Tecnológico de Costa Rica

Escuela de Ingeniería Electrónica

Adversarial Anomaly Detector: Use of Generative Adversarial Networks for the detection of tomato diseases

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Electronics, Major in Digital Signal Processing

Luis Alonso Murillo Rojas

Cartago, December 12th, 2019

Declaro que el presente documento de tesis ha sido realizado enteramente por mi persona, utilizando y aplicando literatura referente al tema e introduciendo conocimientos y resultados experimentales propios.

En los casos en que he utilizado bibliografía he procedido a indicar las fuentes mediante las respectivas citas bibliográficas. En consecuencia, asumo la responsabilidad total por el trabajo de tesis realizado y por el contenido del presente documento.

Luis Alonso Murillo Rojas

Cartago, December 6, 2019

Céd: 2-0696-0826

Instituto Tecnológico de Costa Rica
Escuela de Ingeniería Electrónica
Proyecto de Graduación
Tesis de Maestría
Tribunal Evaluador

Tesis de maestría defendida ante el presente Tribunal Evaluador como requisito para optar por el grado académico de maestría, del Instituto Tecnológico de Costa Rica.

Miembros del Tribunal

M. Sc. Felipe Meza Obando
Profesor Lector

M. Sc. Carl Michael Gruner Monzón
Profesor Lector

Dr. Pablo Alvarado Moya
Profesor Asesor

Los miembros de este Tribunal dan fe de que la presente tesis de maestría ha sido aprobada y cumple con las normas establecidas por la Escuela de Ingeniería Electrónica.

Cartago, December 6, 2019

Instituto Tecnológico de Costa Rica
Escuela de Ingeniería Electrónica
Tesis de Maestría
Acta de Evaluación

Tesis de maestría defendida ante el presente Tribunal Evaluador como requisito para optar por el grado académico de maestría, del Instituto Tecnológico de Costa Rica.

Estudiante: Luis Alonso Murillo Rojas

Nombre del Proyecto: Adversarial Anomaly Detector: Use of Generative Adversarial Networks for the detection of tomato diseases

Miembros del Tribunal Evaluador

M. Sc. Felipe Meza Obando
Profesor Lector

M. Sc. Carl Michael Gruner Monzón
Profesor Lector

Dr. Pablo Alvarado Moya
Profesor Asesor

Los miembros de este Tribunal dan fe de que la presente tesis de maestría ha sido aprobada y cumple con las normas establecidas por la Escuela de Ingeniería Electrónica.

Nota final de la Tesis de Maestría: _____

Cartago, December 6, 2019

Resumen

El tomate es uno de los principales vegetales a nivel mundial debido a su versatilidad de uso y a su impacto económico. Sin embargo, el cambio climático ha provocado que el manejo de plagas y enfermedades sea cada vez más complicado. Es por ello que la implementación de técnicas no invasivas para el diagnóstico temprano de enfermedades en el campo de cultivo representa una solución viable para el control de plagas y enfermedades, evitando efectos secundarios tales como afecciones al medio ambiente. En este proyecto se presenta un estudio de algorítmos semi-supervisados tales como los modelos generativos, con el objetivo de detectar anomalías en grafías de tomate. Además se plantea una propuesta de modelo capaz de detectar anomalias, basandose en las redes generativas adversarias.

Palabras clave: Tomate, aprendizaje profundo, detección de anomalías, auto-codificadores, redes generativas adversarias

Abstract

Tomato is one of the main vegetables worldwide due to its versatility of use and its economic impact. However, climate change has caused the management of pests and diseases to be increasingly complicated. That is why the implementation of non-invasive techniques for the early diagnosis of diseases in the field of crops represents a viable solution for the control of pests and diseases, avoiding side effects such as environmental conditions. This project presents a study of semi-supervised algorithms such as generative models, with the aim of detecting anomalies in tomato spelling. In addition, a model proposal capable of detecting anomalies is proposed, based on adversary generative networks.

Keywords: Tomato, Deep Learning, Anomaly detection, Autoencoders, Generative Adversarial Networks

a Mariángel

Contents

List of Figures ii

Table index iii

List of symbols and abbreviations iv

1 Introduction 1

2 State of the art 3
- 2.1 Related works . . . 3
- 2.2 Methods . . . 4
  - 2.2.1 Deep Feedforward Networks . . . 4
  - 2.2.2 Convolutional Neural Networks . . . 4
  - 2.2.3 Generative models . . . 5
  - 2.2.4 Use of generative models in the anomaly detection . . . 9

3 Proposed solution 11
- 3.1 Data preprocessing . . . 11
- 3.2 Architecture experiment . . . 11
- 3.3 Contribution . . . 12

4 Results and analysis 13

5 Conclusions 20

Bibliography 21

A Capture protocol 24
- A.1 General considerations . . . 24
- A.2 Video considerations . . . 25
- A.3 Instructions for capturing the video . . . 25
  - A.3.1 Color calibration . . . 26
  - A.3.2 Proper use of the gimbal . . . 26
  - A.3.3 Pre-capture application settings . . . 27

List of Figures

2.1 Convolutional Neural Network architecture 5
2.2 Autoencoder architecture 6
2.3 Generative Adversarial Network architecture 8
2.4 AnoGAN 9
2.5 GANomaly 10

4.1 Reconstruction example of the adversarial anomaly detector 13
4.2 Reconstruction example of the sVAE 14
4.3 Reconstruction example of the GM-VAE 14
4.4 GAN training 15
4.5 Experiment test images 16
4.6 Reconstruction evaluation of the Adversarial Anomaly Detectof, sVAE and GM-VAE 17
4.7 Adversarial Anomaly Detector t-SNE evaluation of test image 1 18
4.8 Adversarial Anomaly Detector t-SNE evaluation of test image 2 19

A.1 Tomato crop 24
A.2 Color calibration 25
A.3 Whitebalance configuration 26

Table index

4.1 Reconstruction time evaluation 15
4.2 Reconstruction metric evaluation 16

List of symbols and abbreviations

Abbreviations

AAD	Adversarial Anomaly Detector
CNN	Convolutional Neural Networks
GAN	Generative Adversarial Networks
GM-VAE	Gaussian-Mixture Variational Autoencoder
KL	Kullback-Leibler
MLP	Multilayer perceptrons
MVN	Matrix-variable normal distribution
STD	Standard deviation
sVAE	Spatial Variational Autoencoder
VAE	Variational Autoencoder

General notation

$\mathbf{A}$	Matrix. $\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nm} \end{bmatrix}$
$\mathbb{C}$	Complex numbers set.
$\mathrm{Im}(z)$ o z_{Im}	Imaginary part of the complex number z
j	$j = \sqrt{-1}$
$\mathrm{Re}(z)$ o z_{Re}	Real part of the complex number z
$\mathcal{T}[\cdot]$	Transformation performed by a system
$\underline{\mathbf{x}}$	Vector. $\underline{\mathbf{x}} = [x_1\ x_2\ \ldots\ x_n]^T = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$
y	Scalar.
z^*	Conjugate Complex of z

Chapter 1

Introduction

The diseases and pests control are becoming more relevant in the latest years due to the consequences that climate change brings, sucha as [8] the alteration of stages and rates of development of pathogens, the modification of host resistance and changes in the physiology of host-pathogen interactions. These issues cause changes in the geographical distribution and growth of plant species.

All these situations have the potential to cause catastrophic plant diseases, leading to the loss of food crops, which according to [27] aggravates the deficit of food supply. In this sense, the tomato has a key role, due to its impact in the diet of people. As stated in [6], the tomato represents the most economically important vegetable crop worldwide due to the precense in the diet of millions of people, hence being one of the most produced vegetables after potatoes and before onions.

In [23] is mentioned the use of non-invasive methods to deal with crop disease. Normally, to combat this problem, the farmers use chemical agents as pesticides which has environmental consequences and health issues to the people that consume the vegetable.

In Costa Rica, the disease that causes the most damage to the tomato crops is the *Phytophthora infestans* and the most important pest in the whiteflies [3].

In the development of automated applications for disease detection requires the use of a labeled dataset, which implies invest time consulting with experts for the labeling process. For that reason, this work explores methods to reduce this time as much as possible. The approach followed is the detection of anomalies where a set of data, in its majority, represents the healthy plants and the outliers can be seen as the affected plants or the anomalies.

The general goal of this project is the evaluation of different algorithms for anomaly detection, in the context of tomato images. The specific goals are the following:

- To design a dataset to train the machine learning models.
- Selection of an anomaly detection algorithms.

- To evaluate the selected methods for detection anomaly in tomato images.

The rest of the work is structured as follows:

The state of the art chapter 2 covers related works for the detection of anomalies in different contexts like in agriculture or health care; as well as additional concepts that underline this project.

In chapter 3 the solution strategy to evaluate different deep learning architectures able to detect anomalies is described.

Chapter 4 presents a discussion of the performance of the different architecture and which of them are the best option for the detection of diseases in tomato.

Finally, the conclusions and the future work are summarized in chapter 5.

Chapter 2

State of the art

2.1 Related works

The use of deep learning in agriculture has a great pontential in a variety of applications such as soil mapping, crop type classification, crop monitoring, pest detection and management, among others [17]. Most of the work related to the detection of diseases and pests in tomato make use of deep learning techniques, and specifically the supervised learning algorithms. In [2] a proposal that makes use of CNN classifiers and object detectors like Faster R-CNN, Region-based Fully Convolutional Network or Single Shot Multibox Detector for the disease detections in tomato in South Korea. One of the problems that this project faced was the false positives in the classification. In [13] a filter is proposed to reduce this issue.

In [4] the use of an adversarial autoencoder for the event detection in applications such as public security, health monitoring or intrusion detection. This corresponds to an unsupervised generative method just able to replicate regular data. This method has the advantage that the network is able to learn the main features that represent normal data.

In [5] a new architecture proposal is presented to combine the variational autoencoder (VAE) and the generative adversarial networks, for the detection of anomalies in MR brain images. This new architecture is compared with other approaches like general VAE, spatial VAE, and AnoGAN. The authors claim that their VAEGAN architecture outperforms the other architectures.

2.2 Methods

2.2.1 Deep Feedforward Networks

The Deep Feedforward Networks, also known as multilayer perceptrons (MLPs), is the simplest structure of artifitial neural network. The problem that this kind of network tackles is the approximation of some function f^*. As mentioned in [15], a common example is in the implementation of a classifier where $y = f^*(x)$ maps the input x to some category represented in y. In order to achieve that, the feedforward networks defines a mapping of the form $y = f(x, \theta)$, and during the training, this network adjusts the set of parameters θ that best approximate the function.

These networks are typically composed of many different functions, which represent the layers of the network. For example, we could have four layers that are connected in cascade, in the form $f(x) = f_{(4)}(f_{(3)}(f_{(2)}(f_{(1)}(x))))$. The length of these chains of functions can be seen as the depth of the network, where $f_{(1)}$ is the first layer, also known as the input layer, and $f_{(4)}$ is the last layer or the output layer. Between the first and the output layers are the hidden layers. During training, generally the output values corresponding to some input are explicited provided, but it is not specified what values the hidden layers should have. The learning algorithm decides how to use those hidden layers to generate the desired output.

2.2.2 Convolutional Neural Networks

The convolutional neural networks (CNN) [20] are an extension of the deep feedforward networks, that are specialized in the processing of grid-like data. As its name suggests, this kind of network makes use of the mathematical operation called convolution. The convolution is an operation on two functions of a real-valued argument and is defined as follows:

$$s(x,y) = \sum_u \sum_v x(u,v)w(x-u,y-v) \tag{2.1}$$

In terms of CNN, the first argument of the convolution ($x(u,v)$) is referred as the input and the second argument ($w(x-u,y-v)$) as the kernel. The output of this operation is known as a feature map. The common architecture of a CNN, as shown in figure 2.1, is composed of a convolutional layer, then a pooling layer (which subsampled the data) and in the end, there is typically a fully connected network.

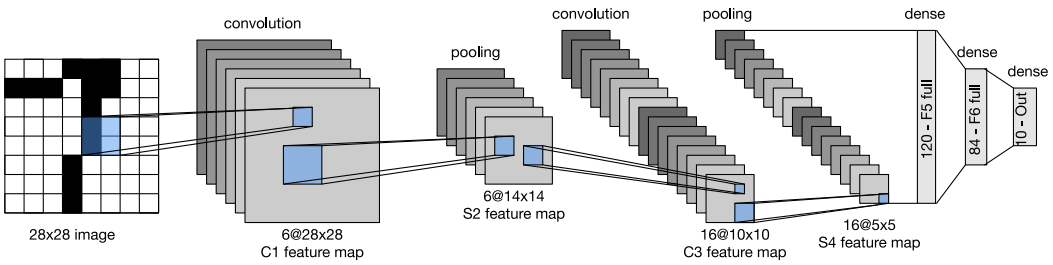


Figure 2.1: Convolutional Neural Network architecture. Reprinted from [9].

2.2.3 Generative models

In machine learning, there are mainly three types of learning algorithms: supervised learning algorithms, unsupervised learning algorithms, and semi-supervised learning algorithms. The supervised learning algorithms make use of labeled data for its training process. In the last years, this kind of algorithm is presenting significant results in applications like object detection [21].

The unsupervised learning algorithms are the ones that do not need a structured dataset and they can find the underlying pattern by itself. One particular type of semi-supervised learning algorithms are the generative models which work in a specific domain data. The generative models are algorithms that are capable of learning the probabilistic data distribution of some datasets, allowing them to generate new samples similar to the ones in that dataset. Two types of generative models are the variational autoencoders (VAE) and the generative adversarial networks (GAN).

Autoencoders

In a similar way as the CNNs, the autoencoders are an extension of the deep feedforward network, with the difference that in this case, the goal is to replicate the input in the output. This type of networks are divided in two parts: the encoder that tries to generate a latent space h where some features are extracted from the input data $h = f(x)$; and the decoder that takes the latent space generated by the encoder as input and reconstructs the data $r = g(h)$. One of the main applications of this type of neural network is the dimensionality reduction, image compression, image denoising or image generation. The image generation case is covered in more detail in the next section with the variational autoencoders as part of the generative models.

Variational Autoencoder

The objective of the variational autoencoders [19] is the generation of data samples from

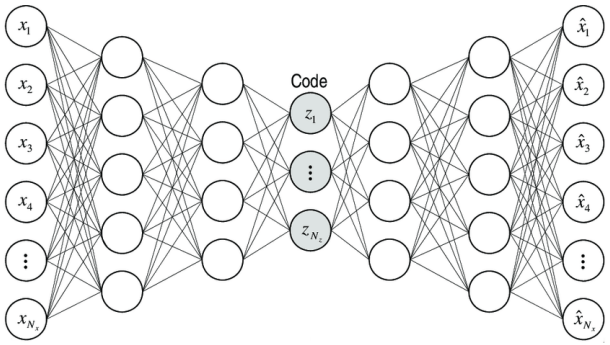


Figure 2.2: Autoencoder architecture. Reprinted from [7].

a learned latent space. This latent space is obtained from a large dataset. In order to achieve this generation process, these autoencoders must try to learn the probability distribution $P(x)$ of the data.

From a probabilistic perspective, the latent variables will be drawn from a prior $P(z)$ and the generated data has a likelihood of $P(X|z)$ that is conditioned by the latent space. So the goal here is to model the data distribution as follows:

$$P(X) = \sum_{z} P(X|z)P(z) \tag{2.2}$$

However, this integral is computationally untractable, due to the impossibility of computing all elements in the latent space. To avoid this, VAEs try to infer the distribution $P(x)$ from data using $P(z|X)$. Variational inference approximates the distribution $\mathbb{P}(z|X)$, using a simpler distribution, where a common choice is a Gaussian distribution. Then, with a parametric inference model $Q(z|X)$ that maps the input data into the latent space; the difference between the distribution $P(z|X)$ and $Q(z|X)$ is calculated using the Kullback-Leibler divergence.

$$\begin{aligned} D_{KL}(Q(z|X)\|P(z|X)) &= \sum_{z} Q(z|X) \log \frac{Q(z|X)}{P(z|X)} \\ &= \mathbb{E}\left[\log \frac{Q(z|X)}{P(z|X)}\right] \\ &= \mathbb{E}[\log Q(z|X) - \log P(z|X)] \end{aligned} \tag{2.3}$$

Using $P(z|X) = \frac{P(X|z)P(z)}{P(X)}$, 2.3 can be rewritten as:

$$\begin{aligned} D_{KL}(Q(z|X)\|P(z|X)) &= \mathbb{E}\left[\log Q(z|X) - \log \frac{P(X|z)P(z)}{P(X)}\right] \\ &= \mathbb{E}[\log Q(z|X) - \log P(X|z) - \log P(z) + \log P(X)] \end{aligned} \tag{2.4}$$

$P(x)$ does not depend on z, hence it can be taken out of the expectation:

$$\begin{aligned}\implies \log P(X) - D_{KL}(Q(z|X)\|P(z|X)) &= \mathbb{E}[\log P(X|z)] - \mathbb{E}[\log Q(z|X) - \log P(z)] \\ &= \mathbb{E}[\log P(X|z)] - D_{KL}(Q(z|X)\|P(z))\end{aligned} \tag{2.5}$$

Since the D_{KL} is always positive, 2.5 can be written as:

$$\log P(X) \geq \mathbb{E}[\log P(X|z)] - D_{KL}(Q(z|X)\|P(z)) \tag{2.6}$$

The first term of the loss function can be seen as the reconstruction error and the second term corresponds to the KL error [12].

Spatial Variational Autoencoder

Consist of an improvement of the typical variational autoencoder. Whereas in the classic VAEs the latent space are vectors where their components have a dimension of 1x1, in the spatial VAE the idea is to extend these latent variables to have a higher dimension and, in that way, to be able to capture more spatial features of the input data.

In [28] a spatial variational autoencoder is proposed, where the latent variables are sampled from a matrix-variable normal (MVN) distribution. The authors claim that this architecture outperforms the original VAEs due to the capture of richer structural and spatial information from data.

Gaussian-Mixure Variational Autoencoder

This architecture corresponds to another variant of the VAEs models. In this case, the prior distribution $\mathbb{P}(z)$ is a Gaussian-mixture that allows the net to perform unsupervised clustering of the data. This autoencoder, proposed in [11], has the potential of grouping the data, and each group can represent or share a specific feature of the original data. It has a competitive performance in comparison with the regular VAEs.

Generative adversarial networks

The generative adversarial networks (GAN) were proposed by Ian Goodfellow [14] in 2014. The basic idea behind GANs is in the contest of two models: on one side, there is the generator G that tries to learn the probability distribution of the data and for the other side, a discriminator D that decides if the input data is real or generated by G. The goal of the generator is to try to create images as real as possible that provokes the discriminator to make mistakes. This game is described as the minmax value function in 2.7.

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}\,(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))] \tag{2.7}$$

A well trained GAN model is able to reach the Nash equilibrium, where the discriminator

has an accuracy of around 0.5, which means that it is not able to discern between fake or real data; and the generator should reach value loss of approximately 0.7.

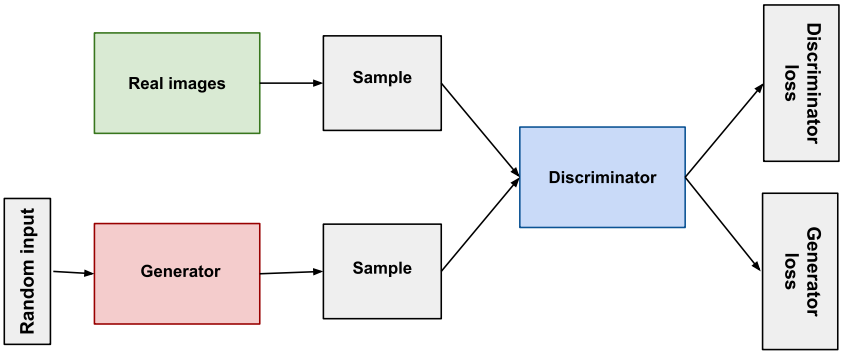


Figure 2.3: Generative Adversarial Network architecture. Reprinted from [1].

One of the challenges that the GANs face is precisely the training process, since depending on the initial conditions the Nash equilibrium is never reached. If the capacity of the model is too small, the model is susceptible to collapse. Another possible scenario is when the learning rate of the model is too aggressive, causing that the net never converges.

Architectures based on GANs

In [24] a series of architectures based on GANs is presented, along with some metrics to evaluate their performance.

- Convolution base GAN: The original GAN is implemented based on the multi-layer perceptron but it has been proven that CNN are better than the MLP in extracting features to the images. This kind of network is known as Deep Convolutional Generative Adversarial Network (DCGAN).

- Conditional GAN: Normally, the generator in the GAN receives as input some random noise, which sometimes makes the model prone to collapse. It is for this reason that in the conditional GAN a variable C is introduced as input to the generator and also to the discriminator, with the objective of add some constraints and, therefore, to have more control in the latent space. The type of constraint will depend on the type of data that the GAN is dealing with.

- Autoencoder based GAN: This type of GAN architecture is presented in [22] where the idea is to make use of an adversarial training to the autoencoder performing variational inference by matching the aggregated posterior latent space of the autoencoder with an arbitrary prior distribution. This allows to overcome one of the main challenges of the autoencoders, when it is not able to correctly learn the data distribution.

2.2.4 Use of generative models in the anomaly detection

The generative models discussed so far, find application in the detection of anomalies. The general idea is to train a model capable of learning the data distribution of a regular dataset. In that way, the model should be able to reconstruct o generate a query image as similar as possible and then compare the original with the reconstructed images. If for some reason that is not the case, there is a high probability that the regions that the model was not able to generate, correspond to an anomaly. All the autoencoders described before can be used in this way, as well as the GANs.

For the specific case of the GANs, in [10] different GAN based architectures is presented, with the purpose of anomaly detection:

AnoGAN

This GAN is first trained with just regular data and is intended to learn the manifold of the data x. Then, with the generator trained, each time that some image has to be evaluated, an iterative process is performed in order to find the latent variables that generate the output $G(x)$ most similar to the query image. This iterative process has the disadvantage that is too time-consuming [26].

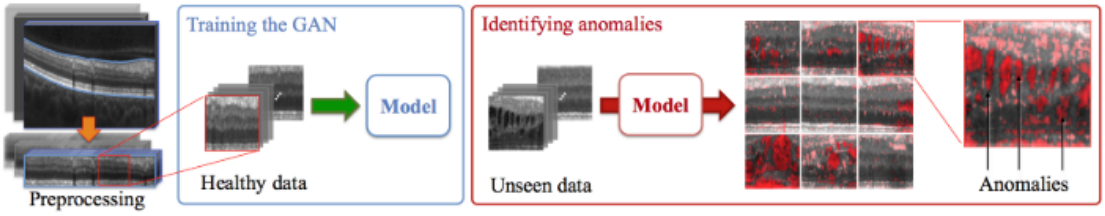


Figure 2.4: AnoGAN traing with healty images first and then reconstruction of unseen images. Reprinted from [26].

GANomaly

This architecture is inspired by the AnoGan but tries to overcome the long detection times. In order to do that, it makes use of an encoder that is able to learn the latent space variables that the generator receives during the GAN training. This has the advantage of having a faster GAN training and reduces the times to generate a similar image to the query image. The generator also has an encoder at the end of its structure that helps during training to learn the manifold of the input data x.

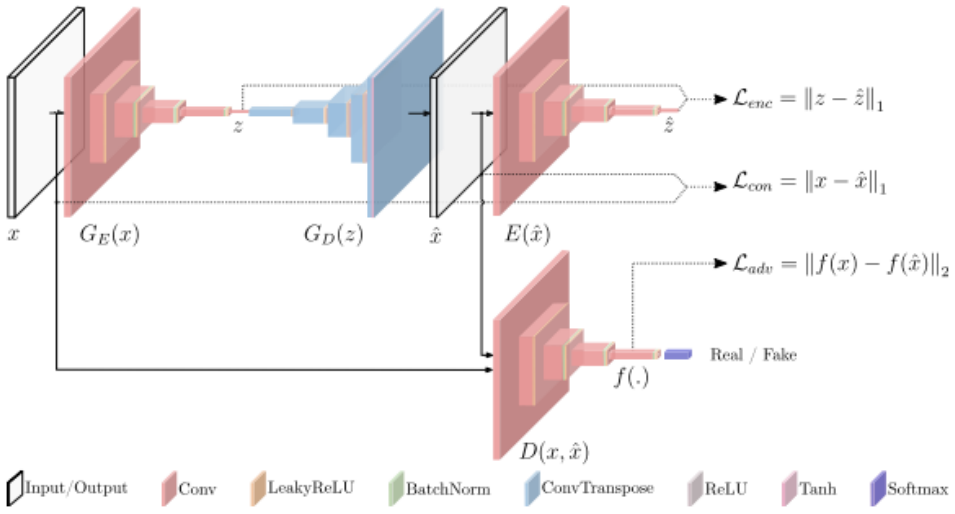


Figure 2.5: GANomaly architecture and loss functions. Reprinted from [10].

Chapter 3

Proposed solution

3.1 Data preprocessing

The data acquired for this project used a camera of an iPhone 8 that captures in a resolution of 1920×1080. The tomato data was captured in tomato crops located in the Alajuela province of Costa Rica (10°01'39.2"N 84°18'31.7"W). In the appendix A is described the capture protocole.

The comparison of anomaly detection models uses 191520 images with a size of 50×50 each. For this set of images, 80% was used for testing and 20% used for validation. This training and validation data contains only healthy tomato plants. A set of 798 images is used for testing, and in this case, it also depicts unhealthy tomato plants.

Before using these images for training the models, a data normalization was applied to linearly map the image values to the range between -1 to 1. Additionally, image data augmentation as rotation, zoom, width shift, among other image transformations were implemented.

3.2 Architecture experiment

In order to evaluate the selected architectures for anomaly detection in tomato plants, an experiment of three architectures are proposed: one to explore the adversarial anomaly detector architecture, another to evaluate the spatial variational autoencoder and a third experiment to make use of the gaussian-mixture variational autoencoder.

All the architectures use the same dataset and for each one its reconstruction of an input image is tested by measuring the distance between the original and reconstructed images using the mean square error (MSE). The expected behavior here is that, as the model is only trained with healthy images of tomato, if some input image presents a possible disease, the model should not be able to reconstruct the affected area, and that

reconstruction error or dissimilarity is an indication of the presence of an anomaly.

3.3 Contribution

The architecture proposed in this project is an adversarial anomaly detector inspired in the AnoGAN. This architecture is composed of the typical generator and discriminator of a GAN, but with the addition of an encoder $E(x) = z$ that is connected with the generator input.

The training process of this approach has two stages: First the the GAN is trained with the conventional methods, but exclusivelly using healthy images of tomato, where the generator receives random noise as input for the latent variables. The second stage is training the encoder that has to learn how to map the query input image into the latent space.

During training of the encoder, the generator and discriminator models remain unchanged. Also, the hidden layers of the discriminator are used to extract features of the synthetic images and the input images. The goal is to train the encoder in a way that it forces the generator to create images with similar features as the ones of the query image.

The proposed loss function for this anomaly detector model is the following:

$$\mathcal{L}\left(\mathbf{E}(x)\right) = (1-\lambda)\cdot\mathcal{L}_R\left(\mathbf{E}(x)\right) + \lambda\cdot\mathcal{L}_D\left(\mathbf{E}(x)\right) \tag{3.1}$$

where λ weights both terms of the function and $\mathbf{E}$ is the Encoder. $\mathcal{L}_R$ (reconstruction error) and $\mathcal{L}_D$ (discriminator error) are defined as:

$$\mathcal{L}_R\left(\mathbf{E}(x)\right) = \|\mathbf{x} - G\left(\mathbf{E}(x)\right)\| \tag{3.2}$$

$$\mathcal{L}_D\left(\mathbf{E}(x)\right) = \|\mathbf{f}(\mathbf{x}) - \mathbf{f}\left(G\left(\mathbf{E}(x)\right)\right)\| \tag{3.3}$$

where $\mathbf{f}(x)$ represents the feature extractor of the discriminator.

This approach overcomes one of the disadvantages of the AnoGAN that is its long computational time. This modification of the AnoGAN architecture is similar to the one presented in [25].

Chapter 4

Results and analysis

From a qualitative evaluation of the three architectures: Adversarial Anomaly Detector, sVAE, and GM-VAE; the one presenting the best reconstruction is the Adversarial Anomaly Detector. For the two autoencoders, the reconstructions suffer from blurring (see figures 4.2 and 4.3), making it difficult to evaluate the presence of an anomaly and hence confirming one of the main problems of this kind of autoencoders. The problems in the autoencoders could the caused for the bias the learned data distribution has, due to the simpler data distributions underlying the model.

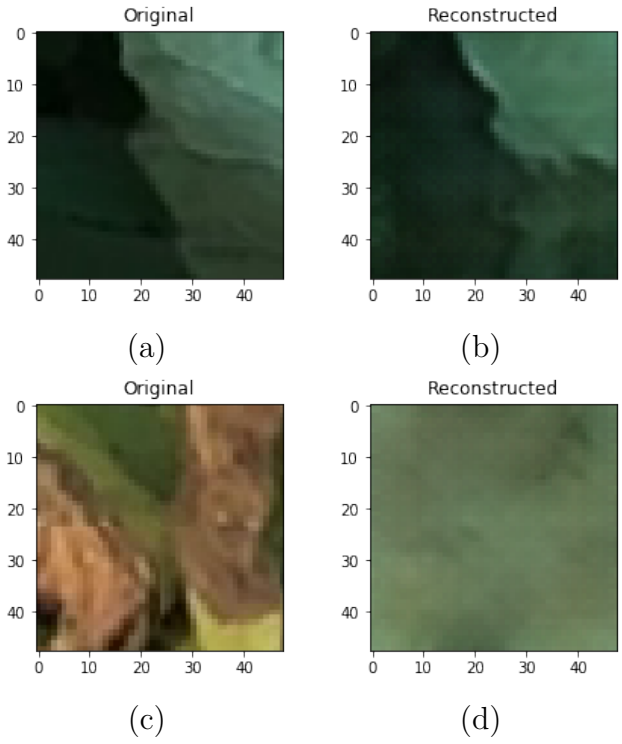


Figure 4.1: Reconstruction example of the adversarial anomaly detector: a) origianl healthy sample, b) reconstructed healthy sample, c) original test sample and d) reconstructed test sample.

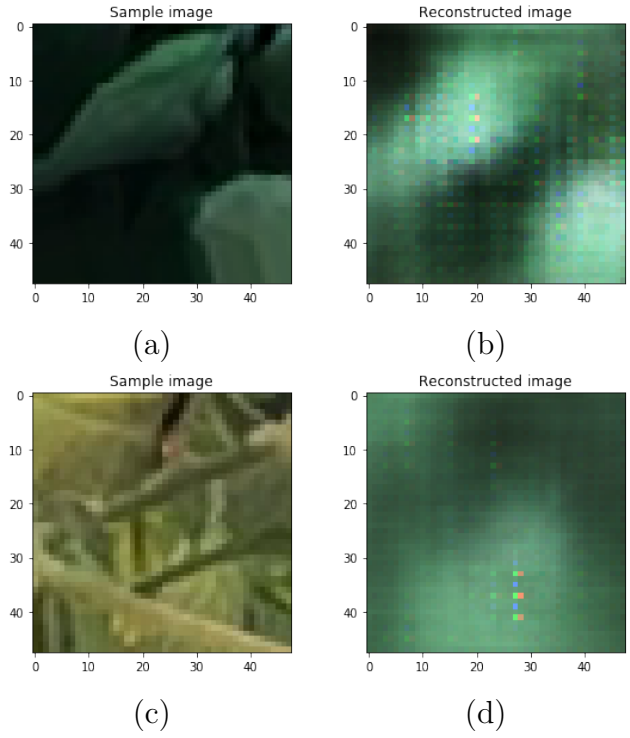


Figure 4.2: Reconstruction example of the sVAE: a) original healthy sample, b) reconstructed healthy sample, c) original test sample and d) reconstructed test sample.

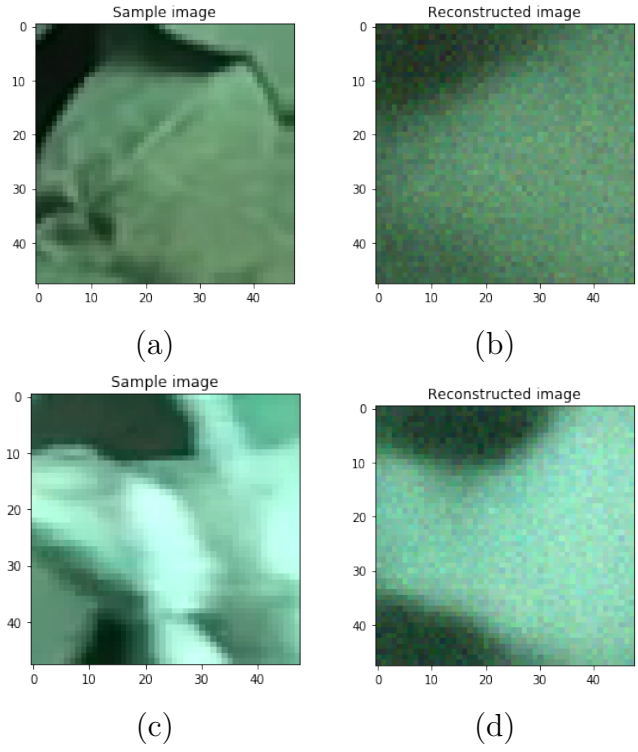


Figure 4.3: Reconstruction example of the GM-VAE: a) original healthy sample, b) reconstructed healthy sample, c) original test sample and d) reconstructed test sample.

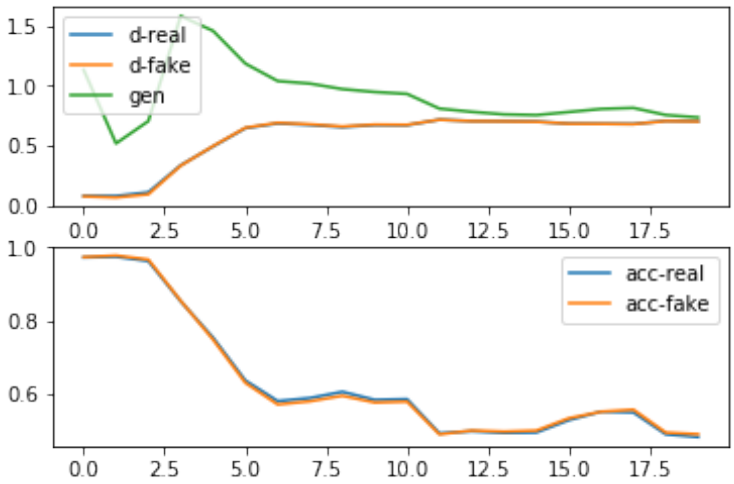


Figure 4.4: GAN training: the top graph shows the loss of the generator and the discriminator. The bottom graph shows the accuracy of the discriminator for real and fake images.

The group of healthy samples are extracted from figure 4.5.a and for the anomalies samples are extracted from figure 4.5.b. In the case of the AnoGAN, the training of the model is unstable, but with the available tomato data, it was possible to achieve the Nash equilibrium (see figure 4.4). The generated images of the AnoGAN are better than the ones in the autoencoders as can be seen in table 4.2 with the values of mean and standard deviation (STD) of the healthy and anomaly samples. Its main disadvantage is the time needed to learn a set of latent variables to reconstruct some images. This problem is resolve with the introduction of an encoder in the AnoGAN architecture as shown in the table 4.1 where the reconstruction times are shown for the different models.

Table 4.1: Reconstruction time evaluation.

Model	Reconstruction time (ms)
sVAE	160.8
GM-VAE	1383.1
AnoGAN	6320.6
Adversarial Anomaly Detector	255.3

Figure 4.6 has the reconstruction scores distribution of regular and anomalous samples for the three models. This figure also shows that the best model for the reconstruction of regular samples is the Adversarial Anomaly detector, however, there are still some samples that are indiscernible between regular or anomalous. For that reason explore more metrics would be needed.

Table 4.2: Reconstruction metric evaluation.

Model	Mean (healthy)	STD (healthy)	Mean (anomalies)	STD (anomalies)
sVAE	10410.6	7020.3	19858.8	18464.1
GM-VAE	15162.2	7597.3	9357.3	6071.7
AAD	3839.5	3251.1	8996.1	3308.8

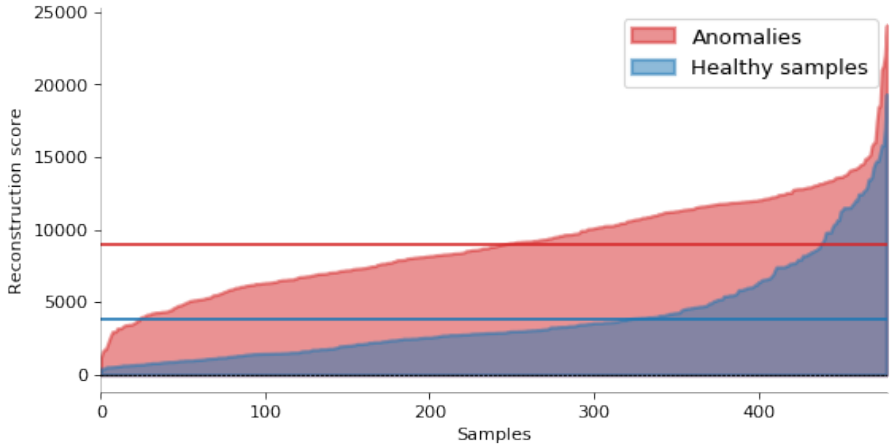
(a)

(b)

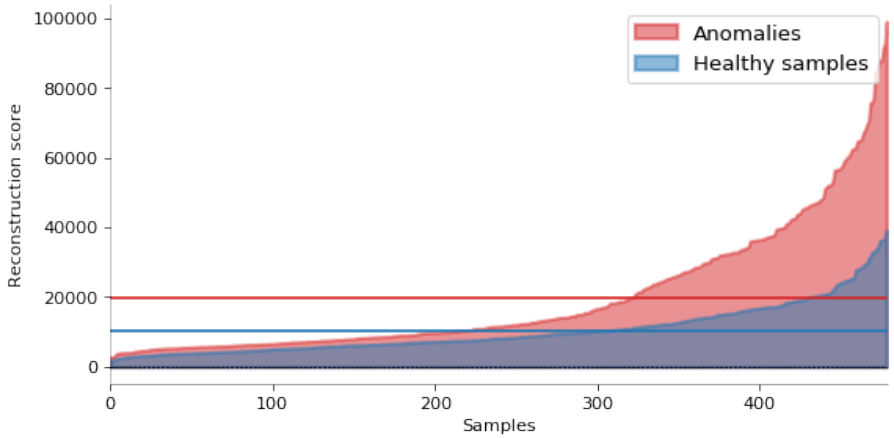
Figure 4.5: Experiment test images: a) healthy image and b) anomlous image.

In figures 4.7 and 4.8 depict the data distribution between regualr and anomalous samples. Figure 4.7 shows a defined group of data that represents the healthy samples and a smaller amount of points around this healthy group that represents anomalous data. In figure 4.8 we have the opposite case, where the number of anomalies is larger than the number of healthy samples. For the first case, the test images depict a tomato plant in a healthy

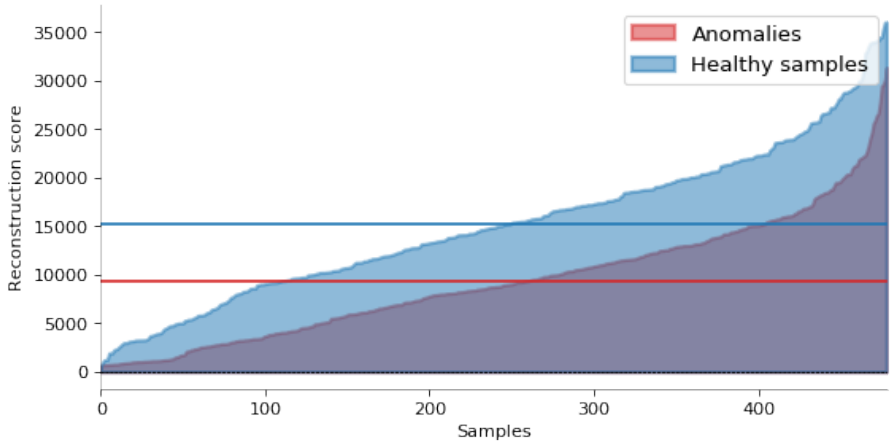
state, and for the second case, the tomato plant presented several affections, fact that is reflected in each of the graphs mentioned before.



(a)



(b)



(c)

Figure 4.6: Reconstruction evaluation of the a) Adversarial Anomaly Detectof, b) sVAE and c) GM-VAE. The horizontal lines represents the mean value of the healty samples and the anomaly samples.

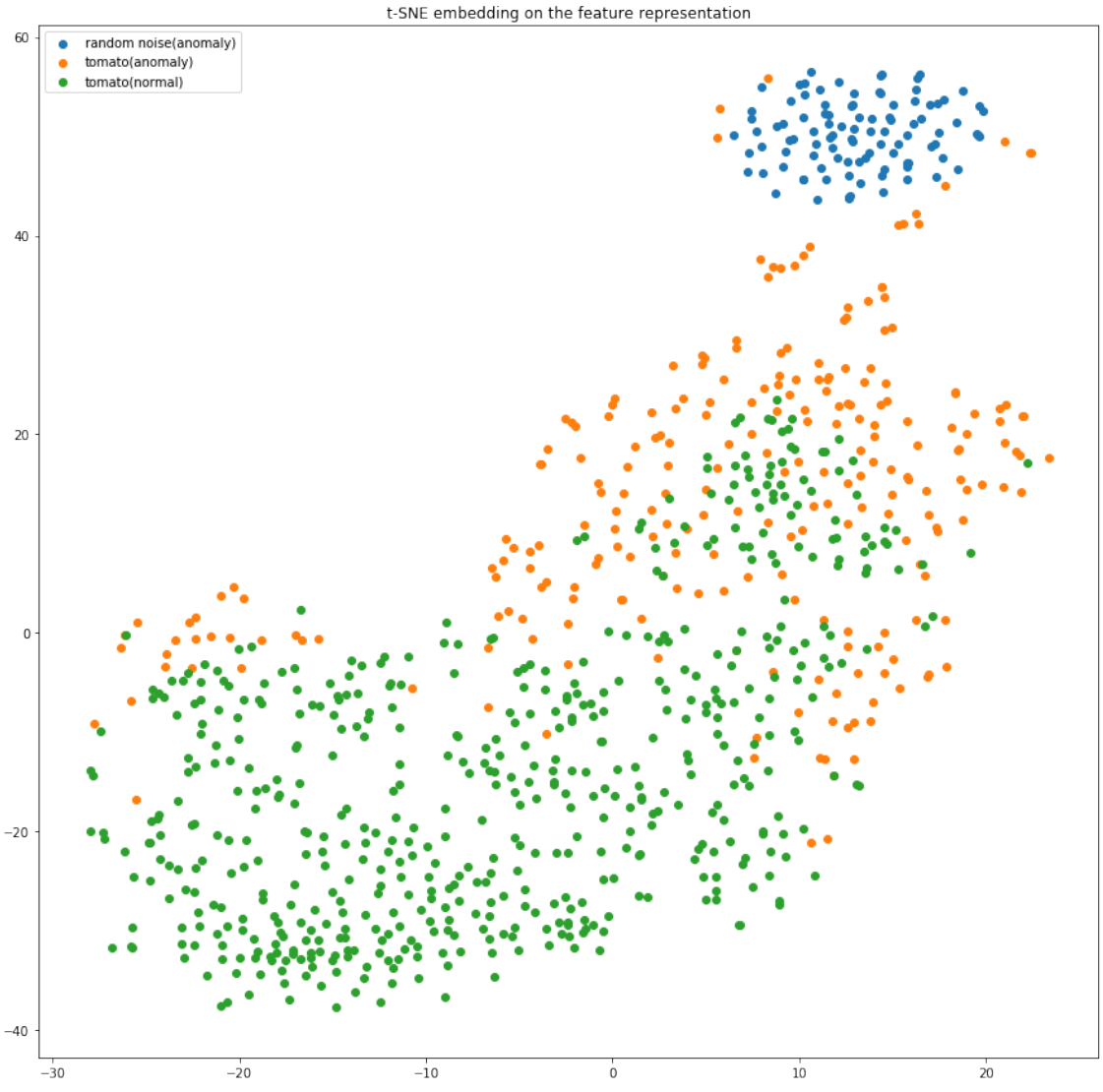


Figure 4.7: Adversarial Anomaly Detector t-SNE evaluation of test image 1

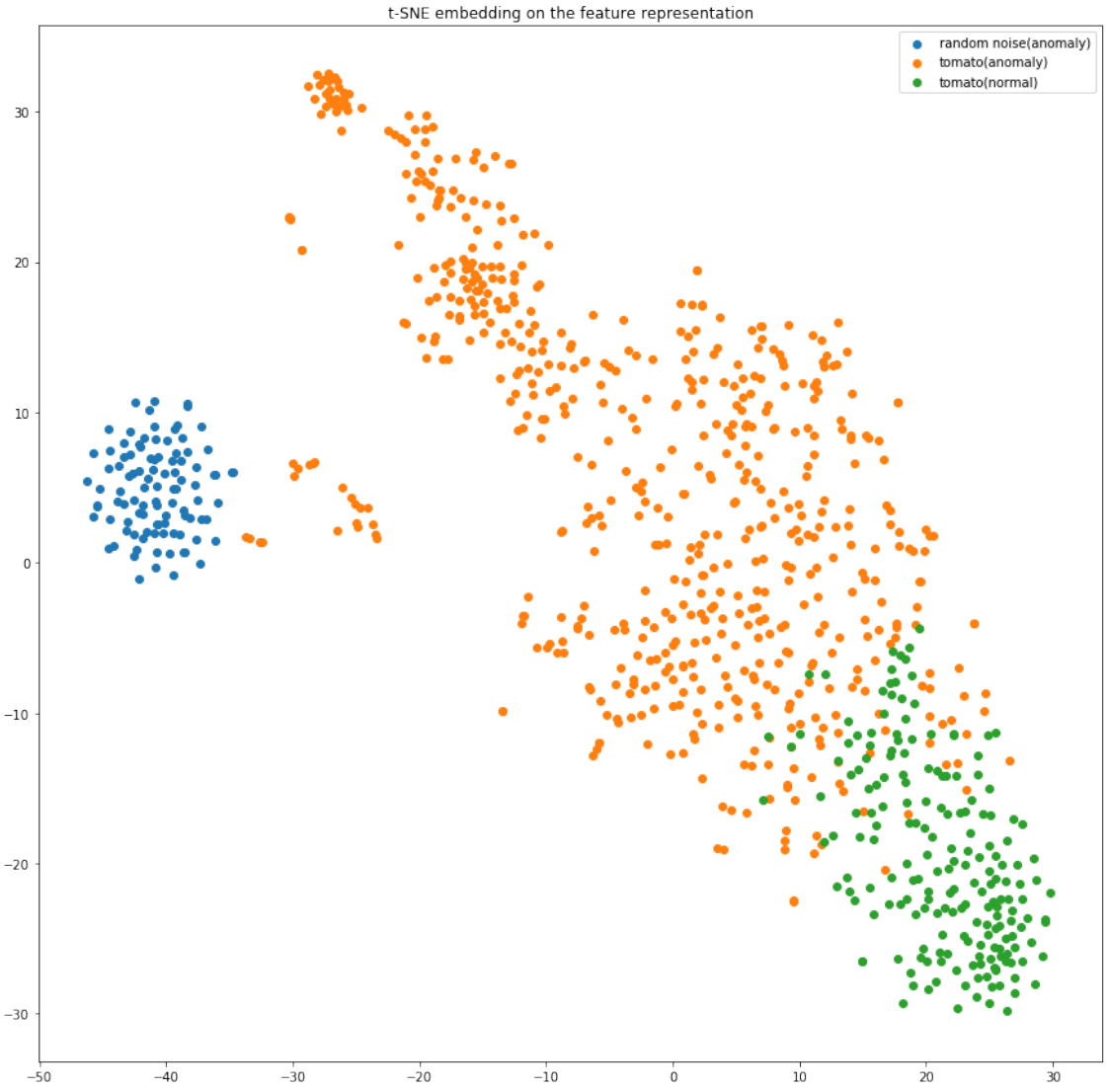


Figure 4.8: Adversarial Anomaly Detector t-SNE evaluation of test image 2

Chapter 5

Conclusions

The experiments performed so far have shown a tendency of the variational autoencoder architectures to blur the reconstructed images. On the other side the AnoGAN has more promising results with better reconstruction images.

A modification to the AnoGAN architecture is proposed, allowing to considerably improve the reconstruction time.

As future work, the objective must be to improve the generator of the GAN. One possible strategy is in the implementation of a disentangling representation of the generator latent space [16]. Another possible approach is the use of a regularized in the latent space of the generator, in a similar way as proposed in [18]. In order to validate the possible improvement in the generator, metrics that evaluate the GAN performance needs to be explored, along with metrics for the evaluation of the anomalies.

Finally, a segmentation process should be implemented in order to mark possible anomalous regions.

Bibliography

[1] . [Online]. Available: https://developers.google.com/machine-learning/gan/gan_structure.

[2] "A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition", *Sensors (Switzerland)*, 2017, ISSN: 14248220.

[3] M. de Agricultura y Ganadería de Costa Rica, "Agrocadena de tomate", pp. 1–80, 2007. [Online]. Available: www.mag.go.cr.

[4] by Asimenia Dimokranitou, G. Tsechpenakis, J. Yu Zheng, and M. Tuceryan, "Adversarial Autoencoders for Anomalous Event Detection", *Master thesis of Purde University*, no. May, 2017, ISSN: 15264602. DOI: 10.1021/acs.biomac.8b00588.

[5] C. Baur, B. Wiestler, S. Albarqouni, and N. Navab, "Deep autoencoding models for unsupervised anomaly segmentation in brain MR images", in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11383 LNCS, 2019, pp. 161–169, ISBN: 9783030117221.

[6] V. Bergougnoux, "The history of tomato: From domestication to biopharming", *Biotechnology Advances*, vol. 32, no. 1, pp. 170–189, 2014, ISSN: 07349750.

[7] S. W. Canchumuni, A. A. Emerick, and M. A. C. Pacheco, "Towards a robust parameterization for conditioning facies models using deep variational autoencoders and ensemble smoother", *Computers Geosciences*, vol. 128, pp. 87–102, 2019. DOI: 10.1016/j.cageo.2019.04.006.

[8] S. M. Coakley and H. Scherm, "Cambio Climatico Y Manejo De Las Enfermedades En Plantas", pp. 399–426, 1999.

[9] "Convolutional neural networks", *Convolutional Neural Networks in Visual Computing*, pp. 89–116, 2017. DOI: 10.4324/9781315154282-4.

[10] F. Di Mattia, P. Galeone, M. De Simoni, and E. Ghelfi, "A Survey on GANs for Anomaly Detection", 2019. arXiv: 1906.11632.

[11] N. Dilokthanakul, P. A. M. Mediano, M. Garnelo, M. C. H. Lee, H. Salimbeni, K. Arulkumaran, and M. Shanahan, "Deep Unsupervised Clustering with Gaussian Mixture Variational Autoencoders", 2016. arXiv: 1611.02648.

[12] C. Doersch, "Tutorial on Variational Autoencoders", 2016. arXiv: 1606.05908.

[13] A. F. Fuentes, S. Yoon, J. Lee, and D. S. Park, "High-Performance Deep Neural Network-Based Tomato Plant Diseases and Pests Diagnosis System With Refinement Filter Bank", *Frontiers in Plant Science*, vol. 9, p. 1162, 2018. [Online]. Available: www.frontiersin.org.

[14] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets", in *Advances in Neural Information Processing Systems*, vol. 3, 2014, pp. 2672–2680.

[15] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT Press, 2017.

[16] I. Higgins, L. Matthey, X. Glorot, A. Pal, B. Uria, C. Blundell, S. Mohamed, and A. Lerchner, "Early Visual Concept Learning with Unsupervised Deep Learning", 2016. [Online]. Available: http://arxiv.org/abs/1606.05579.

[17] A. Kamilaris and F. X. Prenafeta-Boldú, "Deep Learning in Agriculture: A Survey", Tech. Rep.

[18] T. Karras, S. Laine, and T. Aila, "A Style-Based Generator Architecture for Generative Adversarial Networks", Dec. 2018.

[19] D. P. Kingma and M. Welling, "Auto-encoding variational bayes", in *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*, 2014. arXiv: 1312.6114.

[20] Y. Lecun, "Object Recognition with Gradient Based Learning", AT&T Shannon Lab, Tech. Rep., 1999.

[21] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen, "Deep Learning for Generic Object Detection: A Survey", *International Journal of Computer Vision*, 2019, ISSN: 15731405. DOI: 10.1007/s11263-019-01247-4.

[22] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, "Adversarial Autoencoders", 2015. arXiv: 1511.05644.

[23] D. S. Park, "Characteristics of Tomato Plant Diseases", 2017.

[24] "Recent Progress on Generative Adversarial Networks (GANs): A Survey", *IEEE Access*, vol. 7, pp. 36 322–36 333, 2019, ISSN: 21693536.

[25] T. Schlegl, P. Seeböck, S. M. Waldstein, G. Langs, and U. Schmidt-Erfurth, "f-AnoGAN: Fast unsupervised anomaly detection with generative adversarial networks", *Medical Image Analysis*, vol. 54, pp. 30–44, May 2019, ISSN: 13618423.

[26] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery", in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10265 LNCS, 2017, pp. 146–147, ISBN: 9783319590493.

[27] R. N. Strange and P. R. Scott, "Plant Disease: A Threat to Global Food Security", *Annual Review of Phytopathology*, vol. 43, no. 1, pp. 83–116, 2005, ISSN: 0066-4286.

[28] Z. Wang, H. Yuan, and S. Ji, "Spatial variational auto-encoding via matrix-variate normal distributions", in *SIAM International Conference on Data Mining, SDM 2019*, 2019, pp. 648–656, ISBN: 9781611975673. DOI: 10.1137/1.9781611975673.73. arXiv: 1705.06821.

Appendix A

Capture protocol

The following document describes the capture protocol proposed for the thesis project Adversarial Anomaly Detector, which aims to be a guide to be followed by potential collaborators. It is important that the collaborator follow this guide in order to have the best data quality for the project.

A.1 General considerations

The data will be captured in the tomato crop field. Typically these plantations are divided into a large number of grooves, so it is intended to create videos that cover each of these grooves. The distance between the groove ranges from 1.8 to 2 meters. Figure [FIGURE] shows an example of a tomato plantation.

Figure A.1: Tomato crop in Costa Rica.

A.2 Video considerations

- The video resolution must be 1920x1080 pixels.
- Make use of a video stabilizer like a Gimbal. (For example the DJI Osmo Mobile 2)
- The video must be recorded in color.
- The duration of the video will depend on the length of the groove.
- The distance of the chamber from the plants should be approximately 1.5 meters.
- The video should cover the entire structure of the plant, from its base to the highest branches. If it is not possible to cover the entire plant in the same video, it will be done in different videos, covering in one of them the upper part of the plant and in another the lower part.
- Use the camera of a cell phone with Android or iOS operating systems.
- Use a mobile application that is capable of controlling the cell phone's front camera. Suggested applications: Filmic Pro.
- Utilizar una tarjeta de calibración de color previo a realizar los videos.

A.3 Instructions for capturing the video

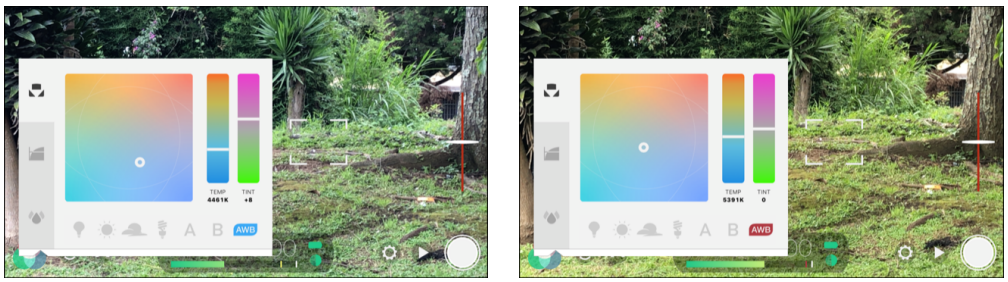
As mentioned earlier, the videos cover each groove that confirms tomato plantations. Below are instructions to correctly capture the videos:

Figure A.2: Colo checker gray card.

A.3.1 Color calibration

Using a gray scale calibration card, the camera's white balance should be adjusted as follows.

1. Enter to the Filmic Pro application.
2. Position the calibration card in front of the camera and zoom in to focus on the card only.
3. Go to the camera settings and select the section related to white balance.
4. Select the option to perform an automatic white balance.
5. Wait a moment for the camera's white balance to be automatically adjusted and then select the option to set the white balance.
6. It is important to mention that this process must be carried out in the place where the capture is intended and it is also recommended to repeat the steps every 30 minutes, to anticipate possible changes in the lighting that the place presents.



(a) (b)

Figure A.3: Whitebalance configuration: In a) the automatic white balance is displayed and in b) the fixed white balance.

A.3.2 Proper use of the gimbal

The following are some recommendations for optimal use of the gimbal.

1. Make use of a tripod that allows you to make a gimbal grip with both hands.
2. Tilt the device slightly forward in order to have greater stability during video capture.
3. Make slow and constant movements. The suggested way of walking with the gimbal is to slightly bend the knees to have greater cushioning of the steps and avoid sudden movements of the camera.

A.3.3 Pre-capture application settings

In order to avoid as much as possible the effect of blurring that can cause the taking of a moving video, it is recommended to set the frame rate of the application to 120 FPS and thus achieve a slow motion effect.