UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA MECÁNICA

MACHINE LEARNING APPLIED TO WELDING DISCONTINUITY DETECTION AND CLASSIFICATION ON MULTIPASS WIRE FEED FLUX-CORED ARC WELDING

TESIS PARA OPTAR AL GRADO DE
MAGÍSTER EN CIENCIAS DE LA INGENIERÍA, MENCIÓN MECÁNICA

JUAN PABLO ROMERO CAMPOS

PROFESORA GUÍA:
VIVIANA MERUANE NARANJO

MIEMBROS DE LA COMISIÓN:
PATRICIO MENDEZ PINTO
RUBÉN FERNÁNDEZ URRUTIA

SANTIAGO DE CHILE
2022

RESUMEN DE LA TESIS PARA OPTAR
AL GRADO DE MAGÍSTER EN CIENCIAS DE LA INGENERÍA
POR: JUAN PABLO ROMERO CAMPOS
FECHA: 2022
PROF. GUÍA: VIVIANA MERUANE NARANJO

APRENDIZAJE DE MAQUINAS APLICADO A LA DETECCIÓN Y CLASIFICACIÓN DE DISCONTINUIDADES EN SOLDADURA POR ARCO CON NÚCLEO FUNDENTE CON ALIMENTACIÓN DE ALAMBRE MULTIPASO

La soldadura es un proceso multifísico y dinámico utilizado en la mayoría de las aplicaciones metalúrgicas. Las malas soldaduras pueden crear discontinuidades que pueden producir fallas en la integridad o incluso daños catastróficos, por lo que las inspecciones de soldaduras se consideran fundamentales. Sin embargo, la mayoría de las veces este proceso lleva un largo período en el que la productividad disminuye, por lo que es fundamental explorar nuevas tecnologías para ayudar con el proceso de inspección.

Este estudio se centró en el uso del aprendizaje profundo para predecir y clasificar las discontinuidades de soldadura mediante el análisis de las señales eléctricas durante la soldadura con un fuerte objetivo en su aplicación y relevancia en la industria, se seleccionó Multipass Flux Core como el tipo de soldadura a estudiar ya que es utilizado en la mayoría de las aplicaciones civiles en América del Norte. Las tres discontinuidades Falta de fusión, Falta de penetración y socavación en las que se centra principalmente este proyecto fueron seleccionadas debido a su criticidad también en aplicaciones estructurales.

Las discontinuidades seleccionadas fueron inducidas deliberadamente en cupones de soldadura en ciertos lugares para comparar los datos de voltaje y corriente entre buenas y malas soldaduras. Los resultados experimentales mostraron que es posible inducir discontinuidades manualmente, y estos resultados fueron validados mediante el uso de pruebas no destructivas. Se probaron varios algoritmos de aprendizaje profundo para identificar y clasificar las discontinuidades inducidas.

Los resultados de estos algoritmos mostraron que es posible separar las soldaduras buenas de las malas (con muescas y falta de fusión). El mejor algoritmo de detección de discontinuidades fue Spectrogram Varational Auto Encoder con un porcentaje de precisión de 99,5 % en el conjunto de prueba. La incertidumbre de las predicciones se midió utilizando las propiedades estocásticas del espacio latente, este mostró estar alineado con respecto al reporte PAUT.

Finalmente, se creó un clasificador utilizando la parte del codificador del Spectrogram Varational Auto Encoder como algoritmo de reducción de dimensionalidad que alimentó una red neuronal densa. Los resultados mostraron un 96 % por ciento de precisión para clasificar entre clases buenas, falta de fusión y falta de penetración en el conjunto de prueba. Estos resultados mostraron potencial para una implementación industrial que podría ayudar a reducir el tiempo de inactividad y de inspección.

RESUMEN DE LA TESIS PARA OPTAR
AL GRADO DE MAGÍSTER EN CIENCIAS DE LA INGENERÍA
POR: JUAN PABLO ROMERO CAMPOS
FECHA: 2022
PROF. GUÍA: VIVIANA MERUANE NARANJO

MACHINE LEARNING APPLIED TO WELDING DISCONTINUITY DETECTION AND CLASSIFICATION ON MULTIPASS WIRE FEED FLUX-CORED ARC WELDING

Welding is a multi-physics and dynamic process used in the majority of metallurgic applications. Bad welds may create discontinuities that can produce integrity failures or even catastrophic damage, for this reason, welds inspections are considered fundamental. Nevertheless, most of the time this process takes a long period in which the productivity decreases, therefore exploring new technologies to help with the inspection process is paramount.

This study focused on the use of deep learning to predict and classify welding discontinuities by analyzing the electrical signals during the welding with a strong aim in its application and relevance in the industry, Multipass Flux Core was selected as the type of welding to study since is used in most civil applications across North America. The three discontinuities Lack of fusion, Lack of penetration, and undercut that this project is mainly focused on were selected due to their criticality also on structural applications.

The discontinuities selected were purposely induced into weld's coupons at certain locations to compare the voltage and current data between good and bad welds. The experimental results showed that to manually induce discontinuities is possible, and this results were validated by using non destructive test. Several deep learning algorithms were tested to identify and classify the induced discontinuities.

The results of these algorithms showed that it is possible to separate good from bad welds (with undercut and lack of fusion). The best discontinuity detection algorithm was Spectrogram Varational Auto Encoder with a 99,5 % percent of accuracy in the test set. The uncertainty of the predictions was measured by using the the stochastic properties of the latent space, this showed to aligned with respect to the PAUT report.

Finally, a classifier was created by using the Encoder part of the Spectrogram Varationa Auto Encoder as a dimensionality reduction algorithms which fed a Dense neural network. The results showed a 96 % percent of accuracy to classify between good, lack of fusion, and lack of penetration classes in the test set. This results showed potential for an industrial implementation that could help to reduce inspection downtime.

Le dedico este trabajo especialmente a mi madre la cual me ha apoyado muchiso estos años.
No podre nunca pagarle por tanto.

Agradecimientos

Le doy las gracias de forma especial a la Profesora Viviana que me a apoyado y me sigue apoyando mucho tanto en esta tesis como otros proyectos. Además de agradezco al profesor Patricio por su candidez y buena disposición para mí. Al profesor Rubén y Williams que los conozco desde hace muchos años y han sido grandes mentores y amigos. Además, le agradezco a Eduardo Rocha y Francisco Basaure que sin su ayuda esta tesis nunca habría podio haberse realizado.

Table of content

Índice de Tablas

Índice de Ilustraciones

Introducción

Welding is a multi-physics and dynamic process used in the majority of metallurgic applications. Bad welds may create discontinuities that can produce integrity failures or even catastrophic damage, for this reason, welds inspections are considered fundamental. Nevertheless, most of the time this process takes a long period in which the productivity decreases, therefore exploring new technologies to help with the inspection process is paramount.

Currently machine and deep learning offer a new framework for automatic detection of outliers and classification algorithms, which in sum with the cost reduction of both acquisitions devices and data warehouse, make it suitable for new research approaches that could contribute to help the industry.

This project studies the relation between the current and voltage measurements of welds with their condition of health, that is to say, if a discontinuity occurs or not, with the support of machine and deep learning algorithms.

Objectives and scope

General objective

Create an algorithm based on machine learning techniques to identify and classify welding discontinuities.

Specific objectives

- Design an experimental setup that allows us to obtain voltage and current welding data under different weld health conditions.
- Relate discontinuities induced during the experiment with current measurements and non-destructive reports.
- Perform a benchmark of all the main machine learning techniques for detection and classification.

Scope

- The discontinuities that will be studied in this project are The following: Undercut, Lack of fusion, and lack of penetration.
- The material selected corresponds to ASTM A36.
- The Acquisition system is given.

Capítulo 1

Background

This chapter presents the main theoretical background needed to understand the current thesis project. Section 1 to 3 presents welding theoretical background, section 4 to 5 describe the necessary machine learning concepts.

1.1. Arc Welding

Arc welding is one of several fusion processes for joining metals. ”By applying intense heat, metal at the joint between two parts is melted and caused to intermix - directly, or more commonly, with an intermediate molten filler metal. Upon cooling and solidification, a metallurgical bond is created” [2].

Particularly, the heat needed to melt metal is produced by the arc; this is a sustained electric discharge through a ionized gas column called plasma that is generated between two electricity conductors, cathode and anode (considering direct current, DC). When they come into contact to establish the flow of current and then separated by a small distance. When they separate, the air between the electrodes is ionized by the potential difference and that reduces the electrical resistance of the air column.

The electrode is a carbon or tungsten rod, it purpose is to convey the current and sustain the electric arc between its tip and the workpiece. Or, it may be a specially prepared rod or wire that not only conducts the current and sustains the arc but also melts and supplies filler metal to the joint. If the electrode is a carbon or tungsten rod and the joint requires added metal for fill, that metal is supplied by a separately applied filler-metal rod or wire.

Figure 1.1 represents the basic electric circuit on any arc welding. An AC or DC power source, fitted with whatever controls may be needed, is connected by a ground cable to the workpiece and by a ”hot” cable to an electrode holder of some type, which makes electrical contact with the welding electrode. When the circuit is energized and the electrode tip touches the grounded workpiece, and then withdrawn and held close to the spot of contact, an arc is created across the gap. The arc produces a temperature of about 6500°F at the tip of the electrode, a temperature more than adequate for melting most metals.

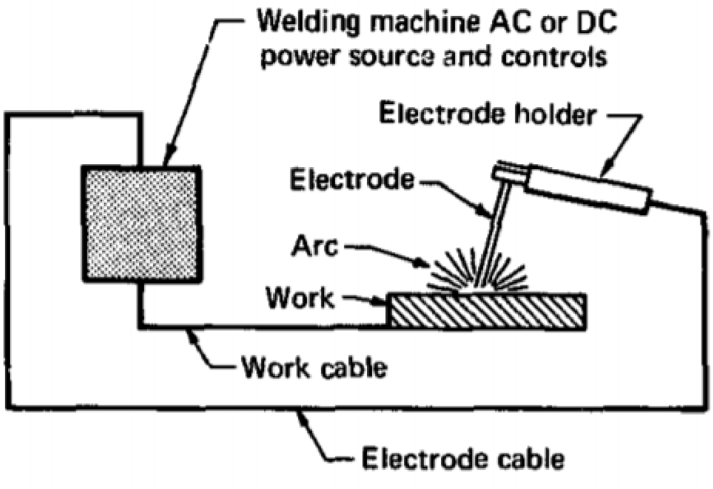


Figura 1.1: Scheme of the basic electric circuit of an arc welding process [2].

1.1.1. Flux cored arc welding

There are many types of arc welding that can be selected depending on the work material, mechanism of the equipment, and the environment, among which can be named the followings: TIG welding, Plasma welding, Shielded metal arc welding, MAG welding, MIG welding, Electrogas arc welding (EGW), etc.

One of the most important types is Flux cored arc welding (FCAW), which uses consumable electrodes that contain flux around it. Therefore, it is a semi-automatic or automatic arc welding process. The coalescence of metal is produced by heating with an arc between continuously fed electrodes and the base metals. The shielding is obtained by the flux in the electrode. Since in general FCAW does not need an external shield gas, it is easier to master, which makes it suitable for construction applications.

Nowadays, FCAW is important in the shipbuilding and civil construction industry as it saves huge labour cost, also it is widely applicable in mechanical industries. FCAW can be used on plain carbon, alloy, stainless and duplex steels. It is used for hard facing and surfacing.

Since welding is mainly performed by humans, the welds could have defects, which might cause several damages. Therefore it is critical to distinguish between good quality and bad quality welds.

1.1.2. Welding defects

As mentioned in the previous sections, Arc welding is a complex, therefore, some defects can arise. Weld defect is defined as any flaw or imperfection that compromises the intended use of a weldment. These are classified according to ISO 6520. This also implies a flaw or imperfection may not compromise the weld, and a weld is said to have a discontinuity when this occurs. therefore, a weld can have a discontinuity and not be considered defective. These acceptable limits are specified in ISO 5817 and ISO 10042.

In general, the most common weld defects are: Cracks, Inclusions, Lack of fusion, Porosity, Undercut, Poor penetration, Burn through, Under-fill, Excess reinforcement, Spatter, Over-roll/Overlap, Whiskers Mechanical damage.

The main three discontinuities that could cause extreme failure and are analyzed in this study are : Lack of fusion, undercut and lack of penetration.

Lack of fusion: Corresponds to a lack of proper fusion between the base metal and the weld metal. It can also appear between adjoining weld beads. This creates a gap in the joint that is not filled with molten metal.

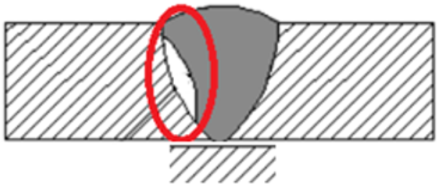


Figura 1.2: Scheme of lack of fusion.

Undercut: The groove formation at the weld toe, reducing the cross-sectional thickness of the base metal. Results in stress concentration weld and work piece.

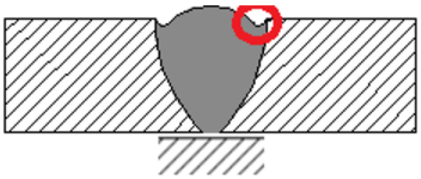


Figura 1.3: Scheme of Undercut.

Lack of penetration: Occurs when the groove of the metal is not filled completely, meaning the weld metal doesn't fully extend through the joint thickness.

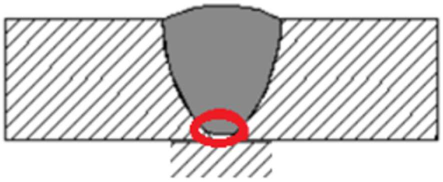


Figura 1.4: Scheme of lack of penetration.

1.2. Machine learning

Machine learning (ML) is the field of study that gives computers the ability to learn without being explicitly programmed.

ML can be understood as computational methods that use experience to improve performance or to make accurate predictions. Experience refers to past information or data that is available to us, which has been labelled and categorized. As with any computational exercise, the quality and amount of the data will be crucial to the accuracy of the predictions that will be made [4].

1.3. Dense neural network

It is a computational model based on a set of simple neural units, each with a different rule or condition. This is called “activation function”, and must be satisfied before propagating the information through the rest of the neuronal units. Each unit neuron or neuron, is connected to the others through links which in turn have associated weights that activate, increase, decrease or deactivate the value of each neuron in the network.

Within a neural network, there are layers, which are sets of neurons by which passes the information for classify or regression. This layers are always hidden (normally their results can be seen); there are only two non-hidden layers, the layer data input, and output. The Figure shows a scheme of the a neural network.

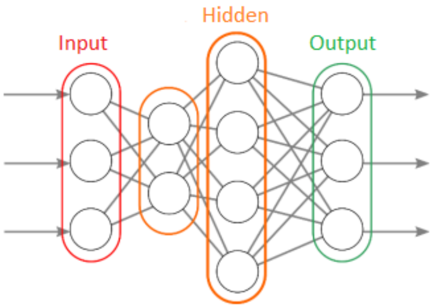


Figura 1.5: Neural Network scheme.

A dense Neural Network are those networks in which all the neurons in one layer are connected to all the neurons in the next. These networks, are obviously more efficient in terms of processing information than regular Neural Network because each neuron has more data to deal with, but in turn, a Dense Neural Network, is slower when processing the data, due to the same reason, the large amount of information that they move from one layer to another.

Regarding the use of neural networks to make predictions from a set of data, it is necessary to divide these into 3 sets, a training set to train the hyper parameters of the network (weights and bias), a validation set (to validate the training behaviour) and a test set to determine if the model obtained after training complies or not with the task to be solved.

The objective of training with the training data set is to optimize a cost or loss function which compares the predicted values with the real labels of the data. For classification tasks, different approaches exist regarding loss functions, activation functions, evaluation metrics and structures.

Today, an algorithm called backpropagation is a frequently used method for training deep neural networks [12]. Backpropagation is a method to calculate gradients with which the weights and biases are updated during training by chain rule. This means than when weights and biases are randomly initialized, a first forward pass through the network is done and the output is calculated. Then, the value of the error or cost function is then passed backwards into the network, adjusting the weights and biases. Then, the following forward pass will deliver a different result and subsequently, by backpropagation of the error, the parameters will then again be updated.

1.3.1. Overfitting, Underfitting and Regularisation

Overfitting a model results in lower performance when faced with data outside the training set. This means that the parameters have adjusted too well to the training data and have lost their ability to generalise the conditions. This may be due to a very complex structure of the trained network (high number of parameters) compared to a small or noisy training data set that is not representative of the phenomena studied. Underfitting, on the other hand, results in low performance due to the inability of the model to capture the data structure. This happens when making very simple models (very few parameters). To avoid these phenomena, various regularisation techniques are used. Dropout, penalty, batch normalisation and cross validation can be used to identify and prevent overfitting. Each technique has its own set of internal parameters and frequently used values that can be easily found in published research.

1.3.2. Hyperparameters

The network architecture is defined by a set of hyperparameters. A short list of basic concepts is presented:

- Number of layers: Depending on the relationships within the analysed data, a deeper network can allow the mapping of more complex functions, since each layer will be able to extract different characteristics from the original data.

- Number of units: for each layer, the number of units must be selected. Deeper architectures with a lower number of units per layer are generally preferred to wide and shallow layers, to allow a more complex understanding of the data.
- Dropout probability: Dropout can be used as a regularization technique, with the probability of dropping a unit of [0.1]. This inherent randomness allows the network to train without relying on specific weights. Dropout can also be used to study the uncertainty of model predictions.
- Activation function: each hidden layer has an activation function, these are non-linear functions such as Sigmoid, Tanh, ReLU or Leaky ReLU, among others. These are shown in Figure 1.6. Each type has its own strengths and weaknesses depending on the type of data analysed. An important feature for selecting the activation function is its derivative, as this is critical for backpropagation. Gradients that disappear or explode are susceptible in deep networks and additional measures can be taken to make the learning process more seamless, such as using different learning rates per layer or using specific initialisation processes.
- Optimisation: the selected optimizer defines the method by which the weights and biases of the network are updated through backpropagation. There are several techniques responsible for minimising the objective function through gradient descent, such as SDG, or through momentum optimization methods such as Nesterov and Adagrad. Adaptive methods, such as Adam and RMSProp, achieve more consistent results than other optimisation techniques by including exponential decay in their operations.
- Number of epochs: an epoch is defined as a complete step forward and backward of all batches of the dataset across the network. The performance of the models depends on how many training epochs you reach. The optimizer and the learning rate, as well as the training and validation error are used to diagnose the number of epochs required for the model to converge.

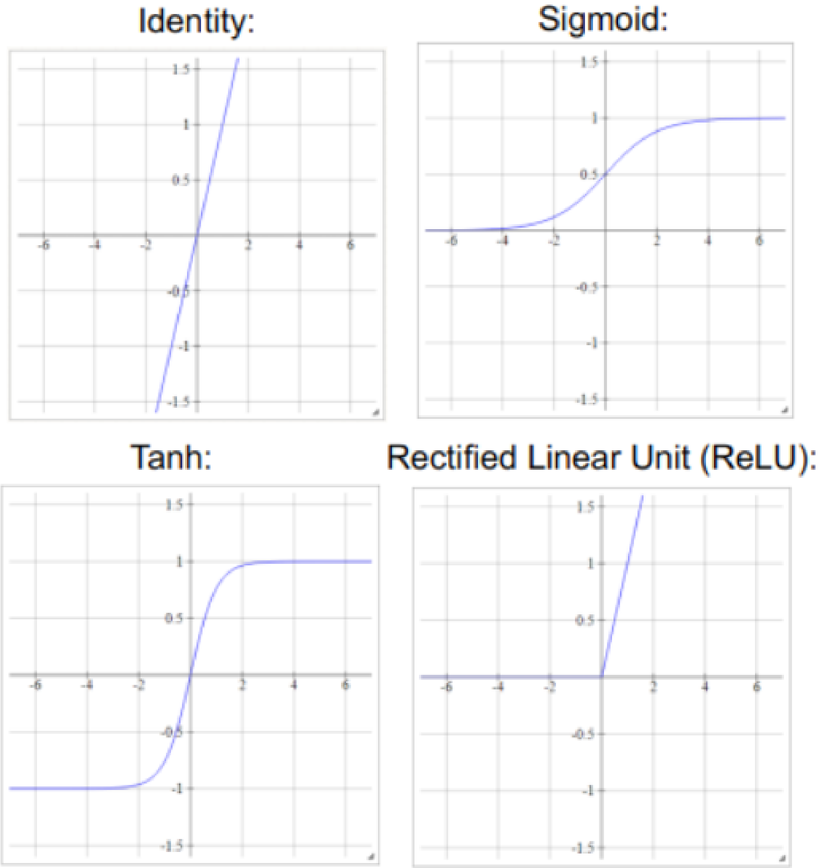


Figura 1.6: Activation functions.

1.3.3. Convolutional Neural Networks

CNNs (convolutional neural networks) emphasize local relationships within data. This makes them especially useful when evaluating spatial data, such as images. The basic idea is that different sensors will extract different information from an image to actually identify the content. A first layer can identify edges, the next shapes, then colors or shadows, and so on..

As the name indicates, this architecture applies a series of convolutional filters to extract abstract representations from an image, called feature extraction to then be processed by a standard fully connected Neural network. The strength of this algorithms is that many types of data can be represented by images (or matrixes), for example, by generating heat maps or spectrograms. Figure 1.7 presents a two-layered convolution which is then fed into the NN-classification layer.

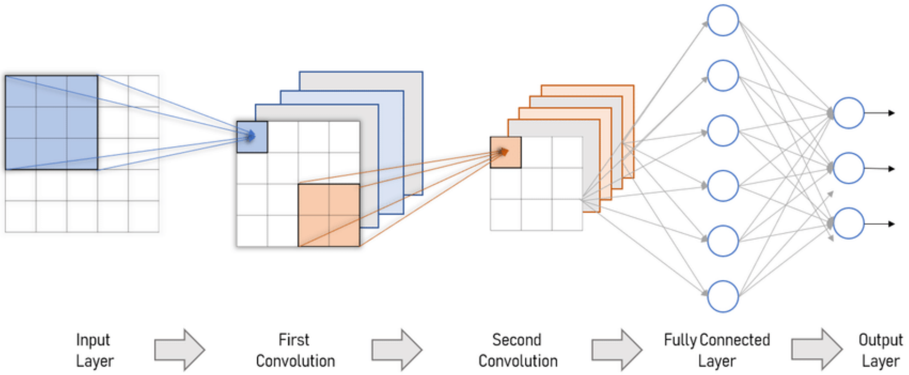


Figura 1.7: CNN scheme example [1].

1.4. Auto encoders

This section presents the implementation of Auto encoder for welding discontinuities identification.

1.4.1. Definition

Auto encoders are artificial neural networks capable of learning dense representations of the input data, called *latent representation* or *coding*, without any supervision. This latent space has a much lower dimension.

Autoencoders consist of 3 parts:

- Encoder: A module that compresses the input data into an encoded representation that is typically several orders of magnitude smaller than the input data.
- Latent Space: A module that contains the compressed knowledge representations and is therefore the most important part of the network.
- Decoder: A module that helps the network decompress the knowledge representations and reconstructs the data back from its encoded form. The output is then compared with a ground truth.

In the following image 1.8 a schematic representation of the Auto encoder can be seen.

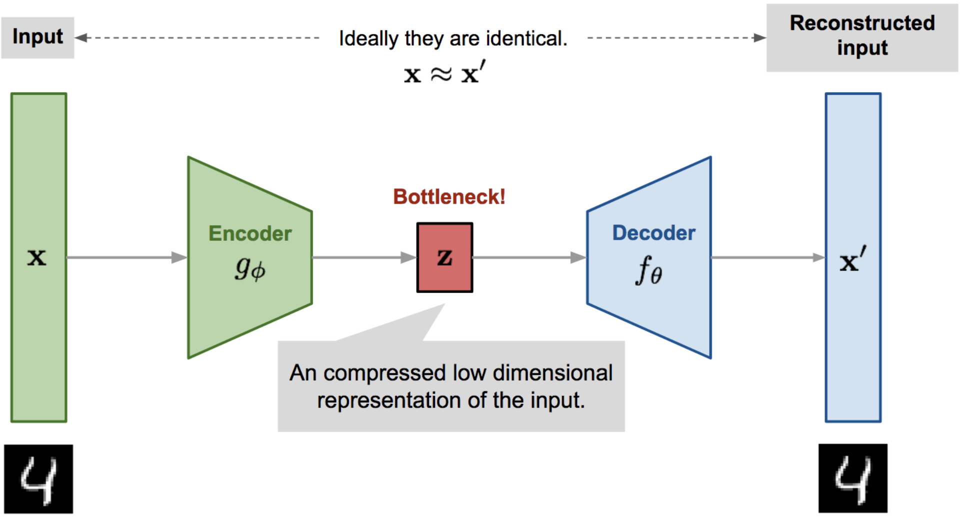

Input
Ideally they are identical.
$\mathbf{x} \approx \mathbf{x}'$
Reconstructed input
$\mathbf{x}$
Encoder
g_ϕ
Bottleneck!
$\mathbf{z}$
Decoder
f_θ
$\mathbf{x}'$
An compressed low dimensional representation of the input.

Figura 1.8: Auto encoder architecture scheme [15].

The encoder is a set of convolutional blocks followed by pooling modules that compress the input to the model into a compact Latent Space. The Latent Space is followed by the decoder that consists of a series of upsampling modules to bring the compressed feature back into the form of an image. In the case of simple autoencoders, the output is expected to be the same as the input.

With respect to the mathematical formulation of an auto encoder, the model can be thought of as consisting of a function $g(.)$ parameterized by ϕ corresponding to the encoder, and a decoder $f(.)$ parameterized by θ. Therefore, given Xa 2d image, the 1d latent space would be $z = g_\phi(X)$ and the reconstruction image $\hat{X}$ would be $\hat{X} = f_\theta(z)$. The following equation represents the mathematical model of a Auto encoder where ϕ and θ are the parameters to be trained.

$$f_\theta(g_\phi(X)) = \hat{X} \tag{1.1}$$

Since $\hat{X}$ should be as close as possible to X, during the training the distance between two images is minimized. The following equation presents metric that is used in Auto encoder for image reconstruction MSE (Mean squared error).

$$L_{MSE}(\phi, \theta) = \frac{1}{N}\sum_{i=1}^{N}(X^i - f_\theta(g_\phi(X^i)))^2 \tag{1.2}$$

Finally, since the decoder builds back the image from its latent attributes, if we train the model just with normal data, and we try to build back abnormal data, difficulties should appear in the reconstruction. The distance can be measured from two images with 1.2. Therefore, by measuring this distance of reconstruction error, a threshold for outlier detection can be defined.

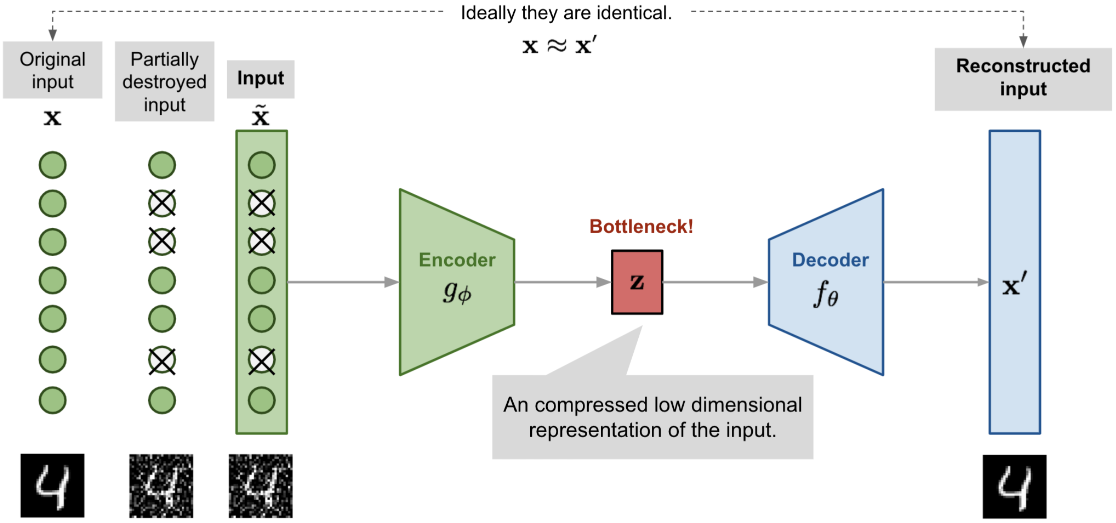
1.5. Denoising Auto encoders.

To avoid overfitting and improve the robustness of the model, Denoising Auto encoder [13] is a modification to the 'vanilla' auto encoder. The input is partially corrupted by adding noises to or masking some values of the input vector in a stochastic manner$\hat{X} \sim M_D(\hat{X}^i|X^i)$. Then the model is trained to recover the original input (not the corrupt one) as it can be seen in the following scheme 1.9.

$$\tilde{X} \sim M_D(\tilde{X}^i|X^i) \tag{1.3}$$

$$L_{MSE}(\phi, \theta) = \frac{1}{N}\sum_{i=1}^{N}(X^i - f_\theta(g_\phi(\tilde{X}^i)))^2 \tag{1.4}$$

For high dimensional input with high redundancy, like images, the model is likely to depend on evidence gathered from a combination of many input dimensions to recover the denoised version rather than to overfit one dimension. This builds up a good foundation for learning robust latent representation.



Ideally they are identical.
$\mathbf{x} \approx \mathbf{x}'$
Original input
$\mathbf{x}$
Partially destroyed input
Input
$\tilde{\mathbf{x}}$
Encoder
g_ϕ
Bottleneck!
$\mathbf{z}$
Decoder
f_θ
Reconstructed input
$\mathbf{x}'$
An compressed low dimensional representation of the input.

Figura 1.9: Denoising Auto encoder architecture scheme [15].

1.6. Varational Auto encoders.

The idea of Variational Autoencoder [10], is actually less similar to all the Auto encoder models, but deeply rooted in the methods of variational bayesian and graphical model. Instead of mapping the input into a fixed vector, maps it into a distribution. Let's label this distribution as p_θ parameterized by θ. The relationship between the data input x and the latent encoding vector z can be fully defined by:

- Prior $p_\theta(z)$
- Likelihood $p_\theta(x|z)$
- Posterior $p_\theta(z|x)$

Assuming that we know the real parameter θ^* for this distribution. In order to generate a sample that looks like a real data point x^i, these steps are followed:

- First, sample a z^i from a prior distribution $p_{\theta^*}(z)$.
- Then a value x^i is generated from a conditional distribution $p_{\theta^*}(x|z = z^i)$.

The optimal parameter θ^* is the one that maximizes the probability of generating real data samples 1.5:

$$(\theta^i)^* = argmax_\theta \sum_{i=1}^{n} log(p_\theta(x^i)) \tag{1.5}$$

$$p_\theta(x^i) = \int p_\theta(x^i|z)p_\theta(z)dz \tag{1.6}$$

Unfortunately it is not easy to compute $p_\theta(x^i)$ in this way, as it is very expensive to check all the possible values of and sum them up. To narrow down the value space to facilitate faster search, a new approximation function is introduced to output what is a likely code given an input x, $q_\phi(z|x)$, parameterized by ϕ.

- The conditional probability defines a generative model, similar to the decoder introduced above. is also known as probabilistic decoder.
- The approximation function is the probabilistic encoder, playing a similar role as above.

1.6.1. KL divergence.

The difference of the VAE model with other autoencoders is that the latent space can be interpreted not as an encoding space, but as a probability distribution (because it does not provide a single encoding, but rather a set of encodings that, with greater or lesser probability, could be the result of the encoder), and this interpretation is also translated to the output space by the decoder. Consequently, the usual loss functions (the representation losses), which are usually given by (clusters of) distances between the inputs and the outputs, are not adequate to measure the error that the network makes in its generation.

In order to solve this problem, a new factor is introduced in the loss function, called KL-divergence (Kullback-Leibler Divergence [9]), which, instead of measuring the distance between points, measures the difference between two probability distributions. When one of these distributions is a normal distribution, the KL-divergence will measure how much this normal distribution resembles the real distribution that we want to approximate.

One of the distributions is clear, it is the sum of normal distributions whose parameters form the latent space of our network, but what is the other distribution? The objective would be to measure the distance with respect to the ”ideal distribution” that generates the outputs that we would like to obtain, but this distribution is not known to us (it is precisely the one that we would like to obtain with this model), therefore conditions so that our network is able to learn a distribution that is good enough are imposed.

For example, if ideal distribution would be, in general, centered at the origin and with deviation 1 is assumed, then the KL-divergence would give 1.7:

$$KL = \sum_{i=1}^{N}(\sigma_i^2 + \mu_i^2 - log(\sigma_i) - 1) \tag{1.7}$$

But if this tern is the only one considered and additional conditions are not imposed, the network will learn to locate all the representations around the origin (approaching the target distribution) and will be unable to differentiate the various classes that can be found within the training set (all the latent representations would be too close to each other to be grouped into distinct pockets).

Therefore, additional information hat the decoder should learn must also be considered during training: it must output the same data that it receives as input and for that the usual loss function can be used. Therefore, one option is that the new loss function considers both facts simultaneously 1.6.1:

$$L_{VAE} = L_{MSE}(\phi, \theta) + KL$$

Where x is the input data, $\hat{x}$ is the output produced by the decoder, Loss is the usual loss function (for example, Euclidean distance between x and $\hat{X}$), and KL is the KL-divergence calculated above (and which depends on the latent representation learned by the encoder through the parameters of the normals).

In this way the network keeps the representation bags close to each other, facilitating the continuity of the representation space, while trying to differentiate the content of one bag from another, ensuring pattern learning. The following scheme shows the graphical representation of a Varational Autoencoder 1.10.

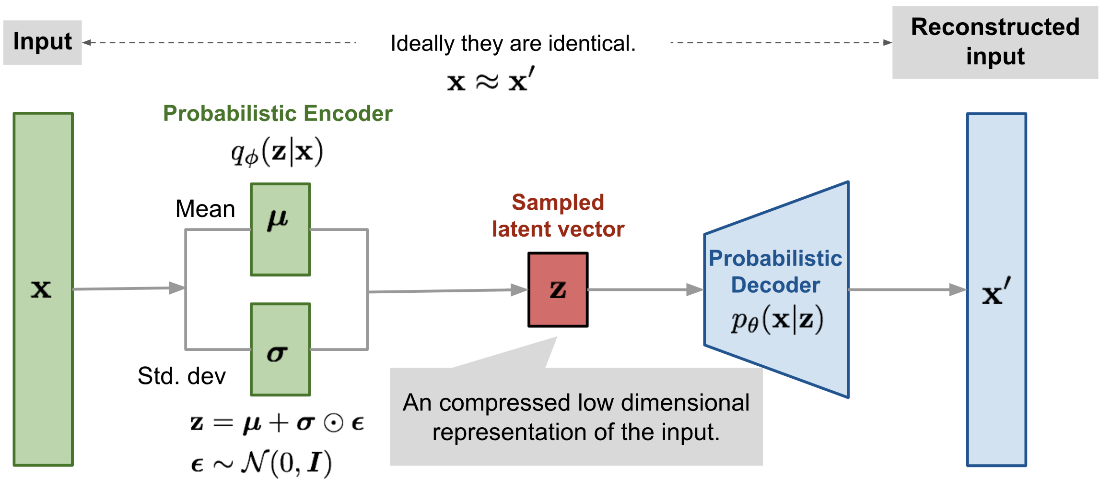

Input
Ideally they are identical.
$\mathbf{x} \approx \mathbf{x}'$
Reconstructed input
Probabilistic Encoder
$q_\phi(\mathbf{z}|\mathbf{x})$
$\mathbf{x}$
Mean
$\boldsymbol{\mu}$
Std. dev
$\boldsymbol{\sigma}$
$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$
$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$
Sampled latent vector
$\mathbf{z}$
An compressed low dimensional representation of the input.
Probabilistic Decoder
$p_\theta(\mathbf{x}|\mathbf{z})$
$\mathbf{x}'$

Figura 1.10: Varational Auto encoder architecture scheme [15].

1.6.2. Review of the state of the art of machine learning applied to detect welding defects

As mentioned before welding defects and discontinuities could lead to a massive failure in machines and structures that may cost time, money and most importantly human life's, therefore, in recent years a lot of work has been done to create efficient algorithms to detect these defects. Machine learning techniques offer a huge framework to explore, a state of the art of these techniques applied to detect welding defects is presented below.

In 1999 Madigan [11] was one of the first studies that tried to develop an algorithm that used electrical parameters to detect defects in welding. This article used automatic GMAW. The defects were intentionally made on production parts in test runs in an automotive parts factory. Five defect conditions were tested: lack of shielding gas, oily parts, the torch located off the joint, thin part sections causing melt-through and large root openings. In the article the algorithm was made by using the standard deviation of a normal sign of the free defects welding to create a threshold to detect the anomalies. Although this work was successful in detecting faults, the algorithms showed a mixed sensitivity to melt-through from large root openings. Also, this method is not accurate enough to detect the criticality of the defect in welding.

In 2001 Wang and Zhao [14], conducted research to automatically monitor the quality of welding produced by plasma arc welding by using the acoustic signal. The main goal of the research was to measure the keyhole maintenance, in the words of the authors “is a critical technique to achieve good quality welds”. To analyze the acoustic signal, an algorithm was carried out that used the Fourier transform to create a threshold. This work was successful in detecting faults in the keyhole maintenance, but it does not delve deeply into other types of failure.

In 2009 Garcia-Allende [3] conducted a research based on feature selection and artificial neural networks applied to arc-welding on-line quality monitoring. optical fibber embedded within the welding torch was used to capture the plasma radiation, with the data a feature extraction was performed by using the Wavelet Transform and then computed by an Artificial Neural Network (ANN) to detect defects (gas flow reduction, incision, and Low welding current). This work was successful in detecting faults by using an ANN, but it did not use the full strength of computer vision to analyze the Wavelets.

In 2014 Zhang and Chen [16]. Created an online welding quality monitoring based on feature extraction of arc voltage signal. The parameter extractions were made by using standardized moments of the signal both in the time and frequency domain and by using a threshold to identify welding's defects. This work showed that the welding defects detection can be improved by using the frequency domain of the signal.

Finally, in recent years there has been an increased interest on using machine learning algorithms, the main two articles that recently have been studied this are: First, (Hou, W., Wei, Y., Jin, Y., Zhu, C. (2019), [6]). in which it was used Convolutional ANN to analysis X-ray images of welding to classify the following welding defects: Lack of penetration, porosity, slag inclusion and crack, and (Huang, Y., Yang, D., Wang, K., Wang, L., Zhou, Q. (2020). Second, [7]) which conducted a research to use information theory and support vector machine algorithms to classify welding defects in GMAW. This last article proved that it is possible to use the voltage and current to classify welding defects, but there is still a need to continue exploring a more diverse range of ML algorithms (because this paper just focused on Suport vector machine) and also, increase the data set to explore deep learning algorithms.

After analyzing the main works carried out on the detection of welding defects it can be concluded that, although in recent years an effort has been made to create more effective algorithms, insufficient data has been tested, especially current and voltage, to train deep learning algorithms. There is no clear concept of the types of defects that must be detected, especially in an industry approach and most current algorithms have trouble being interpretable, being more like a “black box”. To contribute to these problems is why this original work is presented which proposes taking a data set with significant defects for the industry and large enough to test a wide range of algorithms.

Capítulo 2

Methodology

This chapter presents the methodology to achieve the specific objectives.

2.1. Methodology summary

The current thesis project is divided into two main parts. The first part is the data acquisition process, which involves the planning of the experiment, the setup, and the process of measuring current and voltage. The second part is the analysis of the data using machine and deep learning techniques.

The steps needed to accomplish the data acquisition are the following: selecting material to be tested based on its usage in practical applications, building or using a data acquisition system attached to the Lincoln s500 welding machine (The current device present in the ccwj lab), and finally planning the welds to be made.

An intermediate step is necessary before the usage of machine learning algorithms, which is labeling the data; once the experiments are finished and the current and voltage are collected, the plates are sent for an ultrasonic testing, which provides the physical location of each discontinuity.

Finally, once the data is labeled, a machine learning-based algorithm is constructed that may predict the discontinuities. To do so, several steps have to be followed. Fist of all, data visualization has to be carried out in order to have a first approach with a physical intuition; dimensional reduction and feature extraction have to be performed in order to extract the best representation of what is can be seen as a good or bad weld depending on the current and voltage signals.

Figure 2.1 shows the flow chart of the methodology. Each green box represents a major milestone of the project, and each blue box represents the most critical steps during this milestone.

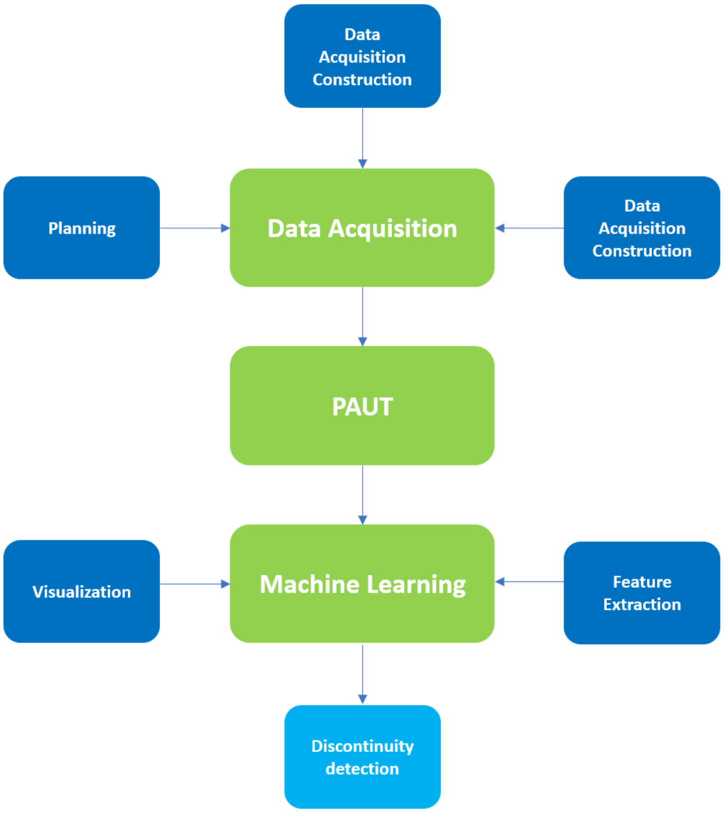


Figura 2.1: General flow chart of the methodology.

2.1.1. Experimental setup

The experimental geometric setup proposed is shown in Figure 2.2. The material selected for the experiment is carbon steel ASTM A36 or (CSA G40.21 44W / 300W), which is commonly used for the structural construction and ship manufacturing [5].

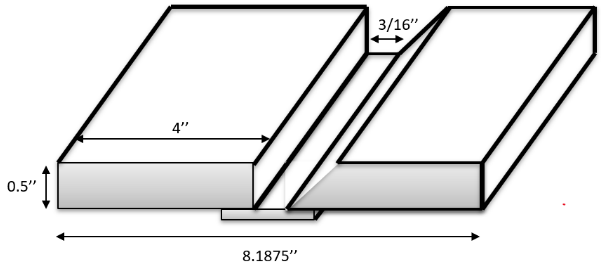


Figura 2.2: General flow chart of the methodology.

The joint configuration consisted of two flat plates 8"x10"x0.5" with 1"x10"x0.25" backing strips, welded by a multipass wire feed welding process (FCAW). Welding was performed in the 1G position. Regarding travel speed, volts and amps values were obtained through a standardized procedure for construction and welding with the type of material selected, which can be seen in the Appendixes.

The device used to collect the data was a National Instrument data acquisition card and the Labview software. Figure 2.3 shows the diagram of the acquisition circuit; the current was taken with a hall sensor connected to the earth cable of the welding, and the voltage was taken with two shunt sensors: one connected to the torch and the other to the earth. All three connections were taken to a transducer to reduce the electric power of welding so as to not destroy the acquisition card.

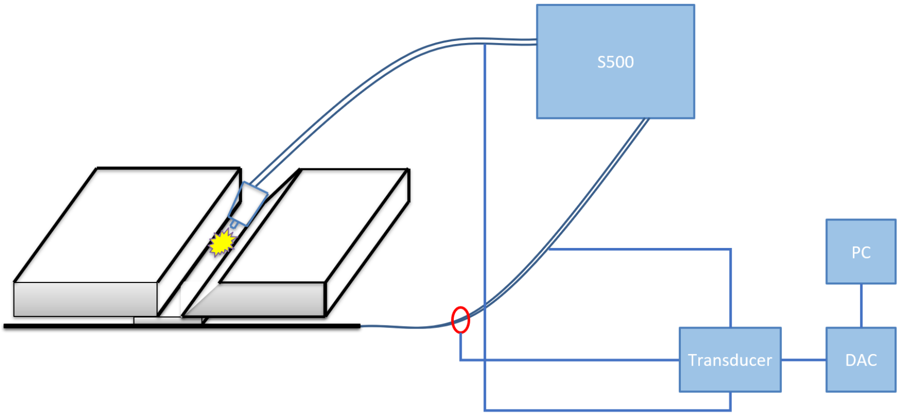


Figura 2.3: Flow chart of the data acquisition system.

The experimental procedure that was used during the lab work to acquire the welding electric parameters consisted of the following steps:

1. Fit up and mark amps, volts, and id numbers in the plait
2. Write down on a whiteboard: date, id number, amps, volts, layer and pass
3. Turn on video recording without a filter lens

   Film whiteboard

   Film ruler over plate to convert pixel over distance
4. Slide filter lens over camera and keep recording
5. Welder rings the bell
6. Start the data acquisition system
7. Start welding
8. Start recording time using a clock watch
9. Complete welding
10. Stop recording data, video and time
11. Clean slag
12. Visual inspection and record defects on plate with soap stone
13. Record on plate pass, id, time, Fill thickness value
14. Populate spreadsheet with picture and defects description
15. Go back to step 2 and repeat

2.1.2. Treatment of the data

Once the data from the laboratory experiments were obtained, the following steps outlined in Figure 2.4 were followed. First, a data warehouse is constructed with measurements which must be appropriately labeled in windows of approximately 0.5 to 1 cm. All this process is done thanks to the video recording which is used to track the position of the torch at any moment during the experiment with the help of the free source Tracker software.

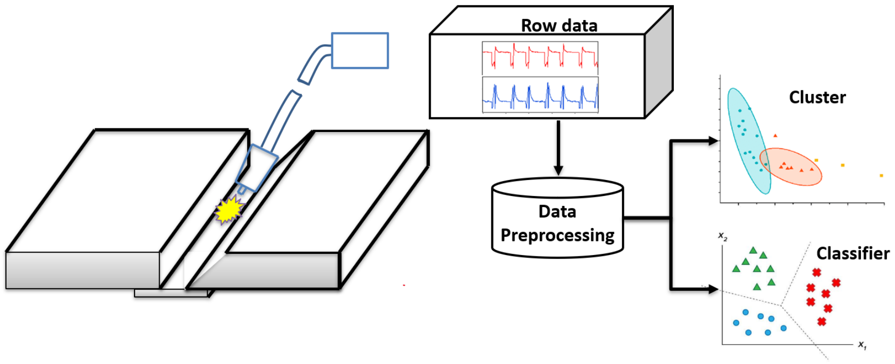


Figura 2.4: General flow chart of the data processing.

Finally, two types of algorithms were created: an anomaly detection and a classifier based on machine learning techniques. The procedure to make this algorithm once the data is appropriately labeled consists of:

- Use feature extraction parameters like mean, variance, peak to peak from the time-dependency data
- Use frequency-domain data by applied Fourier and Wavelet transform and applied feature extraction
- Use traditional machine learning algorithms to create the outlier detection as a model for the current industry status

  One-class support vector machine

  K-means clustering

  Isolation Forests
- Use traditional machine learning algorithms to create a the classifier algorithm

  Support vector machine learning

  Random forest

  Gradient boosting
- Use deep learning algorithms to compare its performances with the traditional ones

  Convolutional neural network

  Denoising Autoencoder

  Variational Autoencoder learning

Finally, once the best algorithms are selected, the algorithm is modified to predict the probability of detecting a failure.

Capítulo 3

Experimental results

This chapter presents the experimental results obtained during the lab measurement.

3.1. Welding results

The following tables 3.1 and 3.2 summarize all the welds made in the laboratory. In total approximately 20 meters of welds were made. The nomenclature used was the following:

- G: Good weld
- Por: Porosity
- Lf: Lack of fusion
- Lp: Lack of penetration
- Un: Undercut
- Wh: Wormhole

As it can be seen in both tables, at the beginning of the process, welding was unstable, which led to several non induced discontinuities. However, from the 6th plate forward, the procedure was mastered. Therefore, using mainly the second table 3.2 for the learning process of the algorithms is proposed.

Tabla 3.1: Welding summarize part 1.

Plate	ID	Layer	Profundity of the weld inches	Quality 1/4	Quality 2/4	Quality 3/4	Quality 4/4
2	9	Root 1	0.360”	G but Por	G but Por	G but Por	Lf
2	10	Hot 1	0240”	G but Por	G but Por	G but Por	G but Por
2	11	Hot 2	0.180”	G but Por	G but Por	G but Por	G but Por
2	12	Fill 1	-	G but Por	G but Por	G but Por	G but Por
2	13	Fill 2	0.08”	G but Por	G but Por	G but Por	G but Por
2	14	Cap 1	0”	G but Por	G but Por	Striation	Striation
2	15	Cap 2	0”	Striation	Striation	Striation	Striation
2	16	Cap 3	0”	Striation	Striation	Striation	Por
3	17	Root 1	0.320”	Salg and Lf	Lf	Lf	Lf
3	18	Hoot 1	0.230”	G	G	G	G
3	19	Hoot 2	0.230”	Lf	G but Por	G but Por	G but Por
3	20	Fill 1	0.150”	G	G	G	G
3	21	Fill 1 pass 2	0.150”	G	G	G	G
3	22	Fill 2	-	-	-	-	-
3	23	-	-	-	-	-	-
3	24	Cap 1	0”	G	G	G	G
3	25	Cap 2	0”	G	G	G	G
3	26	Cap 3	0”	G	G	G	G
3	27	Cap 4	0”	G	G	G	G
4	28	Root 1	0.310”	Pl	Pl	G	G
4	29	Hoot 1	0.150”	Lf	Lf	Lf	Lf
4	30	Hoot 2	0.200”	G	G	Lp	Lp
4	31	Fill 1	0.120”	-	Lf	-	Lf
4	32	Fill 2	0”	G	G	G	G
4	33	Cap 1	0”	-	Wh	-	Por
4	34	Cap 2	0”	-	Wh	Por	Por
4	35	Cap 3	0”	Por	Por	Por	Por
5	36	Root 1	0.380”	Minor Lf	Minor Lf	Minor Lf	Minor Lf
5	37	Hoot 1	0.250’	Lf	Lf	Lf	Lf
5	38	Hoot 2	0.170”	Wh and Lf	Wh and Lf	Wh and Lf	Wh and Lf
5	39	Fill 1	0.090”	Lp	Lp	Lp	Lp
5	40	Fill 2	0.040”	Por and Wh	Por and Wh	Por and Wh	Por and Wh
5	41	Fill 3	0.060”	Lp	Lp	Lp	Lp
5	42	Cap 1	0”	Por	Por	Por	Por
5	43	Cap 2	0”	Por	Por	Por and Wh	Por and Wh
5	44	Cap 3	0”	Por	Por	Por	Por
6	45	Root 1	0.360”	G	G	G	Wh
6	46	Hoot 1	0.230”	G	G	G	G
6	47	Hoot 2	0.160”	G	G	G	Wh
6	48	Fill 1	0.040”	G	G and Por	G	Wh
6	49	Fill 2	0.040”	G	G	G	G
6	50	Cap 1	0”	G	G	G	G
6	51	Cap 2	0”	Por	Por	Por	Por
6	52	Cap 3	0”	Por	Por	Por	Por

Tabla 3.2: Welding summarize part 2.

Plate	ID	Layer	Profundity of the weld inches	Quality 1/4	Quality 2/4	Quality 3/4	Quality 4/4
7	53	Root 1	0.390”	Lf	Lf	Un	Un
7	54	Hoot 1	0.310”	Un	Un	Un	Un
7	55	Hoot 2	0.200”	G	G	G	G
7	56	Fill 1	0.150”	Un	Un	Un	Un
7	57	Fill 1 1/2	0.040”	G	G	G	Wh
7	58	Fill 2	0.020”	G	G	G	G
7	59	Cap 1	0”	G	G	G	G
7	60	Cap 2	0”	G	G	G	G
7	61	Cap 3	0”	G	G	G	G
8	62	Root 1	0.230”	Lp	Lp	Lp	Lp
8	63	Hot 1	0.150”	Lf	Lf	Lf	Lf
8	64	Hot 2	0.150”	Lp	Lp	Lp	Lp
8	65	Fill 1	0.080”	Lf	Lf	Lf	Lf
8	66	Fill 2	0.60”	Lp	Lp	Lp	Lp
8	67	Cap 1	0”	Lf	Lf	Lf	Lf
8	68	Cap 2	0”	G	G	G	G
9	69	Root 1	0.340”	G	G	Un	Un
9	70	Hot 1	0.220”	G	G	Un	Un
9	71	Hot 2	0.110”	G	G	Un	Un
9	72	Fill 1	0.020”	Por	Por	Un	Un
9	73	Fill 2	0.020”	G	G	Lp	Lp
9	74	Cap 1	0”	G	G	Poor	Poor
9	75	Cap 2	0”	G	G	Poor	Poor
9	76	Cap 3	0”	G	G	G	G
10	77	Root 1	0.320”	G	G	Lf	Lf
10	78	Hot 1	0.140”	G	G	Un	Un
10	79	Hot 2	0.130”	G	G	Un	Un
10	80	Fill 1	0.080”	G	G	Lf	Lf
10	81	Fill 2	0.030”	G	G	G	G
10	82	Cap 1	0”	Un	Un	Un	Un
10	83	Cap 2	0”	G	G	G	G
10	84	Cap 3	0”	G	G	Por	Por

3.1.1. Welding current and voltage graphs

This section presents some of the graphs created to visualize the current and voltage signal gathered while the weldings were made.

Good well

As it can be seen the from Figure 3.1 a good quality weld is uniform and straight with no slag, cracking, or holes at all. The Figures 3.2 and 3.10 shown the voltage and current of the experiment with scaled values.

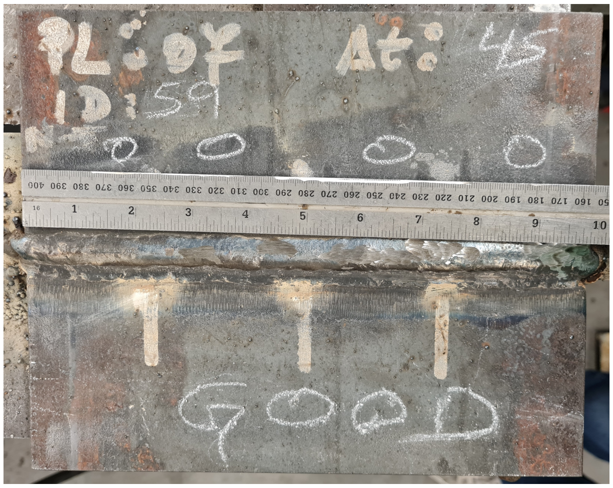


Figura 3.1: Example of a good weld taken from the seven plate.

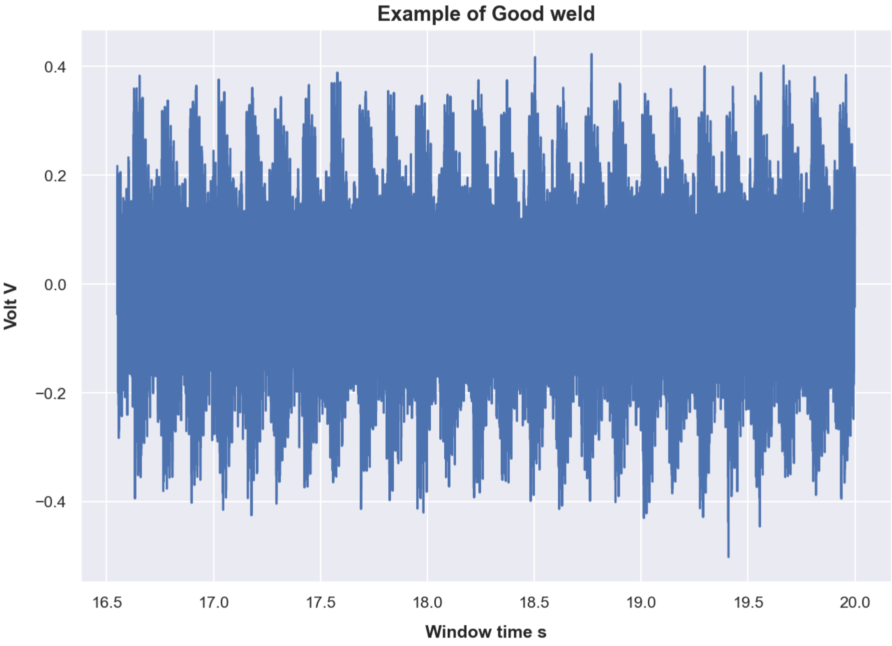


Figura 3.2: Voltage example of a good weld taken from the seven plate.

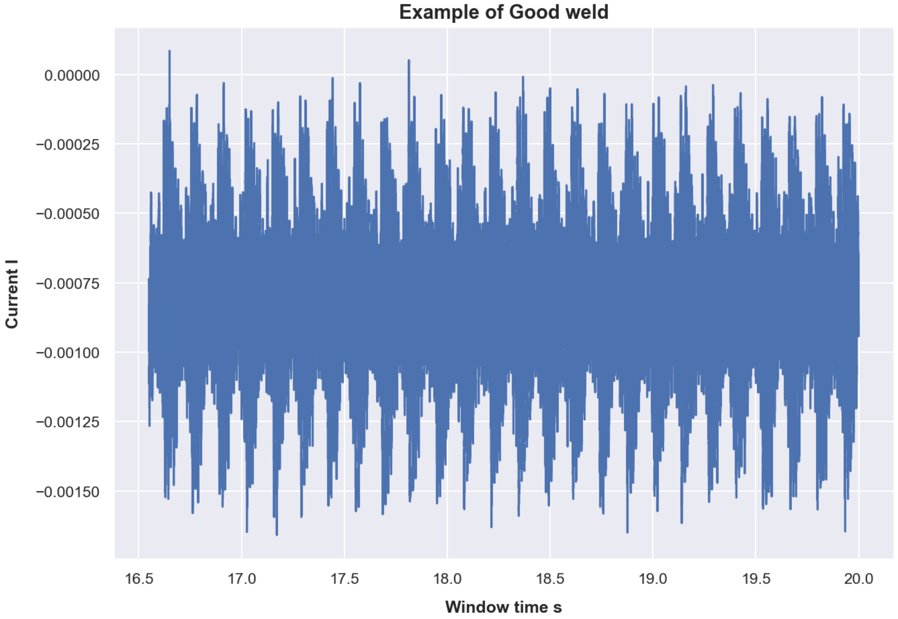


Figura 3.3: Current example of a good weld taken from the seven plate.

Lack of fusion

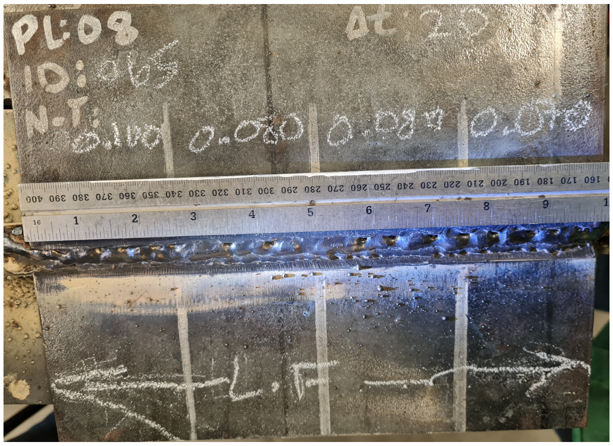


Figura 3.4: Example of a bad weld with lack of fusion taken from the eighth plate.

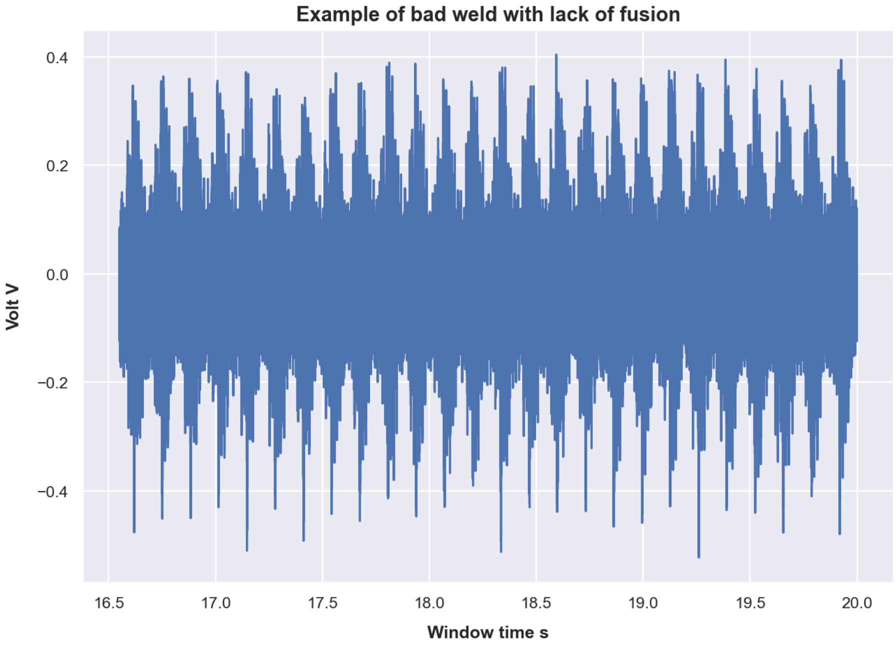


Figura 3.5: Voltage example of a bad weld with lack of fusion taken from the seven plate.

It can be seen from both Figures 3.11 and 3.12 that when lack of fusion is ocurring there is a reduction in the power of the signal, especially it can be seen in the current.

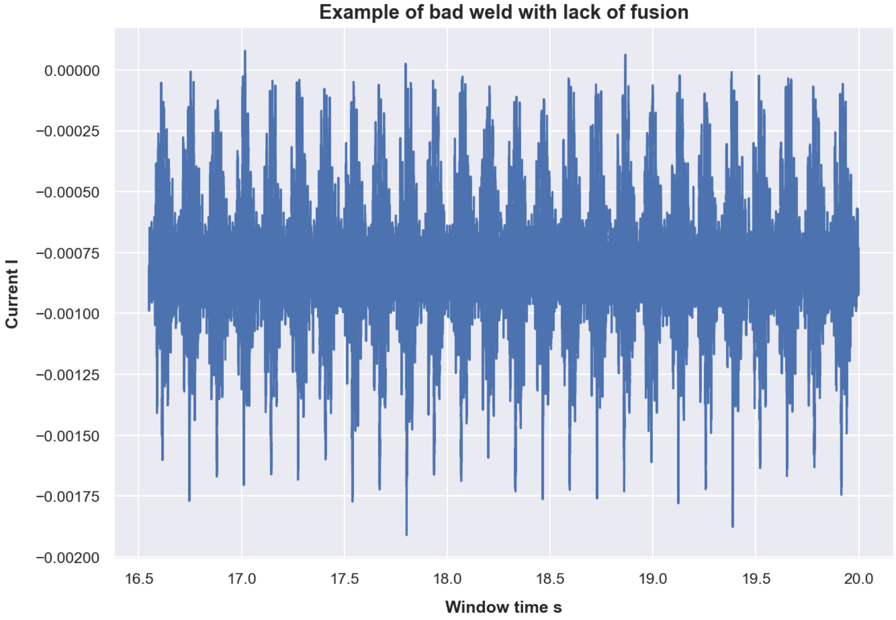


Figura 3.6: Current example of a bad weld with lack of fusion taken from the seven plate.

Lack of penetration

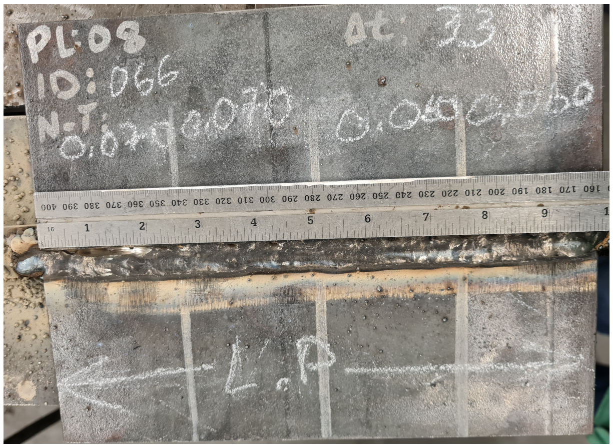


Figura 3.7: Example of a bad weld with lack of penatration taken from the eighth plate.

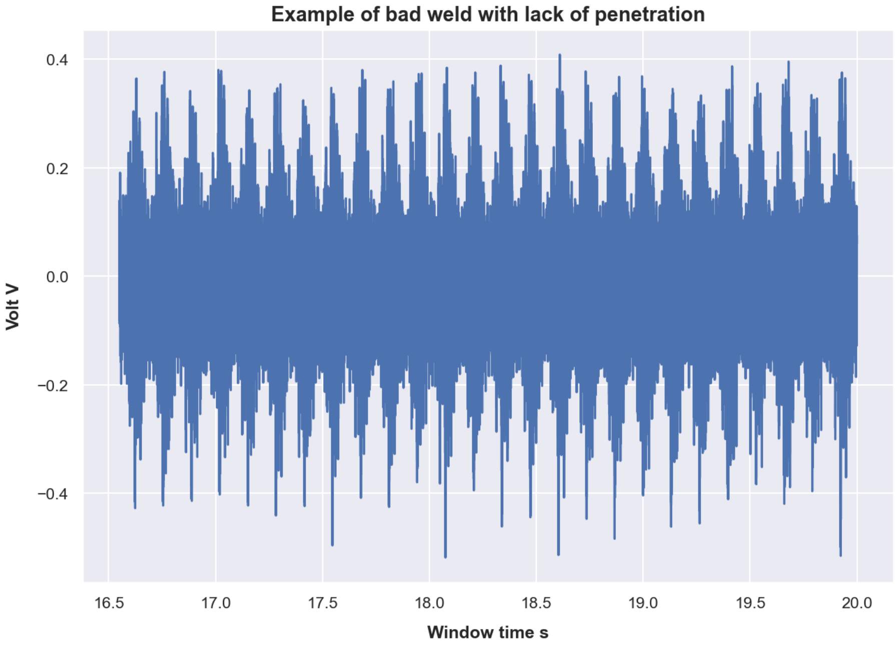


Figura 3.8: Voltage example of a bad weld with lack of penetration taken from the eighth plate.

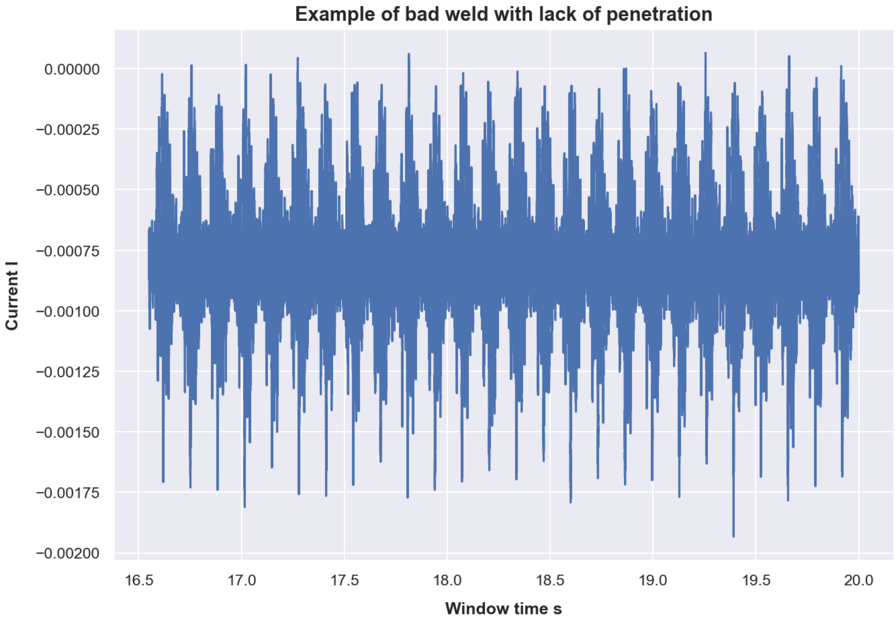


Figura 3.9: Current example of a bad weld with lack of penetration taken from the eighth plate.

Undercut

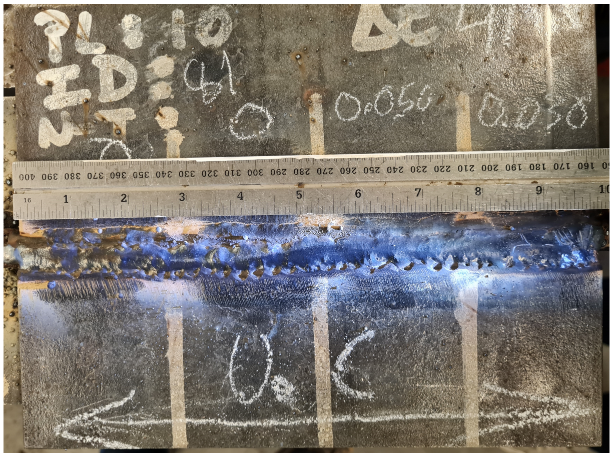


Figura 3.10: Example of a bad weld with undercut taken from the eighth plate.

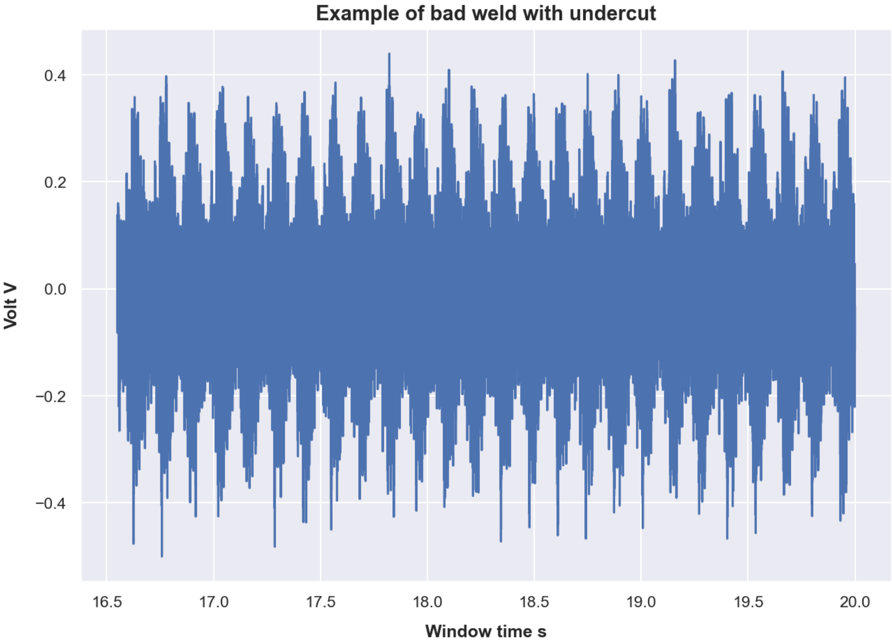


Figura 3.11: Voltage example of a bad weld with undercut taken from the tenth plate.

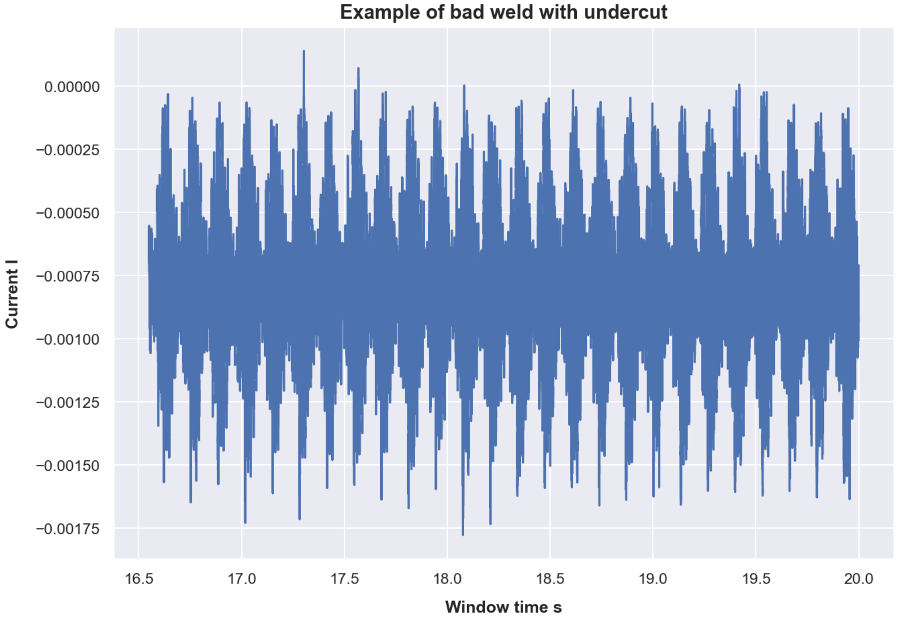
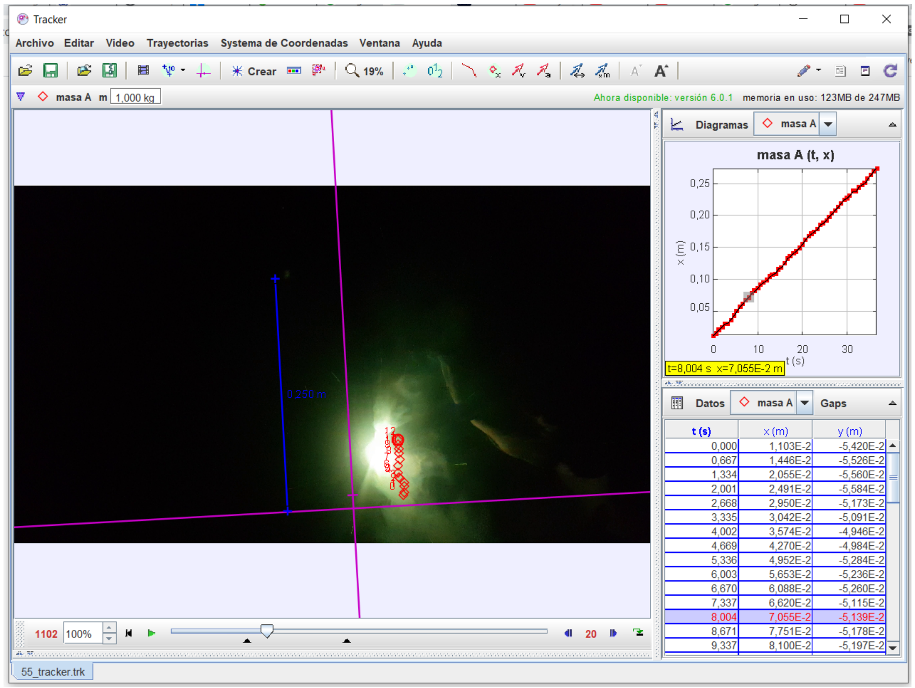


Figura 3.12: Current example of a bad weld with undercut taken from the tenth plate.

3.1.2. Measurement of torch position

The torch position was measured with the objective to appropriate label the data, to do so, the Tracker software it was used as it can be seen from the Figures 3.13 and 3.14.



t=8,004 s x=7,055E-2 m

t (s)	x (m)	y (m)
0,000	1,103E-2	-5,420E-2
0,667	1,446E-2	-5,526E-2
1,334	2,055E-2	-5,560E-2
2,001	2,491E-2	-5,584E-2
2,668	2,950E-2	-5,173E-2
3,335	3,042E-2	-5,091E-2
4,002	3,574E-2	-4,946E-2
4,669	4,270E-2	-4,984E-2
5,336	4,952E-2	-5,284E-2
6,003	5,653E-2	-5,236E-2
6,670	6,088E-2	-5,260E-2
7,337	6,620E-2	-5,115E-2
8,004	7,055E-2	-5,139E-2
8,671	7,751E-2	-5,178E-2
9,337	8,100E-2	-5,197E-2

Figura 3.13: Example of the use of the tracker software for measuring the torch position during the weld.

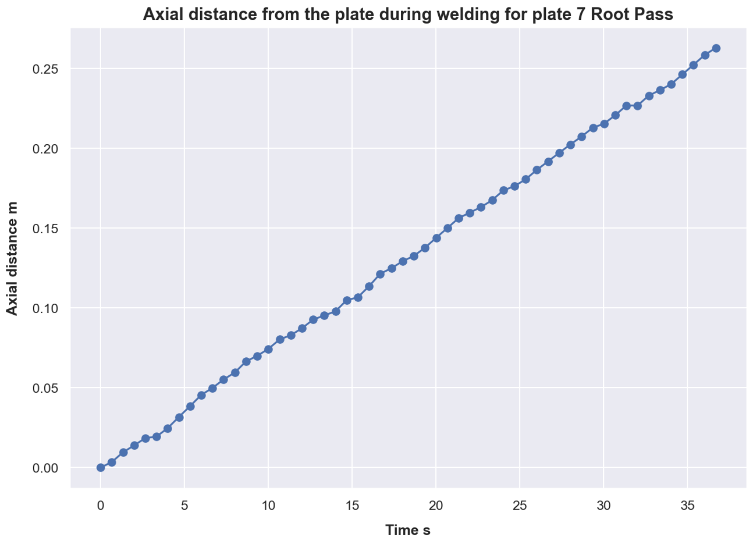


Figura 3.14: Axial distance from the plate during welding for plate 7 Root Pass.

Capítulo 4

Results of classical statistics machine learning for outlier prediction

This chapter presents the results of machine learning techniques applied for outlier detection as a model for current industry status. First of all, an introductory data processing and visualization are shown to then move to basic machine learning algorithms.

4.1. Data preprocessing

Data preprocessing for classical machine learning usage consists of several steps. Once the Tracker software is used, the results are saved in an Excel file. Then a Python routine is created which receives two inputs: the sheet names of the ID Plate where the points from the Tracker software were saved, and the minimal distance to be considered for the windows size. Later, an interpolation of the axial distance curve is computed to fit the points required for the windows sizes. The return of this routine is a NumPy arange with these points. In Figure 4.1 an example of the interpolation curve can be seen.

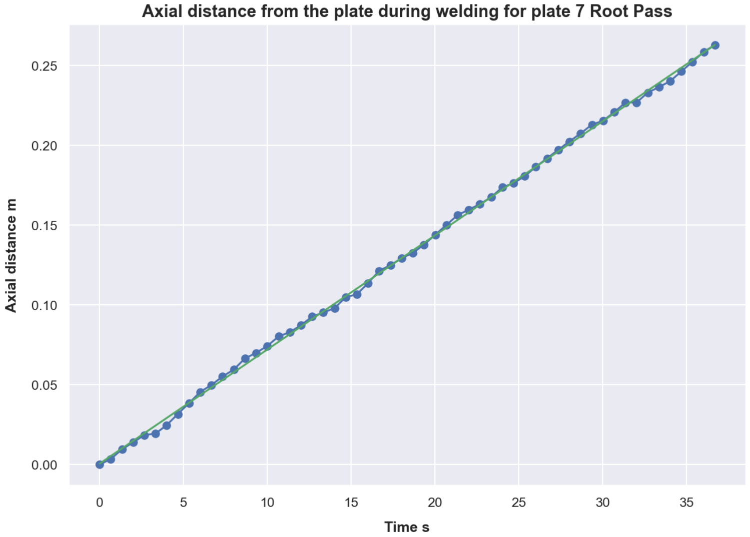


Figura 4.1: Axial distance interpolation from the plate during welding for plate 7 Root Pass.

After the torch velocity is computed throughout all its displacement, a second algorithm is created with Python. This routine takes three arguments: the ID of the plate, the label (good, undercut, lack of fusion or lack of penetration), and the overlap percentage. The output of this algorithm are the temporary windows of the current and voltage saved in an npz file. Some examples are shown in the following figures 4.2 and 4.3.

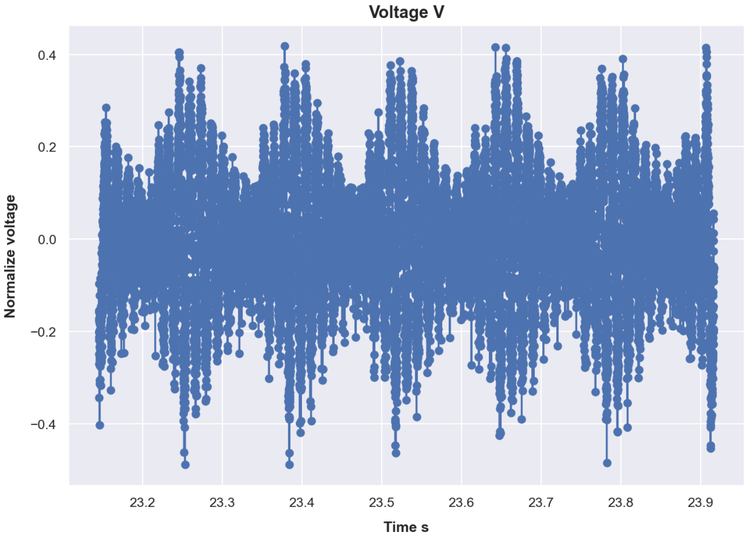


Figura 4.2: Example of voltage window for 50 mm of weld in the plate 7 id 59.

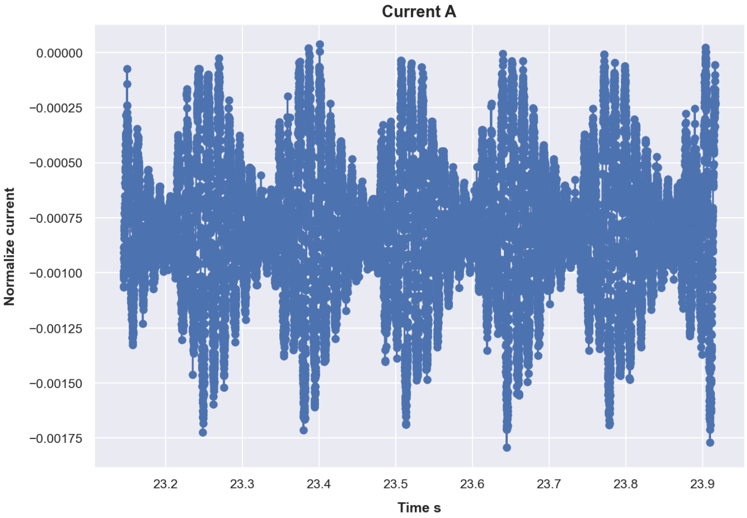


Figura 4.3: Example of current window for 50 mm of weld in the plate 7 id 59.

4.2. Feature extraction and visualization

For the extraction of parameters, two different approaches was used: the first one calculates statistical parameters, mean, variance, max value, min value, RMS, peak-peak, crest, kurtosis, and skew. These nine parameters were taken for the signal, its derivative and its integrative signal. Therefore, the total number of features is 24. The second approach uses Principal Component Analysis (PCA), which is by far the most popular dimensionality reduction algorithm, composed by two main steps: first, identification of the hyper plane in which the data is going to be projected. Second, the identification of the principal components.

The following Figure 4.4 shows the result of the visualization in 3D for good quality welding and lack of fusion welding.

The following sections present the results for each of the proposed classic machine learning algorithms and for each classical statistics parameters extraction.

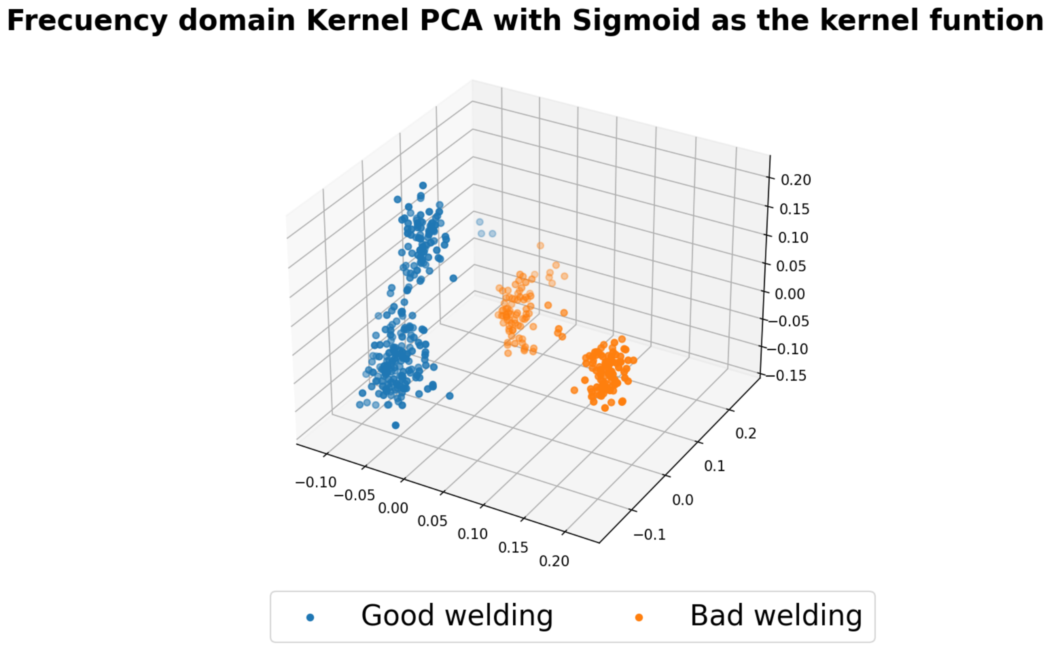


Figura 4.4: Example of visualization of good and bad welds with lack of fusion. Good id: 58, 59 and 60, Bad id: 63 and 65

4.3. Normal statistics Time domain results

This section presents the results for time domain Normal statistics classical machine learning algorithms. Figure 4.5 shows the matrix of confusion for the three proposed algorithms and Table 4.1 summarized the performance of each algorithm.

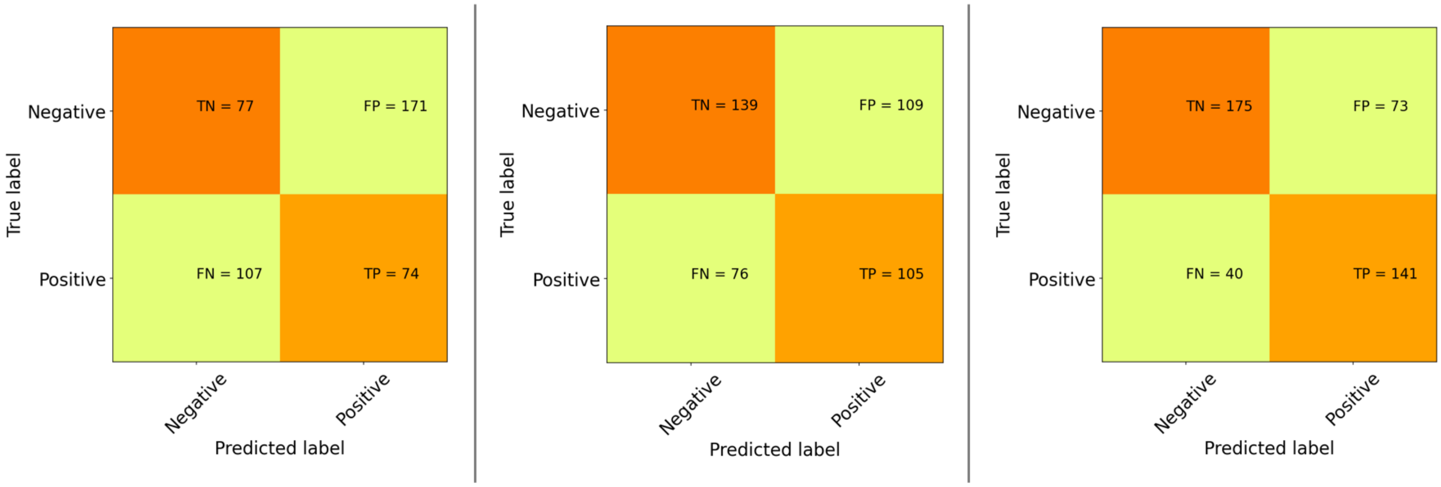


Figura 4.5: Confusion matrix for normal statistics time domain. From left to right: K-means, One-class Support Vector Machine and Isolation Forest. Good id: 58, 59 and 60, Bad id: 63 and 65

Tabla 4.1: Summary of normal statistics results in time domain.

Algorithm	Accuracy	Precision	Recall	F1score
K-means	0.56	0.48	0.57	0.52
Isolation Forest	0.81	0.73	0.86	0.79
Support Vector Machine	0.63	0.56	0.59	0.57

4.4. Normal statistics Frequency domain results

This section presents the results for Frequency domain Normal statistics classical machine learning algorithms. Figure 4.6 shows the matrix of confusion for the three proposed algorithms and Table 4.2 summarized the performance of each algorithm.

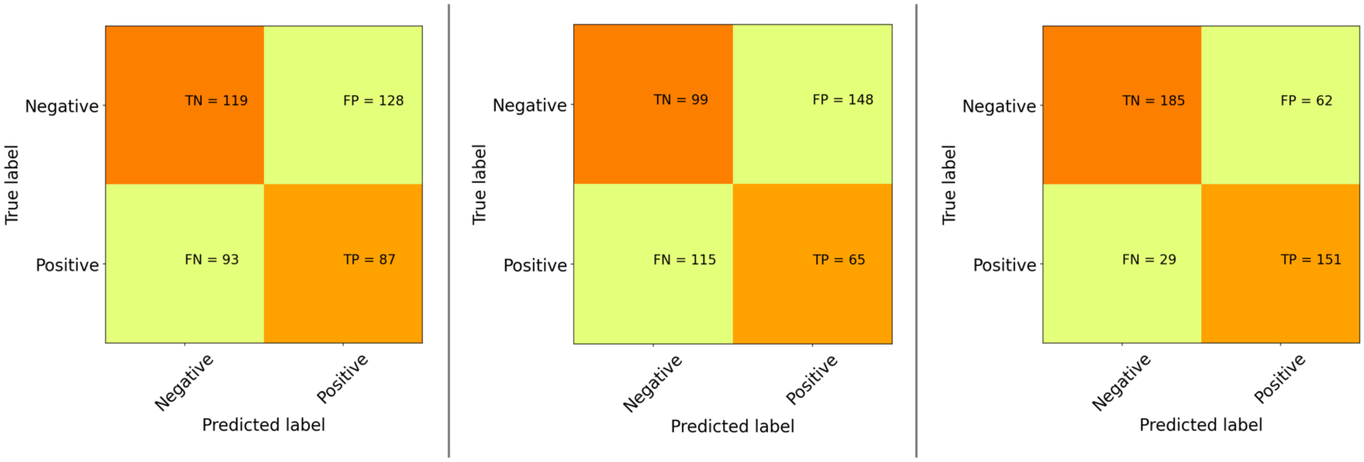


Figura 4.6: Confusion matrix for normal statistics frequency domain. From left to right: K-means, One-class Support Vector Machine and Isolation Forest. Good id: 58, 59 and 60, Bad id: 63 and 65

Tabla 4.2: Summary of normal statistics results in frequency domain.

Algorithm	Accuracy	Precision	Recall	F1score
K-means	0.54	0.46	0.55	0.50
Isolation Forest	0.92	0.84	1	0.91
Support Vector Machine	0.96	0.99	0.91	0.95

Capítulo 5

Data labeling and preprocessing for deep learning.

In machine learning, data labeling is the process of identifying raw data. Today, most practical machine learning models utilize supervised learning, which applies an algorithm to map one input to one output. For supervised learning to work, you need a labeled set of data that the model can learn from to make correct decisions. Data labeling typically starts by asking humans to make judgments about a given piece of unlabeled data.

Thus in this chapter it is presented the identification of the welds to be studied, Correlation of the visual inspection with the phased array of the discontinuities, Axial correlation using video recordings and measured current and voltage, the time windows to be analyzed and the use of transformations to obtain 2d representations of the signals.

5.1. Identification of the welds to be studied.

Through the visual inspection that was detailed by the tables 3.1 and 3.2 , the welds with the most accentuated defects were selected, that is, those in which the quality of the welding is bad enough to not be able to pass the PAUT test and the flaw is visually noticeable in the electrical signals that were taken. The selected welds were 37, 63, 65, 67 lack of fusion, 53, 54, 56 undercut.

Following the same idea implemented with the defective welds, the best welds made were selected, which correspond to beads 57, 58, 59, 60, 61, and 80.

Regarding lack of penetration, it was not possible to observe significant differences to be able to identify them through electrical signals, so this defect will not be considered for the performance of the algorithms. The following images show the chosen welds.

5.1.1. Welding images

Lack of fusion

In the following images show the strings welds selected with lack of fusion produced during the experiment.

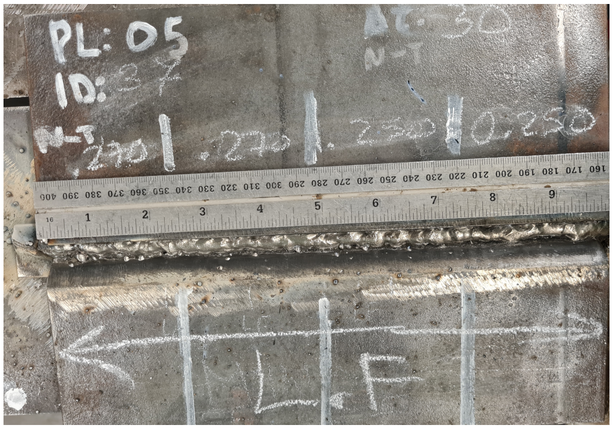


Figura 5.1: Weld 037 Lack of fusion.

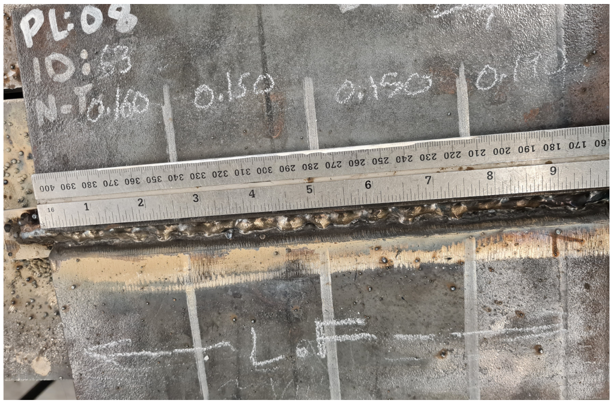


Figura 5.2: Weld 63 Lack of fusion.

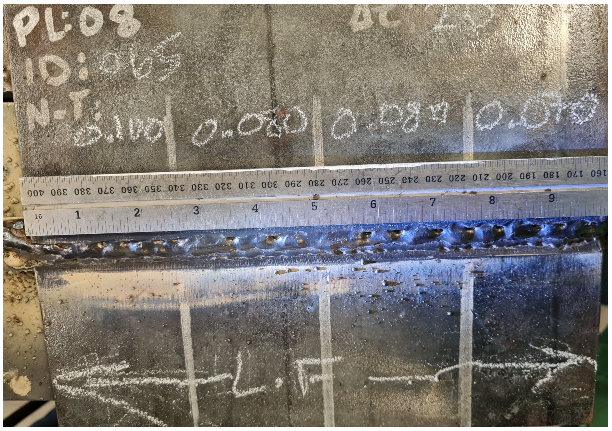


Figura 5.3: Weld 065 Lack of fusion.

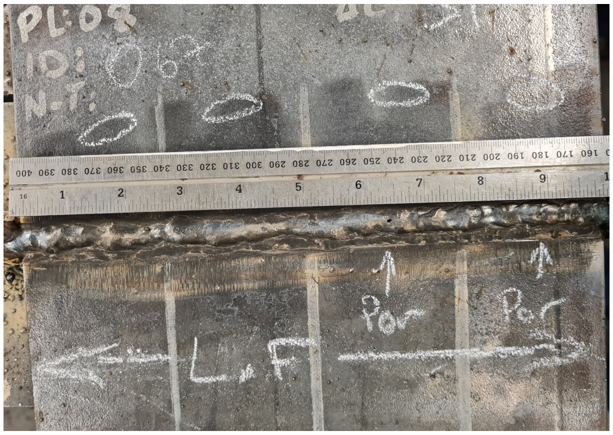


Figura 5.4: Weld 067 Lack of fusion.

Undercut

In the following images show the strings welds selected with undercut produced during the experiment.

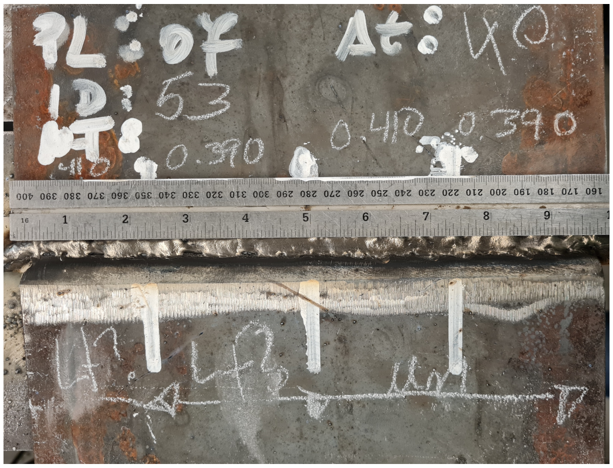


Figura 5.5: Weld 053 Undercut.

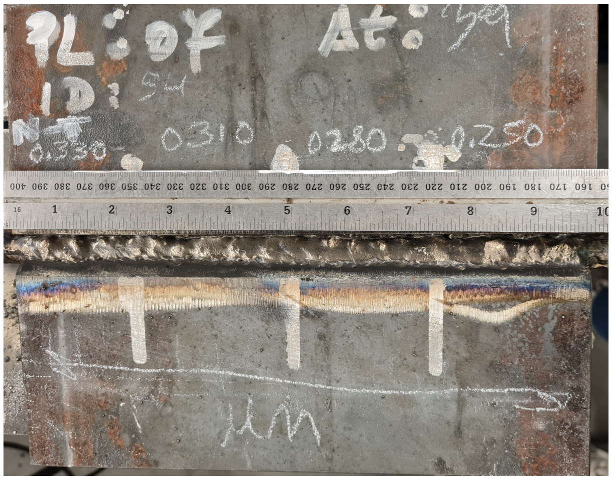


Figura 5.6: Weld 54 Undercut.

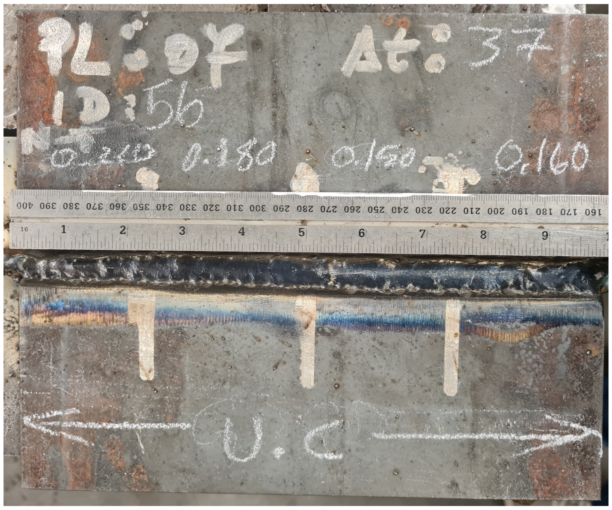


Figura 5.7: Weld 56 Undercut.

Good

In the following images show the strings welds selected with good quality produced during the experiment.

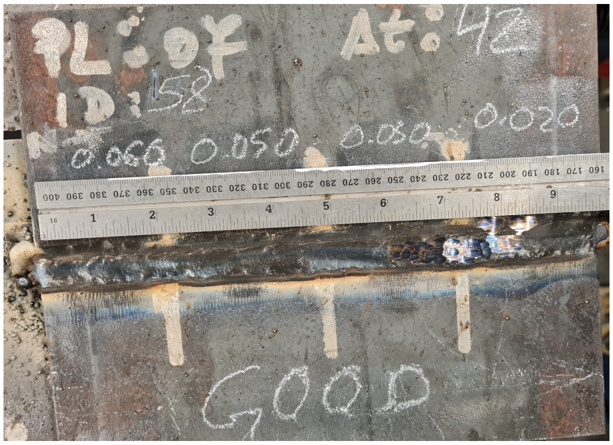


Figura 5.8: Weld 058 Good quality.

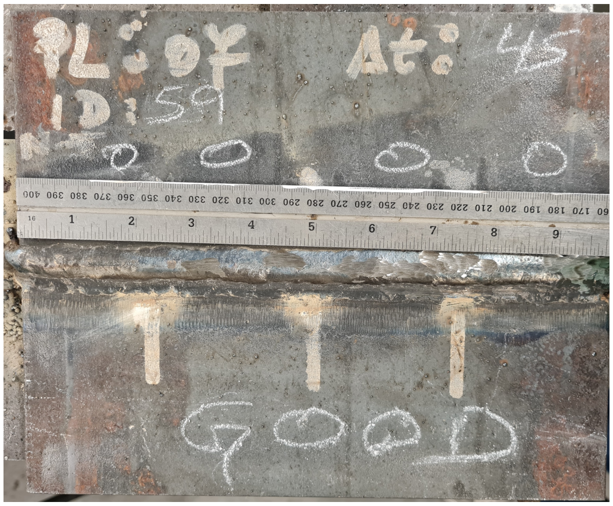


Figura 5.9: Weld 059 Good quality.

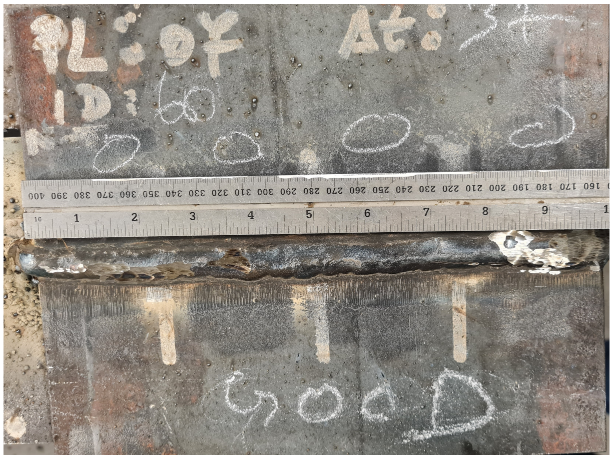


Figura 5.10: Weld 060 Good quality.

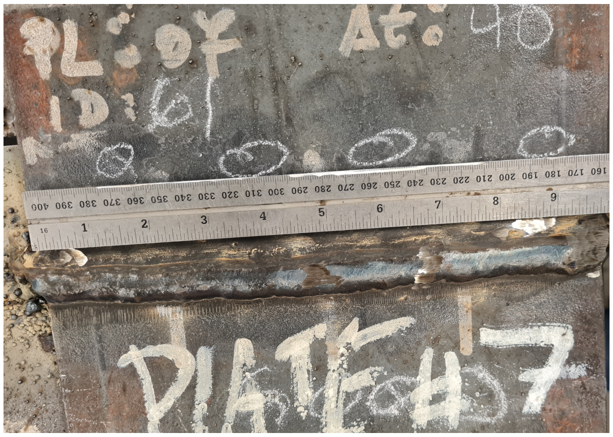


Figura 5.11: Weld 061 Good quality.

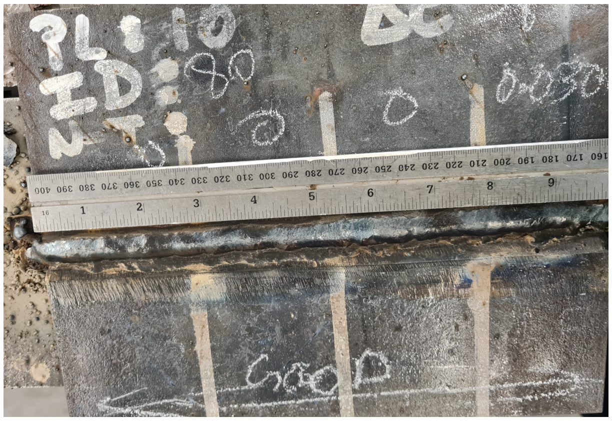


Figura 5.12: Weld 080 Good quality.

5.1.2. Fourier transform.

The Fourier transform is a mathematical transform that decomposes functions depending on space or time into functions depending on spatial frequency or temporal frequency.

The Fourier transform of a function is a complex-valued function representing the complex sinusoids that constitute the original function. For each frequency, the magnitude (absolute value) of the complex value represents the amplitude of a constituent complex sinusoid with that frequency, and the argument of the complex value represents that complex sinusoid's phase offset. If a frequency is not present, the transform has a value of 0 for that frequency.

In order to calculate this transform in the computer, it is used the fast Fourier transform wich is an algorithm that computes the discrete Fourier transform of a sequence, or its inverse. Fourier analysis converts a signal from its original domain (often time or space) to a representation in the frequency domain and vice versa, the following equation it is shown the fast Fourier transform.

$$\sum_{n=0}^{N-1} x_n exp(-i2\pi kn/N) \tag{5.1}$$

5.1.3. Spectrogram.

A 2D representation of frequencies of a given signal with time is called Spectrogram. In a spectrogram representation plot, one axis represents the time, the second axis represents frequencies and the colors represent magnitude (amplitude) of the observed frequency at a particular time. To implement the original signal is break into smaller frames(windows) and it is calculated the Fast Fourier 5.1 transform for each window. This way we will be getting frequencies for each window and window number will represent the time.

Some of the Spectrograms computed for the welds selected are presented in 5.13, 5.14 and 5.15:

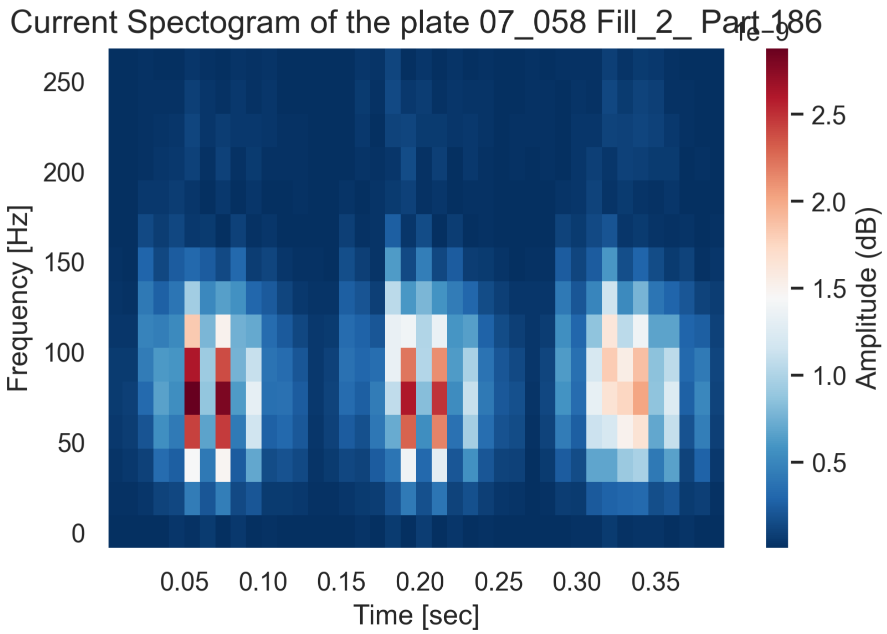


Figura 5.13: Current Spectogram of the plate 07 058 Fill 2 Part 186 (Weld Good quality).

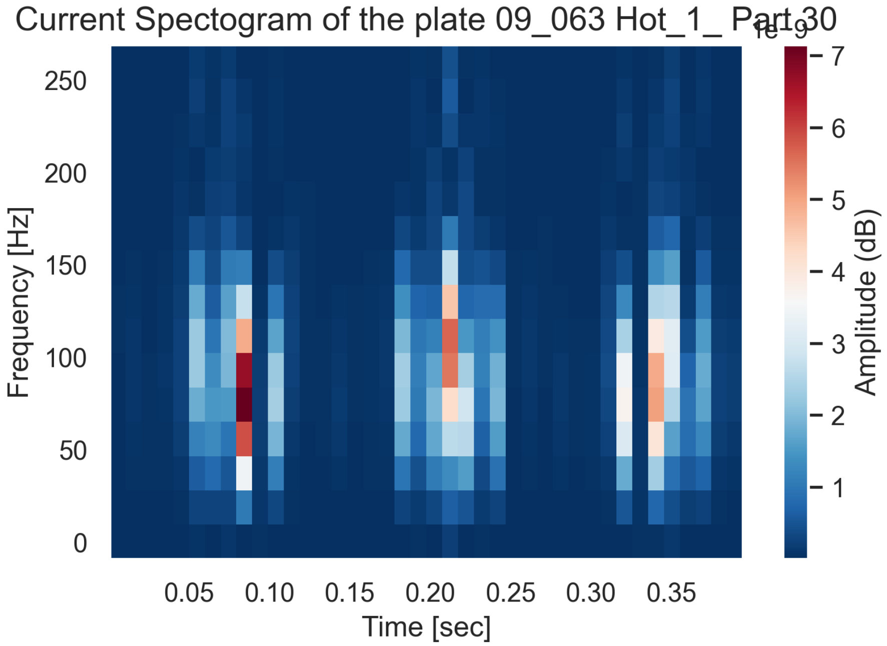


Figura 5.14: Current Spectogram of the plate 09 063 Hot 1 Part 30 (Lack of fusion).

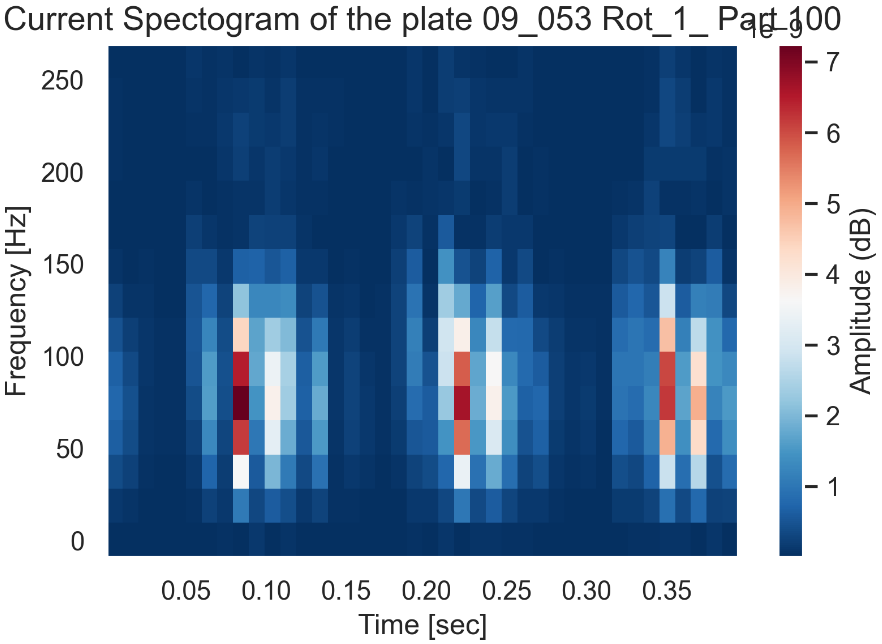


Figura 5.15: Current Spectogram of the plate 09 053 Rot 1 Part 100 (Undercut).

5.1.4. Wavelet transform.

Wavelet series is a representation of a square-integrable (real- or complex-valued) function by a certain orthonormal series generated by a wavelet. A Wavelet is a wave-like oscillation that is localized in time, an example is the Figure 5.16. Wavelets have two basic properties: scale and location. Scale defines how "stretched" or "squished" a wavelet is. This property is related to frequency as defined for waves. Location defines where the wavelet is positioned in time (or space).

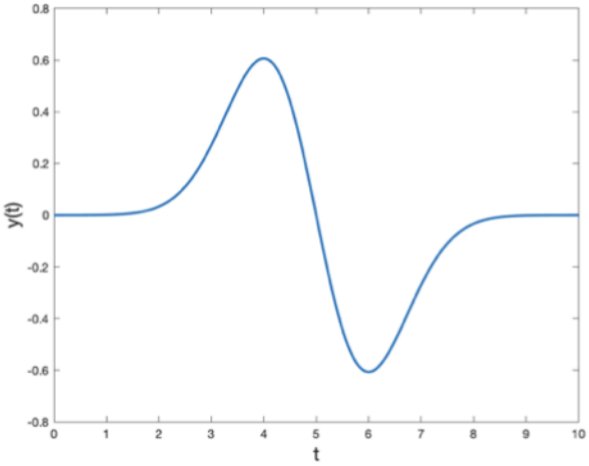


Figura 5.16: The first derivative of Gaussian Function.

The basic idea is to compute how much of a wavelet is in a signal for a particular scale and location, therefore a signal is convolved with a set wavelets at a variety of scales. In other words, is picked a wavelet of a particular scale (like 5.16). Then, is slided this wavelet across the entire signal, where at each time step is multiplied the wavelet and signal. The product of this multiplication gives a coefficient for that wavelet scale at that time step, then increase the wavelet scale and is repeated the process.

A scalogram is the absolute value of the continuous wavelet transform coefficients of a signal. By transforming the signal from the time domain to the frequency domain using the wavelet, the 1-D signal becomes a 2-D matrix, and it could be analyzed at multi resolutions.

Some of the scalograms computed for the welds selected are presented in figures5.17, 5.18 and 5.19.

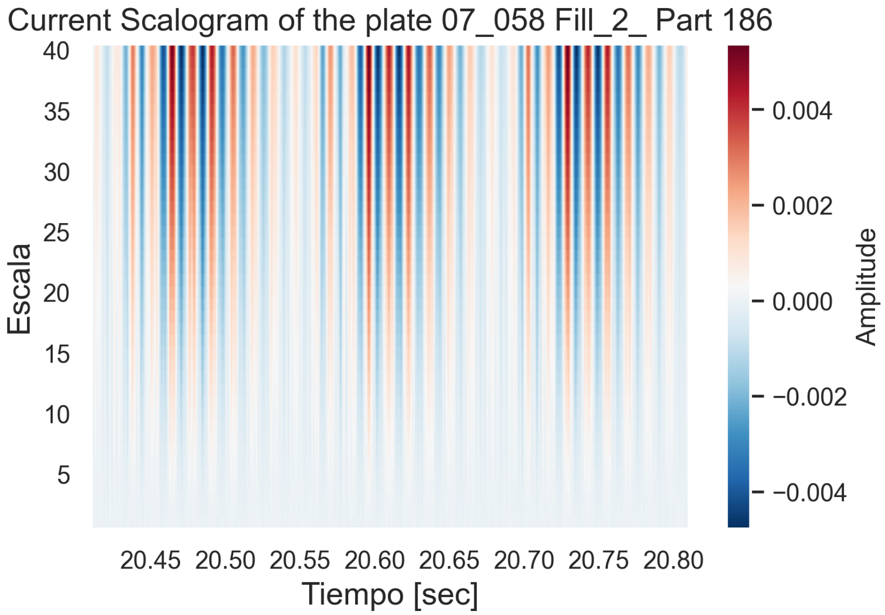


Figura 5.17: Current Scalogram of the plate 07 058 Fill 2 Part 186 (Weld Good quality).

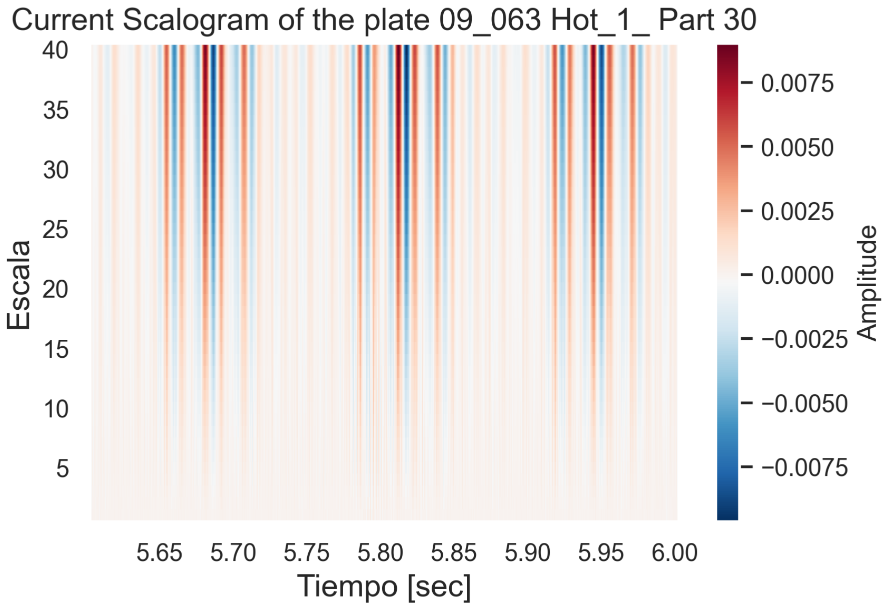


Figura 5.18: Current Scalogram of the plate 09 063 Hot 1 Part 30 (Lack of fusion).

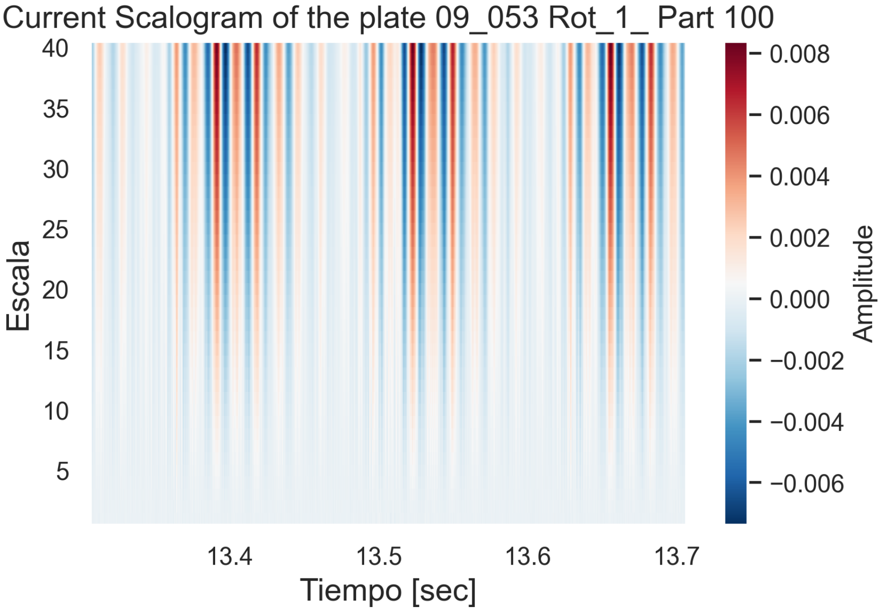


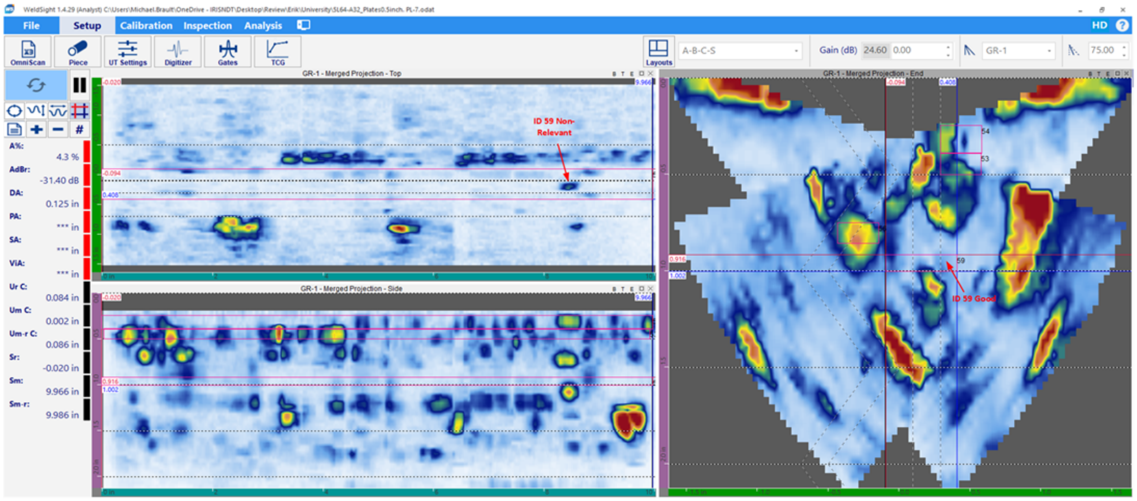
Figura 5.19: Current Scalogram of the plate 09 053 Rot 1 Part 100 (Undercut).

5.2. Phase array inspection.

Phased Array Ultrasonic Testing (PAUT) is an advanced nondestructive examination technique that utilizes a set of ultrasonic testing (UT) probes made up of numerous small elements, each of which is pulsed individually with computer-calculated timing (“phasing”). When these elements are excited using different time delays, the beams can be steered at different angles, focused at different depths, or multiplexed over the length of a long array, creating the electronic movement of the beam. PAUT can be used to inspect almost any material where traditional UT methods have been utilized and is often used for weld inspections and crack detection.

5.3. Relation with the Phased Array

This section presents the relation between the pictures and signals taken during the experiment and the results from the PAUT test. Welds 59 (Good), 63 (Lack of fusion) and 54 (undercut) were chosen. The following Figure shows the PAUT for the selected welds 5.20, 5.2 and 5.6. The colour map is based on the CSA W59-18, Table X.6 (standard covers welding requirements for carbon and low-alloy welded steel construction).



WeldSight 1.4.29 (Analyst) C:\Users\Michael.Brault\OneDrive - IRISNDT\Desktop\Review\Erik\University\SL64-A32_Plates0.5inch. PL-7.odat

A-B-C-S

Gain (dB) 24.60 0.00

GR-1

75.00

A%: 4.3 %

AdBr: -31.40 dB

DA: 0.125 in

PA: *** in

SA: *** in

ViA: *** in

Ur C: 0.084 in

Um C: 0.002 in

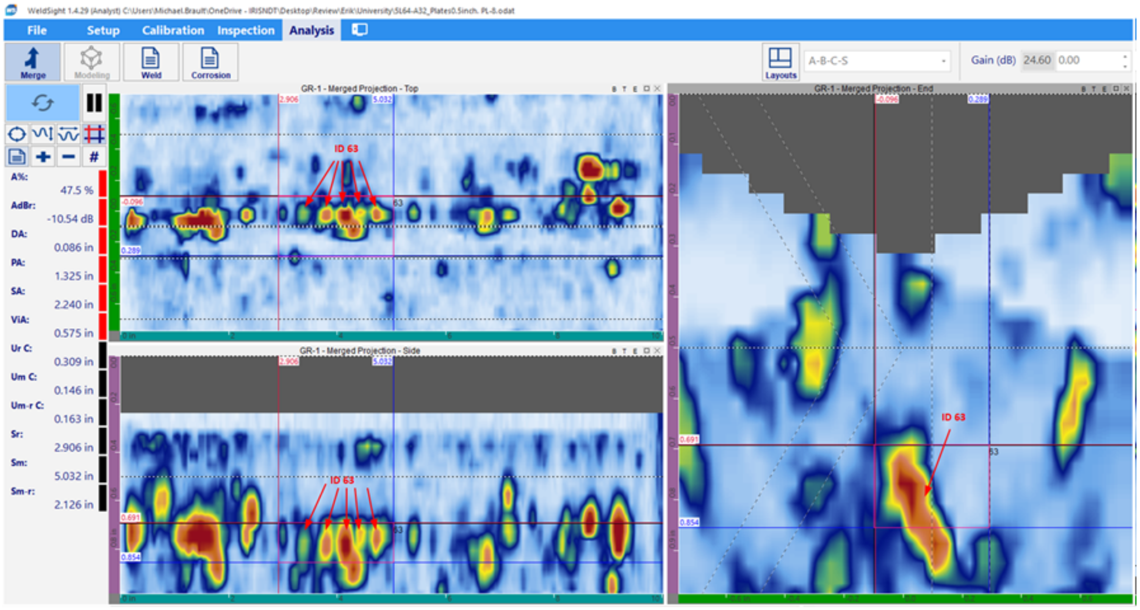
Um-r C: 0.086 in

Sr: -0.020 in

Sm: 9.966 in

Sm-r: 9.986 in

Figura 5.20: PAUT for the 59 weld pass.



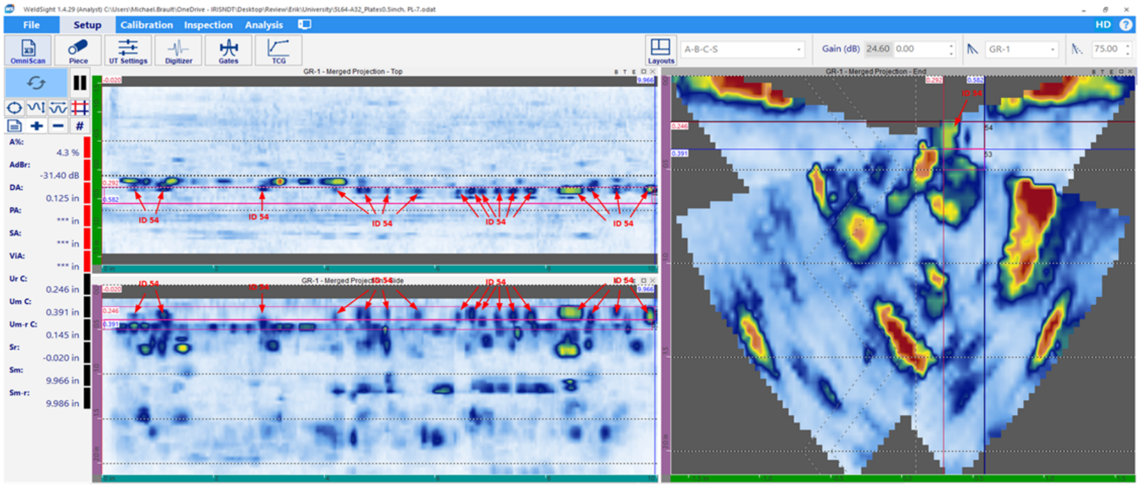
WeldSight 1.4.29 (Analyst) C:\Users\Michael.Brault\OneDrive - IRISNDT\Desktop\Review\Erik\University\SL64-A32_Plates0.5inch. PL-8.odat

A-B-C-S

Gain (dB) 24.60 0.00

A%: 47.5 %
AdBr: -10.54 dB
DA: 0.086 in
PA: 1.325 in
SA: 2.240 in
ViA: 0.575 in
Ur C: 0.309 in
Um C: 0.146 in
Um-r C: 0.163 in
Sr: 2.906 in
Sm: 5.032 in
Sm-r: 2.126 in

Figura 5.21: PAUT for the 63 weld pass.



WeldSight 1.4.29 (Analyst) C:\Users\Michael.Brault\OneDrive - IRISNDT\Desktop\Review\Erik\University\SL64-A32_Plates0.5inch. PL-7.odat

A-B-C-S

Gain (dB) 24.60 0.00

GR-1

75.00

A%: 4.3 %
AdBr: -31.40 dB
DA: 0.125 in
PA: *** in
SA: *** in
ViA: *** in
Ur C: 0.246 in
Um C: 0.391 in
Um-r C: 0.145 in
Sr: -0.020 in
Sm: 9.966 in
Sm-r: 9.986 in

Figura 5.22: PAUT for the 54 weld pass.

Figure 5.20 proves that Picture 5.9 is well labeled because can not be seen any mayor flaws in it. Regarding Figure 5.21 this proves by the red marks and the arrows that picture 5.2 is well labeled in this case as lack of fusion. Finally, Regarding Figure 5.20 his proves by the yellows marks and the arrows that picture 5.6 is well labeled in this case as undercut. The rest of the PAUT can be seen on the annex.

With respect to the time-windows of 0.4 seconds created to train and test the machine learning models, to relate this with the Lab pictures and the PAUT report the ID of the time-window is linked with its time measured from the star of the weld to the end, therefore by using the interpolation of the axial displacement of the torch (like for example this 4.1) is possible to convert time to axial position. The following equation 5.2 is an approximation of the conversion time-space (in inches) where is assumed that the velocity of the torch is constant. Where T_{max}, t_{max}, and t_{min} are the maximum time of entire welding pass, the maximum time of the time windows and the minimal time of the time window respectively.

$$x_{max}, x_{min} = \frac{T_{max}}{10}(t_{max}, t_{min}) \tag{5.2}$$

5.3.1. Summary of the preprocessing.

This subsection finally it is presented the summary of the steps performed to be able to train and run the models.

- Load the current and voltage data, divide the information into temporary windows. A window time of 0.4 seconds and 25 % overlap is selected.
- For each time window obtain the spectrograms and scalograms.
- Normalize the data between (0, 1) for the spectrograms and (-1,1) for the scalograms.
- Split the data into training, validation and test.

5.3.2. Data set for train test

Data set for discontinuities identification

The dataset chosen to train the anomaly detection models corresponds to good welds 58, 59 and 60 with 20 % of data with shuffle equal to True for validation.

With respect to the Test data, the good weld 80 is used, and the poor 63 and 54. Finally, the best model is tested with all the other remaining welds identified in this chapter.

Data set for discontinuities classification

The data set chosen to train the classification models corresponds to good welds 58, 59 and 60 with 20 % of data with shuffle equal to True for validation. With respect to the Test data, the good weld 80 is used, and the poor 63 and 54.

Capítulo 6

Discontinuities detection.

This chapter presents the implementation of the algorithms based on Deep learning of fault identification. For each algorithm, the theory behind it is indicated and its architecture is validated through a theoretical example. Then the algorithms are applied for electrical welding signals. Due to memory problems, it was decided to apply these algorithms only in the current case. In addition, its acquisition is simpler to carry out for this type of algorithms.

6.1. Auto Encoder results

This sections presents the results of discontinuity detection based on Auto Encounter algorithm.

6.1.1. Electrical welding signals implementation

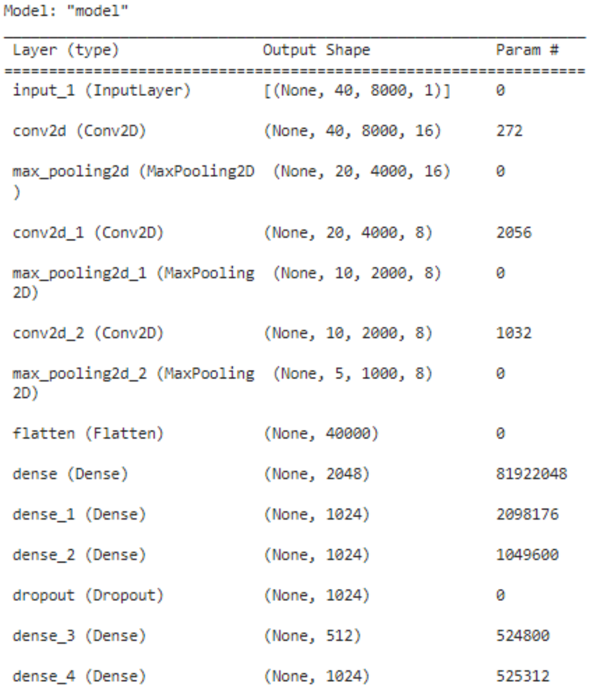
The algorithm seen in the previous background section 1.4 subsection is applied to the case of electrical signals. The architecture was decided base on an artificial problem created with the MNIST data set, this architecture is fixed for the three types of Auto encoders tested to compere it results and performance. The final algorithm ones is selected its hyper parameters are optimized.

The results are presented for each of the two types of calculated images (spectrogram and scalogram).

Welds 58, 59, 60 and 80 were taken, 75 % of them were used for training and the other 25 % were left for validation, the good weld 80 along with welds 54 and 63 that corresponds to the cases of undercut and lack of fusion respectively were used for the test.

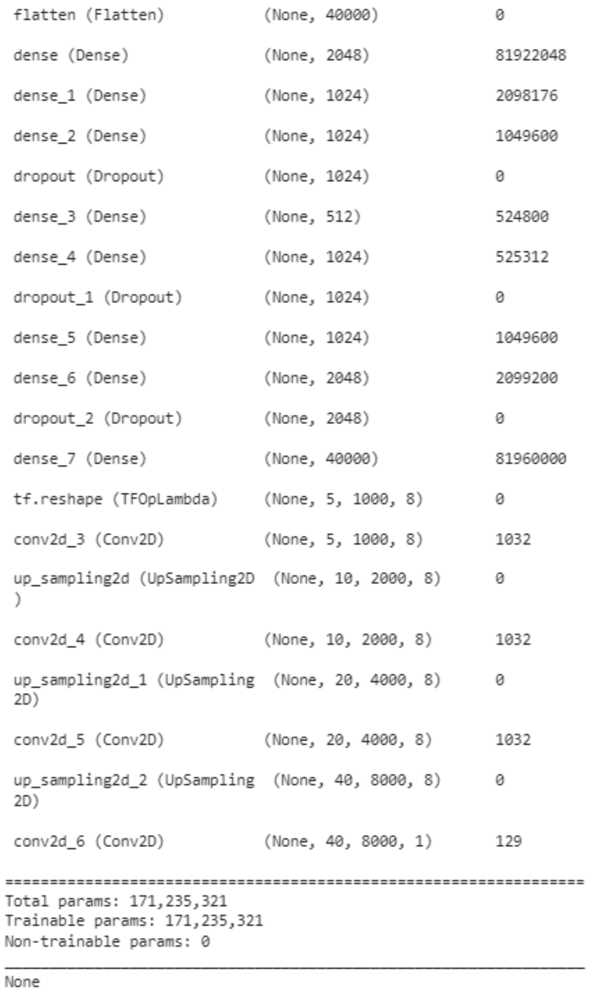
6.1.2. Results scalograms - Auto encoders

The following images 6.1 and 6.2 shows the summary of the model, in total the model has 171,235,321 trainable parameters. The figures 6.3 and 6.4 shows the results, the first one, shows the reconstruction error witch is calculated by using the equation 1.4. The mean of the validation set is used to set the threshold. The final results can be seen in the confusion matrix.



```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 input_1 (InputLayer)        [(None, 40, 8000, 1)]     0         
                                                                 
 conv2d (Conv2D)             (None, 40, 8000, 16)      272       
                                                                 
 max_pooling2d (MaxPooling2D  (None, 20, 4000, 16)     0         
 )                                                               
                                                                 
 conv2d_1 (Conv2D)           (None, 20, 4000, 8)       2056      
                                                                 
 max_pooling2d_1 (MaxPooling  (None, 10, 2000, 8)      0         
 2D)                                                             
                                                                 
 conv2d_2 (Conv2D)           (None, 10, 2000, 8)       1032      
                                                                 
 max_pooling2d_2 (MaxPooling  (None, 5, 1000, 8)       0         
 2D)                                                             
                                                                 
 flatten (Flatten)           (None, 40000)             0         
                                                                 
 dense (Dense)               (None, 2048)              81922048  
                                                                 
 dense_1 (Dense)             (None, 1024)              2098176   
                                                                 
 dense_2 (Dense)             (None, 1024)              1049600   
                                                                 
 dropout (Dropout)           (None, 1024)              0         
                                                                 
 dense_3 (Dense)             (None, 512)               524800    
                                                                 
 dense_4 (Dense)             (None, 1024)              525312    
```

Figura 6.1: Scalogram Auto encoder summary part1.



```
 flatten (Flatten)           (None, 40000)             0

 dense (Dense)               (None, 2048)              81922048

 dense_1 (Dense)             (None, 1024)              2098176

 dense_2 (Dense)             (None, 1024)              1049600

 dropout (Dropout)           (None, 1024)              0

 dense_3 (Dense)             (None, 512)               524800

 dense_4 (Dense)             (None, 1024)              525312

 dropout_1 (Dropout)         (None, 1024)              0

 dense_5 (Dense)             (None, 1024)              1049600

 dense_6 (Dense)             (None, 2048)              2099200

 dropout_2 (Dropout)         (None, 2048)              0

 dense_7 (Dense)             (None, 40000)             81960000

 tf.reshape (TFOpLambda)     (None, 5, 1000, 8)        0

 conv2d_3 (Conv2D)           (None, 5, 1000, 8)        1032

 up_sampling2d (UpSampling2D  (None, 10, 2000, 8)      0
 )

 conv2d_4 (Conv2D)           (None, 10, 2000, 8)       1032

 up_sampling2d_1 (UpSampling  (None, 20, 4000, 8)      0
 2D)

 conv2d_5 (Conv2D)           (None, 20, 4000, 8)       1032

 up_sampling2d_2 (UpSampling  (None, 40, 8000, 8)      0
 2D)

 conv2d_6 (Conv2D)           (None, 40, 8000, 1)       129

=================================================================
Total params: 171,235,321
Trainable params: 171,235,321
Non-trainable params: 0
_________________________________________________________________
None
```

Figura 6.2: Scalogram Auto encoder summary part2.

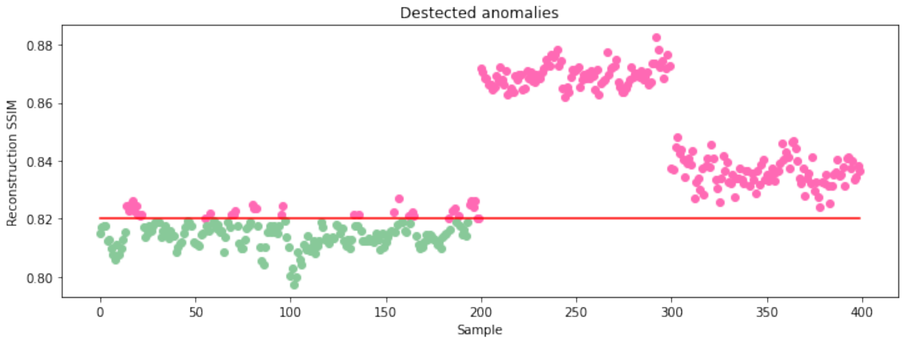


Figura 6.3: Scalogram Auto encoder result.

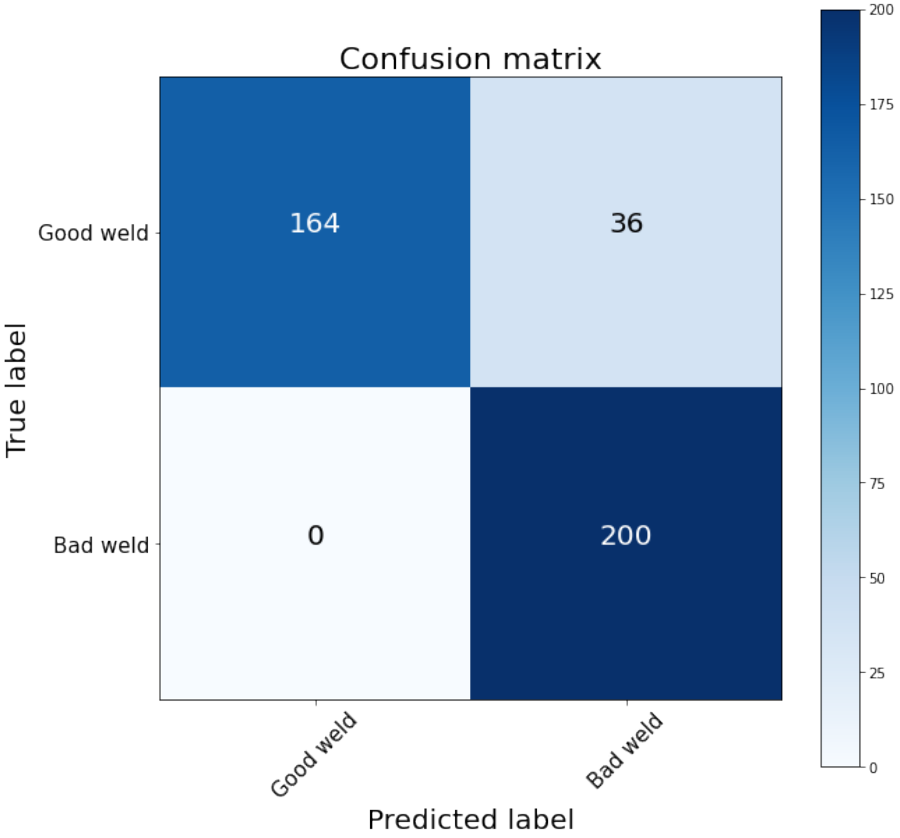
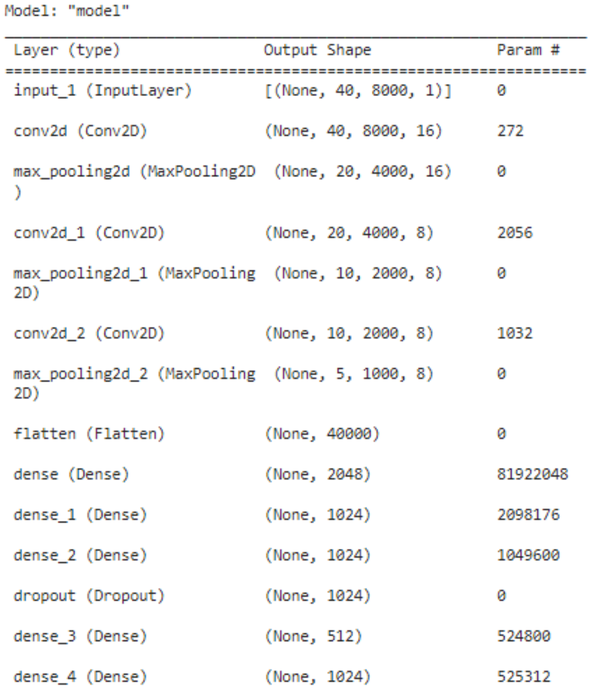


Figura 6.4: Scalogram Auto encoder confusion matrix.

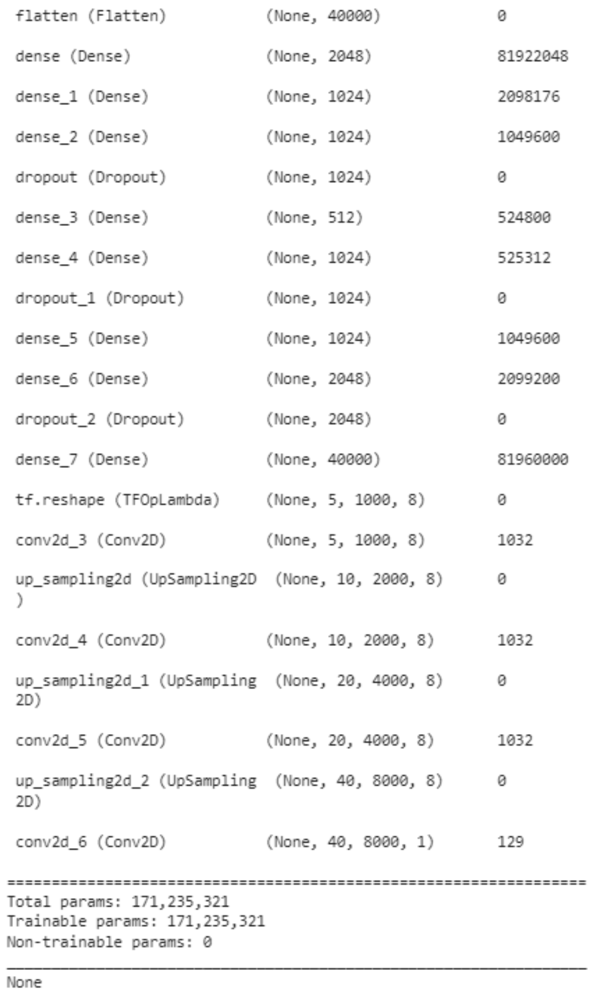
6.1.3. Results spectrograms - Auto encoders

The following images 6.5 and 6.6 shows the summary of the model, in total the model has 171,235,321 trainable parameters. The figures 6.7 and 6.8 shows the results, the first one, shows the reconstruction error and the second and the confusion matrix of the detection.



```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 input_1 (InputLayer)        [(None, 40, 8000, 1)]     0         
                                                                 
 conv2d (Conv2D)             (None, 40, 8000, 16)      272       
                                                                 
 max_pooling2d (MaxPooling2D  (None, 20, 4000, 16)     0         
 )                                                               
                                                                 
 conv2d_1 (Conv2D)           (None, 20, 4000, 8)       2056      
                                                                 
 max_pooling2d_1 (MaxPooling  (None, 10, 2000, 8)      0         
 2D)                                                             
                                                                 
 conv2d_2 (Conv2D)           (None, 10, 2000, 8)       1032      
                                                                 
 max_pooling2d_2 (MaxPooling  (None, 5, 1000, 8)       0         
 2D)                                                             
                                                                 
 flatten (Flatten)           (None, 40000)             0         
                                                                 
 dense (Dense)               (None, 2048)              81922048  
                                                                 
 dense_1 (Dense)             (None, 1024)              2098176   
                                                                 
 dense_2 (Dense)             (None, 1024)              1049600   
                                                                 
 dropout (Dropout)           (None, 1024)              0         
                                                                 
 dense_3 (Dense)             (None, 512)               524800    
                                                                 
 dense_4 (Dense)             (None, 1024)              525312    
```

Figura 6.5: Scalogram Auto encoder summary part1.



```
 flatten (Flatten)           (None, 40000)             0

 dense (Dense)               (None, 2048)              81922048

 dense_1 (Dense)             (None, 1024)              2098176

 dense_2 (Dense)             (None, 1024)              1049600

 dropout (Dropout)           (None, 1024)              0

 dense_3 (Dense)             (None, 512)               524800

 dense_4 (Dense)             (None, 1024)              525312

 dropout_1 (Dropout)         (None, 1024)              0

 dense_5 (Dense)             (None, 1024)              1049600

 dense_6 (Dense)             (None, 2048)              2099200

 dropout_2 (Dropout)         (None, 2048)              0

 dense_7 (Dense)             (None, 40000)             81960000

 tf.reshape (TFOpLambda)     (None, 5, 1000, 8)        0

 conv2d_3 (Conv2D)           (None, 5, 1000, 8)        1032

 up_sampling2d (UpSampling2D  (None, 10, 2000, 8)      0
 )

 conv2d_4 (Conv2D)           (None, 10, 2000, 8)       1032

 up_sampling2d_1 (UpSampling  (None, 20, 4000, 8)      0
 2D)

 conv2d_5 (Conv2D)           (None, 20, 4000, 8)       1032

 up_sampling2d_2 (UpSampling  (None, 40, 8000, 8)      0
 2D)

 conv2d_6 (Conv2D)           (None, 40, 8000, 1)       129

=================================================================
Total params: 171,235,321
Trainable params: 171,235,321
Non-trainable params: 0
_________________________________________________________________
None
```

Figura 6.6: Scalogram Auto encoder summary part2.

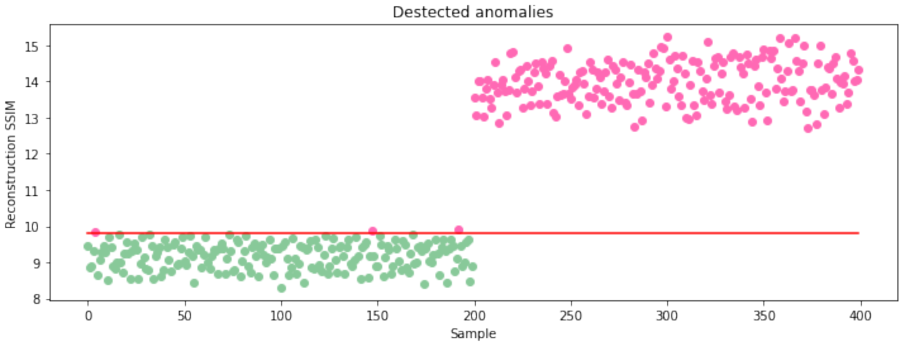


Figura 6.7: Spectrogram Auto encoder result.

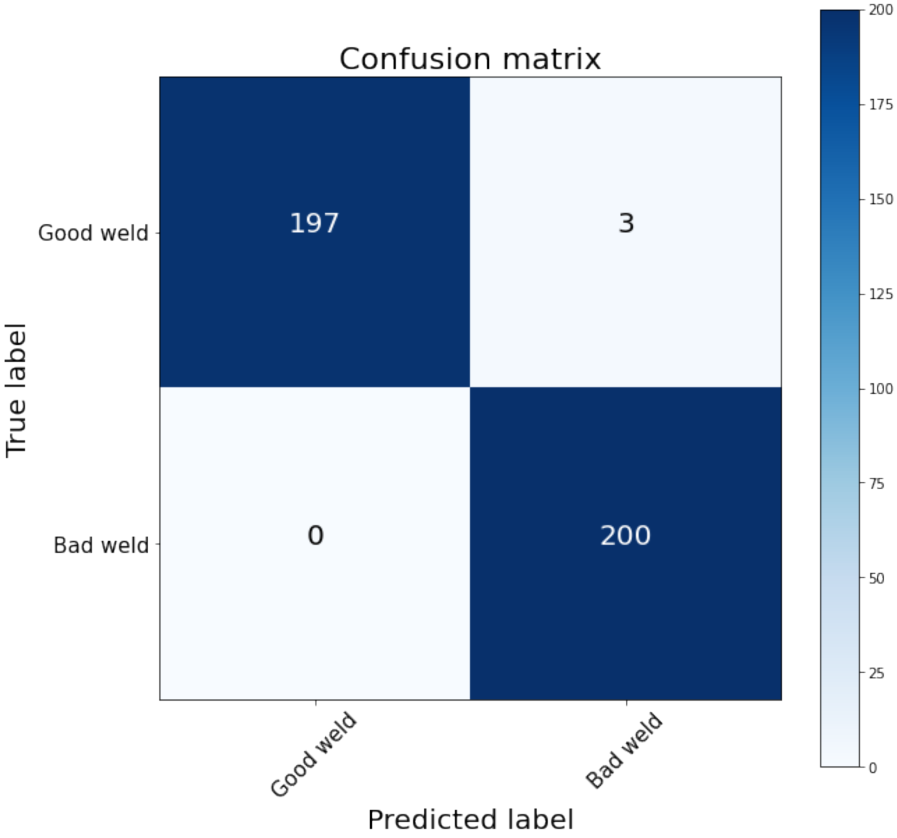


Figura 6.8: Spectrogram Auto encoder confusion matrix.

6.2. Denoising Auto encoders.

To avoid overfitting and improve the robustness of the model, Denoising Auto encoder [13] modification is applied to the previous net shown. This section presents the validation and results of this new implementation.

6.2.1. Results scalograms - Denoising Auto encoders.

Gaussian noise is applied to the images of the scalogram, obtaining a result like the one shown in the Figure 6.9.

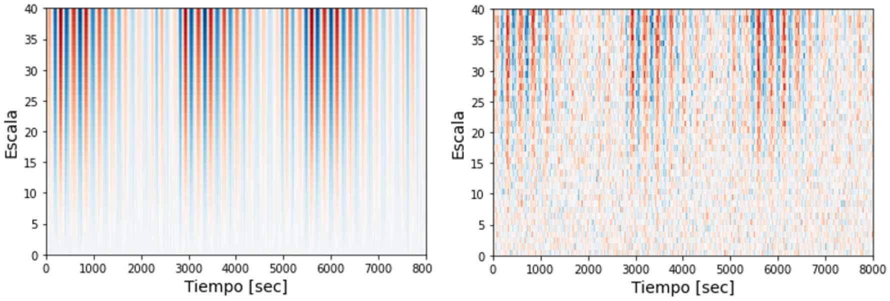


Figura 6.9: Example noisy scalogram for Denoising auto encoder result.

As previously indicated, the architecture has not changed in this new model, so the number of parameters is the same as the previous one. The figures 6.10 and 6.11 shows the results.

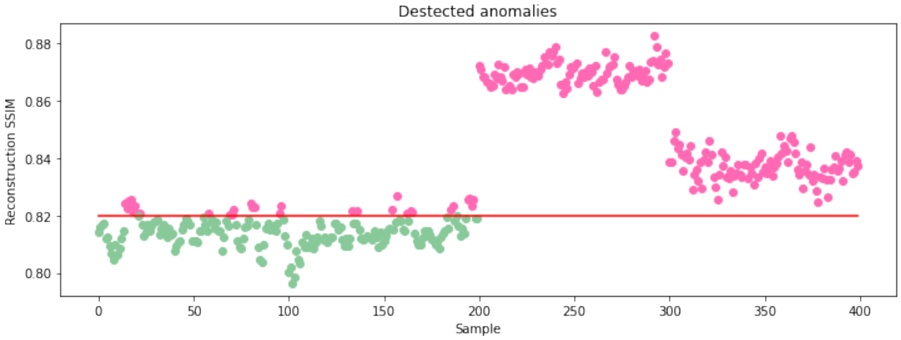


Figura 6.10: Scalogram Denosing Auto encoder result.

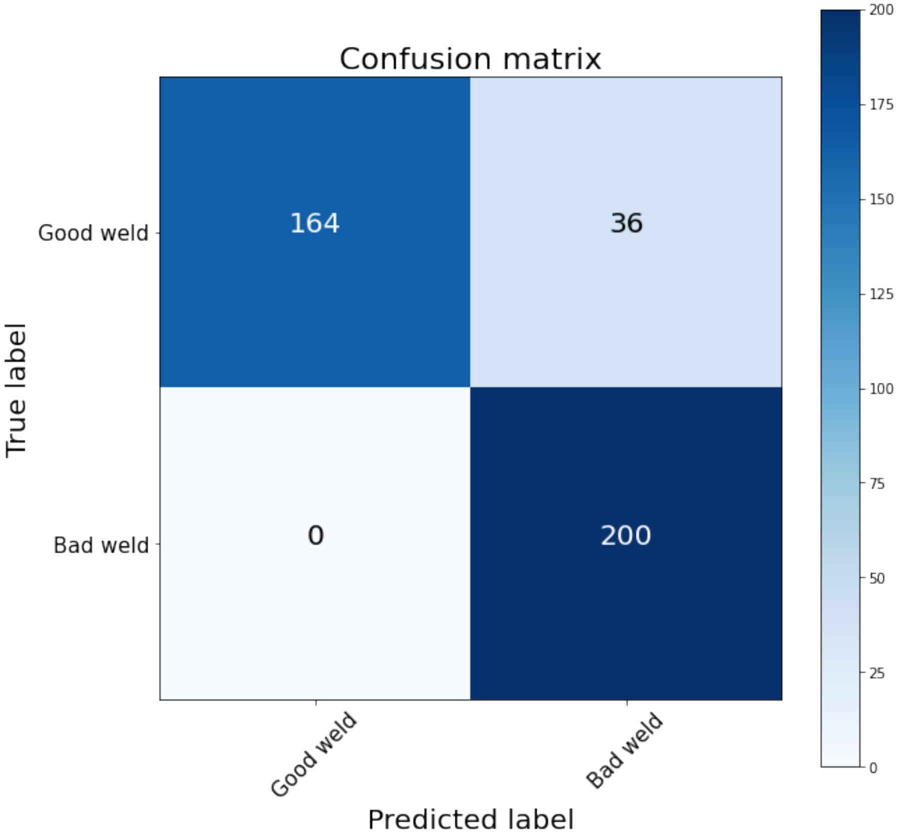


Figura 6.11: Scalogram Denosing auto encoder confusion matrix.

6.2.2. Results spectrograms - Denoising Auto encoders.

Gaussian noise is applied to the images of the spectrograms, obtaining a result like the one shown in the figure 6.12.

As previously indicated, the architecture has not changed in this new model, so the number of parameters is the same as the previous one. The figures 6.13 and 6.14 shows the results.

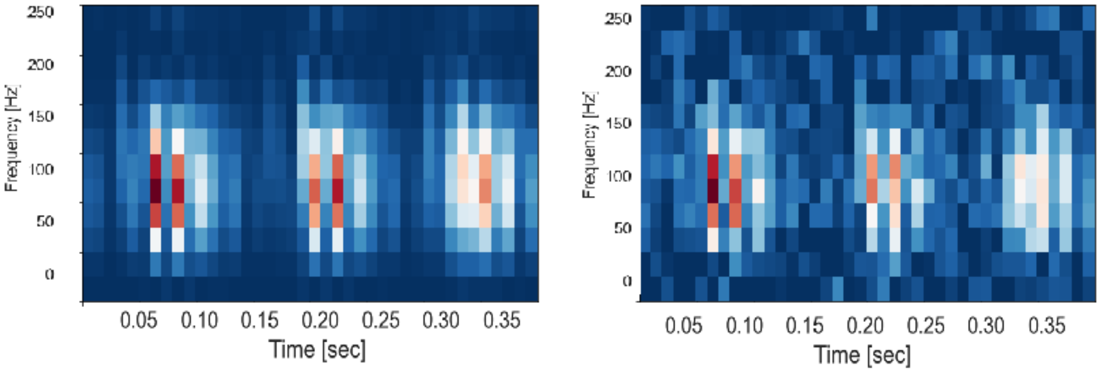


Figura 6.12: Example noisy spectrogram for Denoising auto encoder result.

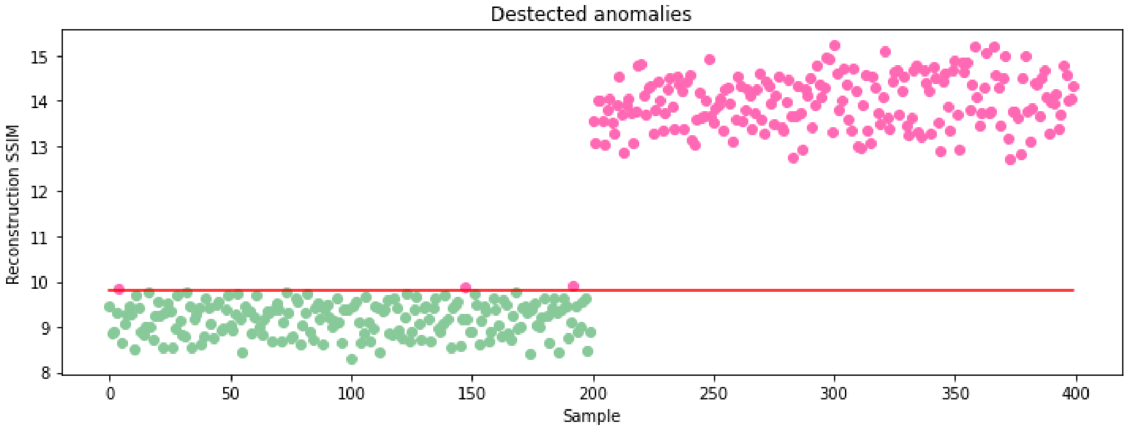


Figura 6.13: Spectrogram - Denosing auto encoder result.

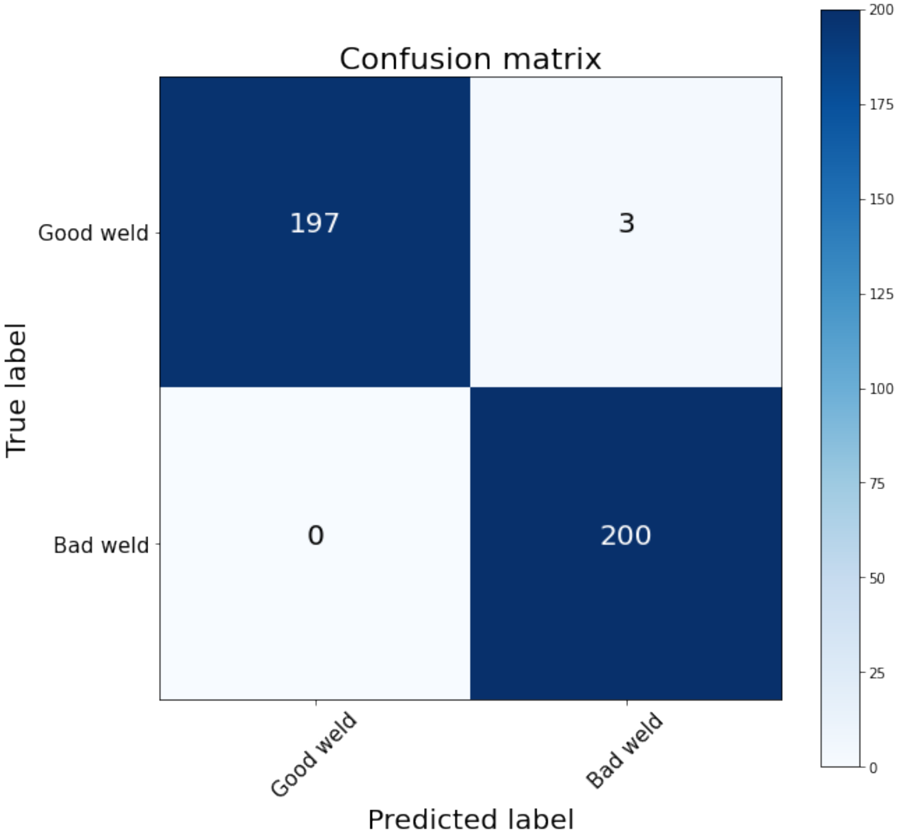


Figura 6.14: Spectrogram Denosing auto encoder confusion matrix.

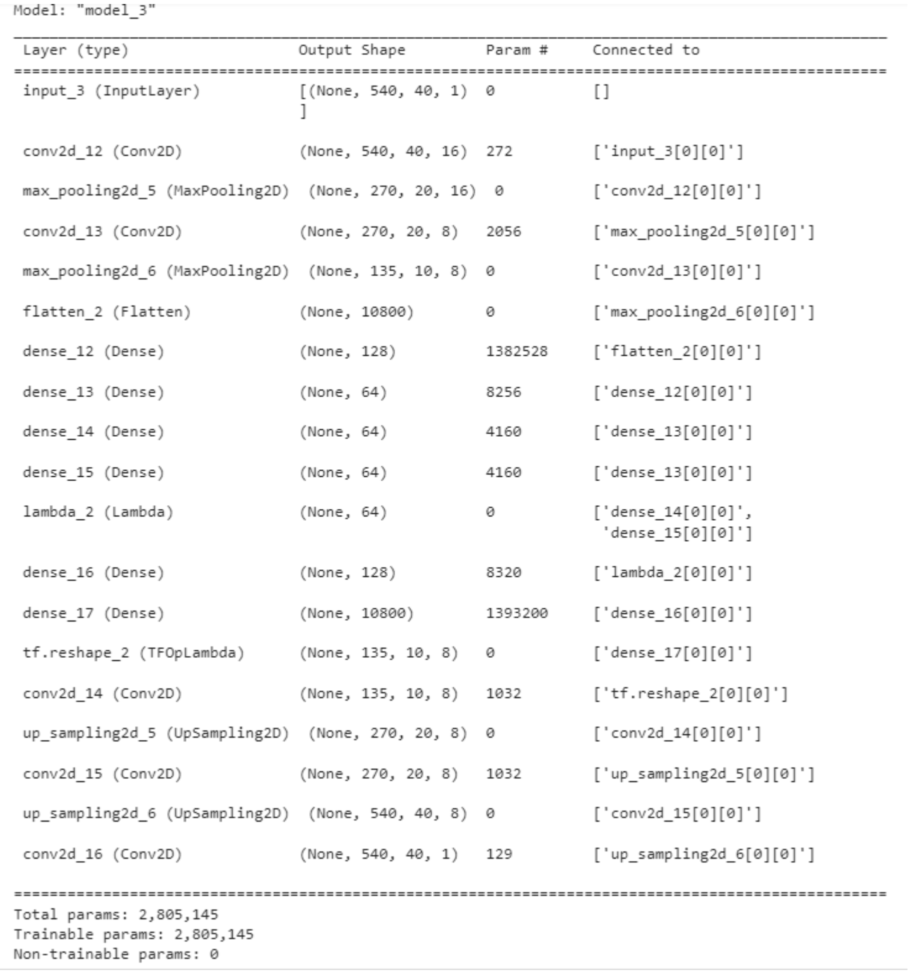
6.3. Varational Auto encoders.

[10], is actually less similar to all the Auto encoder models, but deeply rooted in the methods of variational bayesian and graphical model. Instead of mapping the input into a fixed vector, want to map it into a distribution. This section presents the validation and results of this Varational autoencoder implementation.

6.3.1. Results Varational autoencoder.

When trying to apply the Varational Autoencoder model to the case of scalogram-type data, it was not possible to execute the proposed model due to memory problems.

The following images 6.15 shows the summary of the model for the case of current Spectrograms, in total the model has 2,805,145 parameters to train. The figures 6.16 and 6.17 shows the results.



```
Model: "model_3"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_3 (InputLayer)           [(None, 540, 40, 1)  0           []
                                ]

 conv2d_12 (Conv2D)             (None, 540, 40, 16)  272         ['input_3[0][0]']

 max_pooling2d_5 (MaxPooling2D)  (None, 270, 20, 16)  0          ['conv2d_12[0][0]']

 conv2d_13 (Conv2D)             (None, 270, 20, 8)   2056        ['max_pooling2d_5[0][0]']

 max_pooling2d_6 (MaxPooling2D)  (None, 135, 10, 8)  0           ['conv2d_13[0][0]']

 flatten_2 (Flatten)            (None, 10800)        0           ['max_pooling2d_6[0][0]']

 dense_12 (Dense)               (None, 128)          1382528     ['flatten_2[0][0]']

 dense_13 (Dense)               (None, 64)           8256        ['dense_12[0][0]']

 dense_14 (Dense)               (None, 64)           4160        ['dense_13[0][0]']

 dense_15 (Dense)               (None, 64)           4160        ['dense_13[0][0]']

 lambda_2 (Lambda)              (None, 64)           0           ['dense_14[0][0]',
                                                                  'dense_15[0][0]']

 dense_16 (Dense)               (None, 128)          8320        ['lambda_2[0][0]']

 dense_17 (Dense)               (None, 10800)        1393200     ['dense_16[0][0]']

 tf.reshape_2 (TFOpLambda)      (None, 135, 10, 8)   0           ['dense_17[0][0]']

 conv2d_14 (Conv2D)             (None, 135, 10, 8)   1032        ['tf.reshape_2[0][0]']

 up_sampling2d_5 (UpSampling2D)  (None, 270, 20, 8)  0           ['conv2d_14[0][0]']

 conv2d_15 (Conv2D)             (None, 270, 20, 8)   1032        ['up_sampling2d_5[0][0]']

 up_sampling2d_6 (UpSampling2D)  (None, 540, 40, 8)  0           ['conv2d_15[0][0]']

 conv2d_16 (Conv2D)             (None, 540, 40, 1)   129         ['up_sampling2d_6[0][0]']

==================================================================================================
Total params: 2,805,145
Trainable params: 2,805,145
Non-trainable params: 0
```

Figura 6.15: Spectrograms Varational auto encoder summary part1.

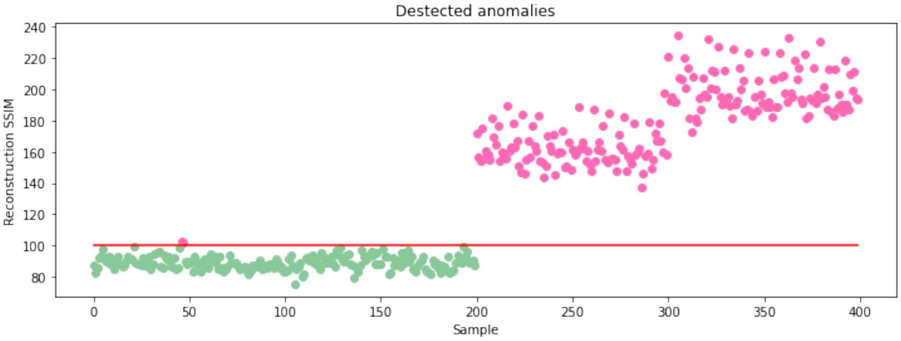


Figura 6.16: Spectrogram Varational auto encoder result.

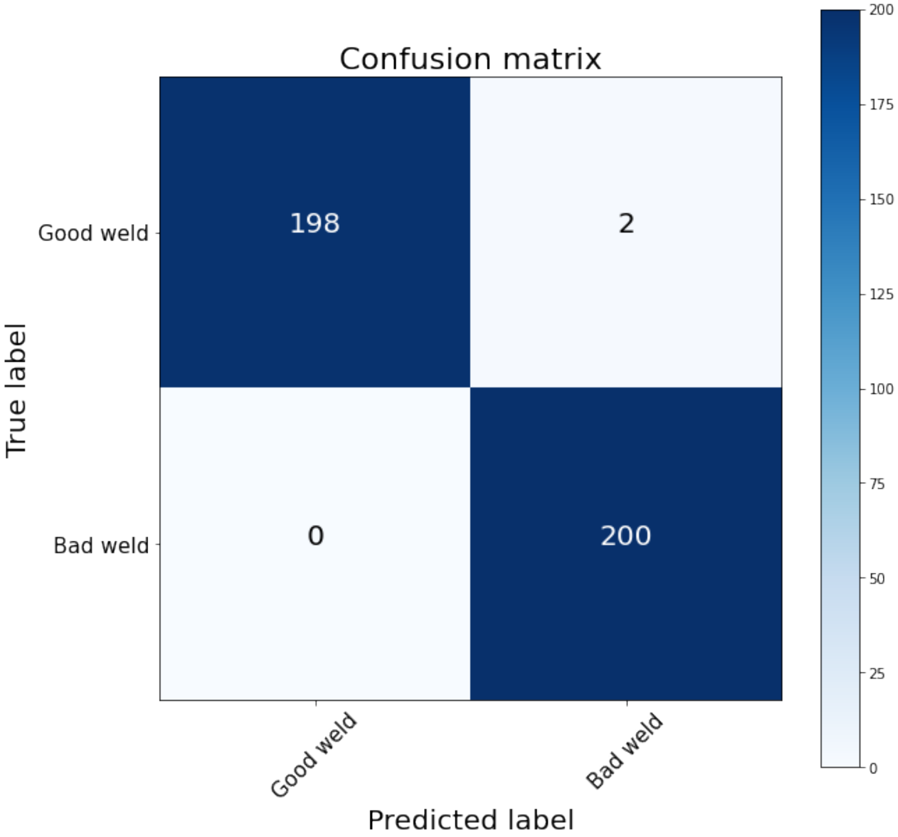


Figura 6.17: Spectrogram Varational auto encoder confusion matrix.

6.4. Explore the state of the art of the industry.

Currently the industry uses statistical exploration for conditional analysis of the electrical signal. Normally expert criteria are used to distinguish between good and bad welds, in the case of this study to imitate this effect simple model of decision trees is used. Welds 80, 63, and 54 are considered to compare with the results of the Auto encoder.

Therefore, 200 good data and 80 bad data, 40 lack of fusion data and 40 undercut data are taken for the Isolation Forest algorithm implementation.

The statistical parameters used was, mean, variance, max value, min value, RMS, peak-peak, crest, kurtosis, skew, this nine parameters were taken for the signal, the derived signal and integrated signal, in total 24 parameters.

The results can be seen in the following image 6.18.

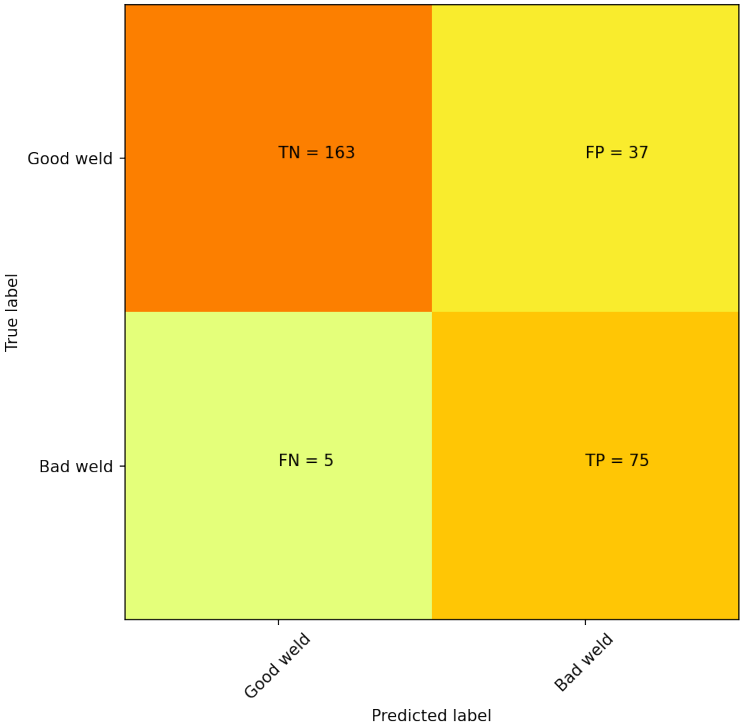


Figura 6.18: Isolation forest experiment confusion matrix.

The results shows that with this type of method it is possible to identify bad welds, but it is not reliable enough to incorporate into a production process. It is even possible to argue serious deficiencies in its scalability due to the urgent need to have a technician specialized in discovering patterns in electrical signals to performs the fine tuning of the code-based algorithm for each new case, that is, for each new technique, equipment , material and welder.

6.5. Discussion and comparison of the implemented methods.

After analyzing the methods implemented throughout this chapter, can be observed that is possible to effectively apply deep learning techniques in detection of discontinuities in welds.

It can be seen Figure 6.3 and 6.4 how the normal Auto encoder is capable of detect lack of fusion and undercut from the scalograms images, also in the case of spectrogram.

The main difference is that scalograms are to heavy representation of the data, that can be seen in 6.5 in comparison to where the scalograms have 171,235,32 parameters to be trained but spectrogram.

In the case of Denoising and specially Varational auto encoder it was not possible to optimize the code to run for the training of the scalograms.

Between these three methods based on Spectrograms-auto encoder, both Denoising and Varational presenting great potential for its application. Is possible to note that these last two are able to satisfactorily generalize the data of good welds when training with strings 58 and 60 and testing it with string 80 for the images 6.13, 6.14, 6.16 and 6.17. Regarding to the actual detection of bad welds, these methods are capable of detect lack of fusion and undercut.

After comparing the two best proposed modes with respect to the simulation of the state of the art of the industry 6.18, it is possible to notice that the Deep learning methodology has advantages, mostly:

- They are easier to scale.
- They are easier to fine tuning for a non-specialized person.
- They are more accurate in their detection of discontinuities.

Regarding its disadvantages, these lie in the need to use more data to train.

If is compared both denosing and varational, although denosing have better metris than varational 6.14 and 6.17, varational presents a better behaviour with respect of the reconstruction error, because as it can be seen from the Figure 6.16 the good weld (80 string) have less variability in comparison of the bad welds (63 and 56 strings) 6.13. Also, varational have the potential to create confidential interval due it probabilistic interpretation. Therefore, it is used this algorithm in the cumming section, final implementation. A summary table is presented for comparison of the algorithms 6.2.

Tabla 6.1: Summary of the discontinuity detection models.

Algorithm	Pros	Contros	F1 Score	Total time for training
Desicition tree	Is the easiest algorithm to implement, and also is easy to interpret.	Doesn't have good enough performance to be implemented	0.794	30 secods
Normal Auto encoder	Is the easiest deep learning algorithm	Is too easy to Overfit and its latent space is not reliable.	0.995	5 Minutes
Denosing Auto encoder	Is an improvement from he NormalAuto Encoder	The latent space is better but the dispercion between good welds is still to	0.995	5 minutes
Varational Auto encoder	Is the best algorithm in terms of performance, and latent space behaviour.	Is mathematicaly much difficult to implement	0.999	8 minutes

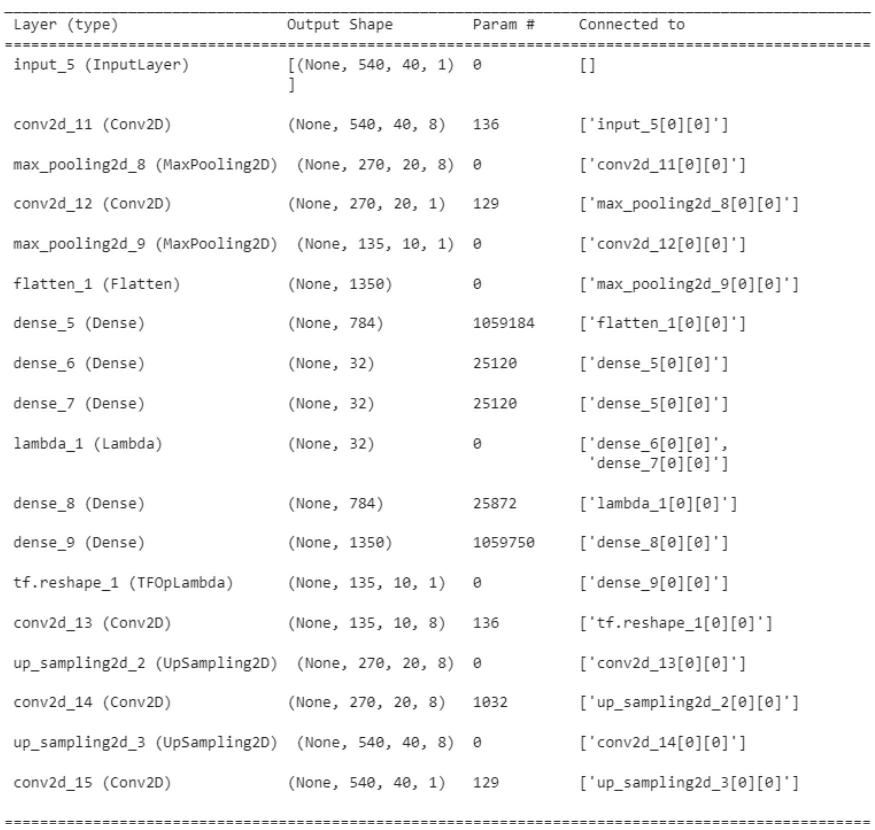
Tabla 6.2: Summary of optimized Number of layers.

Number of layers	Accuracy Mean	Accuracy Standard deviation
2 layers, starting with 16 Conv filters and decreasing by a power of two	0.992	0.04
1 layer of 16 Conv filters	0.975	0.09
3 layers, starting with 16 Conv filters and decreasing by a power of two	0.947	0.09
2 layers, starting with 8 Conv filters and decreasing by a power of two	0.950	0.05

6.6. Final implementation.

The implemented algorithm selected for this section correspond to Spectrogram-Varational auto encoder. The hyper parameters of the net were selected base on sensitivity analysis 6.2.

The representation of the final algorithm can be seen in following images 6.19 and 6.20:



```
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_5 (InputLayer)           [(None, 540, 40, 1)  0           []
                                ]

 conv2d_11 (Conv2D)             (None, 540, 40, 8)   136         ['input_5[0][0]']

 max_pooling2d_8 (MaxPooling2D)  (None, 270, 20, 8)  0           ['conv2d_11[0][0]']

 conv2d_12 (Conv2D)             (None, 270, 20, 1)   129         ['max_pooling2d_8[0][0]']

 max_pooling2d_9 (MaxPooling2D)  (None, 135, 10, 1)  0           ['conv2d_12[0][0]']

 flatten_1 (Flatten)            (None, 1350)         0           ['max_pooling2d_9[0][0]']

 dense_5 (Dense)                (None, 784)          1059184     ['flatten_1[0][0]']

 dense_6 (Dense)                (None, 32)           25120       ['dense_5[0][0]']

 dense_7 (Dense)                (None, 32)           25120       ['dense_5[0][0]']

 lambda_1 (Lambda)              (None, 32)           0           ['dense_6[0][0]',
                                                                  'dense_7[0][0]']

 dense_8 (Dense)                (None, 784)          25872       ['lambda_1[0][0]']

 dense_9 (Dense)                (None, 1350)         1059750     ['dense_8[0][0]']

 tf.reshape_1 (TFOpLambda)      (None, 135, 10, 1)   0           ['dense_9[0][0]']

 conv2d_13 (Conv2D)             (None, 135, 10, 8)   136         ['tf.reshape_1[0][0]']

 up_sampling2d_2 (UpSampling2D)  (None, 270, 20, 8)  0           ['conv2d_13[0][0]']

 conv2d_14 (Conv2D)             (None, 270, 20, 8)   1032        ['up_sampling2d_2[0][0]']

 up_sampling2d_3 (UpSampling2D)  (None, 540, 40, 8)  0           ['conv2d_14[0][0]']

 conv2d_15 (Conv2D)             (None, 540, 40, 1)   129         ['up_sampling2d_3[0][0]']

==================================================================================================
```

Figura 6.19: Spectrograms Varational auto encoder summary final.

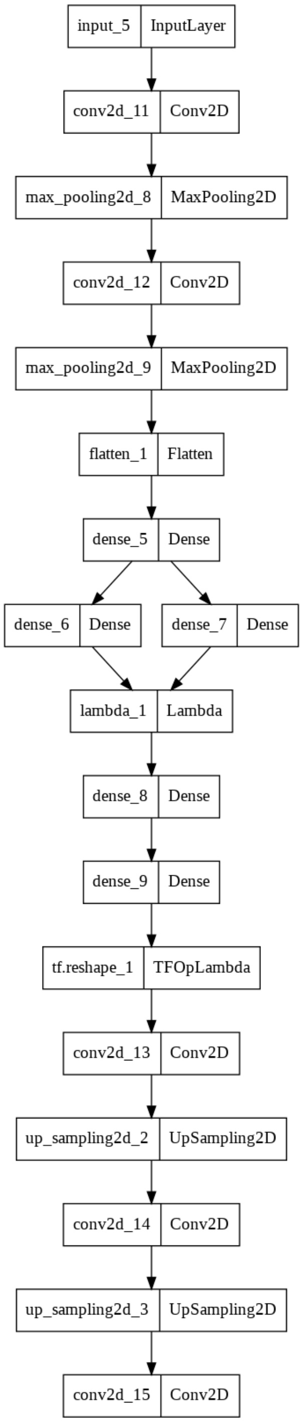


Figura 6.20: Spectrogram Varational auto encoder graph representation final.

Is decided to train the model with a total of 120 epochs to avoid overfitting. The following figure 6.21 shows the loss curves during training where it can be clearly seen that the model is not overfitted.

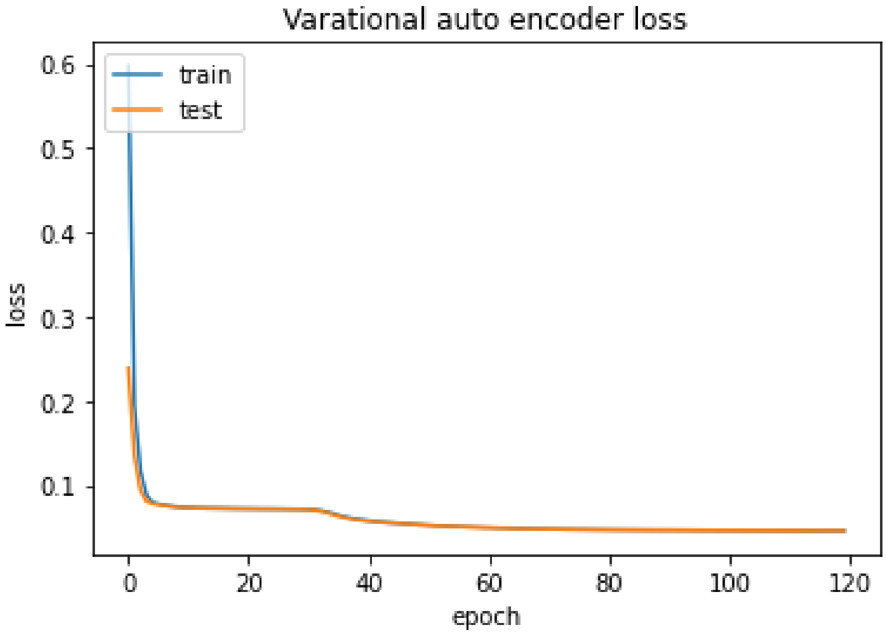


Figura 6.21: Spectrogram Varational auto encoder loss curve.

6.6.1. Confidence interval

A confidence interval is calculated for the measurements, for which the model is applied 100 times for each of the data, obtaining a mean and a standard deviation for each point.

Then the probability that each point is below the threshold is calculated, if this probability is above a percentage selected by the user then it will be considered as a good weld.

For these calculations, the normalized normal distribution $Z \sim N(0,1)$ is used, which is calculated by using the equation 6.1. The probability that this Z distribution is equal to or less than our objective-selected probability p^* is calculated by equation 6.2, In this way, including a measure of uncertainty around the forecast.

$$z = \frac{threshold - \mu_x}{\sigma_x} \tag{6.1}$$

$$P(z \leq threshold) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{\frac{t^2}{2}} dt = \begin{cases} \text{Good weld} & \text{w.p.} \geq p^* \\ \text{Bad weld} & \text{w.p.} < p^* \end{cases} \tag{6.2}$$

An example of this implementation is shown in the following images for a probability of 90, 95 and 99 respectively 6.22, 6.23 and 6.24.

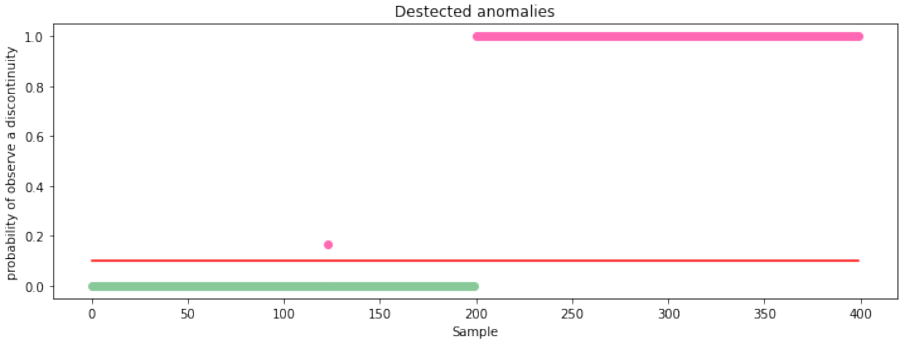


Figura 6.22: Probability curve for 90 % of confidence.

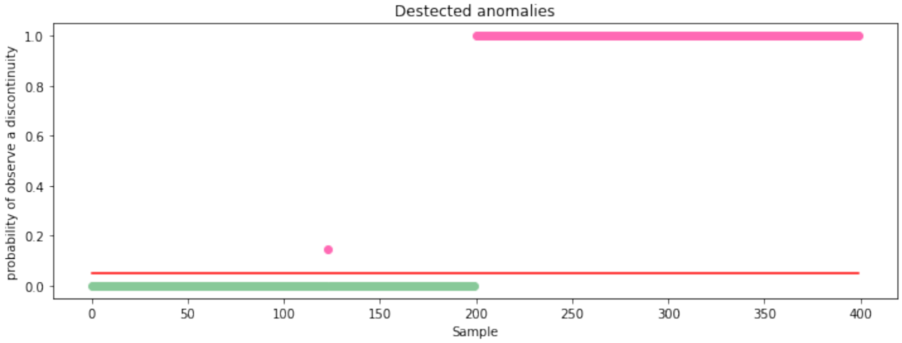


Figura 6.23: Probability curve for 95 % of confidence.

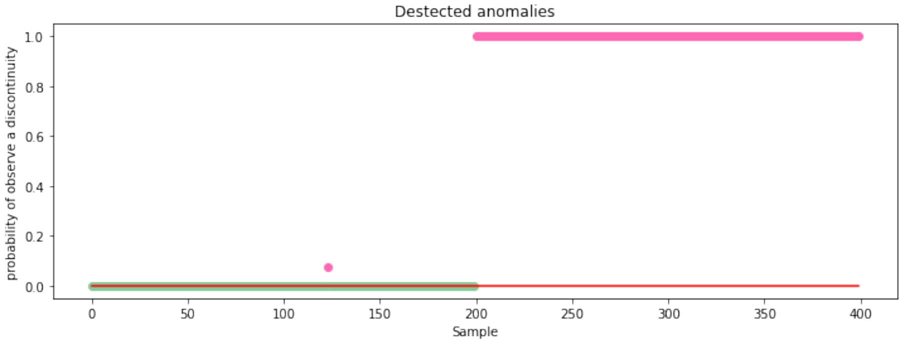


Figura 6.24: Probability curve for 99 % of confidence.

Finally, is decided that, the best way to display the information about the weld is to generate a curve with the interval of confidence, this is implemented by adding 10 points into a new one by using the following equations 6.3 and 6.4 of sum of probabilistic variables.

$$\mu_{sum} = \sum \alpha_i \mu_i \tag{6.3}$$

$$\sigma^2_{sum} = \sum \alpha_i^2 \sigma_i^2 \tag{6.4}$$

The next sub sections shows the result for interval of confidence of 95 percent for all the cases proposed for this study, good, lack of fusion and undercut.

6.6.2. Optimized Results 80 - 57 - 63 - 56 (Good)

The following curves shows the results for good weld Test, in order to be able to see the different, the first 100 samples correspond to the 80 weld (good), the next 100 to the 57 with is also good and finally 63 and 56 are lack of fusion and undercut respectively.

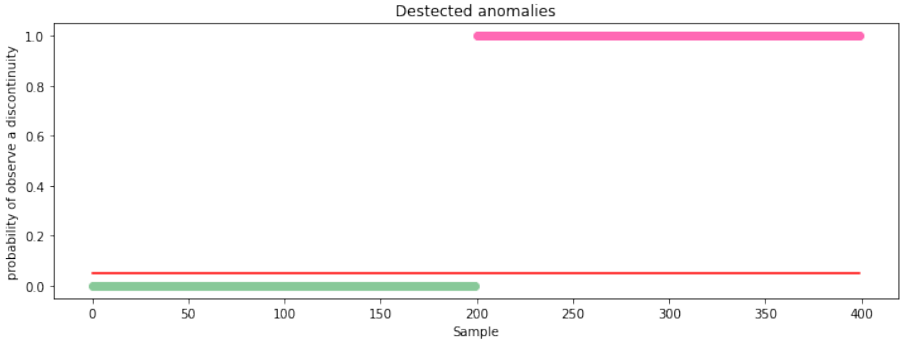


Figura 6.25: Probability curve for 95 % of confidence.

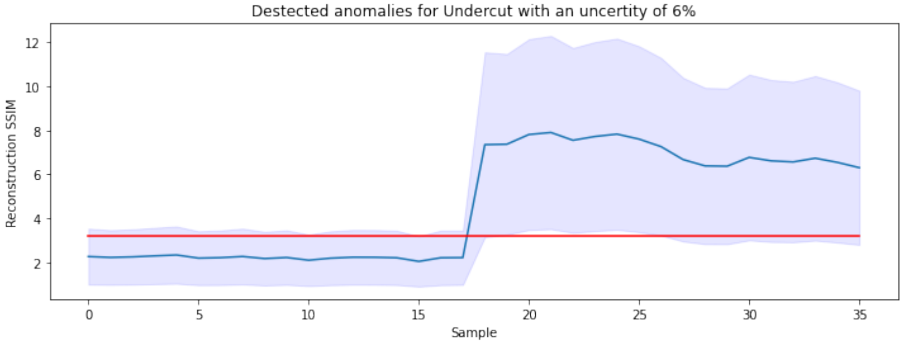


Figura 6.26: Interval of confidence curve.

6.6.3. Optimized Results 80 - 63 - 62 - 65 (Lack of fusion)

The following curves shows the results for lack of fusion weld Test, in order to be able to see the different, the first 100 samples correspond to the 80 weld (good), the next correspond to 63, 62 and 65 witch are lack of fusion welds.

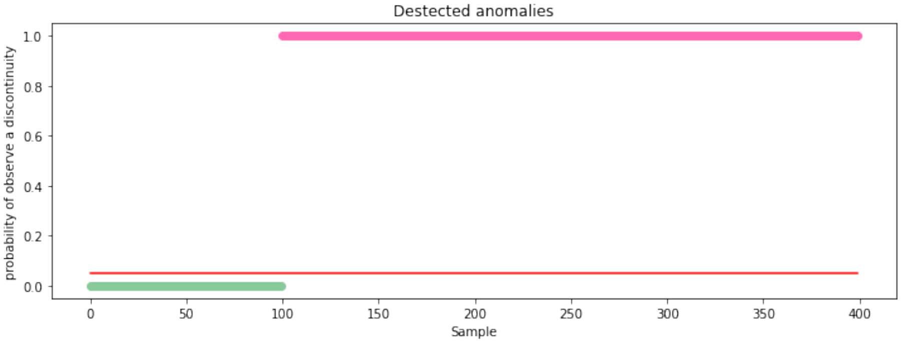


Figura 6.27: Probability curve for 95 % of confidence of observe a discontinuity.

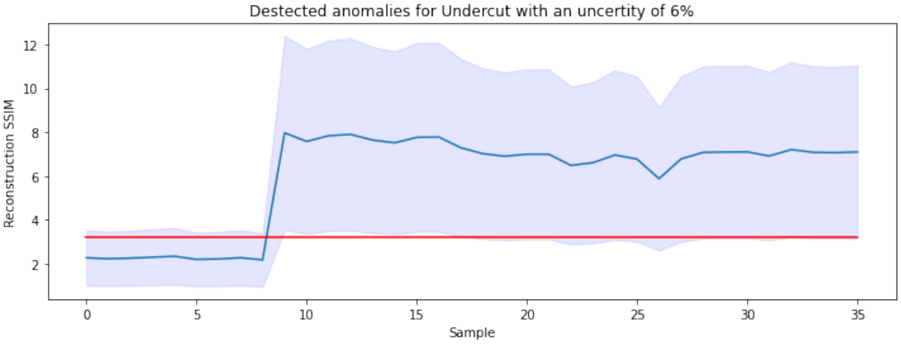


Figura 6.28: Interval of confidence curve.

6.6.4. Optimized 80 - 53 - 54 - 56 (Undercut)

The following curves shows the results for lack of fusion weld Test, in order to be able to see the different, the first 100 samples correspond to the 80 weld (good), the next correspond to 53, 54 and 56 witch are undercut.

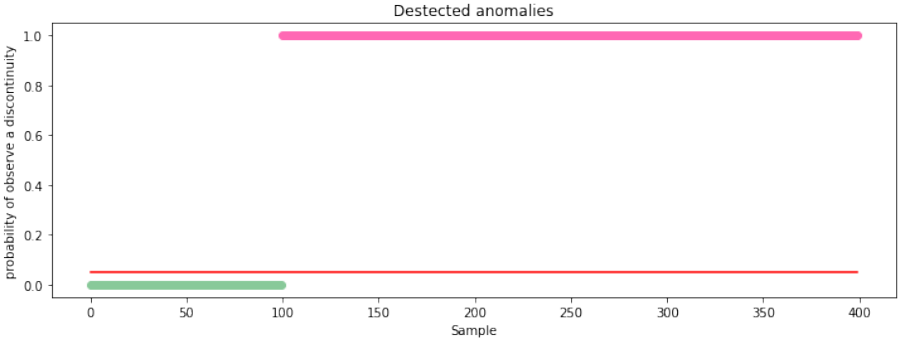


Figura 6.29: Probability curve for 95 % of confidence.

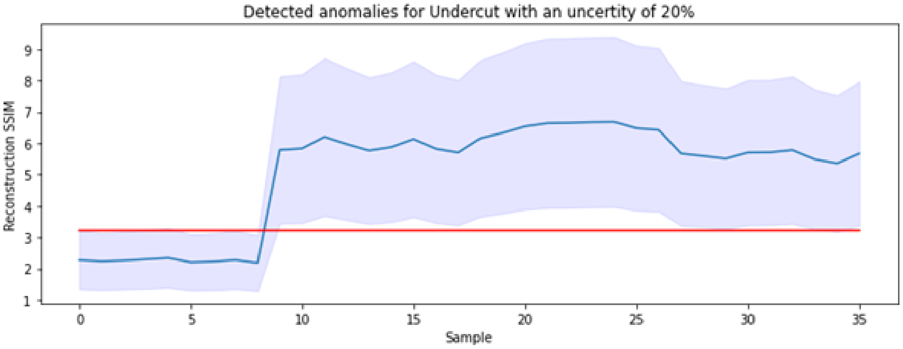


Figura 6.30: Interval of confidence curve.

6.7. Final discussion anomaly detection.

This section shows summary of the models tested during this chapter, as well as a final discussion of the Spectrogram-VAE algorithm implementation. The previous Table 6.2 shows the summary with the pro and cons for each algorithm tested during this chapter, the time of training execution are based on GPU-NVIDIA Tesla K80 hardware.

Regarding the proposed algorithm, Varational auto enconder, from Figure 6.21 can be seen that algorithms is well trained with no appearance over fitting, as mentioned before, from Figure 6.16 can be seen that the dispersion between good and bad welds are significantly different thanks the Normal distributed latent space for good welds.

Regarding the confidence intervals and the provability of observe a discontinuity, thanks to the stochastic latent space of Varational Autoencoder and the central limit theorem [8] Both this probability and the confidence interval can be realized effectively. By analyzing 6.25, 6.26, 6.28, and 6.28 can be observed that to separate good welds and bad welds with lack of fusion is possible with an estimated 6 % of uncertainty. With respect to the undercut case, the estimated uncertainty raised to 20 % witch is expected because this type of failure is even difficult to see in the PAUT report.

Capítulo 7

Discontinuities classification.

This chapter presents the implementation of the algorithm base on Deep learning of fault classification.

7.1. Dense neural network implementation and optimization

This section presents the implementation of Dense Neural Network model for welding discontinuities classification. The proposed model consists in using the Autoencoder encoders layers to compress the current spectrogram 2d images in to small vectors, then this vectors are passed to a Dense Neural Network to classify the type of weld (Good, undercut or lack of fusion).

The data used for the training process correspond to 58, 59, 60 for good welds, 53, 54 for undercut weld, and 62, 63 for lack of fusion weld. Regarding the Loss function and the metric selected for the model, categorical crossentropy and accuracy respectively were used.

The following Figures shows the results for the first model tested. 7.1 and 7.2 shows the Loss function and the Accuracy respectively. The Figure 7.3 shows the confusion matrix of the Test results. The performance metrics for this model were the following: Accuracy: 0.937, Precision: 0.942, Recall: 0.937, and F1score: 0.937.

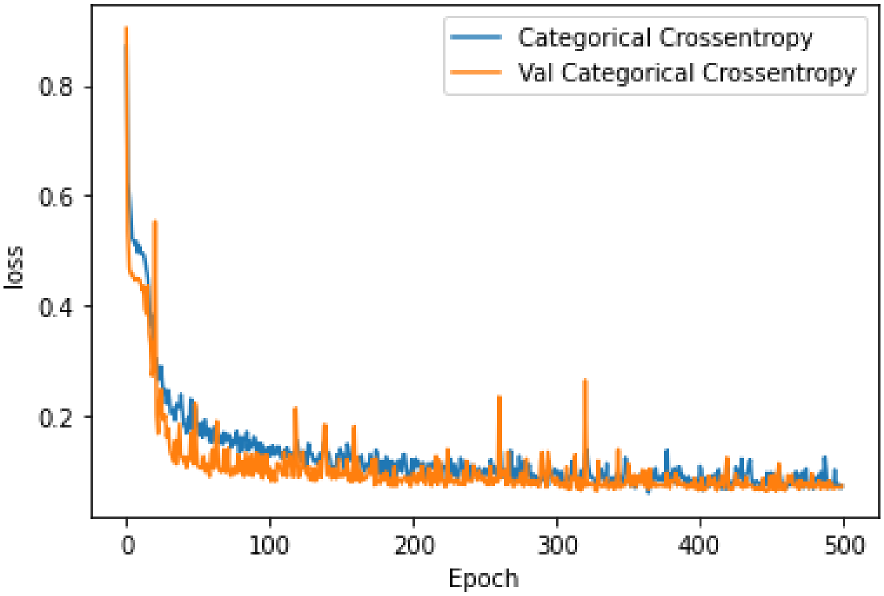


Figura 7.1: Classification Model Loss curve.

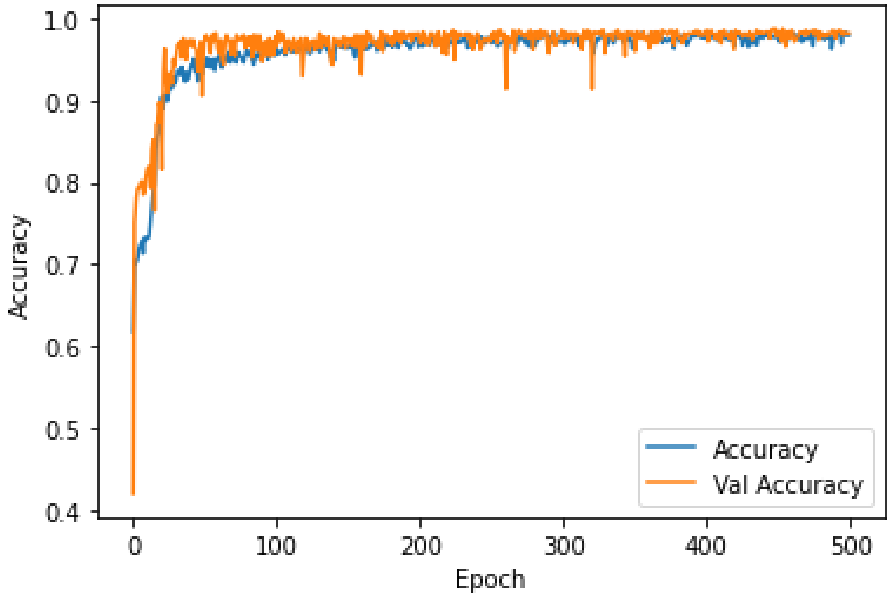


Figura 7.2: Classification Model Accuracy curve.

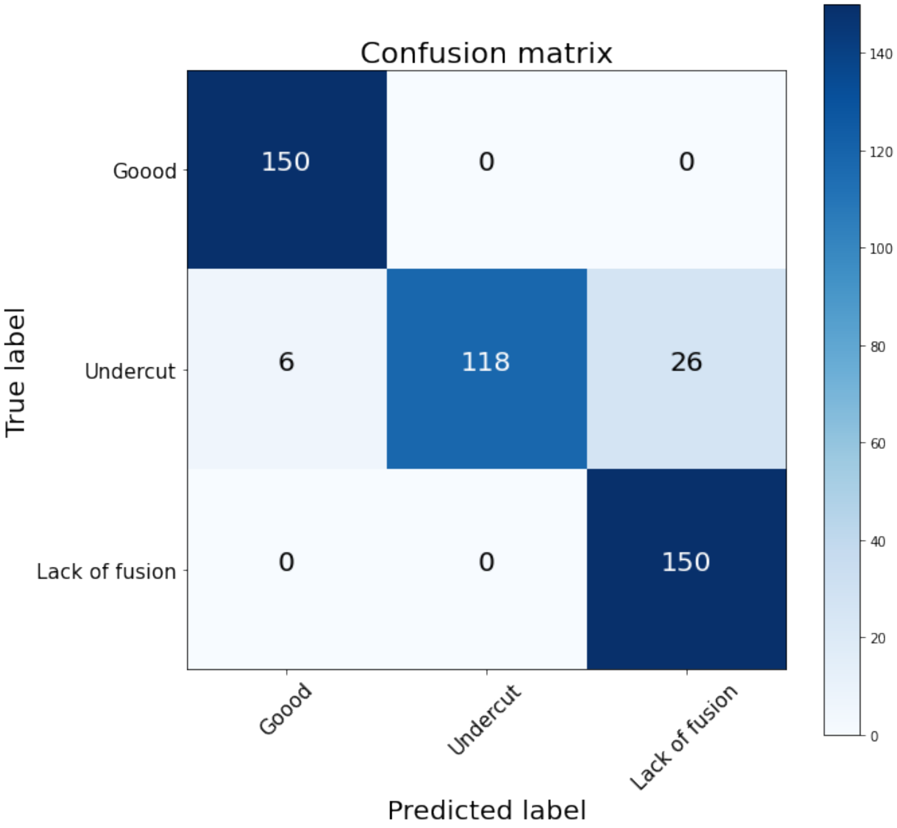


Figura 7.3: Classification Model confusion matrix.

Finally a sensitivity analysis is performed with the model to fine-tuning the hyper parameters, the parameters selected were number of layers, learning rate, optimizer and number of epochs. The next Table summarized the results.

Tabla 7.1: Sensitivity analysis of the proposed network with respect to the learning rate

Learning rate	Accuracy
0.01	0.944
0.001	0.966
0.0001	0.955

Tabla 7.2: Sensitivity analysis of the proposed network with respect to the number of epochs

Number of epochs	Accuracy
200	0.940
500	0.966
700	0.964

Tabla 7.3: Sensitivity analysis of the proposed network with respect to the optimizer

Optimizer	Accuracy
RMSprop	0.940
Adam	0.966
Nadam	0.960

Tabla 7.4: Sensitivity analysis of the proposed network with respect to the number of layers

Number of layers	Accuracy
5 layers, starting with 512 Neurons and decreasing by power of two.	0.937
4 layers, starting with 256 Neurons and decreasing by power of two.	0.966
5 layers, starting with 128 Neurons and decreasing by power of two.	0.962

Tabla 7.5: Sensitivity analysis of the proposed network with respect to the Drop out rate

Drop out rate	Accuracy
0.1	0.971
0.25	0.966
0.5	0.628

7.2. Discussion

As can be seen from the previous section, implementing a sub network that used the decoder of the previous chapter to classify the discontinuities type is possible.

The model is well trained and without signs of over fitting as can be seen from the training curves 7.1 and 7.2.

On the other hand, the result shown in the confusion matrix 7.3 shows that is possible to successfully classify the three states studied in this thesis. Separate the good state from the bad states is relatively simpler than in the other hand, separate the lack of fusion and undercut type where the network gets a bit more confused between.

Regarding the sensitivity analysis displayed in tables 7.1, 7.2,7.3,7.4,and 7.5, to identify the best set of hyper parameters to train the classifier network, the following correspond to the best options:

- Learning rate: 0.1
- Number of epochs: 500
- Optimizer: Adam
- Number of hidden layers: 4
- Drop out rate: 0.1

Conclusión

7.3. Lab work

According to the results presented throughout Chapter 4, the proposed laboratory work has been carried out successfully. The type of weld selected is usually used in civil construction, and the procedure for welding detailed in the annexes was successfully followed. Hence, the results of this laboratory work directly emulate a simplified version of industrial welding. It was possible to intentionally produce welding discontinuities along a stream while capturing current and voltage data. These discontinuities were validated by the non-destructive PAUT test; therefore, the lab work can be considered a success. Likewise, the electrical signals show different patterns for the nominal case (or good welding) and the bad case, in which there are discontinuities.

Improving the data collection system by using a simpler equipment based on Arduino or Raspberry is proposed. Regarding the discontinuities studied, it was only possible to successfully produce lack of fusion and undercut. Consequently, the experiment should be reframed to properly study lack of penetration.

7.4. Identification of discontinuities

Regarding the use of deep learning to identify welding flaws, these algorithms have shown excellent performance, surpassing classical algorithms. 2D representations of electrical signals present great potential for the study of welding reliability, as some previous studies have suggested. Within these 2D representations, the spectrograms and scalograms are the most popular, although with the latter, it is possible to distinguish good welds from bad ones more easily through naked eye, they have poor interpretability and are comparatively heavier than spectrograms. Hence, the scalograms implementation is more complex. On the other hand, although at first glance it is difficult to discern good from bad welds through spectrograms, their implementation through computational vision techniques based on convolutional networks presents an excellent performance.

Regarding the deep learning algorithms implemented, the normal autoencoder presents overfitting tendencies which can be solved by using a denoising autoencoder. However, even these two algorithms present latent vectors that are not robust. By introducing the Bayesian inference and forcing the latent vector to distribute normally, it is possible to produce a robust latent vector where the good welds are tightly clustered and the bad welds more disperse. The F1-score of the proposed model is 99,5 % in the training set, which makes it suitable for implementation in the industry.

Comparing these results with the literature, it is possible to highlight that in this thesis it has been possible to predict very relevant discontinuities for the industry that have hardly been studied in previous works, even more so, considering the type of welding used for this thesis, which until now This date has not had a greater representation in the literature despite being one of the most relevant in construction.

Thanks to the usage of Bayesian inference presented in variational autoencoder, it was possible to successfully use the theorem of central limit to construct confidence intervals and measure the uncertainty in the predictions. These uncertainty measurements accurately predicted that undercut had more uncertainty with respect to lack of fusion (20 % vs 6 %), which is a result that was confirmed in the PAUT report.

The Spectrogram variational autoencoder discontinuity detection algorithm has the potential to simplify and optimize inspection processes, which could lead to a reduction of both time and costs.

The implementation of this algorithm in an industrial environment is proposed to test its potential in helping the inspection process. Finally, since the algorithms were implemented with the current signal due to its easier capturing process, new implementations of this study should focus mainly on the current signal.

7.5. Classification of discontinuities

With regarding the classification of discontinuities, this study provided one of the first algorithms capable of classify good welds from undercut and lack of fusion welds which is an improvement from previous studies that have mainly focused on porosity. Also, its implementation by using the latent space from the Variational Autoencoder is new in the subject and have the potential to be improved. The optimized algorithm had a good performance with a 96 % of accuracy which is an excellent complement from the previous algorithm because they can be executed in cascade in a industrial environment.

Finally, is proposed to work in the future with more complex algorithms and increase the number of both data and discontinuities to classify.

Bibliografía

[1] Sergio Manuel Ignacio Cofre Martel. *A PHYSICS-INFORMED NEURAL NETWORK FRAMEWORK FOR BIG MACHINERY DATA IN PROGNOSTICS AND HEALTH MANAGEMENT FOR COMPLEX ENGINEERING SYSTEMS*. PhD thesis, 2022.

[2] James F. Lincoln Arc Welding Foundation. *The procedure handbook of arc welding*. James F. Lincoln Arc Welding Foundation, 2000.

[3] P Beatriz Garcia-Allende, Jesus Mirapeix, Olga M Conde, Adolfo Cobo, and Jose M Lopez-Higuera. Defect detection in arc-welding processes by means of the line-to-continuum method and feature selection. *Sensors*, 9(10):7753–7770, 2009.

[4] Aurélien Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, 2019.

[5] Ian D Harris. *Plasma arc cutting of bridge steels*, volume 384. Transportation Research Board, 1997.

[6] Wenhui Hou, Ye Wei, Yi Jin, and Changan Zhu. Deep features based on a dcnn model for classifying imbalanced weld flaw types. *Measurement*, 131:482–489, 2019.

[7] Yong Huang, Dongqing Yang, Kehong Wang, Lei Wang, and Qi Zhou. Stability analysis of gmaw based on multi-scale entropy and genetic optimized support vector machine. *Measurement*, 151:107282, 2020.

[8] Oliver Johnson. *Information theory and the central limit theorem*. World Scientific, 2004.

[9] James M Joyce. Kullback-leibler divergence. In *International encyclopedia of statistical science*, pages 720–722. Springer, 2011.

[10] Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. *Advances in neural information processing systems*, 27, 2014.

[11] R Madigan. Arc sensing for defects in constant-voltage gas metal arc welding. *Weld J*, 78:322S–328S, 1999.

[12] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La

Jolla Inst for Cognitive Science, 1985.

[13] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008.

[14] Yaowen Wang and Pengsheng Zhao. Plasma-arc welding sound signature for on-line quality control. *ISIJ international*, 41(2):164–167, 2001.

[15] Lilian Weng. From autoencoder to beta-vae. *lilianweng.github.io*, 2018.

[16] Q Zhou, Y Cheng, J-J Yang, M-F Zhao, L Zhang, X-C Zhang, Z-H Chen, H-H Yan, Y Song, J-H Chen, et al. Pemetrexed versus gefitinib as a second-line treatment in advanced nonsquamous nonsmall-cell lung cancer patients harboring wild-type egfr (ctong0806): a multicenter randomized trial. *Annals of oncology*, 25(12):2385–2391, 2014.