	0.14 ± 0.05	0.23 ± 0.06	0.22 ± 0.10	**0.77 ± 0.01**	0.75 ± 0.01
	CS	0.26 ± 0.01	0.32 ± 0.01	0.43 ± 0.04	0.52 ± 0.02	0.50 ± 0.01	**0.78 ± 0.01**	**0.78 ± 0.01**
	BS	0.00 ± 0.00	0.59 ± 0.04	0.65 ± 0.04	0.69 ± 0.05	0.74 ± 0.04	**0.90 ± 0.01**	0.84 ± 0.01
	Average	0.12 ± 0.04	0.49 ± 0.02	0.58 ± 0.03	0.64 ± 0.03	0.64 ± 0.02	**0.86 ± 0.01**	0.84 ± 0.01

common scab detection, going from 77.6% to 82.4%, respectively. (5) When the change of domain is more significant (*BD2*), the use of a domain adaptation method is compulsory. (6) By comparing adversarial-based methods (ADDA and ours) applied on *DB2*, we can observe that the major difference occurred on the correct detection of the black scurf class (59.2% with ADDA and 66.2% with our proposed method). (7) When the same defect presents very different visual symptoms according to the domain in which it occurs (e.g., black dot or common scab), an unsupervised domain adaptation method may not be sufficient, and a semi-supervised approach could be considered.

Pseudo-label threshold sensitivity. We study the influence of the pseudo-label threshold. This threshold defines the target samples that are taken into account to compute the pseudo-label loss, i.e. only target samples with a pseudo-label probability exceeding the threshold will be used to calculate the pseudo-label loss. Figure 7 shows the mean F1-score of five random experiments on *DB2* dataset when varying the pseudo-label threshold. We observe that the F1-score increases as the pseudo-label threshold increases which confirms our initial assumption that if the network outputs a low probability for the assigned class, the prediction is likely to be inaccurate.

Feature Visualization. We use the t-SNE technique [43] to visualize the learned features extracted by the fully convolutional network (FCN_t) of "Source only", "Deep CORAL", "JAN", "ADDA" and our method. Figures 8 and 9 show the results obtained on *DB1* and *DB2* respectively. We can see that if we only use source samples for training

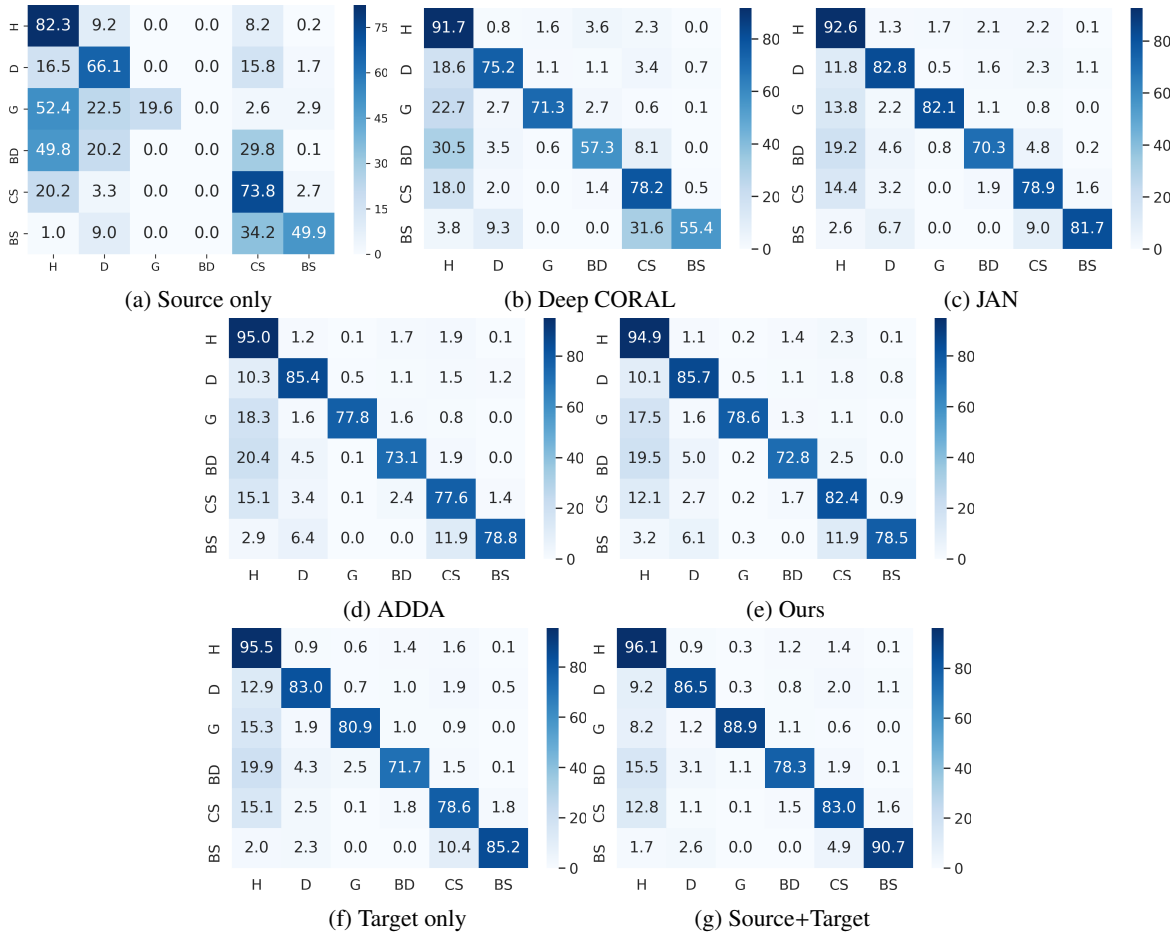


Figure 5: Confusion matrices using the *DB1* dataset for (a) source only model, (b) deep CORAL, (c) JAN, (d) JAN, (e) our model, (f) target only, (g) source+target. By rows: true labels. By columns: predicted labels. Classes: H=healthy, D=damaged, G=greening, BD=black dot, CS=common scab and BS=black scurf.

(Figures 8a and 9a), target features of different classes are mixed together. On the other hand, features are more discriminative when adaptation methods are applied (Figures 8b-8e and Figures 9b-9e). Finally, we can observe that even if visualizations of adapted methods are similar, features of some classes are more compact using our method.

5. Conclusion

In this work, we have presented an effective unsupervised adversarial domain adaptation method to classify potato defects in two different scenarios. Firstly, we have simulated a change in lighting conditions by artificially increasing the brightness of some images. Secondly, a change in a variety of color have been analyzed. Indeed, white potatoes have been used as source samples and red potatoes as target samples. Images from both datasets have been classified into six classes: healthy, damaged, greening, black dot, common scab, and black scurf. A two-stages method has been proposed to adapt a source model to a target dataset. Firstly, a fully convolutional network and a classifier

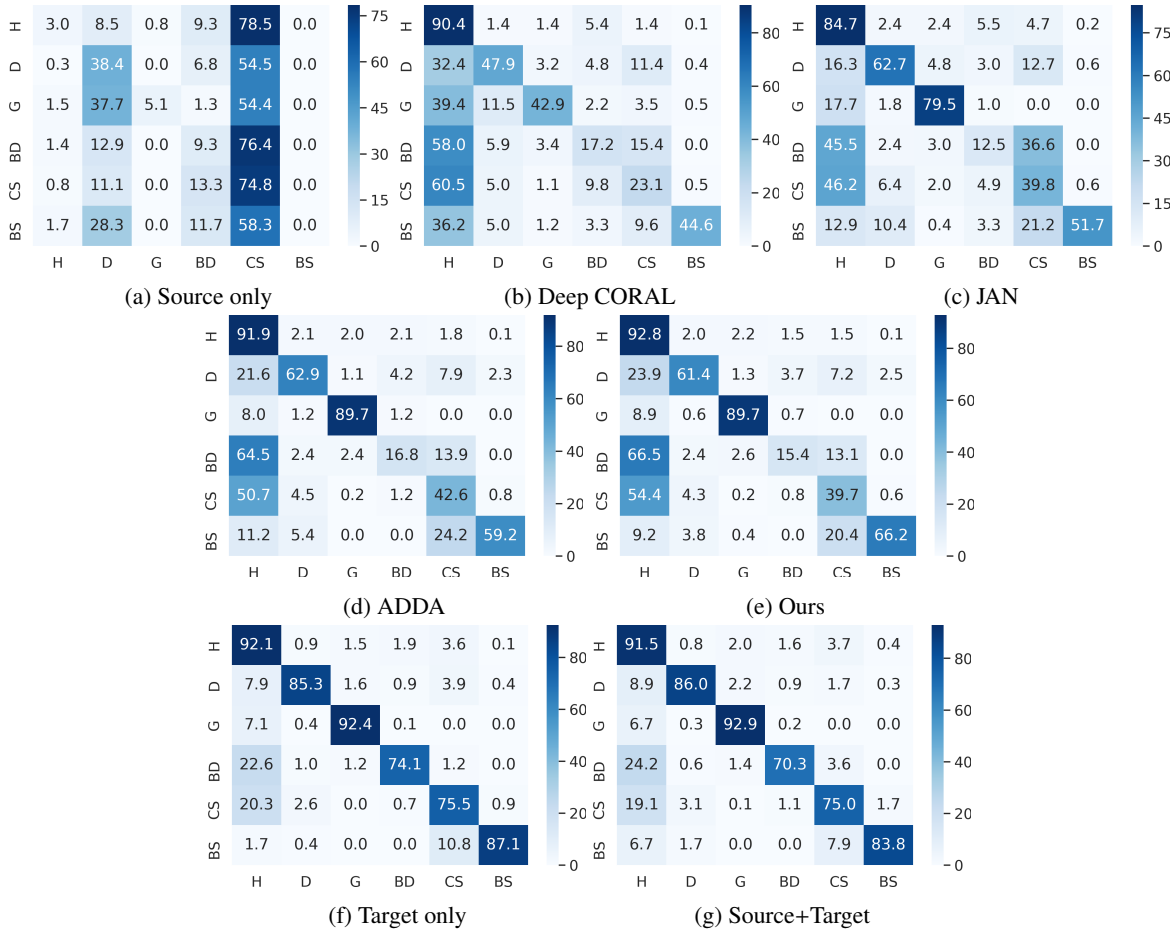


Figure 6: Confusion matrices using the *DB2* dataset for (a) source only model, (b) deep CORAL, (c) JAN, (d) JAN, (e) our model, (f) target only, (g) source+target. By rows: true labels. By columns: predicted labels. Classes: H=healthy, D=damaged, G=greening, BD=black dot, CS=common scab and BS=black scurf.

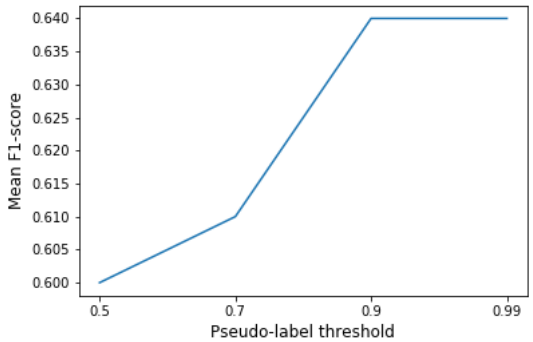
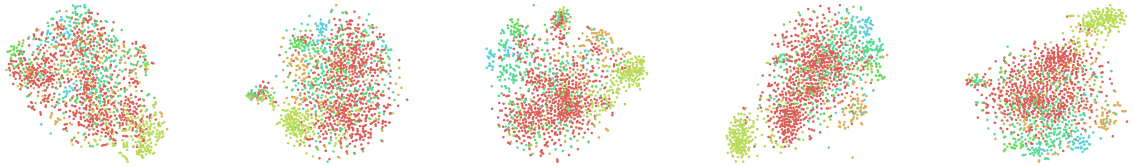


Figure 7: Pseudo-label threshold sensitivity.

have been trained with labeled source images. Secondly, adversarial training has been applied to align source and target distributions. Moreover, a pseudo-label loss has been proposed to train a specific target classifier, despite the

(a) Source only (b) Deep CORAL (c) JAN (d) ADDA (e) Ours

Figure 8: The t-SNE visualization of features of target samples (*DB1* dataset) learned by (a) Source only, (b) Deep CORAL, (c) JAN, (d) ADDA and (e) our method. Each color represents a class, with a total of 6 classes: healthy, damaged, greening, black dot, common scab, and black scurf.



(a) Source only (b) Deep CORAL (c) JAN (d) ADDA (e) Ours

Figure 9: The t-SNE visualization of features of target samples (*DB2* dataset) learned by (a) Source only, (b) Deep CORAL, (c) JAN, (d) ADDA and (e) our method. Each color represents a class, with a total of 6 classes: healthy, damaged, greening, black dot, common scab, and black scurf.

unavailability of target labels.

Experimental results have shown that a domain adaptation method is mandatory to obtain satisfactory results on the target dataset, reaching an average F1-score of 0.84 when the lighting conditions of the acquisition system change. Despite the straightforward implementation of certain discrepancy-based approaches, such as Deep CORAL and JAN, we have shown that these approaches are outperformed by our adversarial-based method. We have also shown that training the target classifier with a pseudo-label loss improves classification results on target images. This is an important difference between our method and ADDA, which directly applies the pre-trained source classifier on the new target images.

Finally, we have shown that our proposed method can be used in a real-world application when the source and target distributions are different. Our main objective is to avoid manual labeling of the new target images by leveraging the knowledge of the annotated source images. In this way, efficient model modifications can be made, which is essential in a real industrial application. Future works consist of evaluating our method in a semi-supervised scenario, especially for cases where the domain change is quite significant (e.g. from white to red potatoes). We will also study an unsupervised heterogeneous domain adaptation approach to address the problem of working with source RGB images and transfer the knowledge to target multi-spectral images. Furthermore, more recent CNN architectures (Inception-v4 [44], Xception [45], ResNeXt-50[46]) will be tested.

References

[1] Faostat, http://www.fao.org, accessed: 2019-05-23.

[2] J. Blasco, N. Aleixos, J. Gómez-Sanchis, E. Moltó, Recognition and classification of external skin damage in citrus fruits using multispectral data and morphological features, Biosystems engineering 103 (2) (2009) 137–145.

[3] M. Barnes, T. Duckett, G. Cielniak, G. Stroud, G. Harper, Visual detection of blemishes in potatoes using minimalist boosted classifiers, Journal of Food Engineering 98 (3) (2010) 339–346.

[4] N. Razmjooy, B. S. Mousavi, F. Soleymani, A real-time mathematical computer method for potato inspection using machine vision, Computers & Mathematics with Applications 63 (1) (2012) 268–279.

[5] G. ElMasry, S. Cubero, E. Moltó, J. Blasco, In-line sorting of irregular potatoes by using automated computer-based machine vision system, Journal of Food Engineering 112 (1-2) (2012) 60–68.

[6] P. Moallem, N. Razmjooy, A multi layer perceptron neural network trained by invasive weed optimization for potato color image segmentation, Trends in Applied Sciences Research 6 (2012) 445–455. doi:10.3923/tasr.2012.445.455.

[7] P. Moallem, N. Razmjooy, B. S. Mousavi, Robust potato color image segmentation using adaptive fuzzy inference system, Iranian Journal of Fuzzy Systems 11 (6) (2014) 47–65. doi:10.22111/ijfs.2014.1748.

[8] M. Brahimi, K. Boukhalfa, A. Moussaoui, Deep learning for tomato diseases: classification and symptoms visualization, Applied Artificial Intelligence 31 (4) (2017) 299–315.

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

[10] A. Picon, A. Alvarez-Gila, M. Seitz, A. Ortiz-Barredo, J. Echazarra, A. Johannes, Deep convolutional neural networks for mobile capture device-based crop disease classification in the wild, Computers and Electronics in Agriculture.

[11] S. Marino., P. Beauseroy., A. Smolarz., Deep learning-based method for classifying and localizing potato blemishes, in: Proceedings of the 8th International Conference on Pattern Recognition Applications and Methods - Volume 1: ICPRAM,, INSTICC, SciTePress, 2019, pp. 107–117. doi:10.5220/0007350101070117.

[12] S. Zhang, S. Zhang, C. Zhang, X. Wang, Y. Shi, Cucumber leaf disease identification with global pooling dilated convolutional neural network, Computers and Electronics in Agriculture 162 (2019) 422–430.

[13] J. Quionero-Candela, M. Sugiyama, A. Schwaighofer, N. D. Lawrence, Dataset shift in machine learning, The MIT Press, 2009.

[14] M. Wang, W. Deng, Deep visual domain adaptation: A survey, Neurocomputing 312 (2018) 135–153.

[15] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, T. Darrell, Deep domain confusion: Maximizing for domain invariance, arXiv preprint arXiv:1412.3474.

[16] M. Long, Y. Cao, J. Wang, M. I. Jordan, Learning transferable features with deep adaptation networks, in: Proceedings of the 32nd International Conference on International Conference on Machine Learning-Volume 37, JMLR. org, 2015, pp. 97–105.

[17] M. Long, H. Zhu, J. Wang, M. I. Jordan, Unsupervised domain adaptation with residual transfer networks, in: Advances in Neural Information Processing Systems, 2016, pp. 136–144.

[18] M. Long, H. Zhu, J. Wang, M. I. Jordan, Deep transfer learning with joint adaptation networks, in: Proceedings of the 34th International Conference on Machine Learning-Volume 70, JMLR. org, 2017, pp. 2208–2217.

[19] B. Sun, K. Saenko, Deep coral: Correlation alignment for deep domain adaptation, in: European Conference on Computer Vision, Springer, 2016, pp. 443–450.

[20] M.-Y. Liu, O. Tuzel, Coupled generative adversarial networks, in: Advances in neural information processing systems, 2016, pp. 469–477.

[21] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, D. Krishnan, Unsupervised pixel-level domain adaptation with generative adversarial networks, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[22] P. Isola, J.-Y. Zhu, T. Zhou, A. A. Efros, Image-to-image translation with conditional adversarial networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1125–1134.

[23] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, V. Lempitsky, Domain-adversarial training of neural networks, The Journal of Machine Learning Research 17 (1) (2016) 2096–2030.

[24] E. Tzeng, J. Hoffman, K. Saenko, T. Darrell, Adversarial discriminative domain adaptation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 7167–7176.

[25] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. A. Efros, T. Darrell, Cycada: Cycle-consistent adversarial domain adaptation, arXiv preprint arXiv:1711.03213.

[26] W. Zhang, W. Ouyang, W. Li, D. Xu, Collaborative and adversarial network for unsupervised domain adaptation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 3801–3809.

[27] M. Long, Z. Cao, J. Wang, M. I. Jordan, Conditional adversarial domain adaptation, in: Advances in Neural Information Processing Systems, 2018, pp. 1645–1655.

[28] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, D. Erhan, Domain separation networks, in: Advances in Neural Information Processing Systems, 2016, pp. 343–351.

[29] J.-Y. Zhu, T. Park, P. Isola, A. A. Efros, Unpaired image-to-image translation using cycle-consistent adversarial networks, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 2223–2232.

[30] M. Ghifary, W. Bastiaan Kleijn, M. Zhang, D. Balduzzi, Domain generalization for object recognition with multi-task autoencoders, in: Proceedings of the IEEE international conference on computer vision, 2015, pp. 2551–2559.

[31] K. Saenko, B. Kulis, M. Fritz, T. Darrell, Adapting visual category models to new domains, in: European conference on computer vision, Springer, 2010, pp. 213–226.

[32] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, et al., Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (11) (1998) 2278–2324.

[33] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, A. Y. Ng, Reading digits in natural images with unsupervised feature learning.

[34] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: Advances in neural information processing systems, 2014, pp. 2672–2680.

[35] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, J. W. Vaughan, A theory of learning from different domains, Machine learning 79 (1-2) (2010) 151–175.

[36] H. Shimodaira, Improving predictive inference under covariate shift by weighting the log-likelihood function, Journal of statistical planning and inference 90 (2) (2000) 227–244.

[37] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1–9.

[38] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, ImageNet: A Large-Scale Hierarchical Image Database, in: CVPR09, 2009.

[39] M. Bekkar, H. K. Djemaa, T. A. Alitouche, Evaluation measures for models assessment over imbalanced data sets, J Inf Eng Appl 3 (10).

[40] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, A. Lerer, Automatic differentiation in pytorch, in: NIPS-W, 2017.

[41] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, A. Smola, A kernel two-sample test, Journal of Machine Learning Research 13 (Mar) (2012) 723–773.

[42] G. French, M. Mackiewicz, M. Fisher, Self-ensembling for visual domain adaptation, in: International Conference on Learning Representations, 2018.

[43] L. v. d. Maaten, G. Hinton, Visualizing data using t-sne, Journal of machine learning research 9 (Nov) (2008) 2579–2605.

[44] C. Szegedy, S. Ioffe, V. Vanhoucke, A. Alemi, Inception-v4, inception-resnet and the impact of residual connections on learning (2017).

[45] F. Chollet, Xception: Deep learning with depthwise separable convolutions, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[46] S. Xie, R. Girshick, P. Dollar, Z. Tu, K. He, Aggregated residual transformations for deep neural networks, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.