Detection of production defects using Machine Learning based Image Classification Algorithms

Pedro Miguel Pinto da Cunha Fernandes

Masters Final Dissertation

Advisor in Procter & Gamble: Eng. Riccardo Dessi
Advisor in FEUP: Prof. Marco Parente

Mechanical Engineering Master's Degree

September 2020

This page was intentionally left blank

"No one can construct for you the bridge upon which precisely you must cross the stream of life, no one but you yourself alone."- Friedrich Nietzsche

This page was intentionally left blank

Abstract

Recently we have seen a fast advancement of processors, computing capabilities and the growing accessibility of technologies that allow for fast data manipulation paired with the growing movement of Internet of Things and the digitization of equipment even in the large-scale manufacturing fields. These factors have allowed to leverage the amount of data and processing power to produce solutions, that are less capital intensive or more capable than traditional methods.

This dissertation developed with Procter & Gamble's Power Oral care division focuses on the development of a machine vision system based on deep learning architectures with the objective of classifying soldering defects in a production environment.

Initially, a state-of-the-art revision is made, the soldering process as a joining process in electronics is analysed as well as the defects that are connected to it. Artificial Intelligence as a technology is introduced as well as the subsects of machine learning and deep learning when relating to image analysis.

Then, this work follows the procedures, train of thought and decision making done during the development process of the image acquisition system, deep learning model, proof of concept vision system and finally the specifications of the system for final implementation.

This page was intentionally left blank

Special Thanks

First, I would like to thank my supervisor Professor Marco Parente for all the support and guidance during the elaboration of this dissertation.

I would like to thank Procter & Gamble for having me during this 6-month period specially to Eng. Riccardo Dessi for the opportunity on working in this highly innovative topic and for his unwavering availability even during chaotic times.

I would like to thank my colleagues with whom I have shared this road to becoming an engineer and without whom I would not have been able to do it.

To all my friends that have followed my path and, in some ways, contributed to it. In particular the ones from Corpo Nacional de Escutas as I believe my experiences there lead me into going into engineering.

To Marta, my companion, for being my best example of work ethic and intellect. For always being there for me, never letting me quit and always pushing me to be greater.

A huge thanks to my older sister Inês whom was always a great example on resourcefulness and determination, thank you for always wanting what is best for me.

Finally to my mother Filomena, words will never be enough to thank you for all that you have given me and showing me that everything is possible if we put our mind to it, no matter the challenge.

Pedro Miguel Pinto da Cunha Fernandes

This page was intentionally left blank

Index

Nomenclature

Abbreviations

PCB – Printed Circuit Board

SMT – Surface Mount Technology

AI – Artificial Intelligence

ML – Machine Learning

ANN – Artificial Neural Networks

CNN – Convolutional Neural Networks

List of Figures

List of Tables

1. Introduction

1.1 Motivation

Recently we have seen a fast advancement of processors, computing capabilities and the growing accessibility of technologies that allow for fast data manipulation paired with the growing movement of Internet of Things and the digitization of equipment even in the large-scale manufacturing fields. These factors have allowed to leverage the amount of data and processing power to produce solutions, that are less capital intensive or more capable than traditional methods.

This master's thesis was elaborated in the scope of the 5th year of the Integrated Master's in Mechanical Engineering, Production, Conception, and Manufacturing specialization of the Faculty of Engineering of the University of Porto. This dissertation regards the development and design of a Deep Learning-based solder defects classification vision system.

1.2 Procter & Gamble

Procter & Gamble has a big portfolio of electrical devices, being these batteries or AC-powered. Soldering plays a huge part when it comes to the desired behaviour of these products. More specifically when discussing Power Oral Care devices, i.e. electrical toothbrushes, that are highly commercialized in the North American and European markets and as such, they must fulfil the quality regulations imposed by the regulatory agencies. For example, when it comes to the United States the Food & Drug Administration labels electronic toothbrushes as class 1 medical devices. This classification enforces a set of quality control regulations when it comes to most steps of production. In the Oral-B IO production line a manual 100% inspection of the soldering quality in implemented. This thesis was then motivated by the desire and constant push for the automation and digitization of processes.

Figure 2 Oral-B iO Product

Figure 1 Procter & Gamble Logo

1.3 Thesis outline

This dissertation is structured in six different chapters. The first one is responsible for the introduction.

In the first half of the second chapter an overview of the soldering process is given as well as an establishment of the most common defects associated with it when it comes to the production of electronic devices and the related quality control according to IPC standards. In the second half the over branching field of Artificial Intelligence is defined. In this chapter some of the areas of machine learning are established in particular the ones that will be paramount for the development work, Artificial Neural Networks and Deep learning.

Chapter 3 presents the steps and decision-making process during the development procedure culminating in a trained deep learning model a developed dataset and an image acquisition system.

In chapter 4 the models are validated, as such an edge computing solution is used for inline deployment of the machine learning model and the results of validation are presented.

Having validated the models, chapter 5 follows the development of a solution for final implementation and presents a solution.

Finally, chapter 6 goes over the fulfillment of the thesis' objectives and provides a conclusion and future works.

2. State of Art

To better understand the span of this dissertation, the first step is to review what the process of soldering entails, specifically when relating to PCB components. After this we will go over the state of the art of machine vision systems for quality inspection and finally give background on machine learning models.

2.1 Soldering

Multiple times in production or industry context it is highly attractive to develop a joint that is structurally resistant and leak tight while avoiding the melting of the base material. For this as mentioned before we could use the non-fusion welding processes that mostly rely on pressure to pack tightly the atoms of the joining materials and achieve a connection. However, these processes have multiple limitations, when it comes to material type, joint geometry, and environment conditions. Adhesive bonding on the other hand fills some of these requirements, having only issues with much less resistance to peeling operations and being more sensible to environmental degradation.

2.1.1 Soldering process

In comparison to brazing the only distinction relies on the working temperature. Specifically, the working temperature of the process, given that braze filters melt at temperatures above 450° C as per convention, while soldering filters melt at lower temperatures. However, the main mechanism for filler dispersion is capillary action in both cases. We distinguish soldering from adhesive bonding which is similar in the use of a liquid filler, yet in the case of adhesives the material origin is usually organic in comparison to brazes or solders that are almost always metallic.

With regards to joint strength due to the mechanical properties of the solder materials as well as a lower adhesion that comes from the solder joint, a brazed joint provides greater adhesion values in most cases. The characteristics of the bond between the solder and base material relates to the surface quality of the latter, as the greater surface roughness prompts greater adhesion, leading to a mechanical interlocking effect. In fact, the solder acts as a solvent in the initial stages of the process, making the capability of this solvent to wet the base material intrinsically related to solder joint quality, being defined as “solderability” of a material. This characteristic depends on multiple parameters: the generation of oxides on the face coat, high thermal conductivity that requires a high working temperature for the process, or the necessity of a controlled atmosphere to prevent the formation of compounds, that could lead to the lowering of solder joint mechanical properties [1].

Apart from the use as a straight adhesive-like joint, soldering is the standard process for the creation of connections between metallic components, that must be permeable to electric current. With the rising presence of electronic devices in our day to day lives, soldering has been a major component of these

manufacturing challenges. It allows for the connection of electric wires to components and the placement and fixing of electronic components on Printed Circuit Boards (PCBs).

2.1.2 Soldering in PCBs

As with all manufacturing processes time, varying global conditions, contaminated source materials and out of tolerance parts are all factors that can contribute to the production of parts that are out of specifications after being soldered. Specifically, when the soldering process is considered in the assembly line of electronic devices.

There are two main methodologies of mounting electronic components in PCBs. Firstly, through-hole technology in this situation the component's leads are placed in holes drilled in the PCB. Secondly surface mount configuration where components are positioned directly on the PCB and fixed relying solely on the solder joint [2].

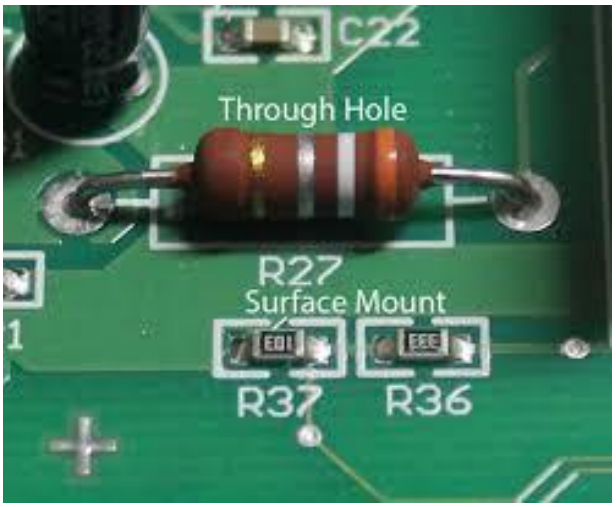


Figure 3 Example of Through-Hole Resistor [3]

Through hole components are subdivided in to two types in relation to the positioning of the leads, axial and radial. A comparison is shown in the image below, the top capacitor contains axially mounted leads and the bottom axially mounted. These configurations are used in different applications. The axial leads

configuration allows the component to stand parallel to the board and as such having less of a height footprint when compared to radial components however occupying a larger area on the board [2].

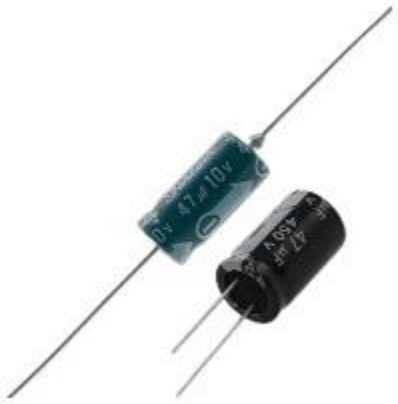


Figure 4 Radial vs parallel leads components

Overall, through hole assemblies are less common compared to the cheaper Surface mount alternative for the following reasons:

- The assembly process is more convoluted as it requires the drilling of holes on the PCB surface.
- The overall component footprint is larger.
- Soldering requires the use of wave soldering or automated hand soldering, compared to the higher repeatability process of reflow soldering used with SMT components.

Even so, Through-hole components offer a greater bond during assembly as the solder joint is not the only factor sustaining the load. As such, they are used in applications where a resistance to mechanical stress is favourable i.e. in the role of connectors [3].

This distinction is relevant as the kind of soldering defects will vary with regards to the type of assembling process.

2.1.3 Production quality control requirements

The production of electronic components in industrialized products are subject to IPC standards. During this sub chapter the IPC-A-160 standard will be analysed and the criteria that must be followed for the soldering joints to be deemed as accepted for a final product will be identified [4].

In this standard soldering joints are marked with a class value, either target, acceptable or defect and within this value the classification can range from 1 to 3 in degrading rates of quality and acceptability. A general standard is defined, regardless of component type or assembly method. The guidelines are as follows for a target classification:

- Solder fillet appears smooth and good wetting is achieved.
- Outline of the lead is easily determined.
- Solder at the part being joined creates a feathered edge.
- Fillet is a concave shape.

These requirements ensure the mechanical reliability of the soldering joint as well as the electrical contact functionality [4].

2.1.4 Soldering anomalies

The main soldering anomalies as per the IPC standard will now be analysed and these will play a determining role in the development process of this project. The focus will be on the anomalies relevant for the work to be done in development. They will be defined by the type of component and assembly method they appear most commonly. Although they are anomalies their presence is classified in acceptable or defect in the same way the overall soldering joint is as there is come marginal allowance for different types of anomalies.[4]

When it comes to a press fit pin for example, a common assembly process for through hole components, the standard for target class 1,2 or 3 requires a 360º soldering fillet around the pin-PCB interface as shown in the diagram. When it comes to the acceptable class 3 outcome a minimal coverage of 330º degrees of the interface is required. Anything apart from this is considered a defect [4].

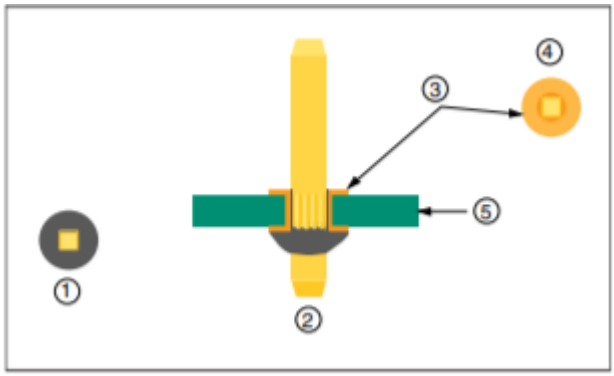


Figure 5 Push-fit pin soldering diagram [4].

2.1.4.1 Exposed base metal

As the name implies this anomaly pertains to when the metal part of the soldering connection appears to be showing and is not sufficiently covered by the solder. Although this can be intentional in design in some applications this is not the general case. Acceptable situations are when this exposed base metal is

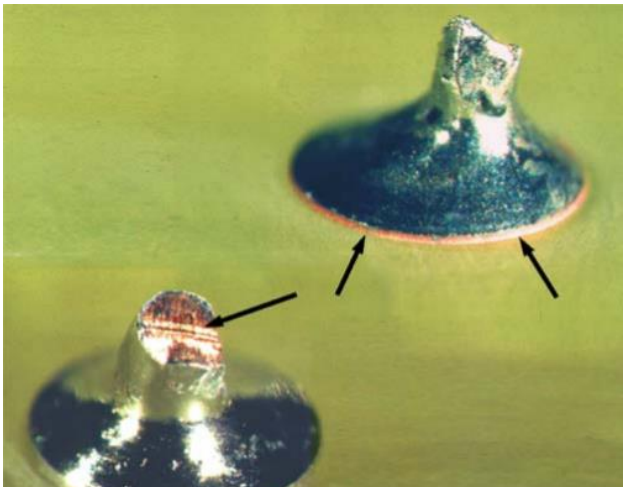


Figure 6 Acceptable exposed base metal [4]

present in vertical edges, i.e. on through hole components the length of connector that is exposed after the connection, or in areas of the connector that are not required for the fillet area of the joint [4].

However, they will be considered a defect when exposed base metal is present in components connectors or wires in the area of the solder joint. This can be caused by insufficient wetting or by erosion due to the assembly process. This defect can be problematic not only for undermining the connectivity of

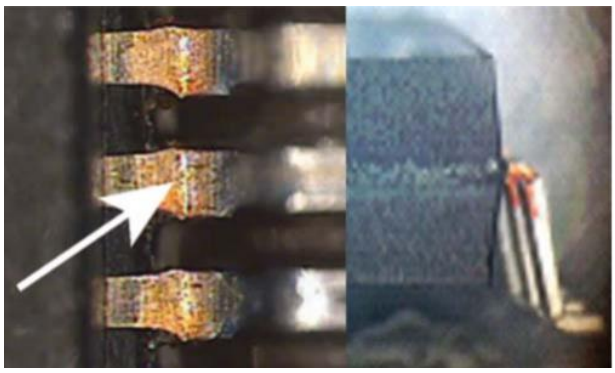


Figure 7 Exposed base metal defect [4].

the device and it's correct operation but for the disruption of the mechanical connection, for example of SMT components that can lead to being dislodge and creating other issues and damages on the product [4].

2.1.4.2 Nonwetting

Nonwetting can be one of the causes of exposed base metal and as such is related to this anomaly. It is defined as the inability of the solder to form a bond with the base metal. Similar to the previous defect these joints are mechanically weaker than the target. Can be caused by temperature issues either when heating the connector or pad or the solder, by the contamination of the area to be soldered for example with a layer of metal oxide, or by the insufficient presence of flux material. In the figure below it is possible to see three of the five pins are insufficiently wet [5].

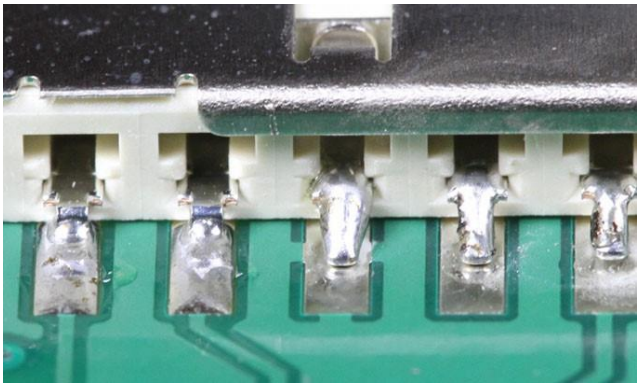


Figure 8 Nonwetting on connector pads[4]

2.1.4.3 Cold Joint

This anomaly is defined by IPC as "A solder connection that exhibits poor wetting, and that is characterized by a greyish, porous appearance. (This is due to excessive impurities in the solder, inadequate cleaning prior to soldering, and/or the insufficient application of heat during the soldering process.)". This type of defect is complicated to identify as it allows for a connection between the two points, even if weaker than ideal, and has the morphology of a target soldering joint. However, the exterior topology of the solder has duller appearance than target. It can lead to serious consequences in a product as it will fail only after some use if missed during quality control [4].

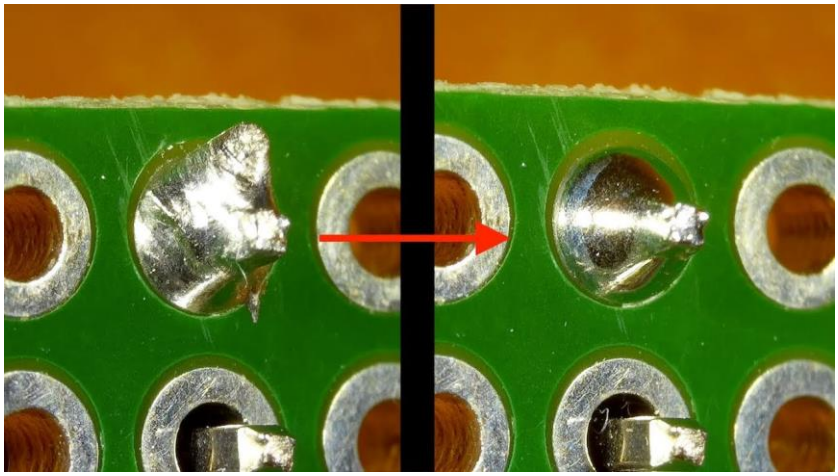


Figure 9 Target joint and cold joint comparison [4]

In the picture a comparison between a target soldering joint and a cold one in a through hole component is shown.

2.1.4.4 Excess solder

This anomaly behaves more like a category within itself due to the highly fluid behaviour of soldering and the pattern or morphology it acquires when present in excess. Different morphologies will cause different issues.

Firstly, the result of excessive soldering remaining in the intended connection and forming a spherical shape. This can be an issue going forward in the product's assembly process if this mound of solder exceeds the available geometric constraints and is also wasteful of the soldering material that can raise overall production costs. Apart from these issues the correct wetting of the joint components is also not guaranteed [5].

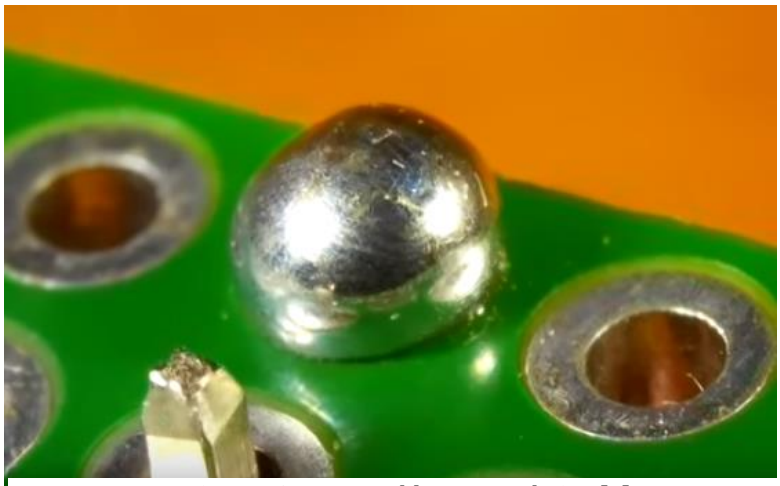


Figure 10 Excess solder accumulation [4]

A more problematic result is solder balling. The formation of small spheres of solder that adhere themselves either to other components or to the board's laminate. These are very complicated to spot during

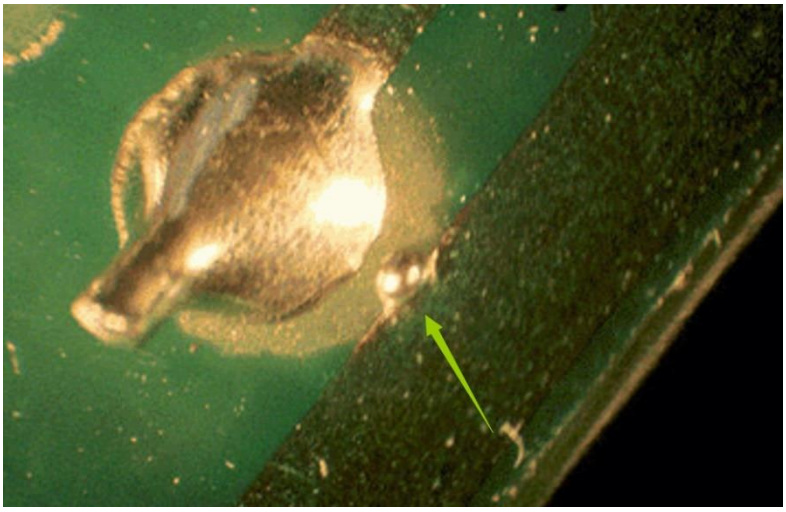


Figure 11 Solder Ball [4]

quality control due to their random location and can cause multiple issues ranging from creating short circuits between different components or if they are dislodge during handling of the product they can cause damage to other components. There can be some allowance to their presence, either if they do not violate some geometric dimensions or if they remain entrapped either beneath a contact or component. Solder balls are a more common appearance in reflow or wave soldering processes and can be caused by multiple factors like the presence of moisture near the PCB that when heated up can cause vapor to release and form the ball. Or due to incorrect setting of parameters in wave soldering, for example when it comes to flux quantities.

When solder solidifies joining two contacts that through specification are not supposed to be connected the anomaly is labelled bridging. This can cause many types of issues in the final product and can be cause by a variety of reasons [6]:

- Excessive solder depositions.
- Soldering pads or contacts too large in comparison to the gap between them.
- Incorrect placement of components.

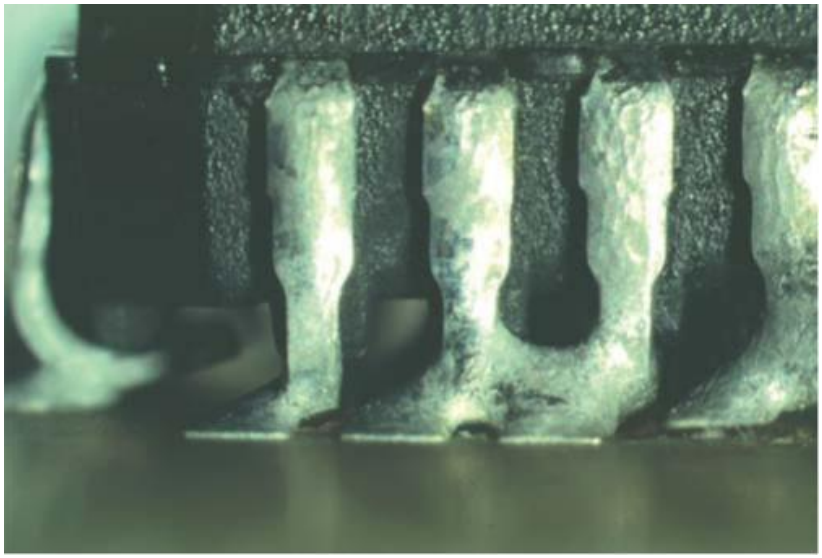


Figure 12 Bridging on connector pads [4].

Finally, when this excessive solder solidifies in the PCB creating a pattern similar to a spider's web, the anomaly is dubbed solder webbing or splashing. If the splashes are positioned in a way, they do not create short circuits, or they are small enough to not cross two areas in a PCB of different electric potential they are considered acceptable. However, when these splashes either show the capability of becoming loose or impact the form, fit or function of the product or cross different electrical potentials they are deemed a

defect. The most common cause for this anomaly is either the insufficient use of flux or the existence of pollutants on the surface of the PCB [5].

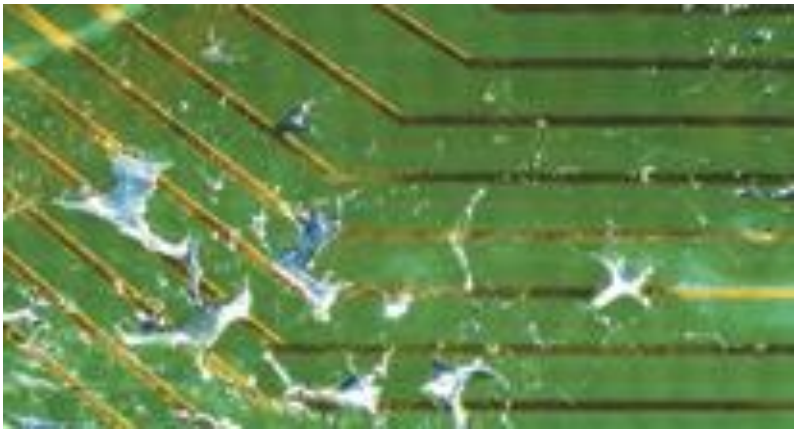


Figure 13 Solder webbing or splashing on PCB
[4]

2.1.4.5 Solder Projections

In some situations, a mutuality of temperature and lack of fluxing material will cause oxide layers to form in the surface of the molten solder. These conditions foster the formation of spike shaped shapes in the soldering during cooling. Depending on the dimensions of these peak-like structures and the constraints of the product they can interfere with the assembly of the product. They can usually be easily identified during inspection [7].

Figure 14 Solder peak [4].

2.2 Machine learning

This chapter will create an introduction to the broad machine learning field of applications. Introducing the various models and their functions as well as the mathematical principals behind this software. Some extra detail will be given on the field of deep learning and image classification using neural networks as this will be the main approach during this work. Hopefully, this chapter will create a theoretical basis that would allow understanding of the development process.

2.2.1 Traditional programming vs Artificial Intelligence vs Machine learning

Initially to more easily follow the nomenclature used in this work it is important to introduce the different concepts in this field of work. Furthermore, Artificial intelligence is the hugely broad term that pertains to any piece of technology, hardware, and software, that analyzes its environment and makes decisions or predictions based on this data. This concept changes very often as the users or technology leaders remove some capabilities of AI like for example some machine vision capabilities like reading bar codes have been removed from the field of AI as not being considered "intelligence". Faced with this difficulty when defining AI Larry Tesler, a computer scientist having worked in many leading companies working with AI, created his famous theorem stating that:

-"Intelligence is whatever machines haven't done yet" [8].

A very famous example of Tesler's theorem in practice is IBM's effort to showcase the capabilities of their supercomputer. In 1997 the IBM supercomputer Deep Blue defeated the professional chess player Garry Kasparov. This was one of the first public displays of the powers of AI. However, this approach differs from the current standards of machine learning practices. The Deep Blue supercomputer was composed of 256 processors that were explicitly programmed to analyze all the possible plays in the chess match and predict the most favorable outcome. After this achievement, the barrier was shifted as this was considered a very "brute force" approach and not real intelligence [9].

Since then the field of AI has been fleshed out and some of its main dimensions are shown in the diagram below. With the scope of this project we are focusing on Machine learning [10].

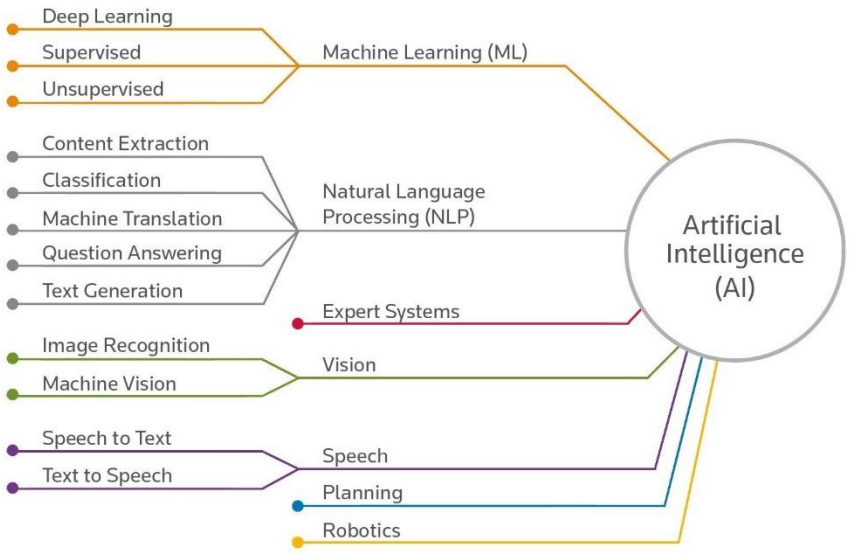


Figure 15 Artificial intelligence diagram [10].

This leads into the distinction between traditional programming and machine learning. This is a very intuitive way to define machine learning.

With the advent of more powerful computing equipment, either when it comes to the ability to process data in higher quantities or in less time, the goal of mimicking human intelligence in software seems closer and closer. At a high-level ML is a field that studies algorithms and models that allow systems to perform tasks without being explicitly instructed and relying on inferences and predictions “learned” from input data.

A common technical definition used for this field is from Tom M. Mitchell a computer scientist in this field.

-“A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E” [11].

At a higher level this means that this kind of computer program learns by analysing data, experience E and with regards to an optimization equation increase its performance factors P to improve at a certain task.

This is a clear distinction when compared with normal programming as shown in the graph below [12].

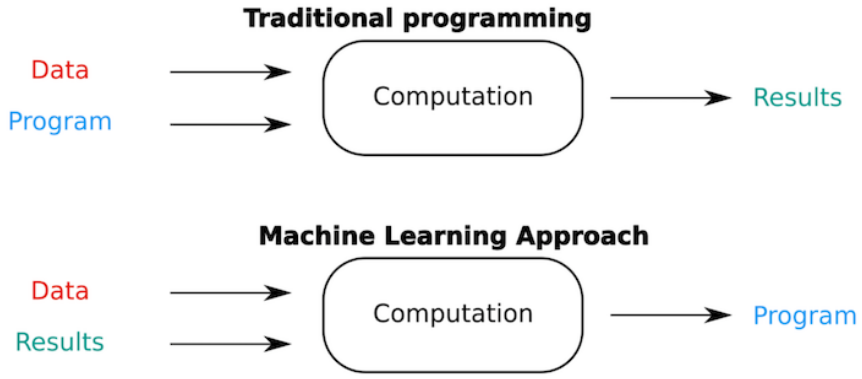


Figure 16 Traditional programming and ML comparison [13].

As with a traditional software we consider the data and the explicitly programmed instructions as and input and expect the results as an output. When it comes to machine learning the data and the results for these data points are known to us and the objective is to output a program capable of generating results for new unknown datapoints [13].

Going forward the different models and approaches inside the machine learning category will be established.

2.2.2 Supervised learning

One way to classify machine learning models is according to the learning approach. Supervised models are the most abundant kind of ML models and the ones most easily identified as ML. They depend on

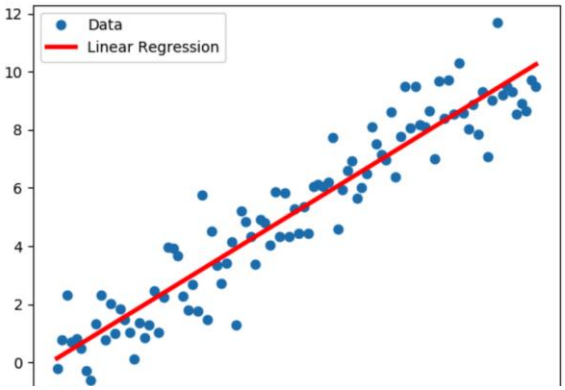


Figure 17 Linear regression example [14].

labelled training data as an input to serve as examples for creating a relationship between this data. A simple example of supervised training model is linear regression where a list of data serves as an input and the linear function that fits these data points is calculated [14].

In fact, most of the supervised learning problems can be defined as a Classification problem, the objective of the problem is to classify the input with a certain label. A very early example is the MNIST dataset, where images of handwritten digits are the inputs and the output is the model's prediction of what this image represents [15]. Or in the other hand as a regression problem, this last one has as a main goal to output a numerical value as a prediction. An example would be the prediction of a student's grade based on metrics like attendance, age, and the sort. The big difference between the two types of problems is that on a regression problem the output does not need to appear in the training data, in the classification approach the labels must be represented and the output of the model is constrained to them.

Many different models fall into this category this work will focus on Neural networks as these are the main players when it comes to image analysis.

2.2.3 Artificial Neural Networks

ANN are machine learning models that, as the name suggests, share a vague resemblance with their biological counterparts. Like in the human brain ANN also consist of neurons connected by each other to form synapses. Depending of the type of component in the ANN the signal being transmitted can be for example a real number, for each node and layer there are weights and activation function that rule the effect of each node of each layer in the final outcome [16].

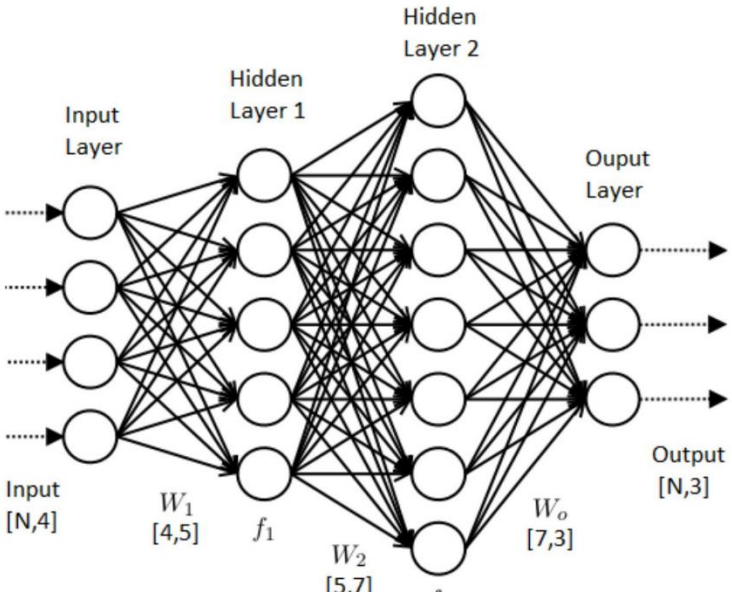


Figure 18 Artificial Neural Network architecture graph [16].

As shown in the graph with each connection between layers the Wn values control the input of the following layer and then this input is passed through an activation function that can trigger another neuron or give out the final prediction. As such the function for each feature X on layer i+1 is [17]:

$$X(i+1) = f(Xi * Wi + bi)$$

Being Xi the past feature Wi the weight of this layer and bi the bias.

Training with this sort of model is achieved by passing the model's prediction and the correct value through a cost function in order to analyze how incorrect this prediction was and backpropagate this result by tuning the weights and biases of each layer in order to minimize this difference. Every training iteration of training and backpropagation is dubbed Epoch [17].

The effectiveness of neural networks is somewhat linked with the number of layers in the model this is an impediment when it comes to processing power as with the increasing number of layers or features there is an exponential computational requirement with the increasing number of weights, biases and activations that need to be calculated. This had been a setback for this type of models until the technology advanced enough to be able to catch up with the theory.

Lately, the growing efforts in developing computing technology as allowed for a widespread use of greater ANN models [18].

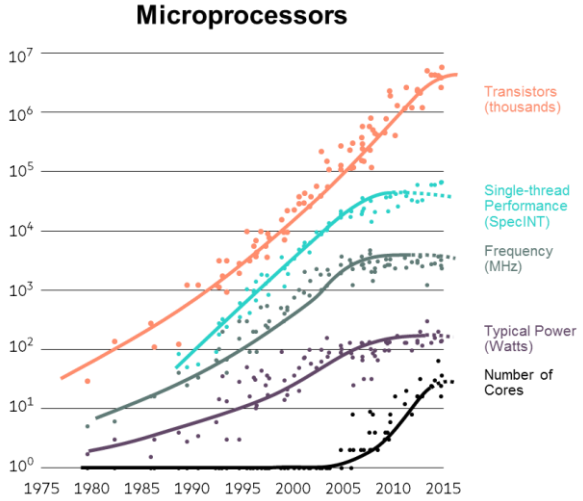


Figure 19 Raise in computing capabilities 1975-2015 [18]

With this a new approach on using ANN models was developed. This approach taking the benefits of ANN and exploiting them by using models with greater number of hidden layers responsible for feature extraction.

2.2.4 Deep Learning

This new capability allowed the use of ANN to solve more complex problems like vision or natural language processing. This technology has been a huge driver of innovation in the field of self-driving automotive. In this particular example Ana I. Maqueda et all use a deep learning neural network paired with an event camera to predict the angle of a vehicle's steering [19].

This project will focus on the image analysis capabilities of Deep Learning models, as such we will be considering the Convolutional Neural Network family of models that are the best performing when discussing these types of tasks [20].

Furthermore, there are two main approaches when it comes to image analysis by neural networks:

- Image classification.
- Object detection.

Both these approaches have the same start point however differ in the result. Image classification based on CNN analyses the image as a matrix of pixel values and depending on the kind of layer does matrix operations with these values to in the end obtain a vector containing a probability value corresponding to each of the classes. The methods by which these matrix manipulations are made will be expanded on later in this chapter. An example of this using the classification of an image of a car can be seen below [21].

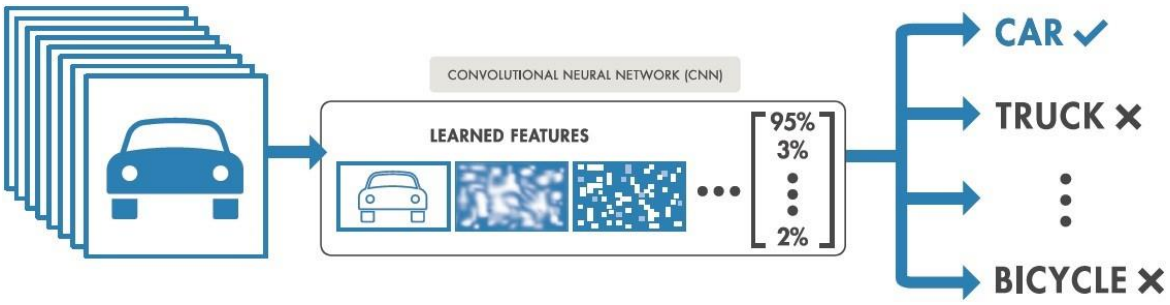


Figure 20 Example of classification workflow [21]

Object detection on the other hand, if we follow for example the R-CNN approach the first layers of the model focus on extracting the Region Of Interest based on the training data input and the classification portion will work only on this predetermined areas. Effectively allowing the model to define the overall position of the object. This solves some issues that the simple classification approach raises but raises some by itself. Firstly, classification algorithm struggle when trying to classify an input with multiple labels. I.e. a picture containing a car and a truck, since the algorithm will look for the maximum value on the output vector and ignore the rest. Object detection on the other hand handles multi classes by classifying different portions of the input this is favorable when faced with a noisy background that will be ignored by the regions of interest. This is highly pertinent when it comes to object avoidance in self-driving cars. However, the training and labeling process requires a greater amount of time as the training data is not only two vectors with the input and the labels but must also contain the coordinate values for the region of interest [22].

Having introduced the two main approaches for image analysis using CNNs some context will be given on the different components of the CNN.

There are three main layer components the convolutional layer, the pooling layer, and the output layer. A successful combination of the three will result in the optimized model.

Initially the data enters the model via the input layer as a tensor shape with the following dimension [23]:

$$t(N, H, W, D)$$

Being N the number of images, H the height of the image, W the width and D the depth, this last one relates to the number of channels of data, i.e. a color image would have value of 3 for depth.

Convolutional layers have a defined height and width and based on this value associated on a weighted matrix they realize the dot product of regions of the image as shown in the graphic below[24]:

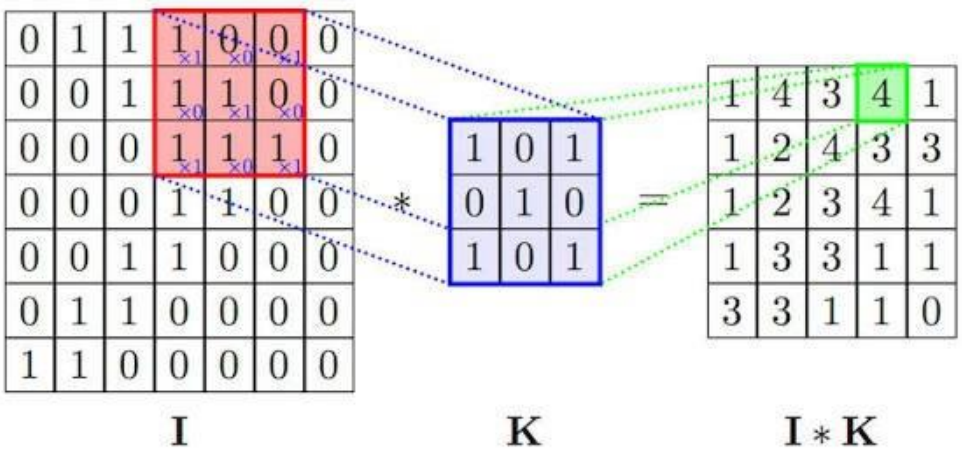


Figure 21 Convolutional layer calculation [24]

In this case a 3*3 Convolutional layer is considered, and this operation concludes in the extraction of the main regions of interest.

In between convolutional layer it is common to insert a pooling layer with the objective of reducing complexity of the data and model and eventually reduce overfitting. This process is done by reducing the input size essentially down sampling it [24].

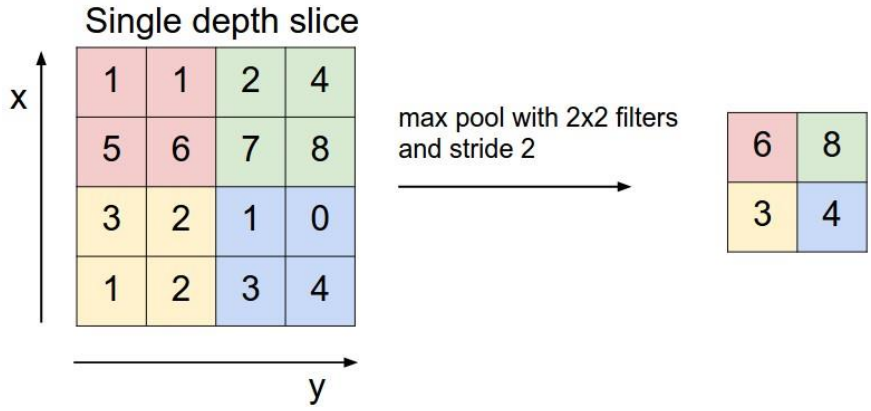


Figure 22 Maximum pooling layer calculation [24]

Here we can see an example of max pooling where the input size is reduced to 2*2.

Finally, regarding the output, depending on the goal for the output dimension some final matrix manipulations can be realized. One example is a flatten layer that converts the input into a 1-dimension vector that can be analyzed [24].

Another parameter to take into consideration and will define the behavior of the model is the activation function picked for each of the convolutional sections. The two most common activation functions will be presented, the Sigmoid and the ReLu [25].

The Sigmoid function is highly used in CNN thanks to it's output being present between 0 and 1 and as such allow us to convert outputs into probabilistic values, this is very useful in image analysis applications. It is defined as follows [25]:

$$f(x) = \frac{1}{1 + e^{-x}}$$

The other and nowadays most used is the Rectified Linear Unit or ReLu. It has a similar working principal as the sigmoid as it returns 0 if faced with negative values but for positive values it returns the value itself. It is defined as:

$$f(x) = \max(0, x)$$

This function has gained in popularity for its help solving the non-linearity issue with CNN as the ReLu has a non-linear characteristic around 0 however it is always either 0 or 1.

Finally, the last parameter to be defined is what algorithm will be used for optimization of our model. These optimizers function with the objective to modify the weights and biases of the network with the objective to minimize a certain cost function that will be defined going forward. There are multiple optimizers in use in this field however the project will go over the two main: Gradient descent and Adam.

Gradient descent is the most common optimization algorithm overall. It's a first -order optimizer and depends on the first order derivative of a loss function. Tries to achieve a minima by applying the following algorithm [25].

$$\theta = \theta - a * \nabla J(\theta)$$

Being θ the weight values and J the loss function. It has fairly easy computation and implementation however it has a tendency to stop at a local minima not global and since it requires the calculation of gradient of the whole dataset it takes a large amount of memory [25].

On the other hand, Adam (Adaptive Moment Estimation) works with momentums of first and second order to control the descent of the algorithm as to not go over the minimum values and decrease the reduction velocity in a controlled manner. It keeps a weighted average of the past gradient calculations.

It takes two values M(t) and V(T) which are the first and second moment, the mean and uncentered variance, respectively. As such each parameter update is equal to:

$$\theta_{t+1} = \theta_t - \frac{N}{\sqrt{V(t)} + \epsilon} * M(t)$$

This optimizer converges rapidly which is very important since the main bottleneck is the time each training iteration takes. However is still computationally costly for each calculation [25].

2.2.4.1 Performance Analysis in Deep learning

Having learned about the building blocks of a deep learning models it is needed now to understand how the training step is regulated and how the performance of a model can be analyzed. First of all, it is paramount to understand what is being used to measure performance. There are two main sets of data when building machine learning models. The initial dataset composed of image data and labels is split into two. The first and most important Training Data is the data the model will use to learn and define the weights and biases. The second, Validation Data is used to analyze the training behavior. The model builder has to consider the composition and class distribution of this training set with the goal being to have a balanced data set with the same number of recordings for each class [26].

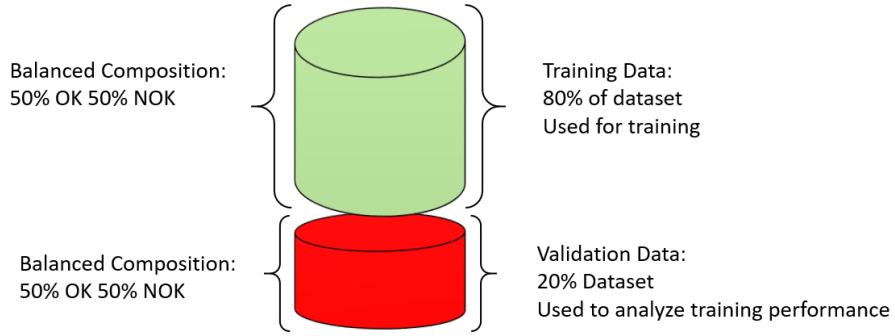


Figure 23 ML dataset distribution

In this example an 80%/20% split is considered.

The most intuitive metric is Classification Accuracy during training this metric is calculated for both the training data and validation data in the following way:

$$Acc = \frac{Number\ of\ Correct\ Predictions}{Number\ of\ Items\ in\ training\ set} * 100$$

The output is a percentage value for the amount of times the model predicted correctly. This metric has a great flaw when handling an unbalanced data set. I.e. if the model contains only 90% of class A and 10% of class B then if the model indiscriminately predicts class A it will arrive in a 90% accuracy which seems like a very high performance. Another hurdle for using this metric is that different classification carries different weights. If for example a good part is marked as a defect this is a much less serious case than a bad part being classified as good and shipped to the customer. One indicator of the training performance is analysing the deviation between the training and validation accuracy. If the first rises and the second takes a must lower value or decreases this may mean that the model is displaying the behaviour of overfitting instead of learning the distinct features of each label it started memorizing the training data and this explains the lack of performance on validation data, images it hasn't yet looked at [26].

The Cross-Entropy loss function penalises wrong classifications and functions only when the output of the algorithm is a probability for each class for each of the data points. The calculation of the Loss is as follows [27]:

$$LogLoss = \frac{-1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} * \log\left(p_{ij}\right)$$

The value y takes a becomes 1 or 0 depending if the sample i belongs to the class j and is the probability calculated by the model. The result can exist from [0, ∞] and as it rises it indicates lower accuracy as the model steps further away from the correct predictions. Two values of loss are usually calculated one for the training data and other for the validation and they also allow the diagnosis of overfitting.

Finally, the most comprehensive metric is the Confusion matrix it outputs a matrix that analyses the whole behaviour of the model. Considering an example for a binary classifier the confusion matrix comes as follows [28]:

Table 1 Confusion matrix example

	Model No	**Model Yes**
True No	No	False Positive
True Yes	False Negative	Yes

This matrix allows us to understand the complete behaviour of the model. The rows indicate the model's prediction and the column indicate the true label of the data samples. Analysing the confusion matrix lets us adapt further training and figure out where our model is struggling.

2.2.4.2 Transfer-learning

As mentioned before the quality and quantity of a dataset is the most crucial factor for the successful solving of a task via a machine learning model. A technique that was developed to help ease this issue is transfer-learning. This technique has grown in popularity used with deep learning models specifically to solve vision or language processing tasks. It consists in the use of a model with a great number of layers that has already been trained in a more general dataset. For example, when it comes to vision problems the ImageNet dataset of 14 million images previously labelled is used to train the models. The overall concept is that the beginning layers of the model are responsible for understanding what features of an image define it. By taking this trained part of a model and connecting it to a classifier with only the intended number of labels allows the user to leverage this model in a much smaller dataset.

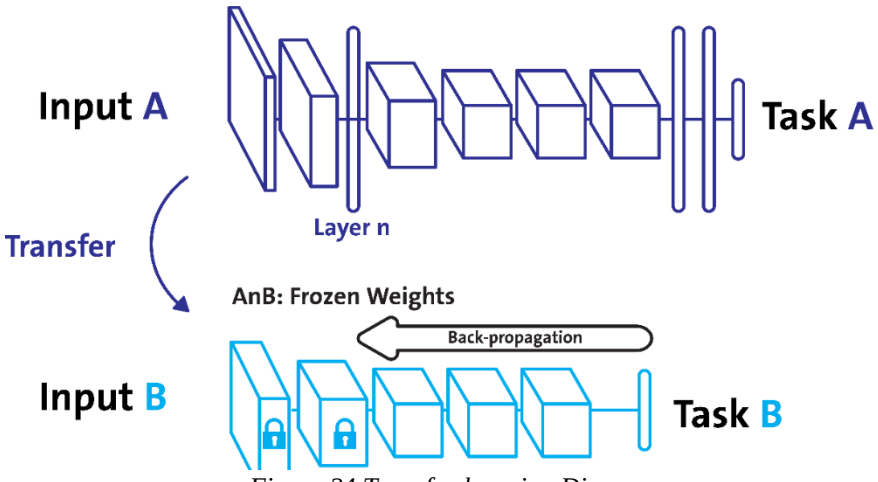


Figure 24 Transfer-learning Diagram

The process of reusing part of a trained model is called freezing, this considers using the same previously converged weights and applying it to the new task. Apart from the advantages mentioned, this technique also allows a reduction in training time and computational consumption required to train this very complex models.

3. Deep Learning System Development

3.1 Problem definition

In this chapter the project motivation and goals are going to be analysed. The project methodology will be outlined, and some background will be provided for the boundary conditions for the development stage.

As aforementioned, the Oral-B iO™ is a brand-new product of the Power Oral Care line. Soldering is a primary process in its assembly specifically for creating the connections between the battery, the drive unit, and the charging coil. In the picture bellow it is possible to analyse the contacts in question which will then be the object of the vision system developed.

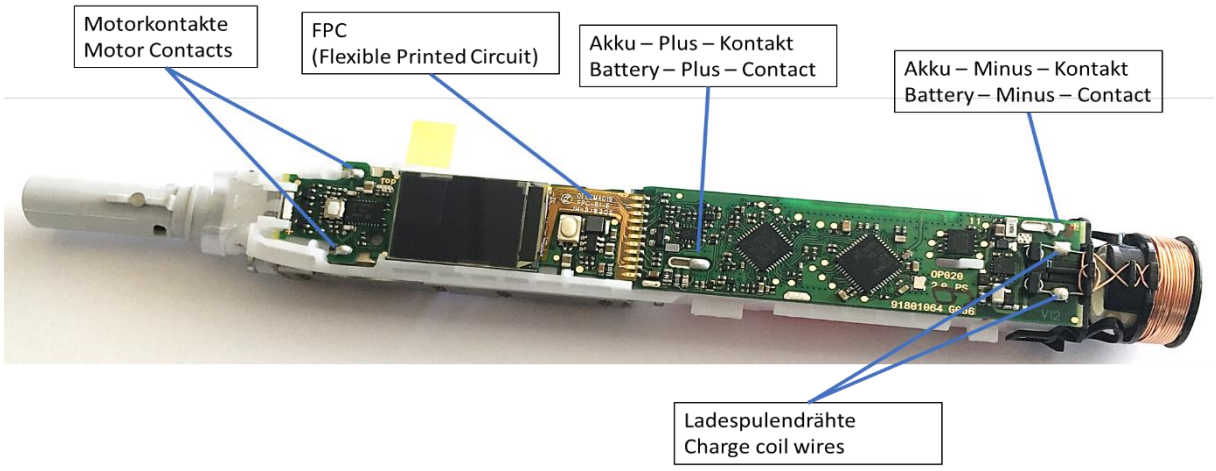


Figure 25 Oral-B iO contacts and PCB

In total there are six contacts of interest and it is regarding them that the vision system will classify.

In the assembly line for the Oral-B iO™ product when it comes to soldering the process involved is selective wave soldering. Selective wave soldering refers to a mass production soldering process where parts are dipped in a container of solder. A pump connected to this container will then pressurize and cause a raise in the surface level of the solder causing the mini wave. To control the spread on this material only through the desired contacts a titanium mask is used to cloak the areas of the PCB where the flow of solder is not intended, this mask also allows for the preheating of the contacts by employment of inserts. Flux is also present in the operation to improve the joints reliability and paired with a control atmosphere, given that nitrogen gas is present as a shielding gas, and with controlled temperature, jointly contribute to a high process capacity and to the production of mechanical and electrically reliable joints. Wave soldering was also selected for this role for its ability to function with multiple types of contacts be it surface mount technology, through-hole or wire-contact interfaces, for the overall low cycle time, when compared with other soldering processes, and for the overall high reliability of the parts [29].

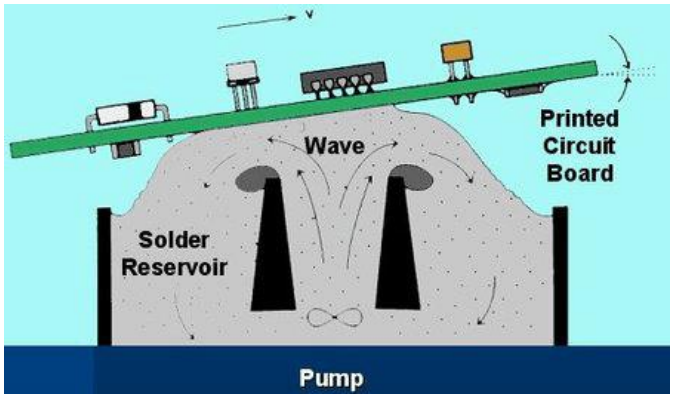


Figure 26 Wave soldering working diagram [42].

Despite all these factors, due to the variability of initial conditions, the cumulative variation of tolerances and the inherent overall variability of soldering, some parts are being rejected by the operators after the manual inspection station, due to the soldering joints being deemed not according to specifications. The definition of these defects is as follows:

- Open joints or unsoldered contacts. Easiest to identify characterized by the lack of wetting of the contact. This morphology can be caused by the lack of flux in the contact.
- Joints that are too high or too low. Insufficient wetting of the contacts or too much soldering that can be problematic in the following steps in assembly.
- Solder splashes. When the solder solidifies past the region of the contact and bridges two areas of different potential in the PCB.
- Soldering peaks. When the solder solidifies forming a peak like structure, can be caused by too high temperature.

During the inspection process the operators utilize a human machine interface to record the classification of the parts. These values are then recorded in a SQL server database, which will be leveraged in the training data creation step of this project.

3.2 Project Goals

The overall goal of the given project was to validate the effectiveness of a Machine Learning algorithm for the identification of soldering defects. Once this background is established and the starting conditions for the development of the project are well defined, the project deliverables can be defined. Therefore, these are:

1. Development of a set-up for inline image acquisition.
2. Leverage SQL database data for labelling these images or utilize offline labelling.
3. Creation of a structured and balanced dataset.
4. Defining and benchmarking algorithm architectures for optimal accuracy.
5. Create a minimum value prototype and inline testing of the vision system.
6. Developing final specifications for line implementation of the inspection system.

3.3 Development methodology

A very important factor that will define the specifications of our model, is the fact that it will be implemented in a production environment. This translates to a very consistent positioning of the subject of the images i.e. the PCB, in line this allows the utilization of traditional vision systems and will also improve and ease the implementation of a deep learning model. The PCB's are transported through the assembly line in Bosch Rexroth Transfer Systems Workpiece Carrier [30] and conveyor belt as is shown in the 3D CAD model's below.

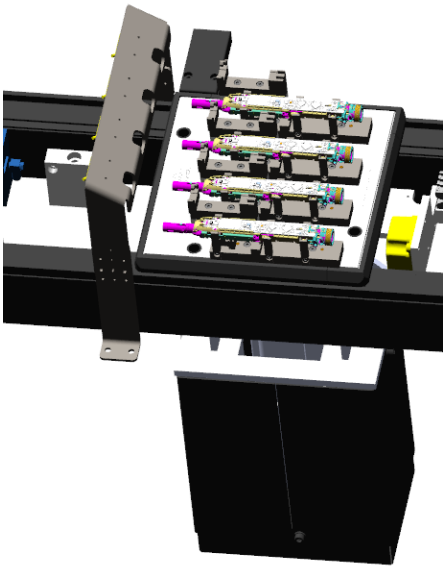


Figure 28 WPC and transport system

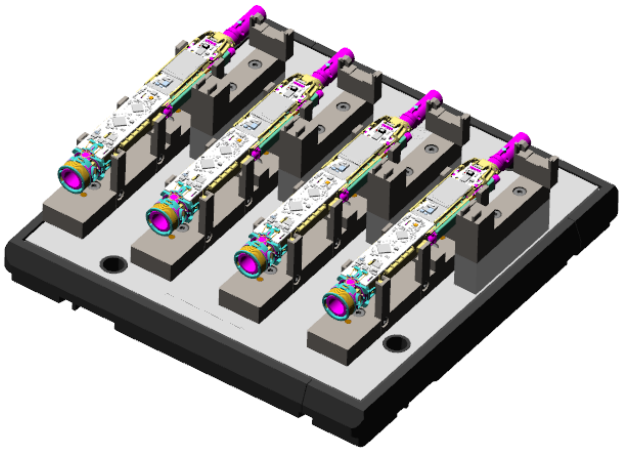


Figure 27 Workpiece Carrier and parts

This provides very useful starting conditions specially since we know that the consistency of the training data is a greatly important success factor for deep learning models.

When discussing deep learning models for image analysis two different approaches were introduced:

- Object detection,
- image classification.

Since the conditions of the assembly line mentioned before are present, this implies that, at the moment of image acquisition, the area of interest, in this case the area of the contacts, will be positioned, considering some geometric tolerances, in a certain position. Considering the characteristics of an object detection model, being composed of two portions, the first extracts the features that it has learned, related to the region of interest, and predicts these positional values. The following portion works on the classification of this area. Even considering the higher adaptability of these models, when there is a great variation of background or the position of the object, in this particular case it is redundant to find this position, given that this is defined by the position of the handle in the workpiece carrier. Accordingly, utilizing a classification model will permit the application of a much simpler labelling process, where text labels instead of bounding boxes coupled with a much lower computational requirement and a greater, even if less flexible, capacity for classification will be implemented.

3.3.1 Training data specifications

As previously noted, the amount of training data and the quality of said data is one of the defining factors for the effectiveness of machine learning applications in general. Taking this into account, the first step would be to build the requirements and specifications of the dataset. From the literature we cannot easily define the object size or image resolution requirement for our pictures. The criteria usually rely on a more empirical approach. This being that if a feature in an image would be identified clearly by the human eye, it would then be applicable in a deep learning module.

Initial tests were undertaken using a Nikon D7000 DSLR (Digital single-lens reflex camera), with the objective of developing an understanding of the possible characteristics of our data, when it comes to the relationship between the area of the workpiece carrier that can be captured and the resolution of each joint. The main relevant characteristics of this camera are shown in the table below.

Table 2 Nikon D7000 camera parameter analysis

Parameter	Value
Image Resolution	4056 x 3040 pixels
Focal Length	16-85 mm Variable
View angle	24-127.5°
Lens	AF-S NIKKOR 16-85mm

The analysed configurations were either: trying to capture four parts, the full workpiece carrier or two parts, corresponding to half of it. This change was controlled by the variation of the lens' focal distance and distance from the parts. The results for pixel size per joint can be seen in the table. These values are directly linked to the fulfilment of our initial requirement of being identifiable by a human observer.

Table 3 Resolution analysis for different part configurations

Number of Parts / Joint Type	**2 Parts**	**4 Parts**
Motor	80x80px	39x39px
Battery	71x191px	30x79px
Coil	91x91px	42x42px

When individually analysing the images, it was understood that the resolution obtained with the picture with four parts was not sufficient to identify changes in the joint and solder morphology.

When it comes to the content distribution of these pictures to maintain the balance of our dataset, an analysis of literature has shown that the dimension of a dataset for the training of a deep learning model varies with the application. When analysing multiple sources for this value some of the conclusions like the one suggested by Goodfellow et all is that to achieve favourable results in deep learning applications a dataset composed of 5000 observations of each of the classes is required. As such since we are considering two classes: Ok and Not Ok. The goal for our final dataset would be acquiring 10000 observations with a 50% distribution of each of the labels [26].

3.3.2 Image acquisition setup development

Having established the requirements for our dataset, the next step is developing a system that would allow for the reliable and consistent acquisition of our picture data. As mentioned, we require a constant positioning of the part on our pictures, since the goal for the vision system is to function in the assembly line, it is advisable for these images to be acquired in these same conditions. So, we established the requirements for the image acquisition setup:

- Capture 4056 x 3040 pixels images.
- Record images with two parts.
- Be able to reliably time the pictures.

- Seamless implementation without line changes.
- Bosch transfer system compatibility.

To fulfil the function of a platform and controller for our image acquisition system the Raspberry Pi was picked. Specifically, the model 3b (datasheet and documentation can be found in the attachments). The Raspberry Pi's small size and development board features will allow to act as an interface between the defined vision sensor and the presence sensor for checking the workpiece carrier position. Essentially

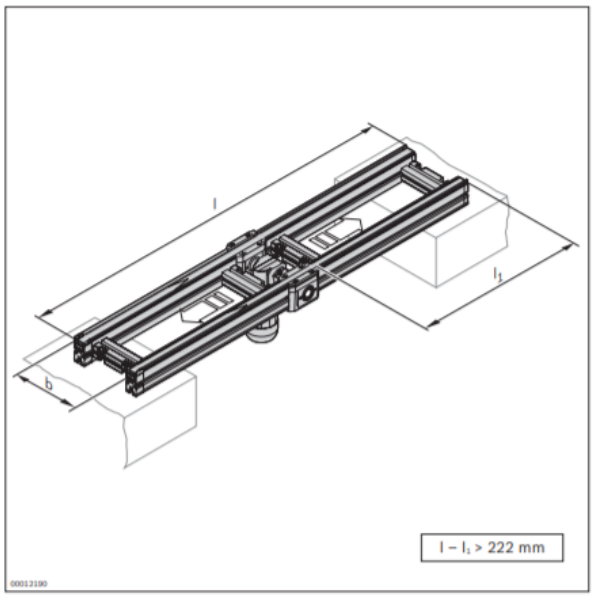
Figure 30 Raspberry Pi HQ camera

Figure 29 Raspberry Pi 3 Development board

functioning as the equipment PLC. Next step is defining how our system will understand when to capture a picture and since this is essential for our pre-determined workflow it is a very important definition.

Most common sensors for this application, that are easily compatible with the 5V system coming from the Raspberry Pi, function mostly using an IR emitter and receiver to either vary the resistance of a resistor (if analogic) or closing a contact and returning the value of a bit (if digital). A well-established supplier of this equipment is Sharp [31].

Analysing the conveyor system dimensions from the Bosch Rexroth catalogue it is possible to define the range of working distances for our sensor.



Material number		3842999717
b (mm)	Track width in direction of transport	160; 240; 320; 400; 480; 640; 800; 1040; 1200
		160 ... 1200[1]
l (mm)	Length	310 ... 6000[2]
l_1 (mm)	Length	90 ... 5770
v_N (m/min)	Nominal speed	0[3]; 6; 9; 12; 15; 18
U (V)	Voltage	See motor data, p. 11-24ff
f (Hz)	Frequency	See motor data, p. 11-24ff
AT	Motor connection S = cable/plug K = terminal box	S; K
MA	Motor mounting R = right L = left M = center	R; L; M[4]

[1] Individual width variants available
[2] l is rounded in accordance with the toothed belt pitch
$l - l_1 > 222$ mm
[3] $v_N = 0$: without motor or gear
[4] When MA = M and b = 160 mm, the max. section load is only 30 kg

Figure 31 Bosch conveyor system specifications

When picking our sensor, the defining parameters are if the signal is analogic or digital and the detection distance. So that we can maximize the information obtained by the sensor analogic was picked. In relation to the detection distance due to the line's dimensions particularly the b value as seen in the diagram two options would fit our specifications: 2 to 15 cm or 4 to 30 cm. Analysing the sensors' characteristic curves, we can see that when the graph approaches the values close to the end of conveyor belt width the voltage values approximate the local minima. And the maximum voltage value is achieved on the range of 0 to 6 centimetres. Considering that the positioning of the sensor has not been defined in our setup the chosen model was the GP2Y0A41SK0F with a detection range of 4 to 30.

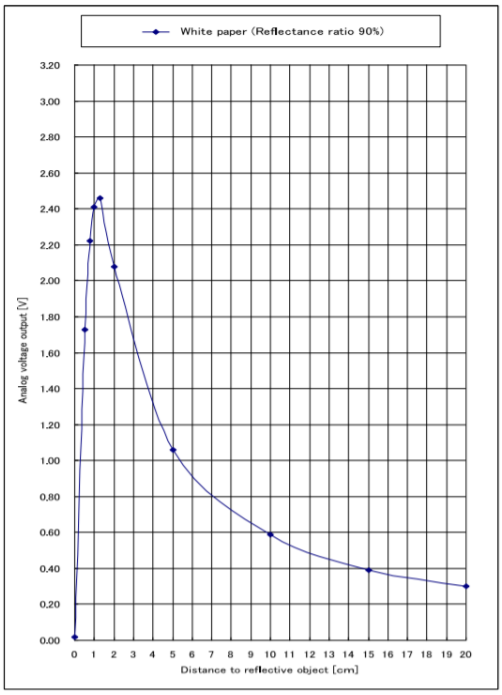


Figure 32 Sensor GP2Y0A51SK0F Characteristic Curve [31].

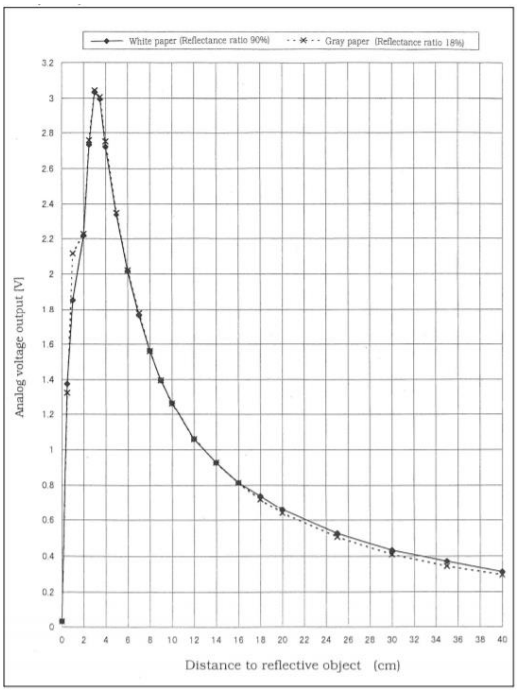


Figure 33 Sensor GP2Y0A41SK0F Characteristic Curve [31].

This sensor would then be connected to the Raspberry Pi's GPIO (General Purpose Input/Output) pins. However, since the development board does not have the capabilities of converting an analog signal to digital inbuilt, an extra component with the function of analog to digital signal conversion had to be implemented. In question a ADS1115 4-Channel 16-bit Analogue Digital Converter [32].

When it comes to vision sensor a Raspberry Pi HQ Camera was considered [33].

This sensor can achieve our requirements when it comes to resolution. Going over the table with some of the specifications we can see that the lens compatibility allows for a C or CS mount lens which are common in industrial vision systems. This type of mount would allow us to utilize the same lens as used with our initial tests with the Nikon D7000 camera when interfacing with an adaptor.

Table 4 Raspberry Pi HQ camera resolution analysis

Parameter	Value
Image Resolution	4056 x 3040 pixels
Sensor size	7.9 mm sensor diagonal
Lens compatibility	CS-mount C-mount

To assemble the components to the final frame mounts where designed to allow the correct positioning of the items. These where designed to be compatible with the very common T-slot extrusion profiles and use T-nuts and bolts for attachment.

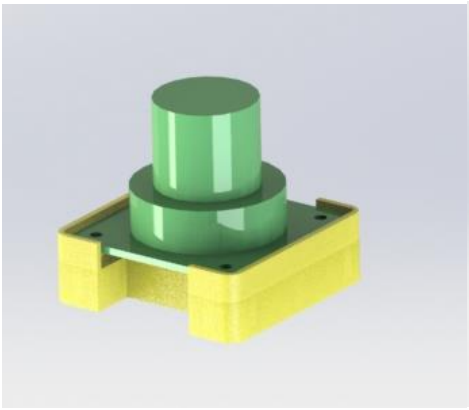


Figure 36 3D printed Camera mount

Figure 35 3D printed sensor mount

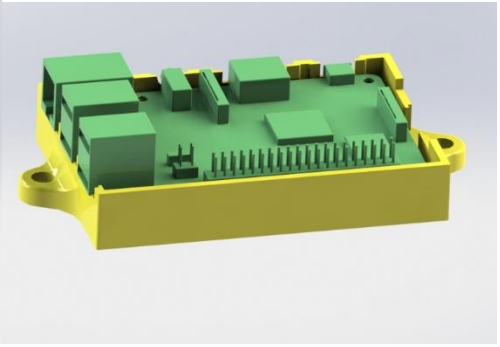


Figure 34 3D printed Raspberry Pi mount

These parts were produced using additive manufacturing in an Ultimaker 3 in Polylactic acid (PLA).

Finally, the frame for mounting the components to the line was designed.

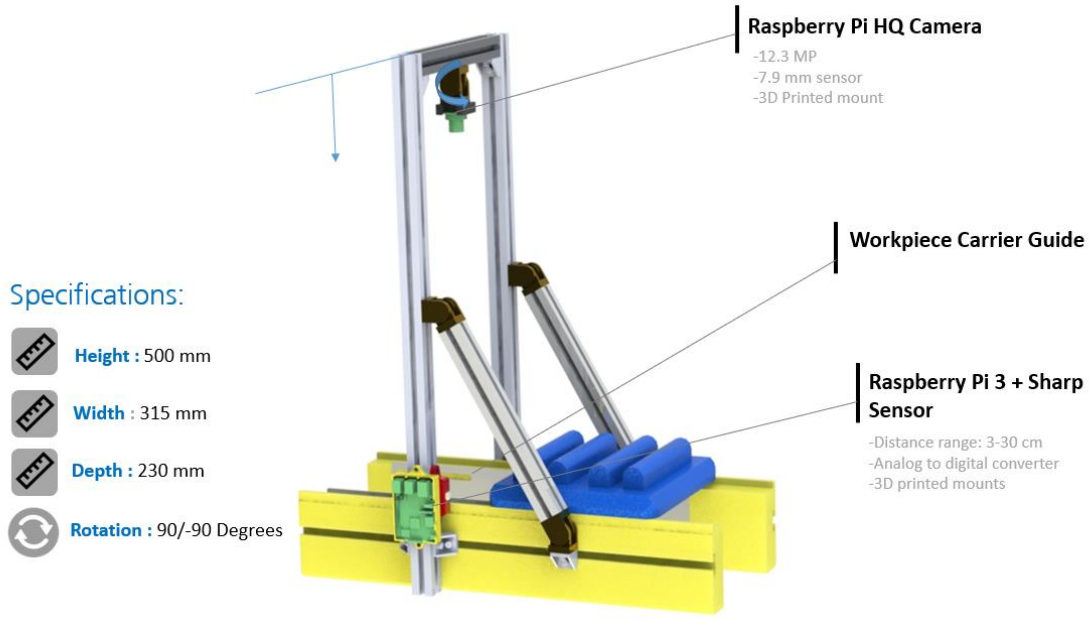


Figure 37 Image acquisition system and specifications

Based around Bosch Rexroth aluminium extrusion profile. Designed to be able to support the load from the camera lens and the to support profiles connected at an angle with the goal to minimize the effects of vibrations from the movement of the conveyor belt in the camera. This setup will allow for adjustable

parameters to be fixed during implementation in the line, the distance to the workpiece carrier and the rotation of the camera.

Initial setups when using the camera paired with the reflex lens and adaptor resulted in below optimal results due to the difference in sensor size between the Raspberry Pi camera and the Nikon D7000 resulting in a larger focus length than ideal. As such other solution for lens had to be tested. Most of the available machine vision C or CS-mount lenses have fixed focal length. To decide on the lens that would satisfy our setup conditions we calculated the ideal focal length.

$$f = \frac{s \times d}{h}\ (mm)$$

With f-Focal length in mm, s- Height or width of sensor in mm, d- distance from object in mm, and h-largest dimension of object in mm

Considering the distance from the object the preestablished 500mm, calculating the height of the sensor when approximate to a square we obtain 5.5mm and finally the length of the chassis that is of interest is approximately 170mm. We obtain a focal length of:

$$f = \frac{5.5 \times 500}{170} \equiv f = 16.18mm$$

Having obtained the focal length, we were able to pick our lens model after some testing.

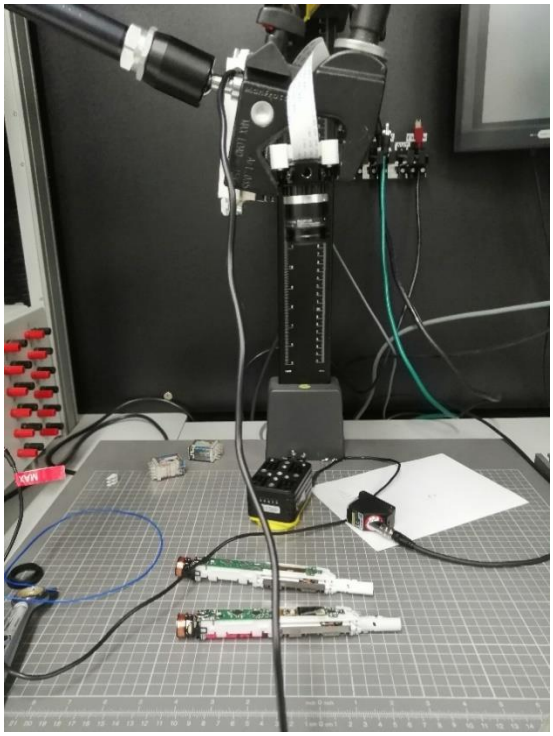


Figure 38 Vision sensor testing setup

By positioning the camera at the set height of 500 mm and testing different lenses with varying focal length, the one that achieved the best results was the Fujinon HF16HA-1B Lens from Fujitsu with 16 mm focal length.

With these parameters defined the image setup was fabricated and implemented in the line. The positioning in the line took advantage of an already present stop gate that would allow for stable picture acquisition. Since we switched to a smaller in size lens and due to some space constraints in the line the size of the frame was also adapted.

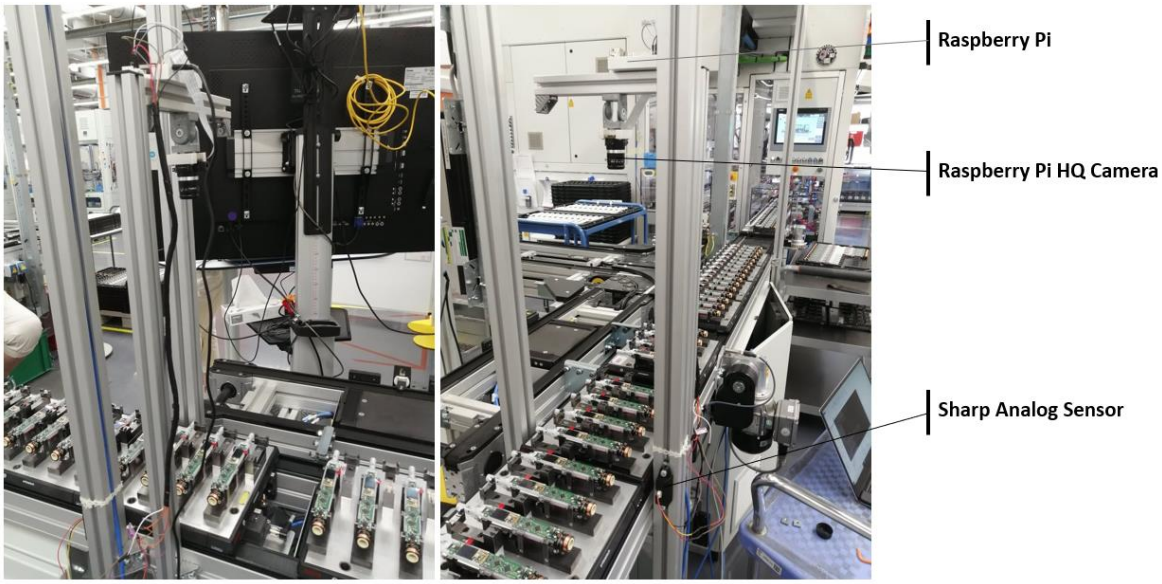


Figure 39 Image acquisition system inline setup

3.3.3 Deep learning model development process

This chapter will go over the process of acquiring training data and building dataset via labelling, choosing the deep learning model architecture and results after training of our built models.

When the process for image acquisition was started by analysing the initial results it was understood that the goal of 5000 observations of Not Ok parts would be a goal not achievable in the duration of this project, due to the defect distribution in the production process, and a smaller dataset would be utilized. One approach to solve this short coming will be the leveraging of a previously mentioned technique of transfer-learning, re-training part of a pre-trained model. The second approach is to create a balanced dataset of images by obtaining artificial not ok parts. This experiment translated into obtaining an equal number of pictures of parts, before the soldering module, where all joints would be of open, and after the soldering module [26].

This resulted in a dataset of 334 parts, these pictures where then automatically cropped into 150x200 pixel size images for each of the joints as shown in the diagram bellow.

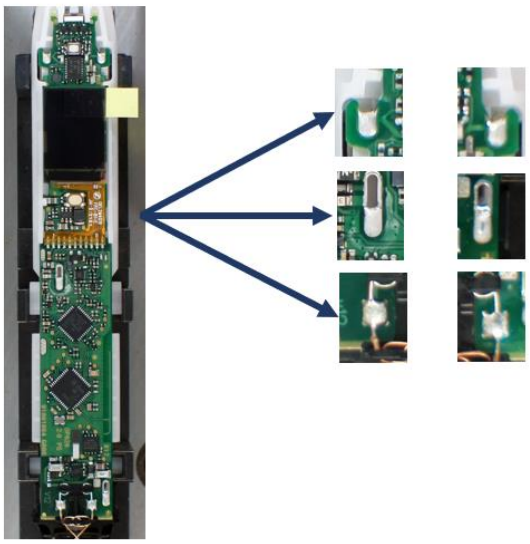


Figure 40 Image segmentation diagram.

The data set was split into training data, 80% of the data set, and validation data 10% of the data. It was then used to benchmark the top five best performing pre-trained transfer learning neural network architectures. The following were the results [34].

Table 5 Multiple CNN architecture benchmarking

Model	Training Results	Validation Results
VGG16	Accuracy=97%; Loss=0.104	Accuracy=98%; Loss=0.094
InceptionResNetV2	Accuracy=94%; Loss=0.234	Accuracy=92%; Loss=0.346
VGG19	Accuracy=96%; Loss=0.125	Accuracy=95%; Loss=0.176
DenseNet169	Accuracy=92%; Loss=0.843	Accuracy=89%; Loss=1.221
NASNetLarge	Accuracy=94%; Loss=0.934	Accuracy=93%; Loss=0.845

The overall best performing model was the VGG16 and the transfer learning of this model was realized by removing the top input layer and changing the input size to (150,200) of our picture size and removing the output dense layer with 1000 output nodes to 2 output nodes – Soldered and Unsoldered. The resulting architecture of this model is as follows.

```
Model: "functional_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         [(None, 100, 200, 3)]     0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 100, 200, 64)      1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 100, 200, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 50, 100, 64)       0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 50, 100, 128)      73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 50, 100, 128)      147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 25, 50, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 25, 50, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 25, 50, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 25, 50, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 12, 25, 256)       0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 12, 25, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 12, 25, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 12, 25, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 6, 12, 512)        0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 6, 12, 512)        2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 6, 12, 512)        2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 6, 12, 512)        2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 3, 6, 512)         0
_________________________________________________________________
flatten_1 (Flatten)          (None, 9216)              0
_________________________________________________________________
dense_1 (Dense)              (None, 2)                 18434
=================================================================
Total params: 14,733,122
Trainable params: 18,434
Non-trainable params: 14,714,688
_________________________________________________________________
```

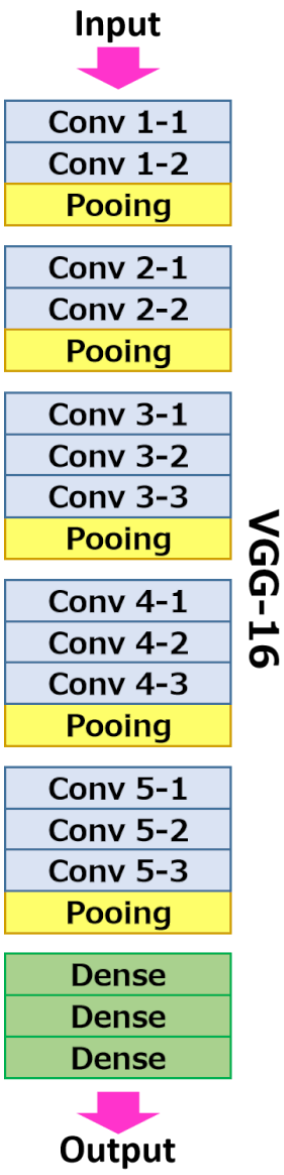


Figure 41 Neural network architecture and diagram

The parameters that remain trainable are the ones referent to the last layer and classifier. These results are overall positive and prove our workflow of using small images of the solder joints and training the model on this data. However, when testing the model with images of unsoldered parts normally occurring after the soldering module showed mixed results. 70% of these images where classified as soldered. This was attributed to the possibility of the deep learning model understanding some features that are constant between pictures taken in the same position i.e. before the soldering module. This could be brightness, geometric positioning, or blurriness. This then leads to a model that learned to look for these features that do not relate to soldering quality. With this in mind, we would then focus on acquiring pictures with constant conditions and normally occurring defects with the image acquisition setup positioned after the soldering module.

Now regarding the creation of the main dataset. A diagram of how the picture acquisition and labelling process is defined is shared below.

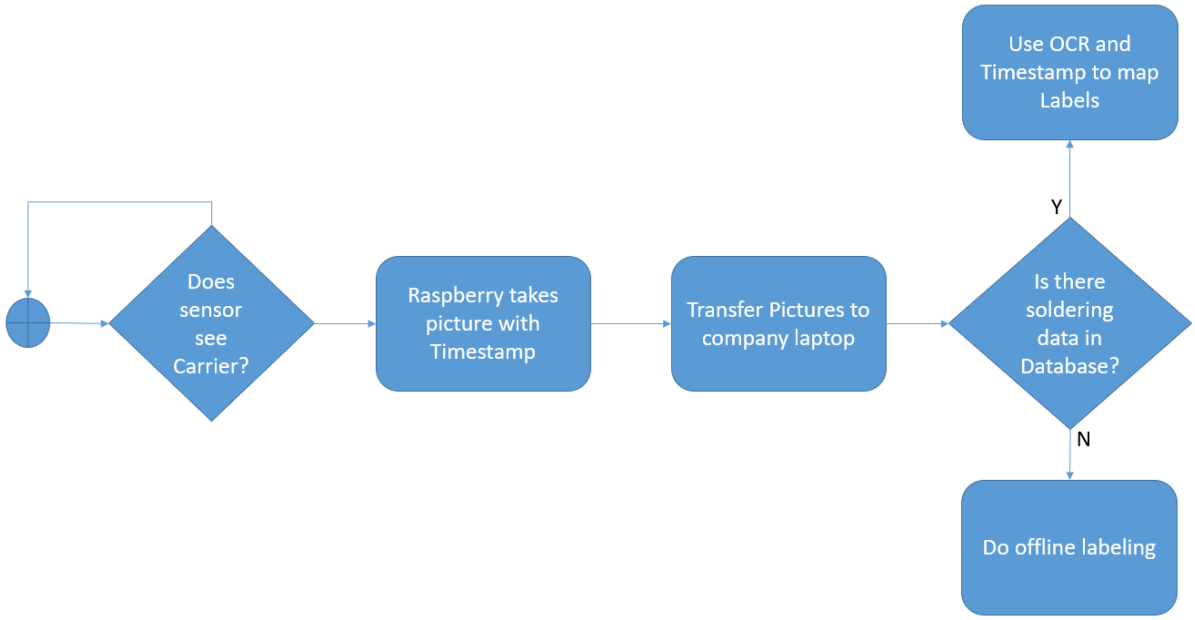


Figure 42 Image acquisition and labelling diagram

As previously mentioned, the operator responsible for quality inspection of the soldering quality uses a human-machine interface to log this data and this value, the Time stamp and the data related to the parts and workpiece carrier is logged in the database. Leveraging this data for this project would allow for faster labelling of the parts and would ensure that the labels are up to standards. Knowing that each workpiece carrier has its unique Carrier ID number, a system based on Optical Character Recognition (OCR) was used. Said process is based on the translation of tags with this number from our images and use this information and the Timestamp, to access the SQL database and connect the data from inspection to our images.

When analysing our final data set, we acquired pictures of 3 811 pairs of parts that translate into 45732 individual joint pictures. The distribution of Not ok and Ok parts in this dataset is shown below.

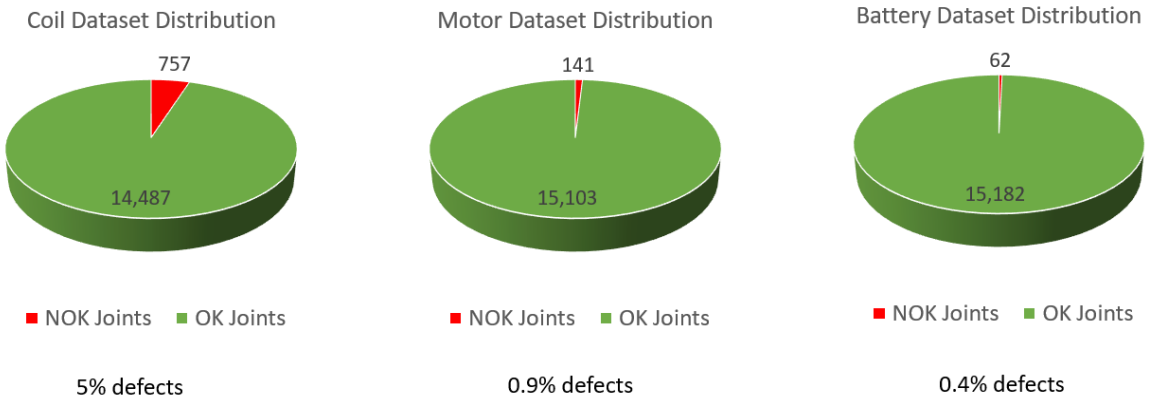


Figure 43 Dataset distribution

As predicted the amount of Not ok parts is very small when compared to overall dataset and below our ideal results. This will create some difficulties in having a deep learning model that is up to standards specially in the joints where this is most predominant: the battery and motor joints.

Having built the dataset, we proceeded to training our previously defined architecture based in VGG16. The results are shown below.

Table 6 VGG16 model trained on all joints results

Parameter	Value
Data distribution	OK: 960; NOK: 960
Epochs	20
Training data results	Accuracy=92%; Loss=0.139
Validation data results	Accuracy=87%; Loss=1.232

However positive these results are, they remain not ideal, both accuracy values are below the target. The accuracy drop is related to the overall bias of the model and dataset towards the coil joint, the most present in the dataset. As such, came the decision to build three different models one to classify each of the analogous pair of joints. These where built with the same overall architecture however the image size was changed for the smaller joint size of the motor and the battery this measure will also allow for an improvement in performance due to the area of interest occupying more of the image. To ensure our models are not biased we will be utilizing the technique of under sampling and utilizing a 50/50 distribution of the dataset. The models where built, and the training results are as follow.

Table 7 VGG16 model trained on motor contacts

Parameter	Value
Data distribution	OK: 141; NOK: 141; 100x200 pixels
Epochs	12
Training data results	Accuracy=98%; Loss=0.053
Validation data results	Accuracy=95%; Loss=0.094

Table 8 VGG16 model trained on battery contacts

Parameter	Value
Data distribution	OK: 62; NOK: 62; 100x200 pixels
Epochs	6
Training data results	Accuracy=98%; Loss=0.023
Validation data results	Accuracy=92%; Loss=0.593

Table 9 VGG16 Model trained on charging coil contacts

Parameter	Value
Data distribution	OK:757; NOK: 757; 150x200 pixels
Epochs	20
Training data results	Accuracy=99%; Loss=0.033
Validation data results	Accuracy=98%; Loss=0.054

This will increase the overall complexity of the vision system however it produces a much more reliable classifier as we can see per the results. Considering the number of epochs, this was smaller when it comes to the battery and motor joints, as when training on the later epochs around epoch 13 for the motor and 8 for the battery, signs of over fitting were showing. As the values for training accuracy kept going up while training loss going down, but the reverse was happening with these values for validation data. As expected, the best performing model was the one built for the coil joint, having a more balanced dataset.

4. Model testing and validation

Having developed the models, there came the motivation to validate its performance against real-time production data directly on the assembly line. This chapter will go over the setup for preparing our system for inline inference in real-time and the results of the overall performance of the model with this data.

Deep learning models have a great computational requirement, that would run too slow on the underpowered CPU of the Raspberry Pi. This factor would prevent the application of the model in real-time, since the inference time would be too long and would not sync with the throughput of the assembly line. To reduce this problem a piece of equipment would be used to accelerate this inference time. One such equipment, and the one that will be used in this project, is the Coral TPU USB accelerator. A product developed by Google with the goal of allowing for the prototyping and edge deployment of deep learning applications [35].

This consists in a USB connected device that contains an integrated circuit designed specifically to accelerate tensor operations, which are the base operations for deep learning models [36].

However, there are software compatibility issues with the full TensorFlow models we are utilizing, as is uses in the activation values and weights float 32-bit data. These must be converted to TensorFlow Lite model that require a full quantization of these values to 8-bit data, this produces smaller and faster running models, in the other hand these models also lose complexity which may cause a loss in performance. Finally, the model must be compiled for the edge TPU.

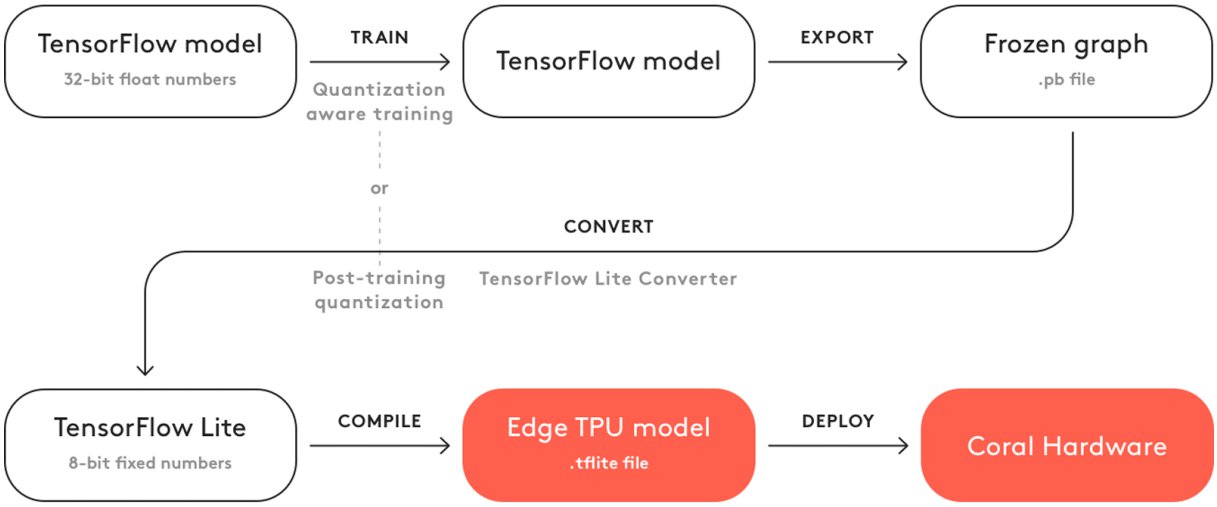


Figure 44 Edge TPU conversion workflow [36].

Two techniques for this conversion are suggested. Quantization aware training that requires retraining of the model and the approximation is done during training. Or post-training quantization that converts the model after the training is done using float 32-bit. Quantization aware training however more complex of

implementation is the one that produces best results as per the documentation and was the one implemented. Accuracy results for a compiled TensorFlow Lite model are shown in the table below.

Table 10 Validation Accuracy results for compiled models

Model	Validation Accuracy
Motor joint	85%
Battery joint	81%
Coil joint	92%

The drop in accuracy is very significative however and will impact performance of the models. This will still allow us the development of a proof of concept of the vision system. After the conversion process tests where undergone for the calculation and comparison of inference times between the different models and hardware configurations.

Table 11 Inference time comparison of hardware when running TFlite model

Hardware	Inference Time
Raspberry Pi	4324ms
Intel Core I9-8950 HK 2.90 GHz	1205ms
Nvidia Quadro P2000	865ms
Raspberry Pi + Coral Edge TPU	100ms

Table 12 Inference time comparison of hardware when running full TensorFlow model

Hardware	Inference Time
Intel Core I9-8950 HK 2.90 GHz	115ms
Nvidia Quadro P2000	2ms

The two tables compare the inference values obtained comparing the inference on one image for the Raspberry pi, Intel CPU, Nvidia GPU and Raspberry Pi and Coral TPU combo. On the first table analysing the performance of the TensorFlow Lite model on these devices having in mind that the lower performance on the CPU or GPU is due to it not being optimized for these devices. There is a clear improvement from the initial 4 seconds to 100ms this would allow for inline inference. On the second table we are considering the inference time of the full model however this is not compatible with the Raspberry Pi by itself or when combined with Coral. The GPU would be the ideal tool for running of inference since there is no drop in accuracy, it is compatible with the full TensorFlow model, and has the shortest inference time.

4.1 Validation results

These systems where then tested on X parts and X pictures of each type of joint and the resulting confusion matrix are shown below.

Table 13 Confusion matrix for the full TensorFlow Motor Joint model

Motor Joint	**Model Good**	**Model Bad**
True Good	94,14%	0.71%
True Bad	0.21%	4.92%

Table 14 Confusion matrix for the full TensorFlow Battery Joint model

Battery Joint	**Model Good**	**Model Bad**
True Good	95,65%	0,11%
True Bad	0	4.24%

Table 15 Confusion matrix for the full TensorFlow Coil Joint model

Coil Joint	**Model Good**	**Model Bad**
True Good	94.34%	0.27%
True Bad	0	5,39%

The first three confusion matrix refer to the full TensorFlow model being ran on the Nvidia Quadro GPU. Going over the results we can see that even the least expected performing model, the battery, shows very positive results as the model only classified 0,21% good parts as bad, creating pseudo scrap. When it comes to the motor, 0,71% bad parts where classified as good which is the worst-case scenario as it means that bad parts would exit the assembly line undetected. This, together with the wrong predictions from the coil model, upon an analysing of images, was traced back to images being highly out of focus.

Table 16 Confusion matrix for TFlite Motor Joint model

Motor Joint	**Model Good**	**Model Bad**
True Good	93,64%	1,09%
True Bad	0,82%	4,46%

Table 17 Confusion matrix for TFlite Battery Joint model

Battery Joint	**Model Good**	**Model Bad**
True Good	93,85%	1,92%
True Bad	0,71%	3,52%

Table 18 Confusion matrix for TFlite Coil Joint model

Coil Joint	**Model Good**	**Model Bad**
True Good	94,89%	0,54%
True Bad	0,27%	4,29%

Examining now the results of real-time inference on the Raspberry Pi and Coral Edge TPU combination. As expected, due to the drop in accuracy resulting from conversion, the outcome is in general

less than optimal and worse compared to the performance of the full TensorFlow model. Even if the great majority of observations were correctly classified. A total of 32 images of bad parts where marked as good.

5. System for final implementation

The prototype has been tested and the working concept has now been proven. The next step is to define the components and specifications for a final implementation of the vision system.

The goal of this stage would be to convert our proof of concept equipment into common inline components that would work with the line Programmable Logic Controller (PLC) fulfilling the function of the Raspberry Pi. The base requirements are as follows:

- Compatible with full TensorFlow models.
- 4056 x 3040 pixels image size for half the workpiece carrier.
- Capture the whole workpiece carrier.
- Sense the presence of the Workpiece carrier.
- Stop the workpiece carrier.
- Link classifier data to workpiece carrier.
- Connect the output of the classifier to the SQL Database.

In the same way in our prototype we used an USB accelerator to remove the computing load from the Raspberry pi's CPU in our final system we will be leveraging a component from Siemens that was developed for the same application. The Siemens TM NPU (Neural processing unit), is a module compatible with the line PLC and allows for the deployment of deep learning models in an edge production environment. This module is capable of accessing TensorFlow machine learning models via SD card input [37].

For the image sensors two Keyence CA-H2100C models were selected, to be placed parallel to each other and be able to capture the whole area of the workpiece carrier, but maintain the initial requirements of resolution for joint. This model allows for 12 Megapixel images and is compatible with C mount lenses as per requirement [38],

To allow the motion control of the workpiece carrier, presence sensing and stopping for image acquisition Bosch Rexroth components were utilized. For the stopping of the carrier a Bosch VE 2/D-60 Stop Gate was selected, as it is compatible with the weight range of our carrier and allows for a dampened stop. For the presence sensor a M8 sensor was nominated as suitable to pair with our workpiece carrier and stop system [39][40].

Finally to fulfil the function of our OCR and tags system we would apply a Siemens SIMATIC RF240R capable of accessing the information present on a RF chip in the workpiece carrier and allowing for the linking of the soldering quality evaluation to the part information and SQL database [41].

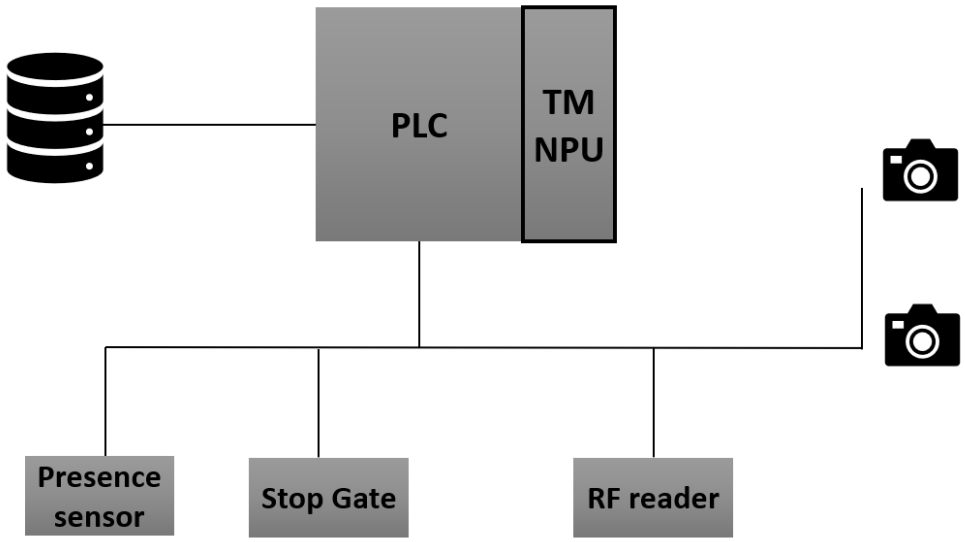


Figure 45 Final implementation architecture.

This diagram represents the working architecture of the vision system integration. The main line PLC is responsible for the transportation control and camera triggering, jointly with the process of separating the images into their individual joints. Feeding this data into the TM NPU module that will then run inference and the output will be stored in the SQL database as well as the evaluation will be stored in the workpiece carrier's RFID chip via the Siemens RF reader.

Having selected the components a way of implementing them inline was also designed.

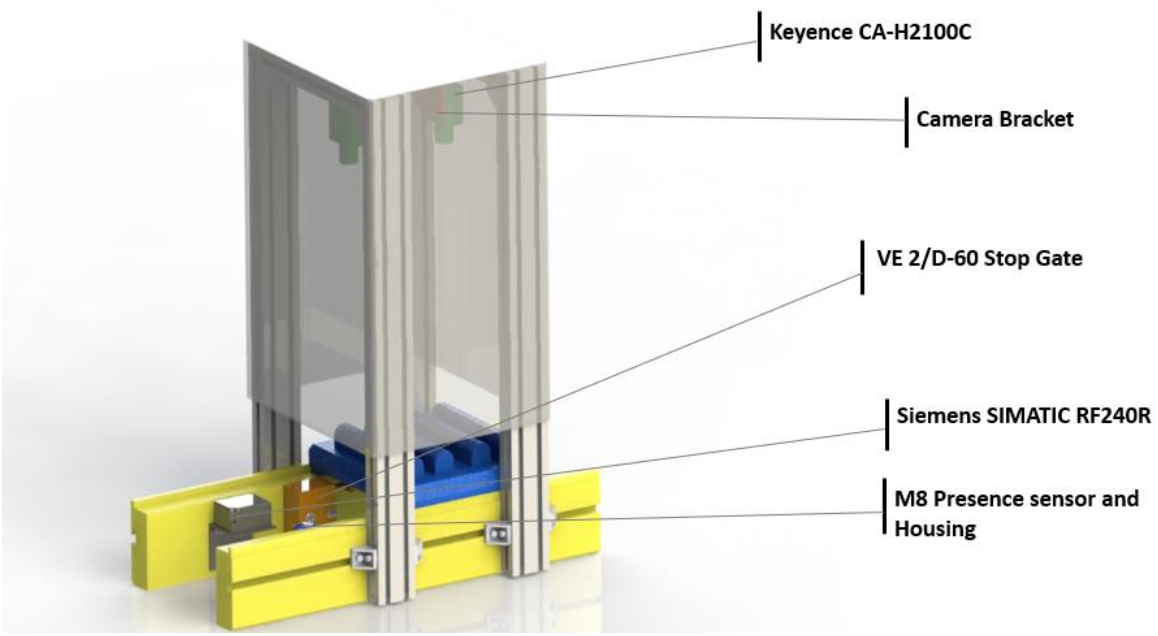


Figure 46 Vision system for final implementation.

The width and quantity of the aluminium extrusion profile was doubled as an effort to ensure long term implementation and reduce the effects of vibrations. An enclosure made of 6mm sheet aluminium is also included that will serve multiple functions:

- Provide stable lighting conditions using external light sources,
- Extra structural rigidity for the frame,
- Mounting point for the camera brackets.

A mounting bracket for the vision sensors was designed, allowing for fine adjustments to the positioning on the height of the sensor relative to the workpiece carrier. It would be mounted to the inside of the enclosure. It is composed of two parts one mounting plate for the camera that is joined using M3 screws and another that connects this mounting plate to the enclosure via M6 bolts.

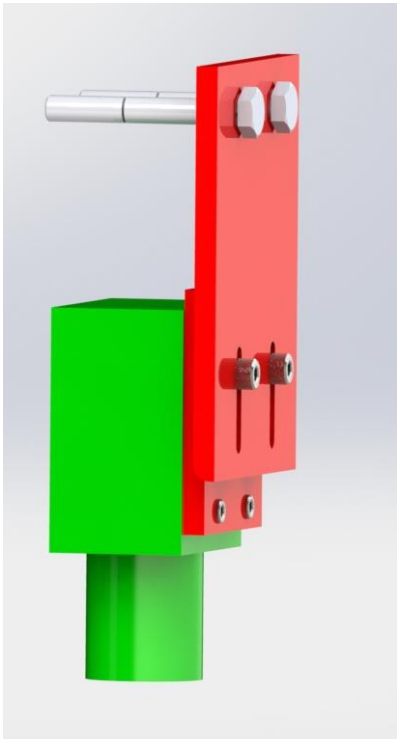


Figure 47 Vision sensor mounting bracket

6. Conclusions and Future work

This project centered on the development of a deep learning-based machine vision system for the identification of production soldering defects in electronics.

The pair growth in data generation and computing capabilities in technology have pushed for constant industrialization of processes and allowed the rise of machine learning solutions for problems that up until now seemed impossible. It is the case of this project where a system was developed for the quality analysis of soldering something that has such a difficult to identify behavior that is too demanding for traditional vision system and usually requires operators to be done.

In this dissertation a vision system capable of reliably acquiring production pictures in real-time. These pictures where then used to build a model that showed performance metrics very close to the results obtained with operators in a very short time and with a very limited data set. This model was also prototyped using edge solutions and off the shelf components.

Finally, the roadmap for full implementation of the vision system was defined.

6.1 Final goals analysis

The overall goals of the project where as follows:

1. Development of a set-up for inline image acquisition.
2. Leverage SQL database data for labelling these images or utilize offline labelling.
3. Creation of a structured and balanced dataset.
4. Defining and benchmarking algorithm architectures for optimal accuracy.
5. Create a minimum value prototype and inline testing of the vision system.
6. Developing final specifications for line implementation of the inspection system.

As we can see the main project goals where fulfilled however with some caveats. When it comes to the dataset, a balanced one as built however due to time constraints it is not as extensive as it would be ideal this could however be rectified with more time for image acquisition. The minimum value prototype was based on the capabilities of the Edge TPU however this equipment required a very significant downgrade of the model, it allowed for a proof of concept, but it is not the ideal solution.

6.2 Future work

As future work for the system implementation and scalability to other products comes:

- Testing the new suggested vision sensors.
- Migrating the architecture to the Siemens PLC environment.
- Further development of the dataset.
- Fabrication and deployment of the vision system.
- Creating a process planification of development and re-training for other products.

7. Index

[1] R. W. Messler, “Chapter 8 - Soldering: A Subset of Brazing,” R. W. B. T.-J. of M. and S. Messler, Ed. Burlington: Butterworth-Heinemann, 2004, pp. 389–446.

[2] “through-hole-vs-surface-mount @ blog.optimumdesign.com.” http://blog.optimumdesign.com/through-hole-vs-surface-mount (accessed Mar. 17, 2020).

[3] B. Illés, O. Krammer, and A. Géczy, “Chapter 1 - Introduction to surface-mount technology,” B. Illés, O. Krammer, and A. B. T.-R. S. Géczy, Eds. Elsevier, 2020, pp. 1–62.

[4] S. Revision, “Acceptability of Electronic Assemblies Standards Should :,” 2014. .

[5] “13 Common PCB Soldering Problems to Avoid @ www.seeedstudio.com.” https://www.seeedstudio.com/blog/2019/08/07/13-common-pcb-soldering-problems-to-avoid/ (accessed Mar. 20, 2020).

[6] “What is Solder bridging on a PCB? @ www.autodesk.com.” https://www.autodesk.com/products/eagle/blog/solder-bridging-pcb/ (accessed Mar. 30, 2020).

[7] “Wave soldering deffects @ www.epectec.com.” https://www.epectec.com/pcb/wave-soldering-defects/ (accessed Apr. 01, 2020).

[8] A. J. Anderson, *Foundations of Computer Technology*. CRC, 1994.

[9] M. Campbell, A. J. Hoane, and F. H. Hsu, “Deep Blue,” *Artif. Intell.*, vol. 134, no. 1–2, pp. 57–83, 2002, doi: 10.1016/S0004-3702(01)00129-1.

[10] “the-ai-paradigm-shift- @ becominghuman.ai.” https://becominghuman.ai/the-ai-paradigm-shift-53fa07ae3ab2 (accessed Apr. 02, 2020).

[11] C. Grosan and A. Abraham, *Machine Learning*, vol. 17. 2011.

[12] “ai-deep-learning-primer @ www.zerotosingularity.com.” https://www.zerotosingularity.com/deep-learning-day/ai-deep-learning-primer/ (accessed Apr. 10, 2020).

[13] “which-machine-learning-algorithm-to-use @ medium.com.” https://medium.com/@mjamilmoughal786/which-machine-learning-algorithm-to-use-bd9f7dc479c4 (accessed Apr. 10, 2020).

[14] H. Tran, “A Survey of Machine Learning and Data Mining Techniques used in Multimedia System,” no. 113, pp. 13–21, 2019, doi: 10.13140/RG.2.2.20395.49446/1.

[15] “MNIST Database @ yann.lecun.com.” http://yann.lecun.com/exdb/mnist/ (accessed Mar. 20, 2020).

[16] “the-artificial-neural-networks-handbook-part-1 @ medium.com.” https://medium.com/coinmonks/the-artificial-neural-networks-handbook-part-1-f9ceb0e376b4 (accessed Apr. 10, 2020).

[17] Y.-S. Park and S. Lek, “Chapter 7 - Artificial Neural Networks: Multilayer Perceptron for Ecological Modeling,” in *Ecological Model Types*, vol. 28, S. E. B. T.-D. in E. M. Jørgensen, Ed. Elsevier, 2016, pp. 123–140.

[18] “How the future of computing can make or break the AI revolution @ www.weforum.org.” https://www.weforum.org/agenda/2019/06/how-the-future-of-computing-can-make-or-break-the-ai-revolution/ (accessed Mar. 30, 2020).

[19] A. I. Maqueda, A. Loquercio, G. Gallego, N. Garcia, and D. Scaramuzza, “Event-Based Vision Meets Deep Learning on Steering Prediction for Self-Driving Cars,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, no. Dl, pp. 5419–5427, 2018, doi: 10.1109/CVPR.2018.00568.

[20] C. Biernacki, “HCP: A Flexible CNN Framework for Multi-Label Image Classificatio,” vol. 22, no. 7, pp. 719–725, 2000.

[21] “simple-image-classification-using-deep-learning-deep-learning-series @ medium.com.” https://medium.com/intro-to-artificial-intelligence/simple-image-classification-using-deep-learning-deep-learning-series-2-5e5b89e97926 (accessed Apr. 12, 2020).

[22] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 580–587, 2014, doi: 10.1109/CVPR.2014.81.

[23] H.-I. Suk, “Chapter 1 - An Introduction to Neural Networks and Deep Learning,” S. K. Zhou, H. Greenspan, and D. B. T.-D. L. for M. I. A. Shen, Eds. Academic Press, 2017, pp. 3–24.

[24] “convolutional networks @ cs231n.github.io.” https://cs231n.github.io/convolutional-networks/ (accessed Apr. 15, 2020).

[25] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, “Chapter 10 - Deep learning,” I. H. Witten, E. Frank, M. A. Hall, and C. J. B. T.-D. M. (Fourth E. Pal, Eds. Morgan Kaufmann, 2017, pp. 417–466.

[26] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[27] “loss_functions @ ml-cheatsheet.readthedocs.io.” https://ml-cheatsheet.readthedocs.io/en/latest/loss_functions.html#cross-entropy (accessed Mar. 10, 2020).

[28] V. Kotu and B. Deshpande, “Chapter 8 - Model Evaluation,” V. Kotu and B. B. T.-P. A. and D. M. Deshpande, Eds. Boston: Morgan Kaufmann, 2015, pp. 257–273.

[29] “wave-soldering @ www.ourpcb.com.” https://www.ourpcb.com/wave-soldering.html (accessed Apr. 01, 2020).

[30] “Index @ Www.Boschrexroth.Com.” https://www.boschrexroth.com/en/xc/home/index.

[31] “diagram @ global.sharp.” [Online]. Available: http://global.sharp/products/device/lineup/selection/opto/haca/diagram.html.

[32] “adafruid 4 channel adc breakouts @ learn.adafruit.com.” https://learn.adafruit.com/adafruit-4-channel-adc-breakouts/ (accessed May 10, 2020).

[33] R. Pi, “High Quality Camera,” no. April, 2020, [Online]. Available: https://static.raspberrypi.org/files/product-briefs/Raspberry_Pi_HQ_Camera_Product_Brief.pdf.

[34] “keras transfer learning @ keras.io.” https://keras.io/api/applications/ (accessed May 20, 2020).

[35] Coral, “USB Accelerator datasheet | Coral,” vol. 0, no. September, pp. 1–6, 2019, [Online]. Available: https://coral.withgoogle.com/docs/accelerator/datasheet/.

[36] “Coral edge tpu @ coral.ai.” https://coral.ai/docs/edgetpu/models-intro/.

[37] “artificial-intelligence @ new.siemens.com.” https://new.siemens.com/global/en/products/automation/systems/industrial/io-systems/artificial-intelligence.html (accessed Jun. 01, 2020).

[38] “keyence datasheet @ www.keyence.com.” [Online]. Available: https://www.keyence.com/products/vision/vision-sys/cv-x100/models/ca-h2100c/.

[39] “ve-2-d-60-stop-gate @ www.boschrexroth.com.” https://www.boschrexroth.com/en/xc/products/product-groups/assembly-technology/transfer-systems/ts-2plus-transfer-system/transportation-control/dampened-stop-gates/ve-2-d-60-ve-2-d-175-ve-2-d-200-stop-gates/ve-2-d-60-stop-gate.

[40] “m8-sensor-with-m8x1-connector @ www.boschrexroth.com.” [Online]. Available: https://www.boschrexroth.com/en/xc/products/product-groups/assembly-technology/transfer-systems/ts-2plus-transfer-system/transportation-control/sensors/m8-sensor-with-m8x1-connector.

[41] “6GT2821-4AC10 @ mall.industry.siemens.com.” https://mall.industry.siemens.com/mall/en/WW/Catalog/Product/6GT2821-4AC10 (accessed Jun. 20, 2020).

[42] “what is the reason for solder bridge of wave soldering @ www.raypcb.com.” [Online]. Available: https://www.raypcb.com/what-is-the-reason-for-solider-bridge-of-wave-soldering/.