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, November 19th, 2017

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 1, 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 1, 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 1, 2019

Resumen

El tomate es una de las principales vegetales a nivel mundial debido a su versatilidad de uso y a su impacto economico. Sin embargo el cambio climático ha provocado que el manejo de plagas y enfermedades sea cada vez más complicada. Es por ello que es importante la implementación de técnicas no invasivas para el diagnóstico temprano de enfermedades en el campo de cultivo. En este proyecto se presenta un estudio de algorítmos no supervisados tales como los modelos generativos, con el objetivo de detectar anomalías en fotos de tomate. Además se plantea una propuesta de modelo capaz de detectar anomalias, basandose en los redes generativas adversarias.

Palabras clave: Tomato, Deep Learning, Anomaly detection, Autoencoders, Generative Adversarial Networks

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 it is important to implement non-invasive techniques for the early diagnosis of diseases in the crop field. This project presents a study of unsupervised algorithms such as generative models, with the aim of detecting anomalies in tomato photos. In addition, a model proposal capable of detecting anomalies is presented, based on the adversary generative networks.

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

a Mariángel

Agradecimientos

El resultado de este trabajo no hubiese sido posible sin el apoyo de Thevenin, Norton, Einstein y mi querido amigo Ohm.

Luis Alonso Murillo Rojas

Cartago, December 1, 2019

Contents

List of Figures

Table index

List of symbols and abbreviations

Abbreviations

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
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 strategy controls are becoming more important in the latest years for crop management, due to the consequences that brings climate change. In [6] list a series of problems that climate change provokes, like the alteration of stages and rates of development of pathogens, modifying host resistance and changes in the physiology of host-pathogen interactions. These could cause changes in the geographical distribution and growth of plant species.

All this situation has the potential to cause catastrophic plant disease, which can provoke food crops to be destroyed, which according to [24] has the consequence of aggravating deficit of food supply. In this sense, the tomato has a key role, due to its big impact in the diet of people. As stated in [5], the tomato represents the most economically important vegetable crop worldwide.

[20] states the importance of the use of non-invasive methods to deal with crop disease. Normally, to combat this problem, the agricultors make use of chemicals like pesticides which leads to negative impacts like environment consequences and health issues to the people that consume the vegetable.

In Costa Rica, the disease that causes the most damage is the Phytophthora infestans and the most important pest in the whiteflies [2].

The general goal of this project is the study of different unsupervised learning algorithms for anomaly detection purposes in tomato images. As part of the specific goals are the following:

- Define a capture protocol of tomato images to the creation of the dataset that shall be used in the training of the generative models.
- Study and selection of unsupervised learning algorithms.
- Evaluation of the generative models for the detection of anomalies in tomato images.

The state of the art section will cover all the related works for the detection of anomalies

in different contexts like in agriculture or health care; also will be explained the main concepts that are the basis of this project.

In the proposed solution section is described the solution strategy to evaluate different deep learning architecture that is able to detect anomalies.

The results section 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 conclusion has a final analysis of the project and establish the future work.

Chapter 2

State of the art

2.1 Realated work

The use of deep learning in agriculture has a great pontential in a variety of application such as soil mapping, crop type classification, crop monitoring, pest detection and management, among others [14]. 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 [1] is presented a work that make use of CNN classifiers and object detectors like Faster R-CNN, Region-based Fully Convolutional Network or Single Shot Multibox Detector for the detections of disease in tomato in South Korea. One of the problems that this project faced was the false positives in the classification which in [10] is proposed a filter to reduce this issue.

In [3] propose the use of an adversarial autoencoder for anomaly detection. This corresponds to an unsupervised method that is just able to replicate healthy data. This method has the advantage that the network is able to find the main features that represent normal data.

In [4] presents a new architecture proposal where combine the variational autoencoder 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 its VAEGAN architecture slightly outperforms most of the other architectures.

2.2 Literature review

2.2.1 Deep Feedforward Networks

The Deep Feedforward Networks, also known as multilayer perceptrons (MLPs), has a big importance in the field of Machine Learning. The problem that this kind of network

tackle is the approximation of some function f^*. As mentioned in [12], 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 should be able to learn the set of parameters θ that best fits to 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 functions 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, and $f_{(4)}$ is the last layer or the output. Between the first and the output layers are the hidden layers. During the training, generally is provided explicitly the output values given some input, but is no specified what values should have the hidden layers. The learning algorithm should decide how to use those hidden layers to generate the desired output.

2.2.2 Convolutional Neural networks

The convolutional neural networks (CNN) [17] 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 a mathematical operation called convolution. The convolution is an operation on two functions of a real-valued argument and is defined as follows:

$$s(t) = \int x(a)w(t-a)da \tag{2.1}$$

In terms of CNN, the first argument of the convolution ($x(a)$) is referred as the input and the second argument ($w(t-a)$) as the kernel. The output of this operation is known as the 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 is in charge of subsampling the data) and in the end, there is typically a fully connected network.

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 are the type of algorithms that make use of labeled data for its training process. In the last years, this kind of algorithm is presenting very important results in applications like object detection [18].

The unsupervised learning algorithms are the ones that do not need a structured dataset and can find a pattern by itself. One particular type of unsupervised learning algorithms

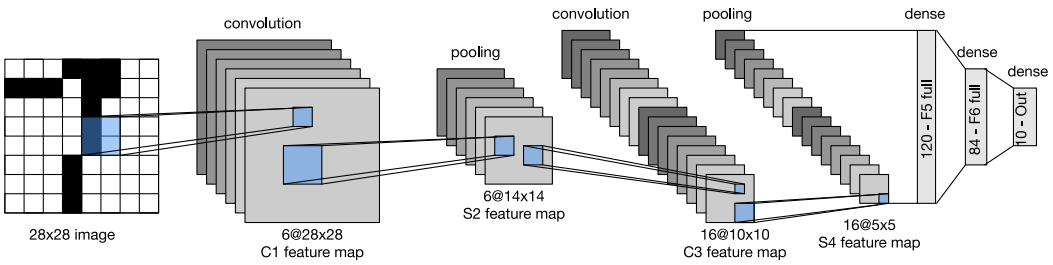


Figure 2.1: Convolutional Neural Network architecture

are the generative models. The generative models are powerful algorithms that are capable of learning the probabilistic data distribution of some datasets, allowing them to generate new samples. Two of the most important 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 into two parts: the encoder that tries to generate a latent space h where is extracted some features of the input data $h = f(x)$; and the decoder that takes the latent space generated by the encoder as input and reconstruct the data $r = g(h)$. One of the main applications of the type of neural networks is the dimensionality reduction, image compression, image denoising or image generation. The image generation case shall be cover in more detail in the next section with the variational autoencoders as part of the generative models.

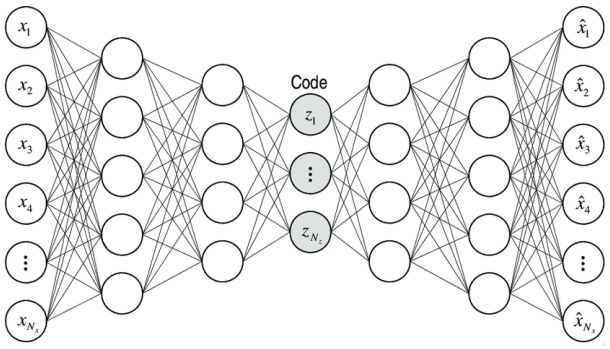


Figure 2.2: Autoencoder architecture

Variational Autoencoder

The objective of the variational autoencoders [16] is the generation of data samples from

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

From a probabilistic perspective, the latent variables will be get 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 de data distribution as follows:

$$\mathbb{P}(\mathbf{X}) = \int_z \mathbb{P}(\mathbf{X}|z)\mathbb{P}(z)dz \tag{2.2}$$

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

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

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

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

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

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

Therefore, the loss function corresponds to:

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

The first term of the loss function can be seen as the reconstruction error and the second term corresponds to the KL error [9].

Spatial Variational Autoencoder

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

In [25] is proposes a spatial variational autoencoder, 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 [8], has the potential of group the data, and each group can represent or share a specific feature of the original data, besides to have a competitive performance in comparison with the regular VAEs.

Generative adversarial networks

The generative adversarial networks (GAN) were proposed by Ian Goodfellow [11] in 2014. The basic idea behind GANs is in the competence of two models: from one side is the generator G that tries to learn the probability distribution of the data and for the other side, a discriminator $\mathbb{D}$ that decides if the input data is real or generated by $\mathbb{G}$. 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 equation 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.

One of the major challenges that the GANs has is precisely the training process, where sometimes is really hard to reach the Nash equilibrium. In some cases, if the capacity of the model is not enough, the model is susceptible to collapse. Another possible scenario is when the learning rate of the model is too aggressive, provoking that the net never converges.

Architectures based on GANs

In [21] presents a series of architectures based on GANs, along with some metrics to evaluate its performance.

- Convolution base GAN: The original GAN is implemented based on the multi-layer

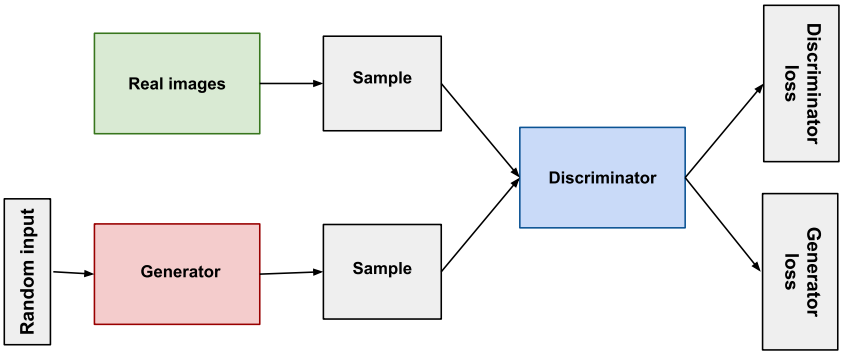


Figure 2.3: Generative Adversarial Network architecture

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 of 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, have more control in that latent space. The type of constraint will depend on the type of data that the GAN is dealing with.

- Autoencoders based GANs: This type of GAN architecture is presented in [19] 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 will allow overcoming one of the main challenges of the autoencoders, where sometimes is not able to learn correctly the data distribution.

2.2.4 Use of generative model in the anomaly detection

The generative models discussed so far have their utility in the detection of anomalies. The general idea is to train a model to be able to learn the data distribution of a normal dataset. In that way, the model should be able to reconstruct o generate a query image as similar as possible. 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 [7] is presented different GAN based architectures, with the purpose of anomaly detection:

AnoGAN

This GAN is first trained with just normal data and be able 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 more similar $G(x)$ to the query image. This iterative process has the disadvantage that is too time-consuming [23].

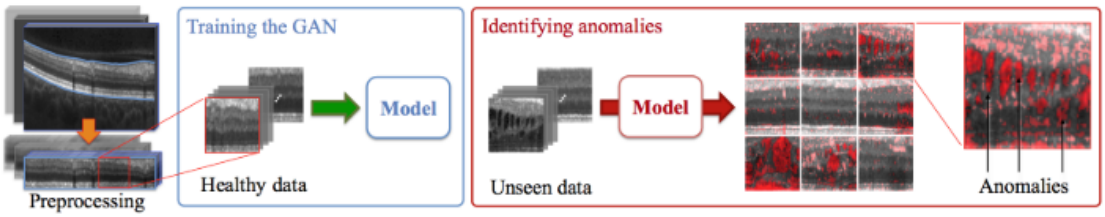


Figure 2.4: AnoGAN traing with healty images first and then reconstruction of unseen images[23]

GANomaly

This architecture is inspired by the AnoGan but tries to overcome the long detection time that it has. In order to do that, make 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 in the training for the learning of the manifold of the input data x.

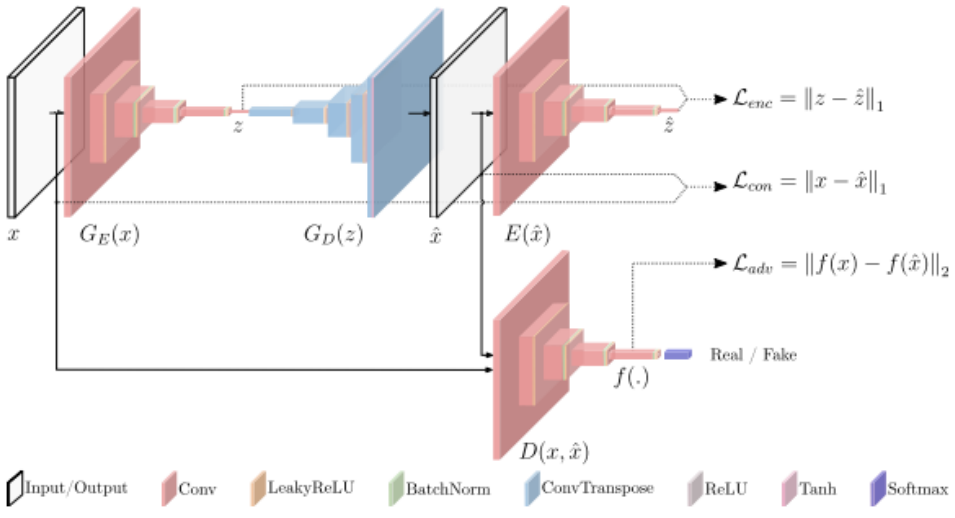


Figure 2.5: GANomaly architecture and loss functions [7]

Chapter 3

Proposed solution

3.1 Data preprocessing

The data acquisition for this project has used a camera that captures in a resolution of 1920x1080. The tomato data was captured in tomato crops located in the Alajuela province of Costa Rica.

For the purpose of the architectures experiment was used 191520 images of a dimension of 50x50 each. For this set of images, 80% was used for testing and 20% used for validation. This training validation data represents only healthy tomato plants. Also was used 798 images for testing and in this case, there are also unhealthy tomato photos.

Before using these images for the training of the models, a normalization was applied to convert the image values to be in a range between -1 to 1. Finally, some image data augmentation like rotation, zoom, width shift, among other image transformations were implemented.

3.2 Architecture experiments

In order to evaluate the different architectures for the anomaly detection in tomato plants, three experiments were 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 experiment used the same dataset and for each architecture is tested its reconstruction of an input image. 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 shouldn't be able to reconstruct the affected area, and that reconstruction error or dissimilarity shall be 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 common 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 is the common training of a GAN model, but with just healthy images of tomato, where the generator receives random noise as input for the latent variables. The second stage is the training of an encoder that shall be able to map to the latent space that the generator needs needs to reconstruct the query input image.

In the training of the encoder, the generator and discriminator models are not trainables. 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 makes that the generator creates 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 represents the weights of the different terms of the function and $\mathcal{L}_R$ (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 bad time performance. This modification of the AnoGAN architecture is similar to the one presented in [22].

Chapter 4

Results and analysis

From the experiments of the three architectures: Adversarial Anomaly Detector, sVAE, and GM-VAE; the one that presented the best reconstruction is the Adversarial Anomaly Detector (see figures 4.1, 4.2 and 4.3). For the cases of both autoencoders, the reconstructions suffer from lots of blurring, making 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 provoked for the series of gaps the learned data distribution has due to the initial assumption of make use of simpler data distributions as a base.

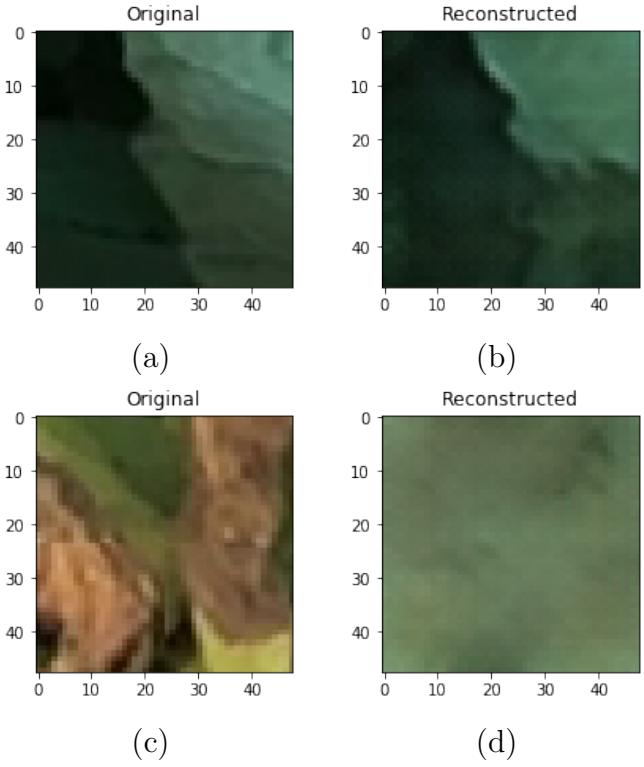


Figure 4.1: Reconstruction evaluation of the adversarial anomaly detector: a) origianl healty sample, b) reconstructed healty sample, c) original test sample and d) reconstructed test sample.

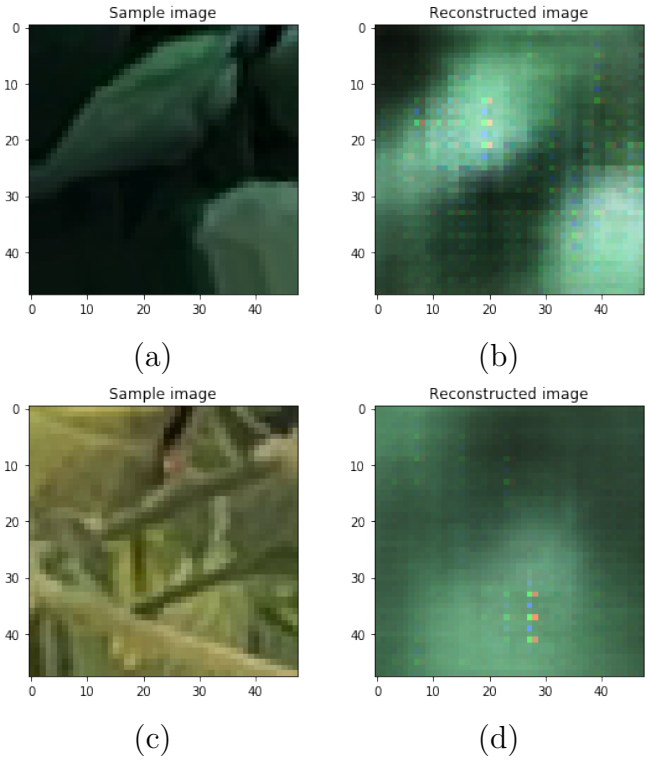


Figure 4.2: Reconstruction evaluation of the sVAE: a) origianl healty sample, b) reconstructed healty sample, c) original test sample and d) reconstructed test sample.

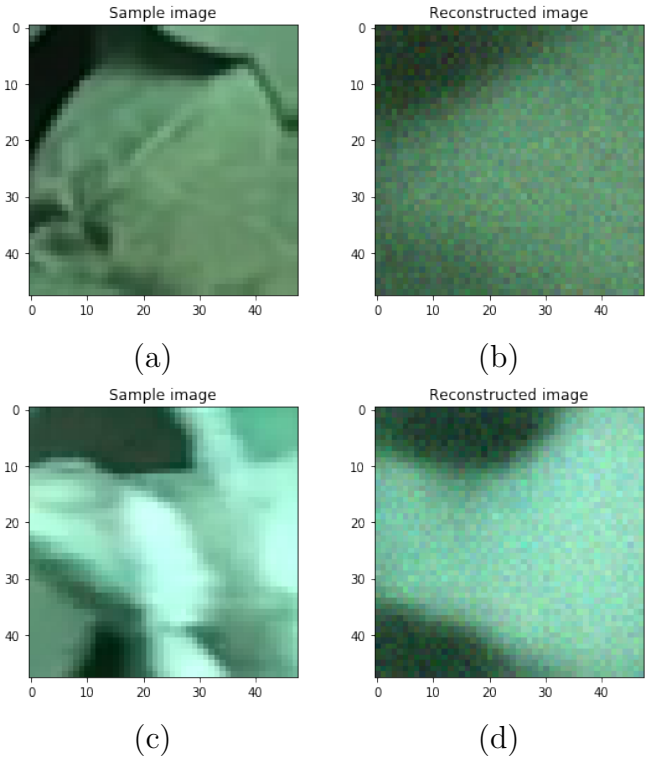


Figure 4.3: Reconstruction evaluation of the GM-VAE: a) origianl healty sample, b) reconstructed healty sample, c) original test sample and d) reconstructed test sample.

In the case of the AnoGAN, one of the most problematics parts was the training of the model, but with the available tomato data, it was possible to achieve the Nash equilibrium. The generated images are much better than the ones in the autoencoders. 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.

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

In figures 4.4 and 4.5 is possible to see the data distribution between normal and anomaly samples. For the case of the figure 4.4 we can see a defined group of data that represents the healthy samples and a smaller amount of points around this healthy group that represents the anomaly data. In the figure 4.5 we have the opposite case, where the amount of anomalies is bigger than the number of healthy samples. Is important to mention that for the first case the test images was an image of 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.

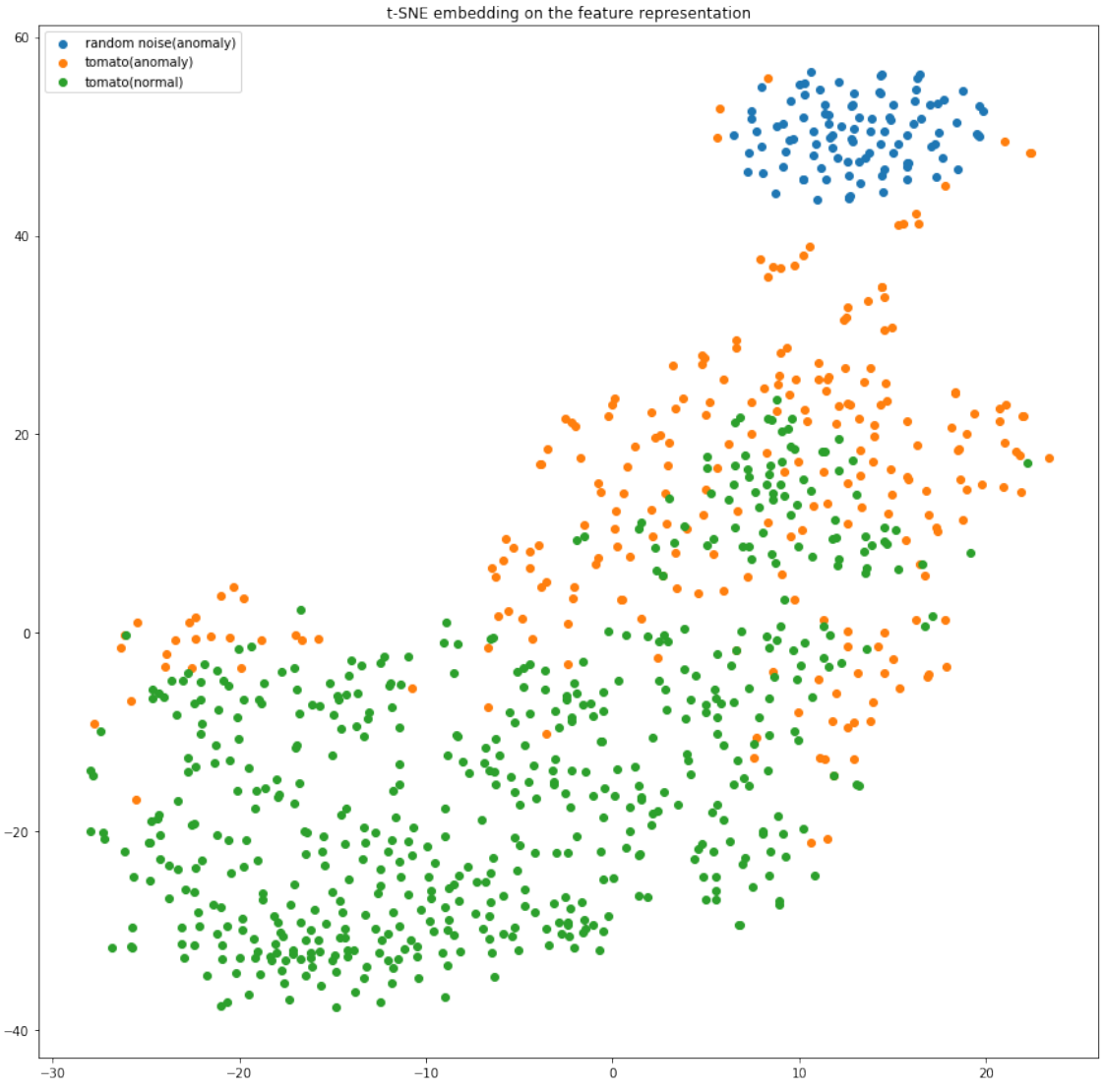


Figure 4.4: Adversarial Anomaly Detector t-SNE evaluation of test image 1

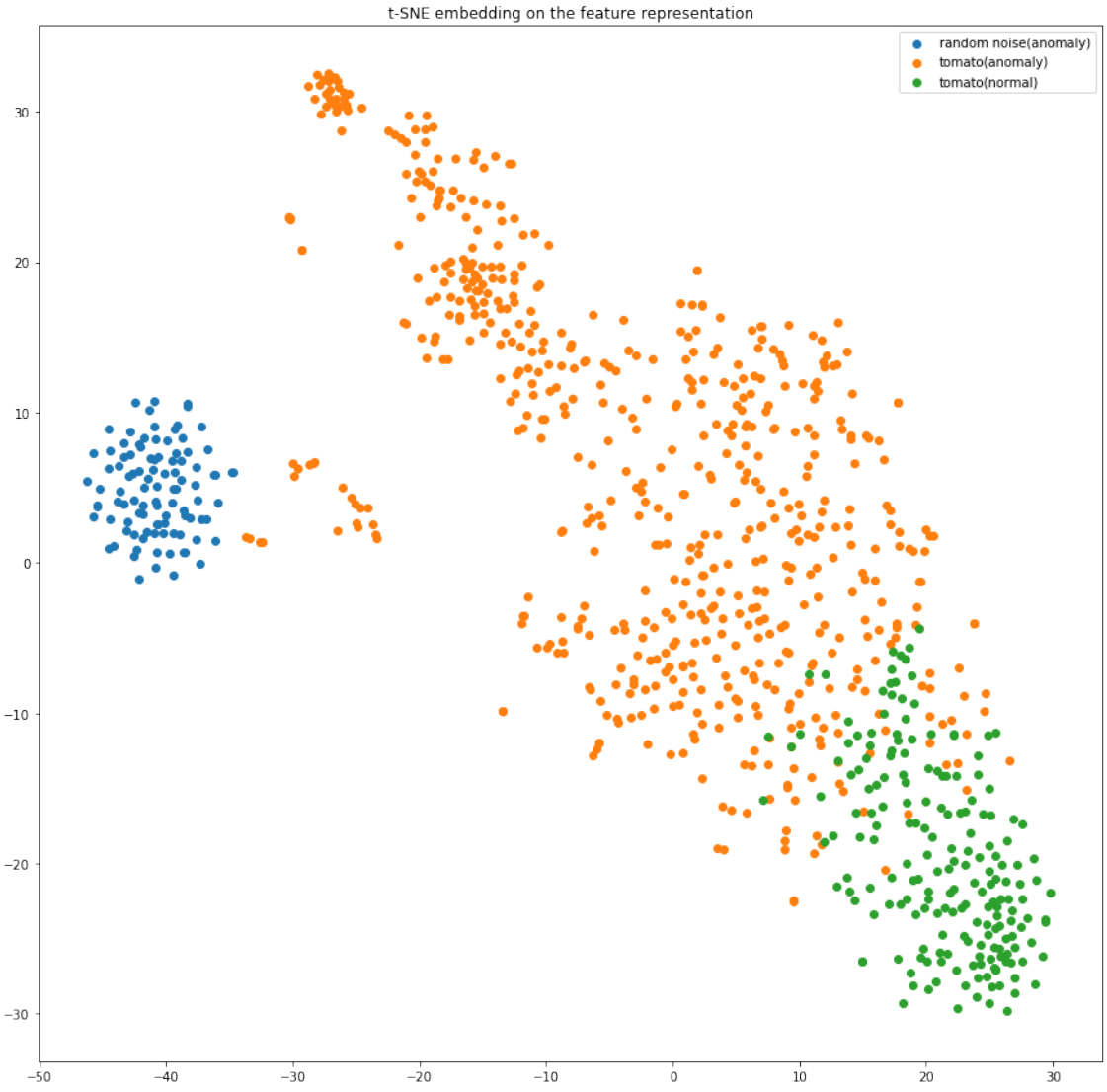


Figure 4.5: Adversarial Anomaly Detector t-SNE evaluation of test image 2

Chapter 5

Conclusions

This project has been able to corroborate that the variational autoencoder architectures do not have the needed performance to achieve a decent anomaly detection due to its blur issues. On the other side the AnoGAN has more promising results with better reconstruction images.

A modification to the AnoGAN architecture was tried, allowing it to improve considerably 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 [13]. Another possible approach is the use of a regularized in the latent space of the generator, in a similar way as proposed in [15].

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

Bibliography

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

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

[3] 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.

[4] 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.

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

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

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

[8] 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.

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

[10] 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.

[11] 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.

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

[13] 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.

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

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

[16] 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.

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

[18] 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.

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

[20] D. S. Park, "Characteristics of Tomato Plant Diseases", no. October 2016, 2017.

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

[22] 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.

[23] 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.

[24] 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.

[25] 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

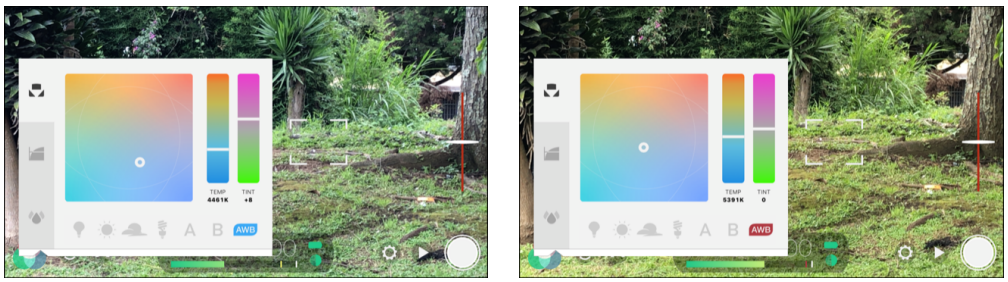
As mentioned earlier, the videos cover each groove that confirms tomato plantations. Below are instructions to correctly capture the videos:

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.