A deep learning approach for detection and localization of leaf anomalies

Calabrò, D.; Lupo Pasini, M.; Ferro, N.; Perotto, S.

A deep learning approach for detection and localization of leaf anomalies

Davide Calabrò[1], Massimiliano Lupo Pasini[2], Nicola Ferro[3], Simona Perotto[3]

October 7, 2022

[1] Dipartimento di Matematica
Politecnico di Milano
Piazza L. da Vinci, 32, I-20133 Milano, Italy
davide.calabro@mail.polimi.it

[2] Oak Ridge National Laboratory
1 Bethel Valley Road, Oak Ridge, TN, USA, 37831
lupopasinim@ornl.gov

[3] MOX – Dipartimento di Matematica
Politecnico di Milano
Piazza L. da Vinci, 32, I-20133 Milano, Italy
{nicola.ferro, simona.perotto}@polimi.it

Abstract

The detection and localization of possible diseases in crops are usually automated by resorting to supervised deep learning approaches. In this work, we tackle these goals with unsupervised models, by applying three different types of autoencoders to a specific open-source dataset of healthy and unhealthy pepper and cherry leaf images. CAE, CVAE and VQ-VAE autoencoders are deployed to screen unlabeled images of such a dataset, and compared in terms of image reconstruction, anomaly removal, detection and localization. The vector-quantized variational architecture turns out to be the best performing one with respect to all these targets.

1 Introduction

In the last decades, the agricultural sector has been developing new technologies to maximize the efficiency of available soil resources in order to tackle several issues. Instances are the increasing demand for food due to the growth of the world population [7], as well as the impact of practices which are detrimental for the ecosystem [10]. In these contexts, precision agriculture has recently attracted a lot of interest since playing a significant role in the development of advanced techniques that optimize the soil productivity in a sustainable way [39, 20]. The general goal is to preserve the stability of the ecosystem while fostering the reuse of the soil for future produce. The strong interest in this new way of conceiving agriculture justifies the spread of innovative start-ups, of services devoted to eco-friendly practices, and of software solutions which allow farmers to accurately estimate yields on a simple smartphone or tablet (see, for instance, [2, 3, 4, 5]). In particular, the availability of higher-quality measurements, offered by advanced in field-sensors as well as by satellite or drone data, supported the proposal of breakthrough software solutions using modern deep learning algorithms [15, 19, 9, 36, 30].

In this paper, we focus on the detection of possible diseases in crops. Anomaly detection in plants represents a pivotal procedure in agriculture since an early detection of the disease enables a timely intervention to prevent the anomaly from spreading to the rest of the plant. Additionally, a precise disease localization allows confining the use of pesticides and other treatments to small strategically selected areas of the plant. This minimizes the negative impact of aggressive chemicals on the whole surrounding ecosystem.

Traditionally, the detection of the diseases is carried out manually. This is a time-consuming task and turns out to be expensive in terms of human resources. The development of advanced technologies as well as modern devices has recently offered the possibility to perform disease detection in a more efficient and affordable way by automating the process. For instance, computer vision turns out to be instrumental in analyzing large quantities of images collected by drones in a short time, such as photographs of healthy and diseased leaves in crops [1]. Among the several computer vision techniques, machine learning (ML) methods, with a focus on deep learning (DL) techniques, showed great potentiality in accurately discriminating healthy from unhealthy leaves through suitable classification procedures.

In the supervised learning framework, the DL models need to be fed with a large volume of training images in order to thoroughly span all classes in the dataset [11, 14]. In particular, in the leaf detection context, during the training phase a supervised DL method deals with healthy and unhealthy leaf classes in order to offer an accurate classification. This turns into a not so feasible practice due to the limited availability of images of leaves affected by diseases.

The importance of the selected dataset is confirmed by several contributions in the literature. For instance, in [22] the authors train a convolutional neural network (CNN) to identify 14 crop species and 26 diseases to conclude that only training DL models on increasingly large image datasets allows guaranteeing a reliable

crop diagnosis. In [8], the authors investigate the sensitivity characterizing the supervised learning of CNN models to the accuracy of the data. In particular, it is highlighted that the lack of conformity across different data samples can severely impact the generalizability of CNNs in accurately classifying diseased and healthy leaves.

The idea behind unsupervised DL procedures for anomaly detection is to carry out the training only on the healthy part of the dataset in order to accurately reproduce non-anomalous samples, while minimizing the risk of false negatives [27]. Once validated, the DL model is deployed for anomaly detection on unlabeled images. If the sample is characterized by a reconstruction error below a threshold, very likely it belongs to the same distribution as the training data, and it is thus classified as non-anomalous. Vice versa, if the reconstruction error is large, the data sample is very probably generated from a distribution different from the training data, and the sample is thus classified as anomalous. DL models generally used for unsupervised anomaly detection are convolutional autoencoders (CAEs) [37, 23], which are commonly exploited also for anomaly localization (see, e.g., [35, 38]).

As a third alternative, contributions where supervised and unsupervised approaches are combined are available in the literature. For instance, in [33] the authors use an unsupervised CAE model to derive relevant features from images of leaves, and then feed them into support vector machine (SVM) models for classification. Similarly, in [13] unsupervised learning techniques, such as principal component analysis and K-means, are employed to recover the informative features of samples of tomato leaves. Then, the extracted characteristics are classified by resorting to supervised ML approaches, such as SVM, CNN and K-nearest neighbors.

In this work, we apply unsupervised learning CAE models to detect and localize leaf diseases. Up to our knowledge, this type of data is still scarcely analyzed in the literature with unsupervised DL approaches. Goal of the paper is to compare the performance of three different CAE architectures, namely standard CAEs, convolutional variational autoencoders (CVAEs), vector-quantized variational autoencoders (VQ-VAEs), on the PlantVillage dataset, which collects images of healthy and diseased leaves of several species. The three models are assessed in terms of image reconstruction and anomaly removal, detection and localization. The paper is organized as follows. Section 2 details the three convolutional autoencoders. Section 3 focuses on the workflow adopted for the anomaly detection, while providing the main features of the considered image samples. In Section 4, we gather more technical information concerning the data preprocessing, the model setup, hardware and software specifics. Section 5 discusses the reliability and the performance of the three compared architectures, with a specific emphasis both on the reconstruction of the images and on the anomaly detection/localization. Finally, some conclusions are drawn in the last section, together with possible future developments.

2 Convolutional autoencoders

In this section, we describe the convolutional autoencoder (CAE) architecture together with two recent improved stabilized variants, namely the convolutional variational autoencoders (CVAEs) and the vector-quantized variational autoencoders (VQ-VAEs).

CAE models can be considered as a dimensionality reduction technique, since they allow extracting essential features from image data to enable an effective compression with a minimal loss of information. The architecture of a CAE model consists of three main components: an encoder, a stack of fully connected layers (latent space), and a decoder. A schematic representation of a CAE is provided in Figure 1.
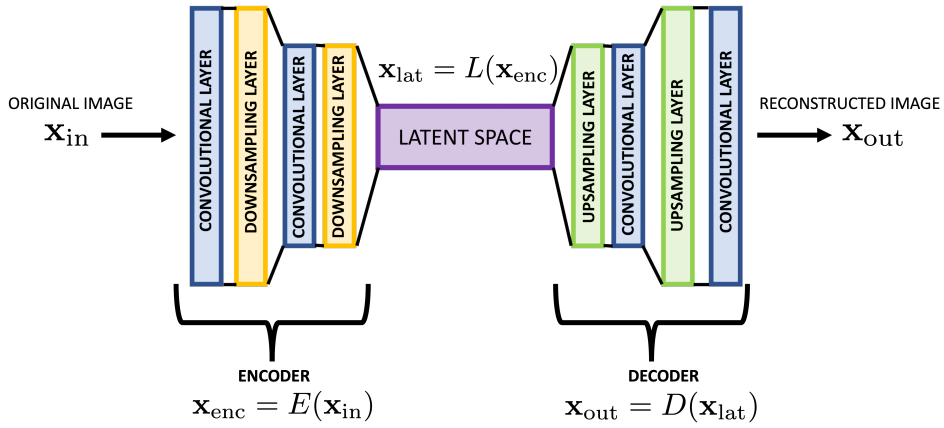

Figure 1: Architecture of a convolutional autoencoder.

An image is generally represented as a three-dimensional (3D) tensor of size (H, W, C), where H and W denote the number of pixels in height and width, respectively, while C is the number of channels (i.e., the depth) of the tensor. An RGB image has three channels, thus resulting into a tensor of size $(H, W, 3)$. Hereafter, we will adopt a vector in $\mathbb{R}^N$ as a representation of an RGB image equivalent to such a tensor, with $N = 3N_c$ and $N_c = HW$ the number of pixels per channel.

An RGB image $\mathbf{x}_{\text{in}} \in \mathbb{R}^N$ represents the standard input to a CAE encoder, which is assembled as a convolutional neural network (CNN) architecture that alternates convolutional with downsampling (average-pooling or max-pooling) layers in order to compress the input image [24]. The encoder transforms an input RGB image into a 3D tensor of size $(\tilde{H}, \tilde{W}, \tilde{C})$, with $\tilde{H} < H$, $\tilde{W} < W$, and $\tilde{C} > 3$. The encoder action is mathematically represented by a nonlinear operator E such that $\mathbf{x}_{\text{enc}} = E(\mathbf{x}_{\text{in}})$, with $\mathbf{x}_{\text{enc}}$ the compressed image.

The encoded image is successively mapped into the latent space, namely $\mathbf{x}_{\text{lat}} = L(\mathbf{x}_{\text{enc}})$, with L a nonlinear operator.

Finally, the re-elaborated compressed image $\mathbf{x}_{\text{lat}}$ is fed into the decoder, which

restores the information lost at the highest attainable fidelity. The decoder is constructed as a CNN architecture that alternates convolutional with upsampling layers in order to progressively re-expand the image up to the original size. The decoder is mathematically represented by a nonlinear operator D, which computes the final CAE output as an RGB image $\mathbf{x}_{\text{out}} = D(\mathbf{x}_{\text{lat}})$.

The nonlinear operators E, L and D are selected as regression models. We denote by $\mathbf{w}_1 \in \mathbb{R}^{N_1}$, $\mathbf{w}_2 \in \mathbb{R}^{N_2}$, and $\mathbf{w}_3 \in \mathbb{R}^{N_1}$ the vectors gathering the corresponding regression coefficients. As a consequence, the generic actions in Figure 1 can be particularized as

$$\mathbf{x}_{\text{enc}} = E_{\mathbf{w}_1}(\mathbf{x}_{\text{in}}), \quad \mathbf{x}_{\text{lat}} = L_{\mathbf{w}_2}(\mathbf{x}_{\text{enc}}), \quad \mathbf{x}_{\text{out}} = D_{\mathbf{w}_3}(\mathbf{x}_{\text{lat}}),$$

with

$$E_{\mathbf{w}_1} : \mathbb{R}^{N_1} \times \mathbb{R}^N \to \mathbb{R}^{N_{\text{lat}}}, \; L_{\mathbf{w}_2} : \mathbb{R}^{N_2} \times \mathbb{R}^{N_{\text{lat}}} \to \mathbb{R}^{N_{\text{lat}}}, \; D_{\mathbf{w}_3} : \mathbb{R}^{N_1} \times \mathbb{R}^{N_{\text{lat}}} \to \mathbb{R}^N,$$

where $N_{\text{lat}} = \tilde{H}\tilde{W}\tilde{C} < N$. The overall action of the CAE can be expressed in a compact form by introducing the vector $\mathbf{w} = \left[\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3\right]^T \in \mathbb{R}^{N_{\text{param}}}$ collecting all the regression coefficients, with $N_{\text{param}} = 2N_1 + N_2$, and the nonlinear operator $C_{\mathbf{w}} : \mathbb{R}^{N_{\text{param}}} \times \mathbb{R}^N \to \mathbb{R}^N$ defined by

$$C_{\mathbf{w}}(\mathbf{x}_{\text{in}}) = D_{\mathbf{w}_3}\left(L_{\mathbf{w}_2}\left(E_{\mathbf{w}_1}(\mathbf{x}_{\text{in}})\right)\right) = \mathbf{x}_{\text{out}}. \tag{1}$$

The training of the CAE over a set of input images $\{\mathbf{x}_{\text{in}}\}$ aims at compressing and successively decompressing each input image by producing an approximation $\mathbf{x}_{\text{out}}$, with a minimal loss of information with respect to $\mathbf{x}_{\text{in}}$. To this goal, the loss function, coinciding with the mean-square error

$$\mathcal{L}_{\text{CAE}}(\mathbf{x}_{\text{in}}, \mathbf{x}_{\text{out}}; \mathbf{w}) = \text{MSE}(\mathbf{x}_{\text{in}}, \mathbf{x}_{\text{out}}; \mathbf{w}) = \mathbb{E}\left[\|\mathbf{x}_{\text{in}} - \mathbf{x}_{\text{out}}\|^2\right], \tag{2}$$

is minimized by varying $\mathbf{w} \in \mathbb{R}^{N_{\text{param}}}$, with $\mathbb{E}[\cdot]$ the expected value and $\|\cdot\|$ the Euclidean norm. In particular, such a minimization is carried out over successive batches of data and by using batched stochastic optimization with automatic differentiation [29].

2.1 Convolutional variational autoencoders

CAEs may suffer from overfitting during the training. CVAE models offer an improvement of convolutional autoencoders, by properly regularizing such a phase. In particular, the peculiar difference between CAE and CVAE models lies in the definition of the latent space. Standard CAE models encode the input as a deterministic vector in the latent space and feed the decoder with an analogous type of data. On the contrary, CVAEs encode the input according to a distribution $\mathcal{D}$ of a continuous variable defined over the latent space, and the decoder is fed with a sample from that distribution. In general, we choose $\mathcal{D}$ as a multi-variate Gaussian distribution characterized by a mean, $\boldsymbol{\mu}_{\text{enc}}$, and by a diagonal variance matrix, whose entries
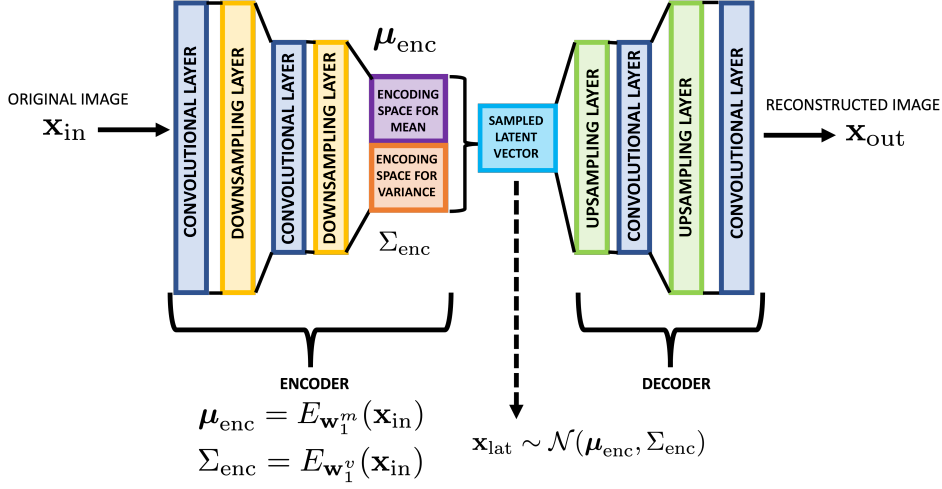
Figure 2: Architecture of a convolutional variational autoencoder.

are gathered in the vector Σ_{enc}. According to a CVAE architecture, vectors $\boldsymbol{\mu}_{\text{enc}}$ and Σ_{enc} are computed by two separate encoders (see Figure 2).

To provide a mathematical formalization of such a process, we introduce vectors $\mathbf{w}_1^m$, $\mathbf{w}_1^v \in \mathbb{R}^{N_1}$ collecting the coefficients associated with the regression encoders $E_{\mathbf{w}_1^m}, E_{\mathbf{w}_1^v} : \mathbb{R}^{N_1 \times N} \to \mathbb{R}^{N_{\text{lat}}}$, for the mean and the variance, respectively, such that

$$\boldsymbol{\mu}_{\text{enc}} = E_{\mathbf{w}_1^m}(\mathbf{x}_{\text{in}}), \quad \Sigma_{\text{enc}} = E_{\mathbf{w}_1^v}(\mathbf{x}_{\text{in}}), \tag{3}$$

with $\mathbf{x}_{\text{in}} \in \mathbb{R}^N$ the input image, and $N_{\text{lat}} = \tilde{H}\tilde{W}\tilde{C} < N$, the dimension of the latent space.

The output of the encoder, $\boldsymbol{\mu}_{\text{enc}}$ and Σ_{enc}, are fed into the latent space that computes the latent vector

$$\mathbf{x}_{\text{lat}} = L_{\mathcal{N}}(\boldsymbol{\mu}_{\text{enc}}, \Sigma_{\text{enc}}) \tag{4}$$

by randomly sampling the multi-variate Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_{\text{enc}}, \Sigma_{\text{enc}})$, with $L_{\mathcal{N}} : \mathbb{R}^{N_{\text{lat}}} \times \mathbb{R}^{N_{\text{lat}}} \to \mathbb{R}^{N_{\text{lat}}}$ the operator associated with the latent space.

Finally, the sample $\mathbf{x}_{\text{lat}}$ is passed to the decoder, represented by the regression operator $D_{\mathbf{w}_3} : \mathbb{R}^{N_1 \times N_{\text{lat}}} \to \mathbb{R}^N$, to generate the reconstructed image

$$\mathbf{x}_{\text{out}} = D_{\mathbf{w}_3}(\mathbf{x}_{\text{lat}}), \tag{5}$$

with $\mathbf{w}_3$ the vector of the regression coefficients characterizing the decoder.

The overall action of the CVAE model can be expressed in a compact form by introducing the vector $\mathbf{w} = \left[\mathbf{w}_1^m, \mathbf{w}_1^v, \mathbf{w}_3\right]^T \in \mathbb{R}^{N_{\text{param}}}$ collecting all the regression coefficients, with $N_{\text{param}} = 3N_1$, and the nonlinear operator $C_{\mathbf{w}} : \mathbb{R}^{N_{\text{param}}} \times \mathbb{R}^N \to \mathbb{R}^N$ defined by

$$C_{\mathbf{w}}(\mathbf{x}_{\text{in}}) = D_{\mathbf{w}_3}(L_{\mathcal{N}}(E_{\mathbf{w}_1^m}(\mathbf{x}_{\text{in}}), E_{\mathbf{w}_1^v}(\mathbf{x}_{\text{in}}))) = \mathbf{x}_{\text{out}}. \tag{6}$$

The involvement of a continuous variable in the CVAE model leads to add a term related to the multi-variate Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_{\text{enc}}, \Sigma_{\text{enc}})$ to the loss function in (2). Thus, the training is performed by minimizing the new loss function

$$\mathcal{L}_{\text{CVAE}}(\mathbf{x}_{\text{in}}, \mathbf{x}_{\text{out}}; \mathbf{w}) = \mathbb{E}\left[\|\mathbf{x}_{\text{in}} - \mathbf{x}_{\text{out}}\|^2\right] + D_{\text{KL}}\left(\mathcal{N}(\boldsymbol{\mu}_{\text{enc}}, \Sigma_{\text{enc}}), \mathcal{N}(\mathbf{0}, I)\right), \quad (7)$$

with D_{KL} the so-called Kullback-Leibler (KL) divergence [18]. We observe that the new contribution in (7) plays the role of a regularizing term by measuring the discrepancy between the distribution returned by the encoder, $\mathcal{N}(\boldsymbol{\mu}_{\text{enc}}, \Sigma_{\text{enc}})$, and a standard Gaussian distribution, $\mathcal{N}(\mathbf{0}, I)$. For more details about the CVAE architecture, we refer the reader to [17].

When the training data is limited, using a regularization defined in a continuous space as in (7) may introduce a strong bias, which results into a severe under-fitting with significant performance deterioration. To overcome this issue, new autoencoding architectures have been developed, as discussed in the following.

2.2 Vector-quantized variational autoencoders

VQ-VAE models offer a solution to the biasing effect triggered when the training data is small, by replacing the continuous regularization in (7) with a new term defined in a discrete space. In particular, this space is characterized by a set of K learnable vectors, $\{\mathbf{e}_i\}_{i=1}^K$, called codebook, with $\mathbf{e}_i \in \mathbb{R}^{\tilde{C}}$. The encoded vector $\mathbf{x}_{\text{enc}} = E_{\mathbf{w}_1}(\mathbf{x}_{\text{in}}) \in \mathbb{R}^{N_{\text{lat}}}$, with $N_{\text{lat}} = \tilde{H}\tilde{W}\tilde{C}$, is remapped into a 3D tensor of size $(\tilde{H}, \tilde{W}, \tilde{C})$ and then cut into longitudinal threads following the third dimension, creating a total of $\tilde{H}\tilde{W}$ fibers, $\mathbf{x}_{\text{enc}}^s \in \mathbb{R}^{\tilde{C}}$, for $s = 1, \ldots, \tilde{H}\tilde{W}$. For each fiber, the vector in the codebook $\{\mathbf{e}_i\}_{i=1}^K$ which is closest to $\mathbf{x}_{\text{enc}}^s$ is detected. This phase can be formalized by the minimization

$$z_q^s(\mathbf{x}_{\text{enc}}) = \mathbf{e}_p = \underset{\mathbf{e}_i}{\arg\min} ||\mathbf{x}_{\text{enc}}^s - \mathbf{e}_i|| \quad s = 1, \ldots, \tilde{H}\tilde{W}, \quad (8)$$

where $z_q^s(\mathbf{x}_{\text{enc}}) \in \mathbb{R}^{\tilde{C}}$ coincides with the quantized representation of the thread $\mathbf{x}_{\text{enc}}^s$ (see Figure 3). The threads $z_q^s(\mathbf{x}_{\text{enc}})$ are eventually aggregated to yield the vector $\mathbf{z}_q^{\text{lat}} \in \mathbb{R}^{N_{\text{lat}}}$.

We mathematically represent all the operations employed to convert the encoded vector $\mathbf{x}_{\text{enc}}$ into $\mathbf{z}_q^{\text{lat}}$ by the nonlinear operator $L_{\mathbf{q}} : \mathbb{R}^{N_2} \times \mathbb{R}^{N_{\text{lat}}} \to \mathbb{R}^{N_{\text{lat}}}$, with $\mathbf{q} \in \mathbb{R}^{N_2}$ the vector collecting the corresponding regression coefficients. Finally, vector $\mathbf{z}_q^{\text{lat}}$ is passed to the decoder for the final image reconstruction. Thus, the overall action of the VQ-VAE model can be expressed in a compact form by introducing the vector $\mathbf{w} = \begin{bmatrix} \mathbf{w}_1, \mathbf{q}, \mathbf{w}_3 \end{bmatrix}^T \in \mathbb{R}^{N_{\text{param}}}$ collecting all the regression coefficients, with $N_{\text{param}} = 2N_1 + N_2$, and the nonlinear operator $C_{\mathbf{w}} : \mathbb{R}^{N_{\text{param}}} \times \mathbb{R}^N \to \mathbb{R}^N$, such that

$$C_{\mathbf{w}}(\mathbf{x}_{\text{in}}) = D_{\mathbf{w}_3}(L_{\mathbf{q}}(E_{\mathbf{w}_1}(\mathbf{x}_{\text{in}}))) = \mathbf{x}_{\text{out}}. \quad (9)$$

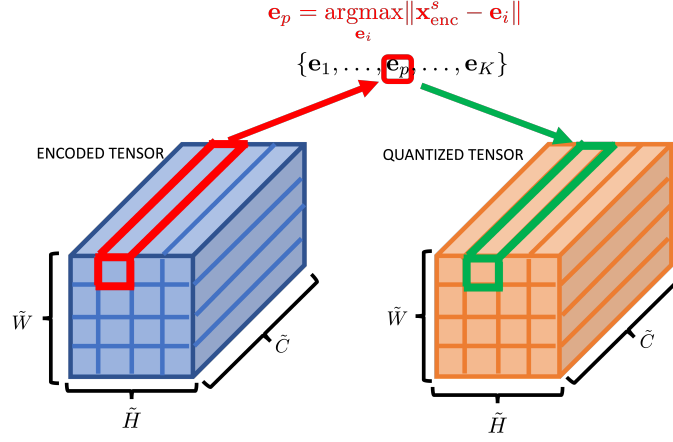An illustration of the VQ-VAE architecture is provided in Figure 4.

$$\mathbf{e}_p = \underset{\mathbf{e}_i}{\operatorname{argmax}} ||\mathbf{x}_{\text{enc}}^s - \mathbf{e}_i||$$

$$\{\mathbf{e}_1, \ldots, \mathbf{e}_p, \ldots, \mathbf{e}_K\}$$

Figure 3: Illustration of the quantization of the encoded vector in terms of the codebook.

$$\mathbf{x}_{\text{enc}} = E(\mathbf{x}_{\text{in}})$$

$$z_q^s(\mathbf{x}_{\text{enc}}) = \underset{\mathbf{e}_i}{\operatorname{argmin}} ||\mathbf{x}_{\text{enc}}^s - \mathbf{e}_i||$$
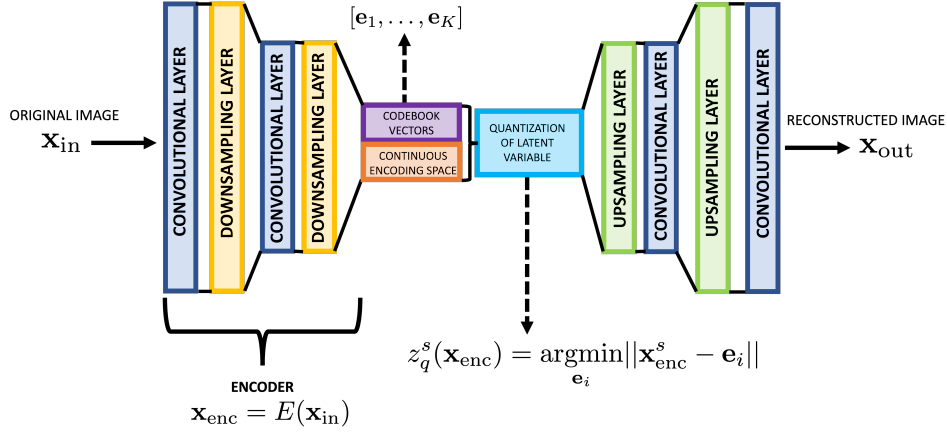
Figure 4: Architecture of a vector-quantized variational autoencoder.

The training of a VQ-VAE model includes the codebook as well as the model learning through a backpropagation process. Similarly to the loss function of CVAE, the training of the VQ-VAE model is carried out by iteratively minimizing a loss function consisting of a reconstruction error enriched by a regularizing term. In particular, the regularization coincides with the codebook learning, which is performed by the minimization in (8).

In particular, the codebook learning is characterized by a bidirectional problem, namely learning codebook vectors $\{\mathbf{e}_i\}_{i=1}^K$ that align to the encoder outputs, and learning encoder outputs $\{z_q^s(\mathbf{x}_{\text{enc}})\}_{s=1}^{\tilde{H}\tilde{W}}$ that align to the codebook vectors [32, 34]. To solve this issue, the regularization term in (8) is thus replaced by

$$z_q^s(\mathbf{x}_{\text{enc}}) = \underset{\mathbf{e}_i}{\operatorname{argmin}} \left\{ ||sg[\mathbf{x}_{\text{enc}}^s] - \mathbf{e}_i||^2 + \beta ||\mathbf{x}_{\text{enc}}^s - sg[\mathbf{e}_i]||^2 \right\} \quad s = 1, \ldots, \tilde{H}\tilde{W}, \quad (10)$$

which combines the so-called codebook alignment loss, $||sg[\mathbf{x}^s_{enc}] - \mathbf{e}_i||$, with the codebook commitment loss, $||\mathbf{x}^s_{enc} - sg[\mathbf{e}_i]||$, with $sg[\cdot]$ the stop gradient operator and β a hyperparameter to be properly tuned. The right-hand side in (10) implements an alternate direction minimization procedure, the stop gradient excluding the tensor it is applied to from the backpropagation.

To sum up, the training of the VQ-VAE model is performed by the minimization of the loss function

$$
\begin{aligned}
\mathcal{L}_{\text{VQ-VAE}}(\mathbf{x}_{\text{in}}, \mathbf{x}_{\text{out}}; \mathbf{w}) \quad = \quad & \mathbb{E}\big[||\mathbf{x}_{\text{in}} - \mathbf{x}_{\text{out}}||^2\big] + ||sg[\mathbf{x}_{\text{enc}}] - z_q(\mathbf{x}_{\text{enc}})||^2 \\
+ \quad & \beta ||\mathbf{x}_{\text{enc}} - sg[z_q(\mathbf{x}_{\text{enc}})]||^2.
\end{aligned}
\tag{11}
$$

For more details about the definition and utilization of the stop gradient operator and on VQ-VAEs, we refer the reader to [34].

3 Unsupervised anomaly detection with convolutional autoencoders

Unsupervised DL-driven detection and localization of anomalies often resort to CAE models to search for patterns in input data that do not conform to images free from defects [27, 35, 23]

To this goal, we train a CAE model only with images without anomalies. Once the CAE architecture is trained, the model is given unlabeled images. This means that the images exhibiting a defect will be reconstructed by removing the anomaly. The reconstruction error between the input and the output image is calculated and used to determine whether the image is affected by an anomaly or is not. In particular, if the reconstruction error is below a user-defined threshold, the input image is classified as free from defects. Otherwise, the image is classified as anomalous. Furthermore, the localization of the anomaly is carried out by analyzing the areas of the image where the pixel-wise reconstruction error is relevant (see Figure 5 for such a workflow particularized to a leaf dataset).

This approach has been employed to classify sets of data in diverse contexts [27, 23], although, up to the authors' knowledge, not in anomaly detection of leaves in smart agriculture applications. In this section, we aim at bridging this gap by complementing the current literature, which is essentially based on supervised DL approaches [33, 22, 8, 28], with an unsupervised technique.

For this task, we consider the dataset PlantVillage, which is freely available online in different versions. In more detail, we employ the dataset in the Mohanthy repository [21], which, in addition to the original pictures, contains preprocessed images, such as grayscale transformation and background removal.
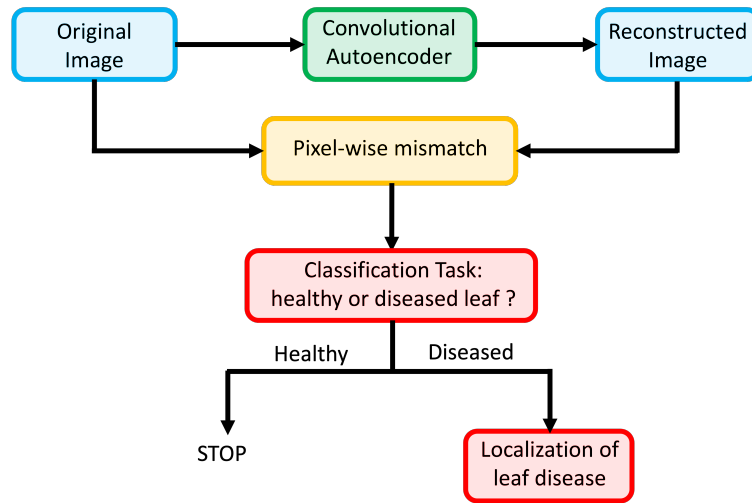
Figure 5: Schema of the proposed unsupervised DL methodology for leaf disease detection and localization.

The dataset consists of 54314 images of healthy and diseased leaves tagged into 38 different categories based on species and illness. Table 1 presents the dataset composition in terms of crop and disease type, and corresponding number of images.

Among the available species, in this work, we focus on cherry and pepper leaves (see Figures 6 and 7 for some samples).

Figure 6: Sample images of healthy (first and second from left) and diseased (first and second from right) cherry leaves.

4   Verification

We numerically assess the performance of the convolutional autoencoders in Section 2 when applied to the context detailed in Section 3.
A cross-comparison among CAEs, CVAEs and VQ-VAEs is carried out in terms of image reconstruction and detection and localization of leaf anomalies.

Crop	Disease	Number of images
Apple	Apple Scab	630
	Black Rot	621
	Cedar Apple Rust	275
	Healthy	1654
Blueberry	Healthy	1502
Cherry	Powdery Mildew	1052
	Healthy	854
Corn	Cercospora	513
	Common Rust	1192
	Nothern Leaf Blight	1162
	Healthy	985
Grape	Black Rot	1180
	Esca	1383
	Blight	1076
	Healthy	423
Orange	Haunglongbing	5507
Peach	Bacterial Spot	2297
	Healthy	360
Pepper	Bacterial Spot	997
	Healthy	1478
Potato	Early Blight	1000
	Late Blight	1000
	Healthy	152
Raspberry	Healthy	371
Soybean	Healthy	5090
Squash	Powdery Mildew	1835
Strawberry	Leaf Scorch	1109
	Healthy	456
Tomato	Bacterial Spot	2127
	Early Blight	1000
	Late Blight	1909
	Leaf Mold	952
	Septoria Leaf Spot	1771
	Spider Mites	1676
	Target Spot	1404
	Yellow Leaf Curl Virus	373
	Mosaic Virus	5357
	Healthy	1591

Table 1: PlantVillage dataset composition.

Figure 7: Sample images of healthy (first and second from left) and diseased (first and second from right) pepper leaves.

4.1 Data preprocessing

Data preprocessing is a fundamental phase of many machine learning and data mining approaches which manipulates, deletes and integrates the original data in order to improve the performance of the selected model.

Concerning the cherry and pepper datasets selected in the previous section, we essentially resort to resizing and data augmentation [31].
Since we perform an accelerated training by GPU hardware, we resize each input RGB image to 256×256 pixels in order to cope with memory limitations of the GPU device. Data augmentation is employed since DL models require large volumes of data for accuracy reasons. To increase the number of images available for training, we resort to the following transformations on the original data:

- image flipping: an image is mirrored around its horizontal or vertical axis[1];

- image rotation: an image is rotated in a clockwise or counterclockwise direction given a rotation angle θ. Since the models accept only squared images, the possible rotation angles are $90°$, $180°$ and $270°$.

The two datasets are split into 3 different subsets, to be used for training, validation and testing (see Table 2 for more details). The training dataset is composed by only healthy leaves images, which are used to optimize the DL models by minimizing the loss functions in (2), (7), and (11), respectively. The validation is successively adopted to identify the best model among CAE, CVAE e VQ-VAE in terms of reconstruction of both healthy and unhealthy leaf images. Finally, the testing dataset consists of pictures of the two classes of cherry and pepper leaves, the goal of this phase being the detection and the localization of leaf anomalies.

4.2 Model setup

We provide the architectural and training setup of the CAE, CVAE and VQ-VAE models. In particular, the three networks share the same architecture for the encoder and decoder parts, while being characterized by a different latent space configuration.

---

[1]The image flipping around the vertical axis is also known as flopping

	Training set	Validation set	Test set	
	Healthy	Healthy	Healthy	Diseased
cherry	681	85	85	85
pepper	1134	113	113	113

Table 2: Dataset splitting used for training, validation and test.

4.2.1 CAE architecture and training setup

The CAE takes in input an image with size $(H, W, C) = (256, 256, 3)$ and passes it to the encoder. The encoder is composed by 5 convolutional layers and 5 max-pooling layers for downsampling, that return a tensor of size $(4, 4, 64)$.

The downsampled tensor is then passed to the layers that define the latent space. The first one is a flatten layer that transforms the three-dimensional tensor into a vector $\mathbf{x}_{enc}$ with 1024 entries. The encoded vector is fed to two fully connected layers, each of them reducing the length of the vector by a factor of 8, so that the vector eventually matches the size of the latent space with 16 entries. In reverse, another pair of fully connected layers increases the size of the vector, each by a factor of 8, yielding the vector $\mathbf{x}_{lat}$ matching the original size (i.e., 1024 entries). The vector is then passed to a view layer that reshapes it into a three-dimensional tensor of size $(4, 4, 64)$ and then fed to the decoder, which consists of 5 convolutional and of 5 upsampling layers. The output $\mathbf{x}_{out}$ with size $(H, W, C) = (256, 256, 3)$ is finally generated.

The ReLU activation function is used for all the convolutional and deconvolutional layers, except for the output layer, where the sigmoid activation function is employed to ensure that the pixel values are within the range $[0, 1]$.

The CAE training is performed over 200 epochs by resorting to Adam algorithm [16], with a learning rate of 10^{-3} for the optimization.

4.2.2 CVAE architecture and training setup

Since the encoder of the CVAE resorts to a probability distribution to describe the input data, the latent space performs the so-called reparametrization trick in order to generate a sample from such a distribution to be eventually passed to the decoder. The CVAE decoder has the same architecture as for the CAE model, and starting from $\mathbf{x}_{lat}$ reconstructs the new image $\mathbf{x}_{out}$ with size $(H, W, C) = (256, 256, 3)$.

The training is performed with the same optimizer and learning rate as for the CAE, although relying only on 100 epochs due to the increased computational complexity of the DL model.

4.2.3 VQ-VAE architecture and training setup

The input image, with size $(H, W, C) = (256, 256, 3)$, is passed to the encoder which is composed by 3 convolutional layers without any max-pooling layer for

downsampling. A tensor of size $(4, 4, 64)$ is thus produced. Each thread of size $(1, 1, 64)$ of the tensor is replaced by the closest vector of the codebook for quantization. The codebook size is $(K, \tilde{C}) = (512, 64)$, meaning that the discrete space of the codebook is made of 512 vectors, each with 64 entries.

The quantization of the latent space yields a tensor with size $(4, 4, 64)$ which is provided as an input to the decoder constituted by 3 convolutional layers without any upsampling layer.

The training phase is performed with the same optimizer and learning rate as for the previous two DL models, by resorting to only 50 epochs due to a further increase in the computational cost of the VQ-VAE architecture.

4.3 Hardware & Software Specifics

The numerical results presented in this work are obtained with the hardware and OS specifics in Table 3.

PC model	Lenovo ThinkPad P14s
CPU	Intel® Core™ i7-10510U CPU @ 1.80GHz × 8
RAM	16 GB DDR4-3200
GPU	NV138 / Mesa Intel® UHD Graphics (Integrated)
OS	Ubuntu 20.04 LTS 64bit

Table 3: Hardware and software specifications.

The DL models are built and trained using the well-known neural network library `Pytorch` [26, 6], which enables a seamless remapping of the tensors from CPUs to GPUs for hardware accelerated training.

5 Discussion

In this section, we compare the three considered autoencoders in terms of reconstruction and classification of both healthy and anomalous leaves, as well as disease localization. These three skills are mutually related. Indeed, the capability of a model to accurately reproduce a non-anomalous image and to clean the defects on the diseased leaves is crucial in view of a reliable classification and localization.

5.1 Reconstruction and anomaly removal performance

We compare the predictive performance of the CAE, CVAE, VQ-VAE models to reconstruct anomalous and normal samples in the test portion of the dataset (see Table 2) for cherry and pepper leaf images. The metric used to quantify the accuracy of the reconstruction is the MSE, whose values - scaled by a 10^3 factor - are collected in Table 4 and Table 5 (second-fourth column). The discrepancy Δ between the MSE values for healthy and diseased samples is also provided in both

tables.

As expected by the architecture of the three models, we can rank VQ-VAE as the best performing architecture, followed by CAE and CVAE, independently of the considered leaf species. Moreover, we can appreciate how the VQ-VAE yields larger values for Δ. This emphasizes the separation of the two classes, with a consequent improvement in terms of leaf classification.

dataset	CAE	CVAE	VQ-VAE	CAE-TE	CVAE-TE
healthy	2.1766	4.8256	1.2380	2.1577	4.6332
diseased	2.1988	5.0732	2.0052	2.1739	4.6441
Δ	0.0222	0.2476	0.7672	0.0162	0.0109

Table 4: Reconstruction accuracy for the cherry dataset: comparison among CAE, CVAE and VQ-VAE in terms of MSE.

dataset	CAE	CVAE	VQ-VAE	CAE-TE	CVAE-TE
healthy	4.1029	6.2846	1.1674	3.9961	6.1512
diseased	4.7811	6.3255	1.8971	4.4350	6.2846
Δ	0.6782	0.0409	0.7297	0.4389	0.1334

Table 5: Reconstruction accuracy for the pepper dataset: comparison among CAE, CVAE and VQ-VAE in terms of MSE.

Since CAEs, CVAEs, VQ-VAEs take different times to train due to the specific complexity, we carry out a further comparison in order to consider such a mismatch. On average, VQ-VAE takes 4 times the training time required by CAE and CVAE, independently of the considered dataset. This is highlighted in Table 6 that shows the total training time (in minutes) measured as the median over 10 different runs for the three models.

dataset	CAE	CVAE	VQ-VAE
cherry	43	49	188
pepper	56	61	220

Table 6: Training time in minutes for cherry and pepper datasets.

Thus, the three DL networks are compared for a fixed computational training time, TE, that we select as the time required by VQ-VAE to complete 50 epochs (fourth column in Table 6). The MSE associated with this investigation is provided in Table 4 and Table 5 (fifth-sixth column). Despite the increased number of exploited epochs, both CAE and CVAE slightly improve the accuracy, while VQ-VAE remains the best performing model both in terms of reconstruction and of class separation.

To corroborate the performed investigation from a qualitative viewpoint, we consider the 4 images of healthy and unhealthy pepper and cherry leaves in Table

7 (first row).

When images of healthy leaves are provided as an input, the autoencoders are expected to reproduce the original image as accurately as possible. Although CAE and CVAE can recover the main characteristics of the leaf, such as color and shape, both the models are unable to reproduce the details at the fine scale, such as veins, chromatic differences and edges (second and third row, first and third column). On the contrary, the VQ-VAE model reproduces the images at a higher resolution, thus outperforming the other two architectures. In fact, now the chromatic differences of the veins of the leaves as well as the detail of the edges are clearly visible in the reconstructed image (fourth row, first and third column).

When images of unhealthy leaves are provided as an input, the autoencoder is expected to maintain the shape of the leaf while cleaning the surface from the alterations due to the disease (anomaly removal). We notice that the CAE and CVAE architectures mildly alter the shape of the leaf and lose each detail at a fine scale (second and third row, second and fourth column). On the contrary, VQ-VAE architecture faithfully reproduces the shape of the leaf as well as the details at a fine scale. Besides being more accurate, the VQ-VAE model has also enhanced generalizability properties. While accuracy and generalizability are both strongly desired in an autoencoder for image reconstruction, generalizability may actually be counterproductive in the context of anomaly removal. In fact, an effective generalizable autoencoder may accurately reconstruct the anomaly along with all the other features of the input image. This issue is acknowledged also by other works in the literature, especially for data where the anomaly covers a very small surface of the image [12].

The generalizability phenomenon is partially observed in the results in Table 7 (fourth row, second and fourth column), where the reconstructed images exhibit a certain heterogeneity in the color distribution. As a matter of fact, the MSE error for the diseased leaves in Tables 4 and 5 is lower for the VQ-VAE with respect to the two other architectures.

To sum up, we can conclude that the VQ-VAE represents the best performing model as far as image reconstruction and anomaly removal are concerned.

5.2 Leaf anomaly detection

In this section, we address the classification between healthy and unhealthy leaves by resorting to anomaly detection. In particular, to quantify the detection capability characterizing the considered unsupervised architectures, we resort to the AUC-ROC score, so that higher values of such a quantity reveal a better classification of the model. We observe that an accurate classification strictly depends on the reconstruction property of the selected DL method.

In Table 8 we provide the values of the AUC-ROC score for the three considered models. CAE and CVAE architectures better distinguish healthy leaves from unhealthy ones on the cherry dataset than on the pepper dataset (second-third columns). This is likely due to different anomalies of the diseases that affect the
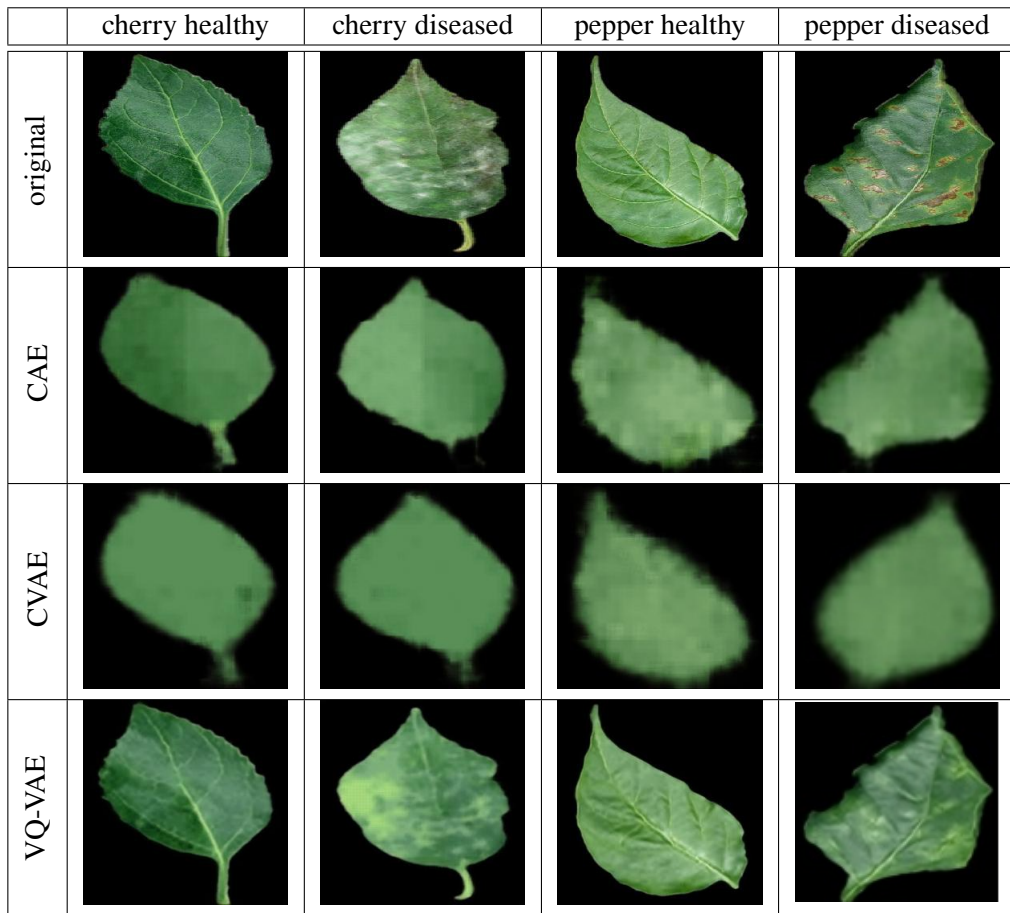
	cherry healthy	cherry diseased	pepper healthy	pepper diseased
original				
CAE				
CVAE				
VQ-VAE				

Table 7: Reconstruction and anomaly removal: comparison among CAE, CVAE and VQ-VAE on both cherry and pepper healthy and diseased leaves.

dataset	CAE	CVAE	VQ-VAE	CAE-TE	CVAE-TE
cherry	0.965	0.946	0.983	0.968	0.941
pepper	0.892	0.819	0.936	0.888	0.844

Table 8: Anomaly detection: comparison among CAE, CVAE and VQ-VAE in terms of AUC-ROC score.

two types of plants. In particular, the uniform distribution of the disease over the whole cherry leaf facilitates a clear distinction between the input (striped) unhealthy and the reconstructed (completely cleaned) images. This results into a good separability of the healthy and unhealthy classes and enables a reliable classification when using CAEs and CVAEs.

In the previous section, VQ-VAE proved to be characterized by enhanced generalizability properties. This is confirmed by the MSE values in Tables 4 and 5 which

are lower for the VQ-VAE when compared with CAE and CVAE (2.50 and 3.33 times lower for the pepper dataset). Despite the superior generalizability feature, VQ-VAE outperforms CAEs and CVAEs in terms of leaf classification due to the highest separation capability, corroborated by the values of the discrepancy Δ in the tables.

Also in the anomaly detection context, we perform a cross-comparison among the three models for a time equivalent training phase. The value of the AUC-ROC score associated with this analysis is provided in the last two columns of Table 8. Similarly to the reconstruction and anomaly removal verification, we can remark that the performance of both CAE-TE and CVAE-TE models do not significantly change by varying the number of epochs. Also in this investigation, VQ-VAE still represents the most reliable method.

5.3 Leaf anomaly localization

As a last check, we assess the anomaly localization capability of the CAE, CVAE and VQ-VAE models. With this regard, the performance is a direct consequence of the reconstruction properties discussed in Section 5.1, the localization step being carried out moving from the distribution of the reconstruction error.

In Table 9, we show such an error for both the healthy and unhealthy leaf samples in Table 7. We adopt a yellow-to-blue color map, after normalizing the images so that the reconstruction error ranges in $[0, 1]$. The anomaly localization provided by CAE and CVAE turns out to be rather unpractical. Indeed, all the details (such as, veins, edges, chromatic differences) at the fine scale which are missed during the reconstruction are highlighted as an error, analogously to the actual anomalies. This undesirable behavior is particularly evident in the CVAE outputs. On the contrary, the high quality guaranteed by VQ-VAE in sharply reconstructing the small-scale features and in mildly recovering the anomalies leads to confine the error to the edges and to the veins for the healthy samples, and to the actual defects for the diseased leaves (we refer to Figure 8 for a high-contrast post-processing of the VQ-VAE reconstruction error for the diseased cherry and pepper leaves in Table 9).

These considerations allow us to conclude that, when applied to PlantVillage dataset, VQ-VAE models perform better with respect to CAE and CVAE, also in terms of anomaly localization.

6 Conclusions and future work

The employment of unsupervised DL techniques on the PlantVillage dataset allowed us to compare the performance of standard autoencoders, such as CAE and CVAE model, with the more sophisticated VQ-VAE architecture, when applied to healthy and unhealthy leaves. The investigation focused on 2 out of the 14 different plant species included in the dataset, namely cherry and pepper leaves affected by
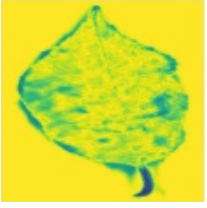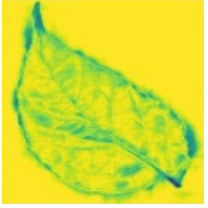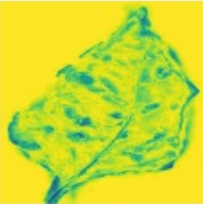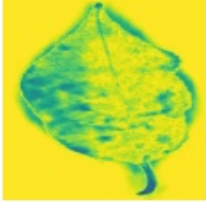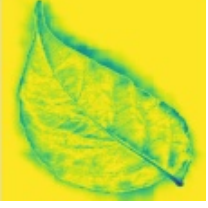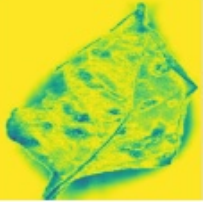
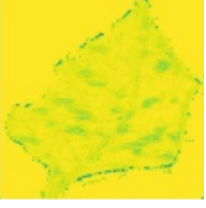|	cherry healthy	cherry diseased	pepper healthy	pepper diseased
Original				
CAE				
CVAE				
VQ-VAE				

Table 9: Anomaly localization: comparison in terms of reconstruction error among CAE, CVAE and VQ-VAE on both cherry and pepper healthy and diseased leaves.
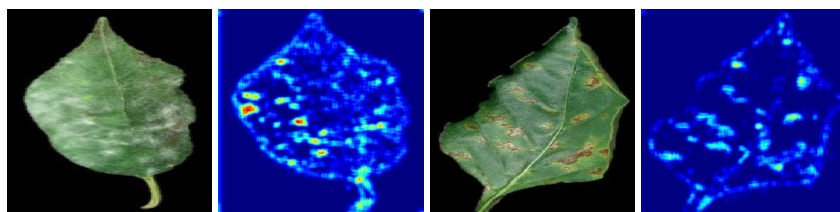

Figure 8: Anomaly localization: comparison between the original image and the high-contrast reconstruction error for the VQ-VAE model, cherry (first and second from left) and pepper (first and second from right) leaves.

powdery mildew and bacterial spot, respectively. A cross-comparison among the three autoencoders has been carried out in terms of: i) image reconstruction for both normal and anomalous samples, ii) anomaly removal from diseased leaves, iii) anomaly detection, iv) anomaly localization.

The results in Section 5 show that VQ-VAEs have a superior predictive performance than CAEs and CVAEs in the autoencoding task (see Table 7), which also reflects into a more accurate image classification of healthy and unhealthy leaves (see Table 8). The improved generalizability of the VQ-VAE models leads to a significant reduction of the reconstruction error on every image (see Tables 4, 5 and 9). This might potentially be counterproductive in the context of anomaly detection because it may reduce the capability of the model to separate images belonging to different classes. This possible risk has already been addressed in other contexts, such as [12]. However, for the specific dataset studied in this work, the gap between the reconstruction errors of the VQ-VAE on healthy and unhealthy images is significantly wider than for CAEs and CVAEs (see Tables 4 and 5), thus allowing the VQ-VAE to retain a superior capacity in accurately separating images of healthy leaves from images of unhealthy leaves.

Future work will be dedicated to apply methodology recently proposed in other contexts [25] to plant anomaly detection. These approaches combine supervised and semi-supervised DL methods to address situations where readily accessible large-scale unlabeled data may contain both known and unknown anomalies,
As a further development of interest, we consider the issue of miss detection of the anomaly due to a highly generalizable autoencoder that accurately reconstructs both normal and anomalous data. The outcome of this future effort will increase the robustness of current DL techniques for anomaly detection.

Acknowledgments

References

[1] Agrio - Precision plant protection. Made easy! `https://agrio.app/`.

[2] Cleverfarm - Accelerating the world's transition to sustainable farming. `https://www.cleverfarm.ag/`.

[3] Dynacrop - Increase farming productivity with satellite monitoring. `https://dynacrop.space/en/`.

[4] Pixofarm - Accurate yield estimation for your orchard. `https://www.pixofarm.com/`.

[5] Plantix - Your crop doctor. `https://plantix.net/en/`.

[6] PyTorch. `https://pytorch.org/docs/stable/index.html`.

[7] United Nations: Department of Economic and Social Affairs - Sustainable development, food security and nutrition and sustainable agriculture: Sustainable development knowledge platform, 2018.

[8] J. Boulent, S. Foucher, J. Théau, and P.-L. St-Charles. Convolutional neural networks for the automatic identification of plant diseases. *Frontiers in Plant Science*, 10, 2019.

[9] A. Chlingaryan, S. Sukkarieh, and B. Whelan. Machine learning approaches for crop yield prediction and nitrogen status estimation in precision agriculture: A review. *Computers and Electronics in Agriculture*, 151:61–69, 2018.

[10] V. H. Dale and S. Polasky. Measures of the effects of agricultural practices on ecosystem services. *Ecological Economics*, 64(2):286–296, 2007.

[11] A. Fernández, J. Bella, and J. R. Dorronsoro. Supervised outlier detection for classification and regression. *Neurocomputing*, 486:77–92, 2022.

[12] D. Gong, L. Liu, V. Le, B. Saha, M. R. Mansour, S. Venkatesh, and A. Van Den Hengel. Memorizing normality to detect anomaly: memory-augmented deep autoencoder for unsupervised anomaly detection. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1705–1714, Seoul, Korea (South), 2019.

[13] S. S. Harakannanavar, J. M. Rudagi, V. I. Puranikmath, A. Siddiqua, and R. Pramodhini. Plant leaf disease detection using computer vision and machine learning algorithms. *Global Transitions Proceedings*, 3(1):305–310, 2022.

[14] W. Jia, R. M. Shukla, and S. Sengupta. Anomaly detection using supervised learning and multiple statistical methods. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 1291–1297, 2019.

[15] A. Kamilaris, A. Kartakoullis, and F. X. Prenafeta-Boldú. A review on the practice of big data analysis in agriculture. *Computers and Electronics in Agriculture*, 143:23–37, 2017.

[16] D. P. Kingma and J. Ba. Adam: a method for stochastic optimization. *arXiv:1412.6980*, 2017.

[17] D. P. Kingma and M. Welling. An introduction to variational autoencoders. *Foundations and Trends in Machine Learning*, 12(4):307–392, 2019.

[18] S. Kullback and R. A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951.

[19] K. Liakos, P. Busato, D. Moshou, S. Pearson, and D. Bochtis. Machine learning in agriculture: a review. *Sensors*, 18(8):2674, 2018.

[20] R. M. Messick, M. J. Heaton, and N. Hansen. Multivariate spatial mapping of soil water holding capacity with spatially varying cross-correlations. *The Annals of Applied Statistics*, 11(1):69–92, 2017.

[21] S. P. Mohanthy. PlantVillage Dataset, 2016.

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

[23] F. Morawski, M. Bejger, E. Cuoco, and L. Petre. Anomaly detection in gravitational waves data using convolutional autoencoders. *Machine Learning: Science and Technology*, 2(4):045014, 2021.

[24] C. Mouton, J. C. Myburgh, and M. H. Davel. Stride and translation invariance in CNNs. In A. Gerber, editor, *Artificial Intelligence Research*, pages 267–281, Cham, 2020. Springer International Publishing.

[25] G. Pang, A. van den Hengel, C. Shen, and L. Cao. Toward deep supervised anomaly detection: reinforcement learning from partially labeled anomaly data. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, KDD '21, page 1298–1308, New York, 2021.

[26] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[27] W. H. L. Pinaya, P.-D. Tudosiu, R. Gray, G. Rees, P. Nachev, S. Ourselin, and M. J. Cardoso. Unsupervised brain anomaly detection and segmentation with transformers. *arXiv:2102.11650*, 2021.

[28] E. Pintelas, I. E. Livieris, and P. E. Pintelas. A convolutional autoencoder topology for classification in high-dimensional noisy image datasets. *Sensors*, 21(22):7731, 2021.

[29] H. Robbins and S. Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22(3):400–407, 1951.

[30] R. Sharma, S. S. Kamble, A. Gunasekaran, V. Kumar, and A. Kumar. A systematic literature review on machine learning applications for sustainable agriculture supply chain performance. *Computers & Operations Research*, 119:104926, 2020.

[31] C. Shorten and T. A. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(60):400–407, 2019.

[32] C. Snell. Understanding VQ-VAE (DALL-E Explained Pt. 1) - ML@B Blog, Sept. 2021.

[33] K. Trang, L. TonThat, and N. G. Minh Thao. Plant leaf disease identification by deep convolutional autoencoder as a feature extraction approach. In *2020 17th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, pages 522–526, 2020.

[34] A. van den Oord, O. Vinyals, and K. Kavukcuoglu. Neural discrete representation learning. *arXiv:1711.00937*, 2018.

[35] S. Venkataramanan, K.-C. Peng, R. V. Singh, and A. Mahalanobis. Attention guided anomaly localization in images. *arxiv.1911.08616*, 2019.

[36] M. C. Vuran, A. Salam, R. Wong, and S. Irmak. Internet of underground things in precision agriculture: architecture and technology aspects. *Ad Hoc Networks*, 81:160–173, 2018.

[37] L. Wang, D. Zhang, J. Guo, and Y. Han. Image anomaly detection using normal data only by latent space resampling. *Applied Sciences*, 10(23):8660, 2020.

[38] Y. Yang, S. Xiang, and R. Zhang. Improving unsupervised anomaly localization by applying multi-scale memories to autoencoders. *arxiv.2012.11113*, 2020.

[39] N. Zhang, M. Wang, and N. Wang. Precision agriculture—a worldwide overview. *Computers and Electronics in Agriculture*, 36(2):113–132, 2002.

MOX Technical Reports, last issues

Dipartimento di Matematica

Politecnico di Milano, Via Bonardi 9 - 20133 Milano (Italy)

70/2022   Andrini, D.; Balbi, V.; Bevilacqua, G.; Lucci, G.; Pozzi, G.; Riccobelli, D.
Mathematical modelling of axonal cortex contractility

68/2022   Orlando, G.; Benacchio, T.; Bonaventura, L.
An IMEX-DG solver for atmospheric dynamics simulations with adaptive mesh refinement

67/2022   Alghamdi, M. M.; Boffi, D.; Bonizzoni, F.
A greedy MOR method for the tracking of eigensolutions to parametric elliptic PDEs

69/2022   Franco, N.R; Manzoni, A.; Zunino, P.
Learning Operators with Mesh-Informed Neural Networks

63/2022   Corti, M.; Antonietti, P.F.; Dede', L.; Quarteroni, A.
Numerical Modelling of the Brain Poromechanics by High-Order Discontinuous Galerkin Methods

66/2022   Antonietti, P.F.; Liverani, L.; Pata, V.
Lack of superstable trajectories in linear viscoelasticity: A numerical approach

65/2022   Dassi, F.; Fumagalli, A.; Mazzieri, I.; Vacca, G.
Mixed Virtual Element approximation of linear acoustic wave equation

64/2022   Massi, M.C., Dominoni, L., Ieva, F., Fiorito, G.
A Deep Survival EWAS approach estimating risk profile based on pre-diagnostic DNA methylation: an application to Breast Cancer time to diagnosis

62/2022   Ciaramella, G.; Halpern, L.; Mechelli, L.
Convergence analysis and optimization of a Robin Schwarz waveform relaxation method for periodic parabolic optimal control problems

61/2022   Gregorio, C.; Cappelletto, C.; Romani, S.; Stolfo, D.; Merlo, M.; Barbati, G.
Using marginal structural joint models to estimate the effect of a time-varying treatment on recurrent events and survival: An application on arrhythmogenic cardiomyopathy