# Comparative Study of Image Processing Algorithms to Detect Defects in Cast Components

Thesis by
[Nikith Muralidhar]

In Partial Fulfillment of the Requirements for the
Degree of
[Master of Science]


DUBLIN BUSINESS SCHOOL
Dublin, Ireland

[2022]
Defended [August 30, 2022]

# DECLARATION

I declare that this Applied Research Project that I have submitted to Dublin Business School for the award of MSc in Data Analytics is the result of my own investigations, except where otherwise stated, where it is clearly acknowledged by references. Furthermore, this work has not been submitted for any other degree.

Signed: Nikith Muralidhar

Student Number: 10586165

Date: 30/August/2022

# ACKNOWLEDGEMENTS

This thesis would not have been possible without the ongoing support, advice, and assistance from my supervisor Charles Ezenwa Nwankire. I express my sincere gratitude to him.

Respect is also due to my father, Mr Muralidhar M, mother, Mrs Jayalalitha, and the rest of my family for their steadfast financial support and ongoing encouragement that helped me get to where I am today. Other family members also deserve particular gratitude, especially my grandmother, for her constant encouragement.

Last, I am indebted to Mr Darshan SM and Ms Harshitha as they helped me find my footing as I started this process and their continued support and encouragement helped me throughout my research.

# ABSTRACT

In the manufacturing industry, the non-destructive evaluation (NDE) of components is crucial. These cast components are susceptible to blowholes and other anomalies. If such flaws are included in the components, the fatigue life will be harmed, which would almost certainly result in catastrophic accidents. Humans currently evaluate cast components by various methods. We propose an automatic approach for detecting faults in casts with the goal of producing a category that will eliminate the need for manual testing. The technique looks for defects in cast components, In the previous years, Image processing technology has advanced significantly. The method proposed utilises Convolutional Neural Networks (CNN) and Support Vector Classifiers (SVC's). This process classifies if the component has a defect or not. According to the hypothesis, human examiners may benefit from the approach because it reduces their workload.

**KEYWORDS:** Image classification, CNN, Deep learning, Machine Learning, Defect detection.

# TABLE OF CONTENTS

# LIST OF ILLUSTRATIONS

# LIST OF TABLES

# NOMENCLATURE

**Castings.** an object made by pouring molten metal or other material into a mould.

**Def front.** Folder in the dataset that has defective cast images.

**FN.** False Negative.

**FP.** False Positive.

**Impeller.** In a centrifugal pump, an impeller is a rotating iron or steel disc with vanes.

**NDE.** is a method of testing and analyzing used by the industry to check for faults, discontinuities, and characteristic changes in a material, component, structure, or system without harming the original part.

**Ok front.** Folder in the datset that contains images with no casting flaws.

**Overfitting.** Something which happens when a statistical model matches its training data exactly.

**Pixel.** a tiny spot of light, one of many from which an image is built, on a display screen.

**RGB.** Red, Green and Blue.

**TN.** True Negative.

**TP.** True Positive.

**Underfitting.** When your model is underfit, it produces accurate but initially erroneous predictions.

**Weld.** Metal components in some cases non metals are joined together by melting the surfaces with a blowpipe, electric arc, or other method, then pressing, hammering, etc. the pieces together.

*Chapter 1*

# INTRODUCTION

A significant aspect of all industrial processes is quality control. Manufacturers must boost their product yield while upholding strict quality control restrictions due to rising rivalry in the manufacturing sector. Intelligent visual inspection devices are becoming crucial in production lines to meet the rising demand for high-quality goods.(Ferguson et al., 2017).

The casting process frequently results in product imperfections that are damaging to the quality of the finished product. Timely screening of these flaws can enable the early identification of defective items during the manufacturing process, saving time and money. Wherever possible, it is preferable to automate quality control to ensure reliable and cost-effective inspection, particularly if defect detection needs to be done numerous times throughout the manufacturing process. Rapid inspection rates, higher quality standards, and the requirement for more quantitative product evaluation that is not constrained by human exhaustion are the main factors that promote the adoption of automated inspection systems (Ferguson et al., 2017).

Finished components go through NDE abbreviated as Non-Destructive Examination. NDE is a method of testing and analyzing used by the industry to check for faults, discontinuities, and characteristic changes in a material, component, structure, or system without harming the original part. The appropriate NDE techniques for castings rely on the potential types of flaws. Surface, subsurface, or interior flaws may exist, but it's important to thoroughly check all three places for discontinuities. *Table 1.1* are the appropriate NDE procedures for the various types of faults.

| **Defect** | **Inspection Method** |
|---|---|
| Surface | Visual testing with Liquid Penetration |
| Sub-Surface | Magnetic Particle testing |
| Internal | Ultrasound or Radiography |

Table 1.1: Types of defects and their inspection methods

## 1.1 Need for Visual Inspection

The manufacturing sector is under pressure to develop and use automated inspection methods for the screening of welding/cast components Castings may experience

discontinuities such as fractures or tearing, inclusions from chemical reactions or foreign material in the molten metal, or porosity brought on by shrinkage or gases, depending on the part's design and processing methods. Each of these flaws has the potential to create a fracture site and act as a stress raiser. This area is susceptible to pressure leading to casting fracture. Crucial structural elements include castings like steering knuckles, control arms, transmission mounts, impellers, and cross members(Chen, Miao, and Ming, 2011). These castings must be of high caliber and be reliable. Today's castings producers need visual and X-ray inspection systems that can operate at production line rates because they are open 24 hours a day, 7 days a week. With imaging technology dynamic analysis can be done, this basically eliminates false rejects. The technician would require minimal training while giving the technician the ability to handle multiple machines at once.

Visual inspection can be done on the following materials

1. Metals
2. Non Ferrous Metals
3. Glass
4. PVC
5. Ceramics
6. Food and cosmetic items

As its observed from the above points, all the industries use one or the other form visual inspection.The most advantageous aspect of visual inspection is that its low cost, requires very little equipment and the technician requires no prior experience to perform this kind of inspection in contrast to other forms of inspections.

### 1.2 Dataset

The dataset has 7348 images in total. These are grayscale images with a size of (300*300) pixels. The series and their respective descriptions have been mentioned in the table below.The images taken are of an cast impeller, these images were taken at PILOT TECHNOCAST, Shapar, Rajkot, India. The dataset is available on `https://www.kaggle.com/datasets/ravirajsinh45/real-life-industrial-dataset-of-casting-product`

Figure 1.1: Samples Images from the Dataset

| Series | Images | Description |
|---|---|---|
| Def_front | 6633 | Defective images |
| Ok_front | 715 | No defects |

Table 1.2: Description of the images

## 1.3 Objectives of the research

**Research Problem**

Manual examination is labor-intensive, subjective, time-consuming, uneven, and could be biased. For helping inspectors assess welds and casts in light of a set of criteria, an automatic inspection system is preferred. In a perfect world, this system would generate assessment results that are more reliable, consistent, effective, and objective than those of human inspectors(Dong, Taylor, and Cootes, 2021).

**Research Question**

The study aims to compare two algorithms CNN's and Support Vector Classifiers

**Hypothesis**

Two image processing algorithms are used in this article to find flaws in cast components.

**Expected Outcomes**

Two image processing algorithms are used in this article to find anomalies in cast components. This method is aimed to locate the flaws in the cast component using image processing and machine learning techniques.The system has the ability to lighten the workload of human inspectors. We use the public cast dataset from Kaggle for our studies. To determine which of the two image categorization models performs the best, we compare them. The proposed method is supported by some encouraging experimental findings, demonstrating its potential for usage in the

industry.

*Chapter 2*

# LITERATURE REVIEW

Nondestructive testing is commonly performed on welded and cast products since weld and cast failure can be fatal. Testing is required for things like pressure tanks, power plants, and aviation equipment. The size and kind of the defect must be determined when radiographic inspection uncovers flaws in the materials and welds. Operators use qualitative and quantitative analysis to make judgments in nondestructive testing. Currently, radiography detection relies on experts to interpret images, which complicates operations and leads to ambiguity. The evaluation of the cast picture quality using computers has greatly advanced radiographic inspection (Zhang, Xu, and Ge, 2004).

The manufacturing sector is runs 24 hours a day, 7 days a week, today's castings manufacturers need inspection systems that can function at production line rates. We attempt to develop a prototype of fully automatic inspection equipment that can be delivered at the manufacturing line as a solution to the difficulty. Finding an automatic system for detecting and recognizing cast and welding flaws is therefore crucial if defect detection is to be efficient, uniform, and intelligent(Feng et al., 2020).

To ensure the quality of the product and the effectiveness of the manufacturing process, surface defect inspection, which looks at the product's surface, is a crucial stage in quality control. Defective products must be identified quickly and taken off the manufacturing line; otherwise, they will negatively impact the assembly line that comes after them and reduce the level of overall quality. The challenges with defect identification by eye-checking include low sampling rates, low accuracy and efficiency, high labor intensity, and significant sensitivity to human experience, physical condition, and emotional state. Machine vision-based automated flaw identification is becoming increasingly popular because it can greatly outweigh these drawbacks. Machine vision-based defect detection systems take surface photos of the product and use image analysis to find the flaws(He and Q. Liu, 2020a). Machine vision-based surface detection technology has been extensively employed in industrial detection, particularly for the detection of metal surface flaws. However, the complicated metal surface's intricate texture and structure make it simple to produce

uneven brightness in the image of the metal surface. Additionally, diverse surface flaws like scratches, stains, and pits exist due to the various processing technologies. The features of these defects vary as well, which makes it extremely challenging to find complicated metal surface flaws. It is crucial to accurately and effectively evaluate complicated metal surface flaws(Zhou et al., 2020). In terms of precision casting, casts' internal structures are typically non-uniform, their elongation and density are low, they frequently have air holes, and faults and problems with derived structures are likely to appear (Lin et al., 2021).

- Crack: Usually occurring at the intersection of the thickest and thinnest region of the cast, the crack is due to high tensile stress and (extra) shearing stress throughout the process.
- Chill hardening: is a condition that is brought on by low temperatures, too-rapid deformation, and too-rapid cooling during deformation. Severe chill hardening can lead to fracture.
- Burr crack: During trimming, a burr fracture developed on the separating surface.

Many deep convolutional network-based approaches for defect identification have been proposed. Deep learning-based methods can be loosely categorized into three types based on how they approach the defect detection problem: pure image classification methods, object detection-based methods, and pixel-level segmentation methods.

The input image is divided into overlapping blocks by pure classification algorithms, which then classify the block images into various categories. A block is classified as defective if it has that many defective pixels or more. The deep learning model's input size determines the block size, which is commonly 256, 128 or 32. For instance, some authors classify the input pavement 256x256 pictures into faulty and non-defective using MatConvNet. Li and Zhao alter GoogleNet's structure in order to perform the categorization. Identical techniques include It's easy to classify things in binary. The classification of picture blocks also employs multi-class classification. One CNN model, for instance, is intended to categorize faulty images of a resistance welding site, a panel of glasses, and a wafer images(He and Q. Liu, 2020b).

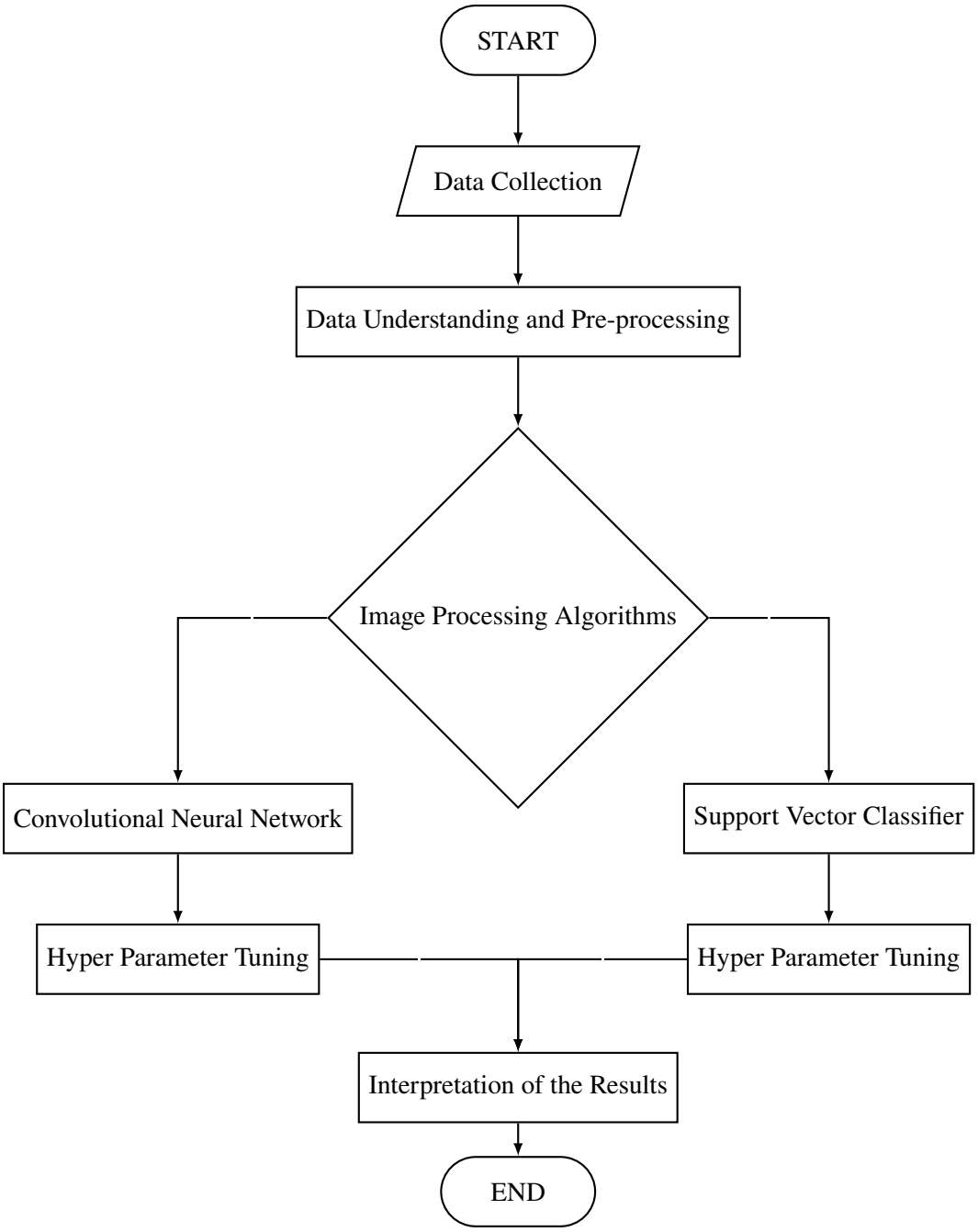
An essential task in computer vision is object detection, which can also be used to solve the defect detection problem. Its objective is to identify the object's type and locate it with a bounding box. To increase accuracy and efficiency, many deep CNN models have also been proposed, including YOLO, SSD, and Faster R-CNN. Faster R-CNN is used by Suh and Cha to identify damages to civil infrastructure. In order to accelerate feature extraction, Young-Jin et al. modify the Faster R-CNN using ZF-net. Similar to this, in order to detect railway subgrade defects, the creators of Faster R-CNN adapt it. In order to find flaws on the sealing surface of containers, such as breaches, dents, burrs, and abrasions, Yiting et al. use the MobileNet-SSD framework(He and Q. Liu, 2020b).

Image comparison-based methods are within the field of machine vision-based fault detection techniques. The approaches that rely on image comparison to find surface flaws compare the differences between the reference image and the test image.

Based on (K. Liu et al., 2020), developed a similarity evaluation index to further increase the precision of the strip steel surface defect detection after extracting the structural and texture components of the test image using an image decomposition algorithm based on the generated reference image and total variation. These techniques, however, fail to take into account the reference picture and test image mismatch problem brought on by improper positioning and visual system variations. In the results of the picture comparison, this problem may lead to significant mistakes. Studies have highlighted the importance of picture registration in a variety of contexts, including medical and remote sensing images.

To this, I propose an image algorithm comparison technique that is able to compare two models and provide with with the best performing image processing model. The methodology of the process can be visualised in the flow chart below.

The flow chart below represents the simplified process flow diagram from the start to the end of the thesis


START
Data Collection
Data Understanding and Pre-processing
Image Processing Algorithms
Convolutional Neural Network
Support Vector Classifier
Hyper Parameter Tuning
Hyper Parameter Tuning
Interpretation of the Results
END

*Chapter 3*

# DATA FLOW CYCLE

## 3.1 CRISP DM

The initial version of the CRoss-Industry Standard Process for data processing was published in 1999. It is remains the most widely used analytic methodology today, according to various surveys. The number of data processing projects available has significantly expanded due to the ubiquitous availability of electronic devices and sensors. Thanks to a plethora of fresh applications and business models, data has been monetized in more ways during the previous two decades. Under the cover of knowledge research, the area of extracting value from data has expanded greatly in size and complexity while simultaneously becoming much more exploratory(Martínez-Plumed et al., 2021). Teams working on data science projects may find it difficult to adhere rigorously to project methodologies. Agile methodologies should aid and can improve process models like CRISPDM and the Analytics Life Cycle. Although process models have been around for a while, they are not well established in modern data science efforts(Schröer, Kruse, and Gómez, 2021). The subject of getting the most value out of data has grown significantly in volume and complexity under the data science umbrella, but it has also become considerably more exploratory. In the latter, data-driven and knowledge-driven stages interact as opposed to the typical data mining process, which begins with stated business goals that translate into a clear data mining task, which ultimately converts "data to knowledge." In other words, the techniques for getting value out of data have changed along with the nature of data. A data mining process model that appears to be independent of industry is called CRISP-DM. There are six iterative phases from business knowledge to implementation as seen in *Figure 3.1*.

*Figure 3.1* succinctly summarizes the main concept, tasks, and outcomes of these stages based solely on the CRISP-DM user guide.

**Step 1. Business Understanding**

Understanding the project's goals and requirements is the main focus of the business understanding phase. With the exception of task number three, the other three tasks in this phase are fundamental project management activities that apply to most projects.

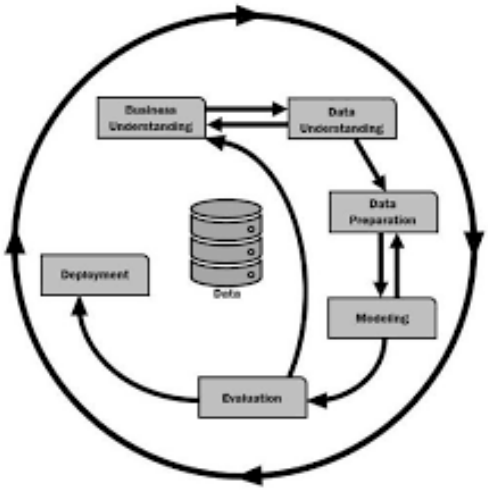



Figure 3.1: CRISP DM Methodology (Birjandi and Alemi, 2010)

- **Objectives of the business**: In our set case, the goal of the business is to use the best image processing algorithm in the manufacturing industry to effectively identify
- **Accessing**: Establish the resources that are available, the project's needs, the risks and contingencies, and perform a cost-benefit analysis.
- **Data mining**: What success means in terms of technological data mining in addition to the business objectives.
- **Plan of the Project**: For each project phase, decide on technology and tools and create specific plans.

**Step 2. Understanding the Data**

The phase of data understanding comes next. In addition to strengthening the basis of business understanding, it concentrates attention on finding, gathering, and analyzing data sets that might assist you in achieving project objectives. Four tasks are included in this phase:

- **Acquiring data**: Collecting the needed data and utilising the right tools to analyse the data.
- **Data understanding**: Analyze the data and note its surface characteristics, such as data type, record count, or field identifiers.

- **Data exploration**: Closer investigation of the data It can be queried, visualized, and the relationships between the data are found.
- **Quality**: How pure or impure is the data? Keep track of any quality problems.

**Step 3. Preparing the Data**

One of the most crucial and time-consuming parts of data mining is data preparation. In actuality, it's estimated that 50–70% of a project's time and effort is often spent on data preparation. This expense can be reduced by investing enough time and energy in the first phases of business and data knowledge, but one would still need to put in a lot of work to get the data ready for mining.

**Step 4. Modelling the Data**

Typically, modeling is done across a number of iterations. Data miners typically run numerous models with the default settings before fine-tuning them or returning to the data preparation stage for any changes needed by their preferred model. A single model and a single execution are rarely sufficient to provide a satisfactory response to a data mining challenge for an organization. This is what makes data mining so intriguing: there are numerous angles you may take when examining a particular issue.

**Step 5. Evaluation**

Ensure that the models created during the modeling phase meet the technical and practical requirements of the data mining success criteria that you previously specified. However, before moving further, you should assess the outcomes of your efforts using the project's initial set of business success criteria. This is essential to ensuring that your company can apply the findings you've generated. Data mining results come in two flavors:

- The CRISP-DM models that were ultimately chosen in the first round.
- Any findings or inferences derived from the data mining process and the models themselves. They are referred to as discoveries.

**Step 6. Deployment**

The process of applying your fresh insights to improve your company is known as deployment. Deployment can also refer to using data mining insights to bring about change inside your organization. Although they might not be formally included

into your information systems, these findings will unquestionably be helpful for planning and making marketing-related decisions. The CRISP-DM deployment phase typically consists of two different kinds of activities:

1. Coordinating and observing the implementation of outcomes.
2. Finishing up assignments including writing a final report and performing a project evaluation.

*Chapter 4*

# INTUITION BEHIND IMAGE PROCESSING

## 4.1 Gentle introduction to CNN

### Understanding an Image

Before initiating image processing, it is important to comprehend what an image actually is. Based on the quantity of pixels, an image's dimensions (height and breadth) serve as a representation. For instance, if an image is 500 × 500 (width x height), then 2,50,000 pixels make up the entire image. This pixel is a location on the image that assumes a certain hue, level of transparency, or color. Typically, it appears as one of the following:

- Gray-scale: A pixel in gray-scale has an integer value between 0 and 255. (0 is completely black and 255 is completely white
- RGB: The pixel is made up of three integers in the range of 0 to 255. (the integers represent the intensity of red, green, and blue).
- RGBA: is an expansion of RGB that includes an additional alpha field that symbolizes the opacity of the image. A pixel can be visualised in the image below *Figure 4.1*

Each pixel of an image must undergo a fixed series of operations during image processing. The initial series of actions are carried out pixel-by-pixel by the image

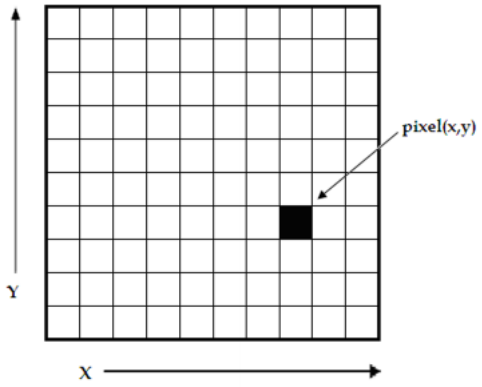



Figure 4.1: Elements of an Image (Lyra, Ploussi, and Georgantzoglou, 2011)

processor on the image. The second operation will start once this is finished in full, and so forth. Any pixel in the image can be used to determine the output value of these procedures. A mathematical expression for a picture is a ***f(x,y)*** which is a 2D signal that fluctuates throughout the spatial coordinates *x* and *y*. The components of an image that include information about color and intensity are called pixels. *X*, *Y*, and *Z* are transformed into spatial coordinates in 3D representations of images. A matrix-shaped arrangement of pixels is used. It is referred to as an RGB image. A mathematical expression for a picture is a *f*, which is a 2D signal that fluctuates throughout the spatial coordinates *x* and *y*.

**Image Classification**

Giving a complete image a label or class is the task of image classification. It is assumed that each image will only have one class. The class to which an image belongs is predicted by image classification models when they receive an image as input. Thematic maps can be made using the output raster from image categorization. There are two ways of categorization: supervised and unsupervised, depending on how the analyst and the machine interact during classification. They both have the option of being pixel-based or object-based.

It is unclear what occurs between the image and the output, and we will go into more detail in later posts. However, most networks deconstruct the image into abstract forms and colors that are then utilized to generate an assumption about the image's content.

**Applications of Image Classification**

- Visual Search
- Detecting logos
- Security for facial recognition
- Industries for anomaly detection

There are typically two types of classification

1. Multi-class: Each image may have more than one label in a classification task known as "multi-label classification," and some images may carry all the labels at once.

2. Binary: The most frequent classification challenge in supervised image classification is single-label classification.

Since our problem statement falls on the supervised learning. We do supervised learning by giving our model signals of training failures, such as the fact that you classified this image as a defect when it actually belonged to a non defect category. This is the situation in which we have tagged dataset with picture and class pairs.

Gradient descent is used to minimize a function called loss when training neural networks. Binary Cross-Entropy (BCE) in binary classification, where $M$ is the number of classes, can be calculated as follows:

$$BCE = -(y \log(p) + (1 - y) \log(1 - p)) \tag{4.1}$$

- M = No of classes
- log = natural logarithm
- y = binary indicator (Defect or Not)
- p = probability

*Chapter 5*

# METHODOLOGY

## 5.1 Description of the Dataset

This dataset contains products from the casting industry. A liquid substance is often poured into a mold that has a hollow chamber in the desired shape during the casting process, and the material is then allowed to solidify. Casting flaws are the reason for collecting this data. An undesirable imperfection in the metal casting process is known as a casting fault. There are many different kinds of casting flaws, including blow holes, pinholes, burrs, shrinkage flaws, flaws in the mold material, flaws in the metal-pouring process, and metallurgical flaws. The casting business does not want defects. Every industry has a quality inspection section with the purpose of eliminating this substandard product. The fact that this examination procedure is manual, however, is the fundamental issue.

Due to human error, this procedure takes a long time and is not entirely accurate. This can be as a result of the entire order being rejected. The result is a significant loss for the corporation. All of these images show the submersible pump impeller. The collection has 7348 images in total. These are all grayscale photos with a size of (300*300) pixels. All of the photos have already had augmentation. Also included were 512x512 grayscale photos. There is no augmentation in this data collection. 519 ok front and 781 def front impeller photos are included in this. The two different types of categories are defective and fine products these can be seen in Figure 5.1.

We have previously divided the data used for training and testing the classification model into two files. def front and ok front subfolders are included in both the train and test folders.

1. Train: def front contains 3758 and ok front contains 2875 images
2. Test: def front contains 453 and ok front contains 262 images

## 5.2 Splitting the data

Even while extracting meaning from raw data is an art in and of itself and requires strong feature engineering skills and subject understanding, great data is meaningless

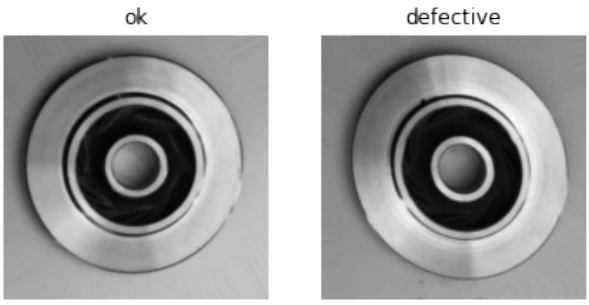



Figure 5.1: Sample images from the dataset

if it is not used efficiently (in exceptional circumstances). The fundamental problem that ML/DL practitioners go with is how to divide the data for training and testing. Even if it seems simple at first glance, its complexity cannot be established unless it is thoroughly investigated. On the model's output, inaccurate training and testing sets may have unintended effects. It may cause the data to be either over- or under-fitted, and our model might end up yielding inaccurate results.

A train set, a test set, and a holdout cross-validation or development (dev) set should be created from the data, respectively. Let's first quickly go over the purpose of these sets and the sorts of information they should include.

1. **Train Set**: The train set would include the information that would be entered into the model. Simply said, this data would provide knowledge to our model. By leveraging the examples in this data, a regression model, for instance, may find gradients to reduce the cost function. Then, these gradients will be used to save expenses and enhance data prediction. From *Figure 5.2* we can see the split of "ok" and "not ok" images

2. **Validation Set**: The development set is used to validate the trained model. This setting is the most important since it serves as the basis for our model evaluation. The model is overfitting and has a high variance if there is a sizable difference between the error on the training set and the error on the development set.

3. **Test Set**: The test set contains the data that were used to evaluate the trained and approved model. It shows how well our entire model works and how likely it is that it will predict an irrational event. To evaluate the efficacy of our method, a variety of assessment criteria (such as precision, recall, accuracy, etc.) may be used.

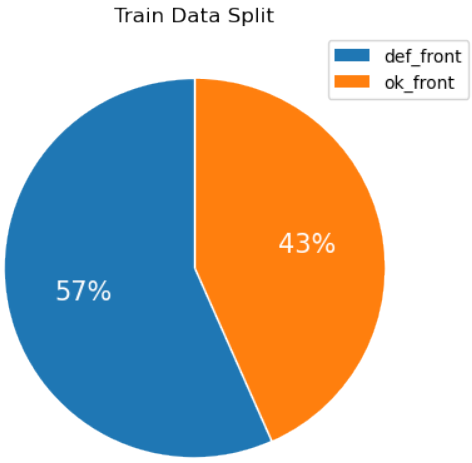



Figure 5.2: Split percentage in the train folder

### 5.3 Image Augmentation

Image augmentation is a technique that creates several modified versions of the same image by applying various changes to the original image. Nevertheless, each duplicate varies from the others in a few key ways, depending on the augmentation techniques you use, such as shifting, rotating, flipping, etc. These little adjustments to the original image just offer a different angle for catching the object in real life; they do not alter the target class. So, we frequently use it to create deep learning models. These picture augmentation methods not only increase the quantity of your dataset but also provide a degree of variance, which helps your model generalize more effectively to unobserved data. Additionally, when the model is trained on fresh, slightly modified photos, it becomes more reliable.

A quick and simple approach to enhance your photographs is with the Keras ImageDataGenerator class. It offers a variety of augmentation methods, including standardization, rotation, shifts, flips, brightness changes, and many others. The Keras ImageDataGenerator class' primary advantage is that it is intended to give real-time data augmentation. Meaning that while your model is still being trained, it is instantly producing enhanced pictures. Every epoch, the model is given fresh versions of the pictures thanks to the ImageDataGenerator class. However, it does not include it in the original corpus of photographs; it simply delivers the altered images. The model would then view the actual photos more than once, overfitting our model without a doubt. The fact that ImageDataGenerator uses less RAM is another benefit. This is because we load all the photos at once when we don't utilize

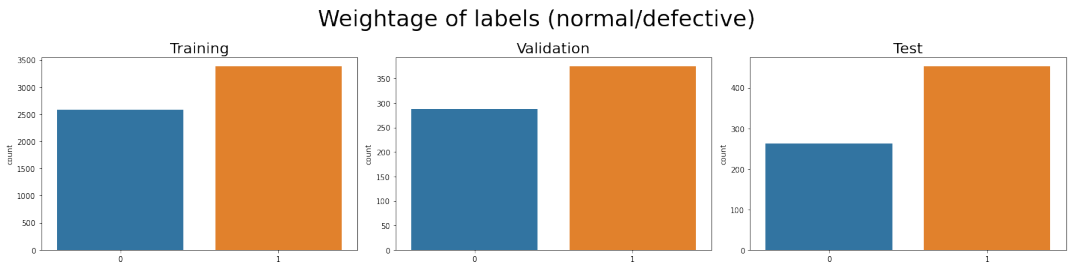



Figure 5.3: Labels count after augmentation

this class. However, while utilizing it, we load the photos in chunks, which uses much less RAM. This method is used only for CNN as it works well only for deep learning algorithms.

Several crucial variables for this strategy include the ones listed below:

- **Directory**: This is the location of the parent folder, which includes a subdirectory containing all of the pictures for the various classes.
- **Target_size**: The input image's size.
- **Color_mode**: If the photographs are black and white, set color mode to grayscale; otherwise, set it to RGB for colorful images.
- **Batch_size**: The size of the data batches.
- **Class_mode**: Set to categorical for 2-D one-hot encoded labels and binary for 1-D binary labels.
- **Seed**: set to replicate the outcome is a seed.

## 5.4 Convolutional Neural Networks [CNN]

Over the past ten years, Convolutional Neural Networks have produced groundbreaking findings in a range of pattern recognition-related domains, including speech recognition and image processing. The reduction of ANN's parameter count is CNNs' most advantageous feature. This accomplishment has inspired academics and developers to use larger models to handle challenging problems, which was not achievable with traditional ANNs. The most crucial presumption regarding problems that CNN solves is that there shouldn't be any spatially dependent features. In other words, we don't need to focus on where the faces are in the photographs

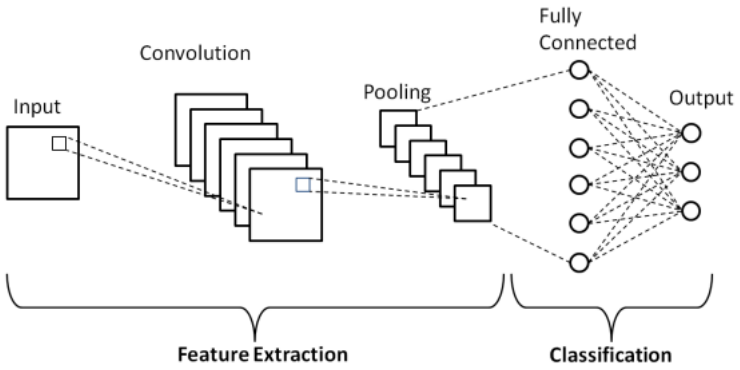



Figure 5.4: Schematic flow of CNN architecture (Phung and Rhee, 2019)

while using a face detection tool, for instance. The schematic flow for the CNN algorithm can be seen in *figure 4.2*

A classifier in machine learning gives a data point a class label. For instance, an image classifier can identify the objects in an image by producing a class label (such as bird or plane). The classifier known as a convolutional neural network, or CNN for short, is particularly effective at handling this challenge. A neural network, or CNN, is an algorithm that finds patterns in data. In general, neural networks are made up of layers of neurons, each of which has its own learnable biases and weights. Let's dissect a CNN into its fundamental components.

- **Tensor**: A tensor can be thought of as an n-dimensional matrix. Tensors in the CNN described above, with the exception of the output layer, will be three-dimensional.
- **Neuron**: One way to conceptualize a neuron is as a system that processes several inputs and produces a single output. The red to blue activation maps shown above are the outputs of neurons.
- **Layer**: A layer is a group of neurons that perform the same operation and share the same hyperparameters.
- **Kernel**: Although particular to each neuron, the kernel weights and biases are adjusted during training to allow the classifier to adjust to the issue and dataset presented.

## 5.5 Understanding Layers of CNN

### Input Layer

The CNN's input image is represented by the input layer, which is the layer on the left. Due to the fact that we use RGB images as input, the input layer includes three channels that are, respectively, the red, green, and blue channels, in contrast if the images are in grayscale we would see only one channel.

### Convolutional Layer

The learnt kernels (weights), which extract features that separate various images from one another, are present in the convolutional layers, which form the basis of CNN. The output of the appropriate neuron from the previous layer and a distinct kernel are combined in an element wise dot product by the convolutional neuron. As there are distinct kernels, this will provide an equal number of intermediate results. The total of all the intermediate findings plus the learnt bias yields the convolutional neuron. The network architecture designers specified the size of these kernels as a hyper-parameter. We must conduct an elementwise dot product using the output of the previous layer and the distinctive kernel that the network learned in order to obtain the output of the convolutional neuron (activation map).

The final step is to do an elementwise sum that includes all 3 intermediate outcomes as well as the bias the network has learned. The activation map for the topmost neuron in the first convolutional layer will then be shown on the interface above as a 2-dimensional tensor. To create the activation map for each neuron, the same procedure must be used.

## 5.6 Understanding Hyper-parameters

1. **Padding**: When the kernel goes beyond the activation map, padding is frequently required. Padding can assist maintain the spatial scale of the input, which enables an architecture designer to create deeper, more performing networks. Padding conserves data at the borders of activation maps, leading to superior performance. Although there are other padding methods, zero-padding is the most widely utilized technique due to its effectiveness, simplicity, and computational efficiency. The method entails symmetrically adding zeros to an input's edges. Many effective CNNs, including AlexNet, use this strategy.

2. **Kernel**: which is also sometimes referred to as the filter size. The picture

classification task is significantly impacted by the choice of this hyperparameter. For instance, lower kernel sizes are able to extract from the input a substantially greater amount of data containing extremely local features. A smaller kernel size also results in a lesser reduction in layer dimensions, as seen on the visualization above, allowing for a deeper architecture. On the other hand, a large kernel size extracts less data, which causes a rapid reduction in layer dimensions and frequently results in worse performance. Larger features can be extracted more effectively from large kernels. In the end, the task and dataset you employ will determine the optimum kernel size, but in general, lower kernel sizes result in greater performance for the picture classification work since an architectural designer can stack more and more layers together to learn more and more complicated characteristics.

3. **Stride**: The stride value specifies how many pixels the kernel should move over each time. For instance, Tiny VGG employs a stride of 1 for its convolutional layers as an example. This implies that the dot product is done on a 3x3 window of the input to generate an output value, then is shifted to the right by one pixel for every subsequent operation. Like kernel size, stride has an effect on a CNN. More features are learned as the stride is shortened since more data is extracted, which also results in larger output layers. On the other hand, increasing stride results in lower output layer dimensions and a more constrained feature extraction.

### 5.7 Activation Function

To determine whether a neuron should be activated or not, the activation function computes a weighted sum and then adds bias to it. The output of a neuron is meant to become non-linear through the activation function. A neural network is just a linear regression model in the absence of an activation function. The activation function alters the input non-linearly, enabling it to learn and carry out more difficult tasks.

1. **Sigmoid Function**: A unique variation of the logistic function, the sigmoid function is typically represented by the symbol $\sigma$ (x)

   Formula for Sigmoid Function

   $$S(x) = \frac{1}{1 + e^{-x}} \tag{5.1}$$

   A mathematical function with a distinctive S-shaped curve is called a sigmoid function. Numerous sigmoid functions that are frequently used include the

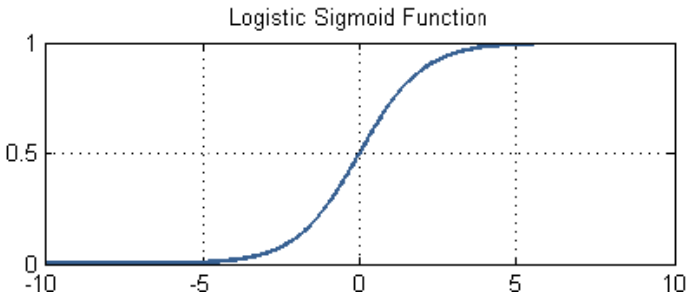



Figure 5.5: Graph of the Sigmoid Function (Chang et al., 2012)

logistic function, the hyperbolic tangent, and the arc-tangent. Numerous machine learning applications that need the conversion of a real number to a probability can benefit from the use of sigmoid functions. A probability score, which can be simpler to work with and interpret, can be created from the output of a machine learning model by adding a sigmoid function as the final layer. The sigmoid function lies between 0 and 1 the graph for the same can be seen in *Figure: 4.3*

**Advantages**

- In contrast to step function, it will provide an analog activation.
- When compared to a linear function, the output of the activation function will always fall between (0,1) and (-inf, inf). So, we've confined our activations to a certain range.
- It works well as a classifier.

**Disadvantages**

- The Y values often respond to changes in X relatively little at either end of the sigmoid function.
- It results in the issue of "vanishing slopes".
- Gradients are sated and killed by sigmoids.

2. **ReLu**: Given that they are so precise, neural networks are extremely common in current technology. The highest performing CNNs currently have an incredible number of layers that can continuously learn new features. These ground-breaking CNNs' non-linearity helps explain in part why they can achieve such astounding accuracy. ReLU injects the model with much-needed nonlinearity. As a result, the output cannot be expressed as a linear

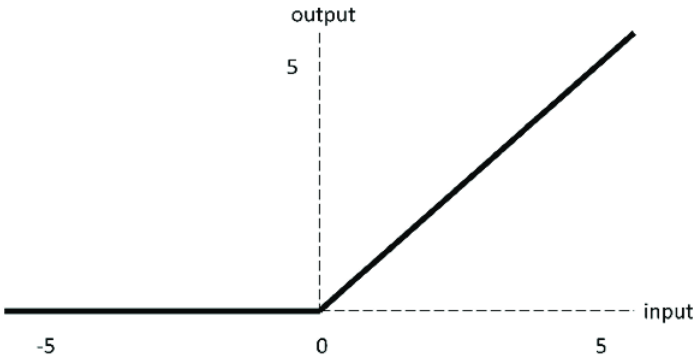



Figure 5.6: Graph of the ReLu Function (H. Sultan, Salem, and Al-Atabany, 2019)

combination of the inputs since non-linearity is required to establish non-linear decision boundaries. Deep CNN designs would become a single, equivalent convolutional layer without a non-linear activation function, which would not perform as well. Because CNNs employing ReLU train more quickly than CNNs using other non-linear activation functions, such as Sigmoid, the ReLU activation function is especially employed as a non-linear activation function. The plot for the ReLu function can be visualised in *Figure: 4.4*

$$f(x) = max(0, x) \tag{5.2}$$

**Pooling Layers**

In various CNN architectures, there are many different types of pooling layers, but they are all designed to gradually shrink the spatial scope of the network, which lowers its parameters and overall computation. Choosing a kernel size and a stride length throughout architecture design is necessary for the Max-Pooling process. Once chosen, the operation only chooses the largest value at each kernel slice from the input, producing a value for the output. The kernel is then slid over the input with the set stride. A convolutional neural network's convolutional layers repeatedly apply learnt filters to input images to produce feature maps that list the features present in the input.

Convolutional layers are particularly effective, and stacking them in deep models enables the input-close layers to learn low-level features (such as lines) while the input-deeper layers learn higher-order or more abstract features, such as forms or particular objects. The fact that convolutional layer feature maps preserve the exact

location of input features is one of their limitations. This implies that slight changes in the feature's location in the input image will produce a different feature map. By altering the convolution's stride over the image, convolutional layers can be used to do down sampling. Using a pooling layer is a more reliable and typical strategy. After the convolutional layer, a new layer called a pooling layer is introduced. For example, the layers in a model might appear like follows after a nonlinearity (such as ReLU) has been applied to the feature maps produced by a convolutional layer:

- Input Image
- Convolutional Layer
- Nonlinearlity
- Pooling Layer

A frequent pattern for layer ordering in a convolutional neural network that may be repeated once or more times in a given model is the insertion of a pooling layer following the convolutional layer. Each feature map is processed separately by the pooling layer, which produces a new set of the same number of pooled feature maps. In order to apply a pooling operation on feature maps, similar to a filter, pooling is done. The size of the pooling operation or filter, which is typically 22 pixels applied with a stride of 2 pixels, is smaller than the size of the feature map. This means that each feature map will always be compressed by a factor of 2, i.e., each dimension is cut in half, making each feature map only contain a quarter as many pixels or values. For instance, a feature map of 66 (36 pixels) that has a pooling layer applied to it will produce an output pooled feature map of 33. (9 pixels). Instead of being taught, the pooling operation is defined. In the pooling operation, there are two often utilized functions:

1. **Average Pooling**: For each patch on the feature map, determine the average value.
2. **Max Pooling**: Determine the highest value for each feature map patch.

A condensed version of the input features is produced by employing a pooling layer and downsampling or pooling feature maps. They are helpful because a pooled feature map with a feature in the same place is produced when a feature's location

in the input, as identified by the convolutional layer, changes little. The ability that pooling adds is referred to as the model's invariance to local translation. Determine the highest value for each feature map patch.

**Flattening Layer**

The flattening stage, which is a necessary part of creating a convolutional neural network, is delightfully easy. It entails converting the pooled feature map produced during the pooling process into a one-dimensional vector. We convert the pooled feature map into a one-dimensional vector because an artificial neural network will now use this vector as input. Or to put it another way, this vector will now serve as the input layer of a convolutional neural network that is chained onto an artificial neural network.

The process of CNN is as follows:

- An image should be represented as a tensor to feed to the network. The resulting tensor has a 15 x 20 x 3 shape(there are 15 rows, 20 columns and 3 color channels). the image goes through a convolutional layer. A filter (also called a "kernel"), usually size 3x3 (commonly used), 5x5 or 7x7, runs through the pixels of an image computing the dot product at each step in the convolution operation and storing the results in a matrix called "feature map".
- The feature map is the structure responsible for recognizing the various patterns (features) related to the image.
- The feature map coming from the filter is the sum of the feature maps collective to each channel. Several filters, each corresponding to a feature map.
- Each filter of a layer is a parameter of the network, it is adjusted with the weights to result in the optimal learning. Multiple feature maps are fed to a new convolutional layer.
- Strides are used to make output maps smaller than the input ones.
- The pooling layers reduce the number of network parameters, attenuating overfitting and computational cost.
- First few layers perform feature extraction and pattern identification in images. And rest works on classification.

Long-running processes can use the fault tolerance strategy of application checkpointing. In this method, in the event of a system failure, a snapshot of the system's state is obtained. Not everything is lost if there is a problem. The checkpoint can be utilized directly or as the beginning of a new run that continues from where the previous one left off. The model weights serve as the checkpoint during the training of deep learning models. These weights can be used as the starting point for continued training or for making forecasts as-is. Through a callback API, the Keras library offers checkpointing functionality. You may specify where to checkpoint the model weights, how to name the file, and under what conditions to checkpoint the model using the ModelCheckpoint callback class. You may enter the API the statistic to track such loss or accuracy on the training or validation dataset. You can decide whether to enhance the score by maximizing or minimizing it. Finally, variables like the epoch or metric can be included in the filename you use to store the weights. The training procedure can then pass the ModelCheckpoint when invoking the *fit()* method on the model.

The amount of training epochs to utilize might be a challenge while developing neural networks. Overfitting of the training dataset can occur when there are too many epochs, and underfitting can occur when there are too few. Early stopping is a technique that enables you to define an arbitrary large number of training epochs and terminate training as soon as the model performance on a hold out validation dataset stops increasing. Early Stopping and checkpoint saving are implemented in our model.

### 5.8 Support Vector Classifier

Support-vector networks and support-vector machines are supervised learning models and accompanying learning algorithms that analyze data for regression and classification. At ATT Bell Laboratories, Vladimir Vapnik and coworkers created One of the most trustworthy prediction methods, the SVM, which is based on statistical learning frameworks or the VC theory proposed by Vapnik and Chervonenkis. An SVM training technique develops a model that, when given a set of training examples, divides new samples into one of two categories, resulting in a non-probabilistic binary linear classifier. SVM maximizes the distance between the two categories by assigning training samples to spatial coordinates (Cortes and Vapnik, 1995).

Support vector machines (SVMs) are effective but flexible supervised machine learning methods used for both classification and regression. They are, nonetheless,

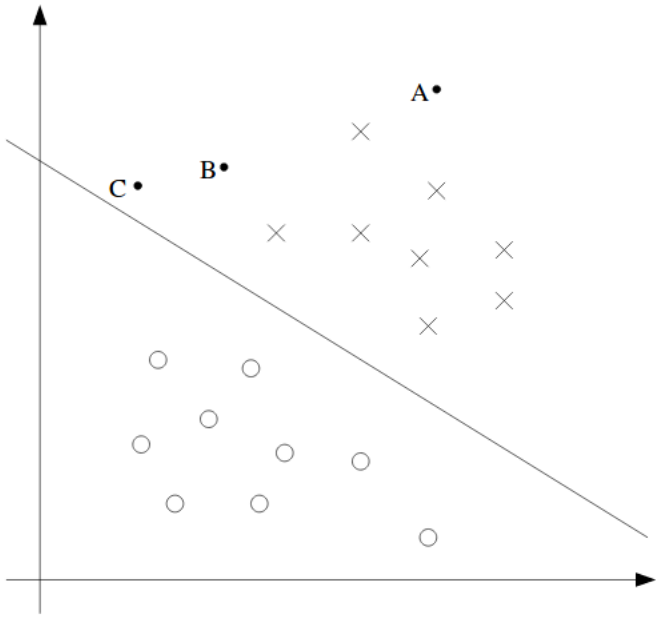



Figure 5.7: SVC Classifier

frequently used in classification problems. SVMs were first introduced in the 1960s, but around 1990 they underwent improvements. Unlike other machine learning algorithms, SVMs are implemented differently. Due to their ability to handle a variety of continuous and categorical variables, they have lately experienced a significant increase in popularity. A simple SVM classifier works by drawing a straight line between two classes. To put it another way, the data points on one side of the line will all be categorized into a single group, whilst the data points on the other side of the line will be categorized into a separate group. This suggests that there are an infinite number of potential lines. The SVM approach is better than various other algorithms like k-nearest neighbors because it chooses the best line to classify your data points. The line that splits the data and is farthest possible from the closest data points is chosen this can be seen in *Figure 5.7*.

The SVM approach aims to find a hyperplane in an N-dimensional space that unambiguously classifies the data points. The size of the hyperplane depends on the quantity of features. Essentially, if there are just two input features, the hyperplane is a line. If there are three input characteristics, the hyperplane transforms into a 2-D plane. It becomes difficult to imagine something with more than three features.

The key SVM concepts are as follows:

- **Support vector**: The data points that are closest to the hyperplane are called support vectors. The separation line will be determined using these data points.
- **Hyperplane**: It is a decision plane or space that is divided up amongst a variety of different items.
- **The margin**: It is the separation between two lines on the closest data points of different classes. It may be calculated using the perpendicular distance between the line and the support vectors. In contrast to a narrow margin, a huge margin is seen positively.

The input data space is transformed into the required format by the kernel that implements the SVM algorithm. The kernel trick is a technique utilized by SVM in which the kernel raises the dimension of a low-dimensional input space. Simply said, the kernel makes a problem separable by increasing the number of dimensions in it. As a result, SVM gets stronger, more flexible, and more precise. SVM uses a number of different kernel types, including the linear, polynomial, and radial basis function(RBF) kernels. In our investigation, we used Linear Kernels, which produced the best results for text categorization. Here is the function of linear Kernel,

$$f(x) = w^T * X + b \tag{5.3}$$

From equation 5.3, the letters b, w, and X represent the training set's anticipated linear coefficient, the weight vector you want to minimize, and the data you're trying to classify, respectively. This equation establishes the decision boundary that the SVM returns.

*Chapter 6*

# ANALYSIS

## 6.1 Model Evaluation

**CNN**

The layers of my CNN model is summarised below using the *.summary()* function in python.

Model: "sequential_6"

| Layer (type) | Output Shape | Param # |
|---|---|---|
| conv2d_30 (Conv2D) | (None, 150, 150, 16) | 800 |
| max_pooling2d_30 (MaxPooling2D) | (None, 75, 75, 16) | 0 |
| conv2d_31 (Conv2D) | (None, 75, 75, 32) | 4640 |
| max_pooling2d_31 (MaxPooling2D) | (None, 37, 37, 32) | 0 |
| conv2d_32 (Conv2D) | (None, 37, 37, 64) | 18496 |
| max_pooling2d_32 (MaxPooling2D) | (None, 18, 18, 64) | 0 |
| conv2d_33 (Conv2D) | (None, 18, 18, 128) | 73856 |
| max_pooling2d_33 (MaxPooling2D) | (None, 9, 9, 128) | 0 |
| conv2d_34 (Conv2D) | (None, 9, 9, 256) | 295168 |
| max_pooling2d_34 (MaxPooling | (None, 4, 4, 256) | 0 |

```
g2D)

flatten_6 (Flatten)          (None, 4096)              0

dense_12 (Dense)             (None, 64)                262208

dropout_6 (Dropout)          (None, 64)                0

dense_13 (Dense)             (None, 1)                 65

=================================================================
Total params: 655,233
Trainable params: 655,233
Non-trainable params: 0
_________________________________________________________________
```

From the Keras library we have used the Early stop function. The snip of the code can be seen below.

**Code:**

```
early_stop = keras.callbacks.EarlyStopping(monitor=
'val_loss', min_delta=1e-6, patience=5)
```

Early stopping is one of the primary tactics used to combat this overfitting scourge known as regularization. The concept is extremely basic. By adjusting the parameters, the model tries to pursue the loss function crazily on the training data. Now, a different collection of data is kept as the validation set. As training progresses, the loss function on the validation set is recorded. When we see that the validation set has not improved, we stop rather than continuing through all the epochs. Early Halting is the term used to describe the early stopping approach based on the validation set performance.

The graphs from *Figure 6.1* show that the model is not overfitting the data because both train loss and val loss concurrently decreased to zero, which leads us to draw this conclusion. Additionally, both train and val accuracy rise in the direction of 100%. We can also see the benefit of the early stopping function. Our model was set to run for 100 epochs, this was stopped at 28 epochs as there was no improvement

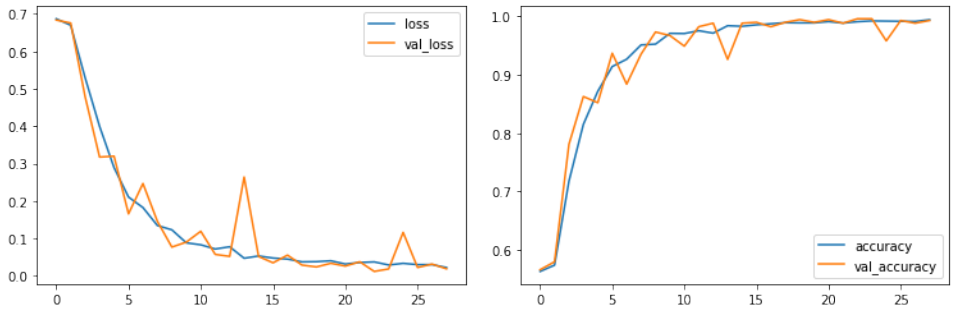



Figure 6.1: Graphs of Train loss and Validation loss

in the validation. The callback function was implemented as it allowed for the following:

- Save my model at regular intervals
- Callback can be used on fit, predict and evaluation of the model

The cut-off probability can be a parameter set based on the company's interest:

- You must choose a cut that yields the highest degree of precision if you wish to minimise the loss brought on by the loss of excellent pieces (i.e., lower the possibility of false positives).
- You must choose a cut that yields the highest degree of precision if you wish to minimise the loss brought on by the loss of excellent pieces (i.e., lower the possibility of false positives).
- You must choose a cut that causes the greatest amount of recall if you want to minimise the danger of false negatives by preventing the client from receiving faulty components as much as possible.
- You can choose a cut that yields a maximum of F1 if you desire a good compromise between the two eventualities mentioned.

Over the majority of the cutoff probability range, accuracy outperforms recall, and the likelihoods of erroneously classified samples are near to 0.5. (generally below, due to the recall being lower than precision).

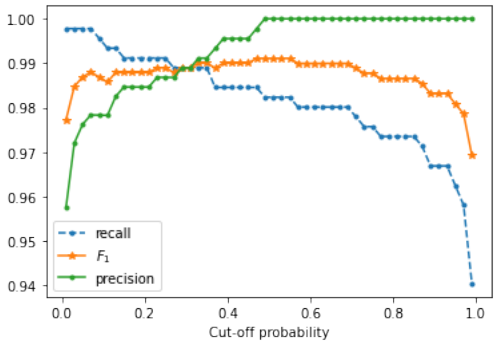



Figure 6.2: Cut of probability

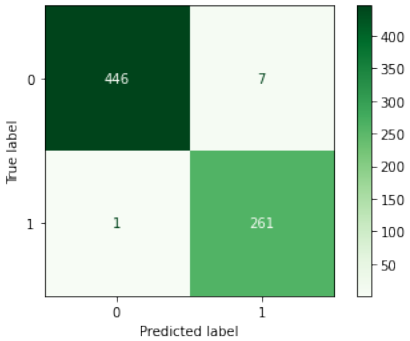



Figure 6.3: Confusion Matrix

**SVC**

Confusion matrix of test dataset on SVC:

The receiver operating characteristic curve (ROC curve) is a graph that displays how well a classification model performs across all categorization levels. More items are classified as positive when the classification threshold is lowered, which raises the number of both False Positives and True Positives. The full two-dimensional region under the complete ROC curve is measured by AUC. An overall assessment of performance across all potential categorization criteria is provided by AUC. AUC may be seen as the likelihood that the model values a randomly chosen positive example higher than a randomly chosen negative example. The Below graph of SVC roc curve has large AUC which indicates good performance of the model.

## 6.2 Performance Metrics

It is essential to choose the proper performance measures for each experimental setting in order to obtain the data required for the verification of any hypothesis or comparison.

Specific assessment measures were determined using these para-metrics:

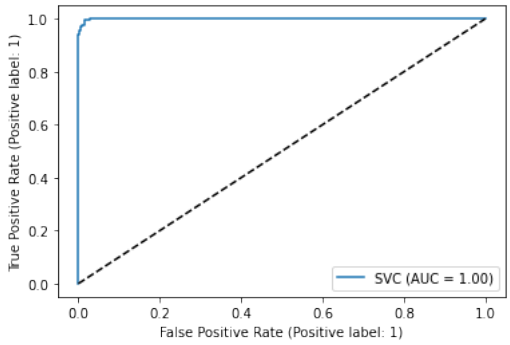



Figure 6.4: AUC curve for SVC

- The proportion of test cases that are correctly classified is known as True Positive (TP).
- The proportion of test instances that are appropriately excluded from the main class is known as True Negative (TN).
- False Negative (FN): the quantity of test cases that are inappropriately included in the main class.
- False Positive (FP): the quantity of test cases that are mistakenly excluded from the main class.

Since accuracy is an obvious metric and has a simple interpretation—just counting successfully detected messages—we have chosen it as the primary criterion for evaluating our classifiers. Additionally, comparisons have been made between the recall and precision of different classifiers. To assess and contrast the detection abilities of the classifiers under investigation, we used the following widely used measures:

1. **Accuracy**:
$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{6.1}$$
2. **Precision**:
$$Precision = \frac{TP}{TP + FP} \tag{6.2}$$
3. **Recall**:
$$Recall = \frac{TP}{TP + FN} \tag{6.3}$$

| | precision | | recall | | f1-score | | support | |
|---|---|---|---|---|---|---|---|---|
| | CNN | SVC | CNN | SVC | CNN | SVC | CNN | SVC |
| Metrics | | | | | | | | |
| 0 | 0.996198 | 0.997763 | 1.000000 | 0.984547 | 0.998095 | 0.991111 | 262.000000 | 453.000000 |
| 1 | 1.000000 | 0.973881 | 0.997792 | 0.996183 | 0.998895 | 0.984906 | 453.000000 | 262.000000 |
| accuracy | 0.998601 | 0.988811 | 0.998601 | 0.988811 | 0.998601 | 0.988811 | 0.998601 | 0.988811 |
| macro avg | 0.998099 | 0.985822 | 0.998896 | 0.990365 | 0.998495 | 0.988008 | 715.000000 | 715.000000 |
| weighted avg | 0.998607 | 0.989012 | 0.998601 | 0.988811 | 0.998602 | 0.988837 | 715.000000 | 715.000000 |

Figure 6.5: Perfromance metrics for CNN and SVC

4. **F1**:

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN} \tag{6.4}$$

## 6.3 Model Comparison

SVM and CNN models' classification performance was contrasted. The suggested model outperforms the other technique in the categorization of datasets, according to the total accuracy of the methods. According to the results in *Figure 6.5*, CNN has the maximum classification accuracy for the dataset. The SVM model just uses the spectral signature of each picture pixel and does not make use of the dataset's rich spatial information. The classification accuracy of the techniques for classifying defects is enhanced by the use of additional spatial information derived from nearby pixels. CNN uses rich spatial characteristics at various scales to represent the spatial structure of the data in order to categorize each pixel in the image since spatial information can increase classification accuracy.

Having a high precision rate means the classifiers are working as they should. A high fractional value of recall denotes the recovery of a sizable share of the total number of pertinent occurrences. It's interesting to see that while SVC's accuracy is lower at 98.88%, CNN's accuracy is greater at 99.86%. Determining the discriminative capacity of risk models requires consideration of the precision, recall, and accuracy values. CNN should thus be chosen in this case rather than SVC.

## 6.4 Deployment

The process of integrating a machine learning model into an already-existing production environment is known as deployment, and it allows you to use data to make useful business decisions. It's a step at the end of the machine learning life cycle.

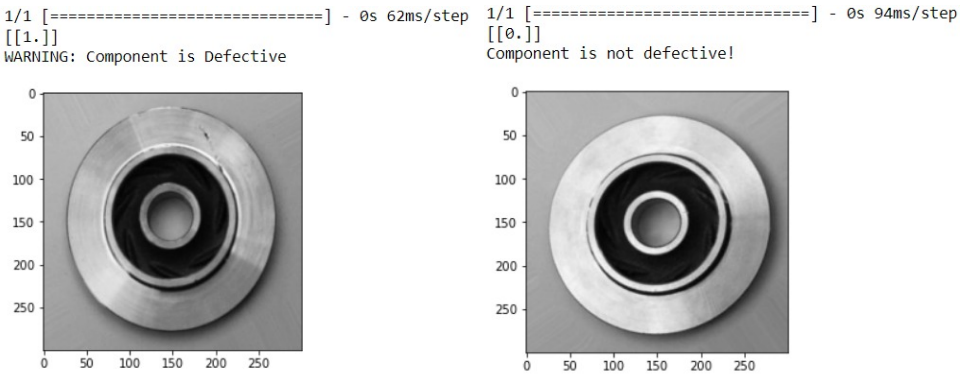



Figure 6.6: Custom Prediction

We have saved the best performing model in cnn_model.hdf5 file. This can be called at anytime and be used to make predictions.

Now that the model has been saved. I have tried to make custom predictions calling the saved model. The code has a for loop and a custom file path for the user to upload. Once the path has been given the model will be able to give a binary output (Defect or not). This can be seen in *Figure 6.6*

*Chapter 7*

# CONCLUSION AND FUTURE WORK

## 7.1 Conclusion

In terms of image recognition, CNN is a strong contender. CNN might be used to analyze sequence data, but they excel in shifting through voluminous picture data to identify non-linear relationships. SVM is a margin classifier that supports several kernels for these classifications. SVM has trouble predicting class labels when the labels are quite large. SVM is also challenging to parallelize, however parallelization is built into the CNN design. When compared to visual perception of the overall categorization of each image, accuracy assessments do not offer the reality. The CNN classification methods demand the definition of a Deep Neural Network Model. To be similar to SVM, this model is defined as a basic model. However, because of the way CNN operates, data must be prepared for kernel window processing. The Kernel window in this investigation was 7 × 7 pixels, based on earlier work. To lower the dimensions of the data that is decreasing noisy bands, the CNN was necessary to utilize specific layers. To expand the dataset training samples, the method needed to be augmented. Despite the accuracy of 99.86% reported by CNN, SVM classifiers have performed better in terms of accuracy in visual. The predictions of both models CNN and SVC can be seen in the below *Figure 7.1* and *Figure 7.2* respectively.

Our technique nevertheless provided excellent results even when only a few photos were utilized for training. This implies that the suggested technique is appropriate for identifying anomalies in both other photos.

Key takeaways:

- Early stopping technique is utilised to avoid unnecessary under-fitting and over-fitting due to useage of too few or too many epochs in our model.
- Data Augmentation The Keras ImageDataGenerator class' primary advantage is that it is intended to give real-time data augmentation. Meaning that while your model is still being trained, it is instantly producing enhanced pictures.
- We built and trained our model with out any pre-trained weights. We used the callback function built in keras to save our model.

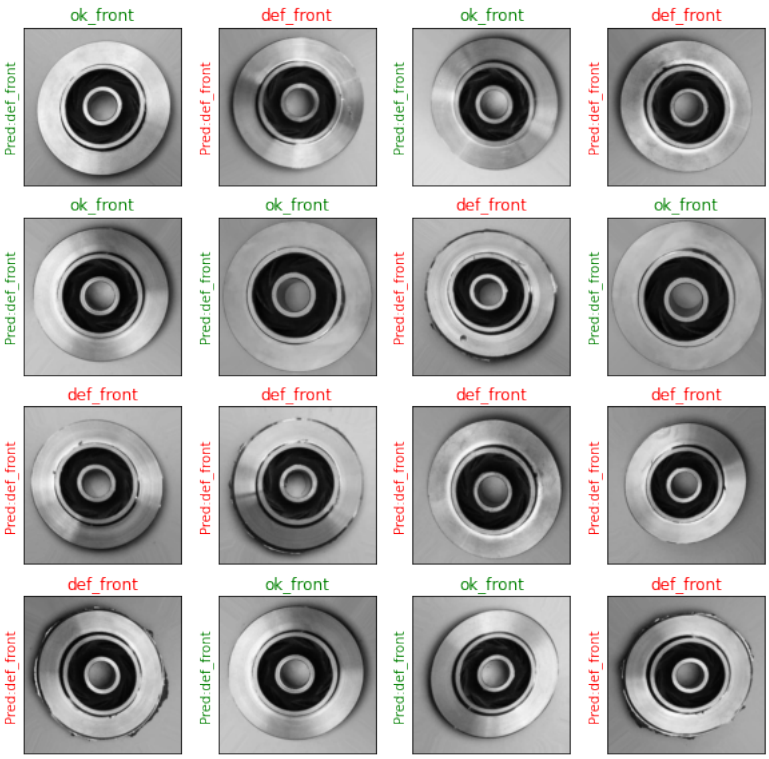



Figure 7.1: Predictions for CNN

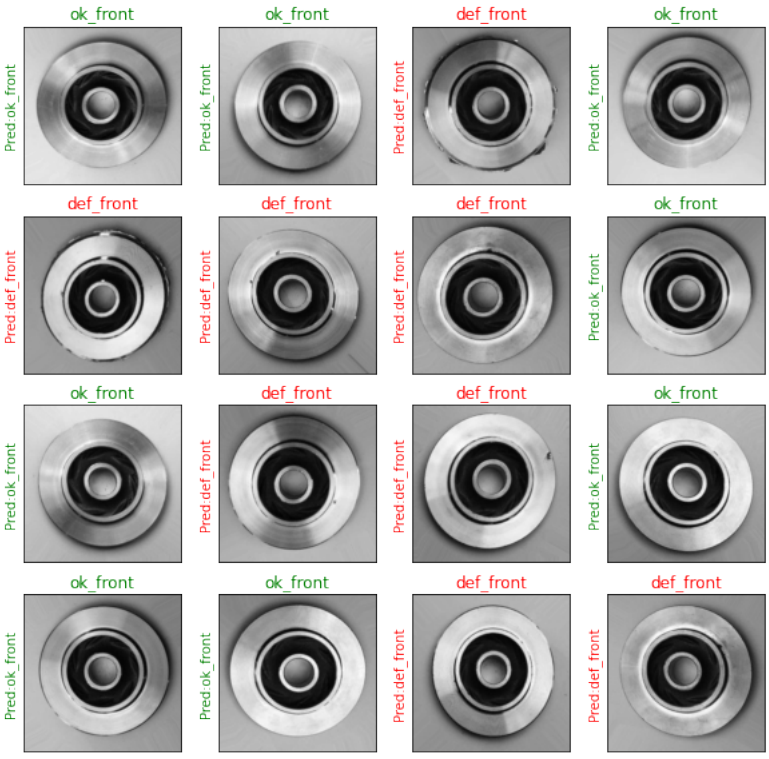



Figure 7.2: Predictions for SVC

## 7.2 Future Work

This paper only discusses the defect classification using two algorithms. In the future, I plan to use Region Based CNN (R-CNN's) which would detect defects in the cast components using image localization technique. This technique essentially would draw a bounded box around the region of interest. In our case a box would be drawn around the defected area. In contrast, there are a limited number of weld/cast images database. As seen in our work we can train our model on newer images to improve its performance. This has the potential to be used in the aerospace and the automotive industry where there's needs to be a high safety standard.

The saved model can also be deployed on Heroku or Streamlit. This would essentially allow the user to upload their own images and the model should be able to predict the class.

# BIBLIOGRAPHY

Birjandi, Parviz and Minoo Alemi (2010). "The impact of test anxiety on test performance among Iranian EFL learners". In: *BRAIN. Broad Research in Artificial Intelligence and Neuroscience* 1.4, pp. 44–58.

Chang, Jeannette et al. (Oct. 2012). "Automated Tuberculosis Diagnosis Using Fluorescence Images from a Mobile Microscope". In: vol. 15, pp. 345–52. ISBN: 978-3-642-33453-5. DOI: 10.1007/978-3-642-33454-2_43.

Chen, Wenfei, Zuohua Miao, and Delie Ming (2011). "Automated Inspection Using X-Ray Imaging". In: *2011IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications*, pp. 1769–1772. DOI: 10.1109/TrustCom.2011.247.

Cortes, Corinna and Vladimir Vapnik (1995). "Support-vector networks". In: *Machine learning* 20.3, pp. 273–297.

Dong, Xinghui, Christopher J. Taylor, and Tim F. Cootes (2021). "A Random Forest-Based Automatic Inspection System for Aerospace Welds in X-Ray Images". In: *IEEE Transactions on Automation Science and Engineering* 18.4, pp. 2128–2141. DOI: 10.1109/TASE.2020.3039115.

Feng, Xiao-Xing et al. (2020). "Identification of X-ray Weld Defects under Artificial Intelligence Framework". In: *2020 5th International Conference on Mechanical, Control and Computer Engineering (ICMCCE)*, pp. 1186–1189. DOI: 10.1109/ICMCCE51767.2020.00261.

Ferguson, Max et al. (2017). "Automatic localization of casting defects with convolutional neural networks". In: *2017 IEEE International Conference on Big Data (Big Data)*, pp. 1726–1735. DOI: 10.1109/BigData.2017.8258115.

H. Sultan, Hossam, Nancy Salem, and Walid Al-Atabany (May 2019). "Multi-Classification of Brain Tumor Images Using Deep Neural Network". In: *IEEE Access* PP, pp. 1–1. DOI: 10.1109/ACCESS.2019.2919122.

He, Zhiquan and Qifan Liu (2020a). "Deep Regression Neural Network for Industrial Surface Defect Detection". In: *IEEE Access* 8, pp. 35583–35591. DOI: 10.1109/ACCESS.2020.2975030.

– (2020b). "Deep Regression Neural Network for Industrial Surface Defect Detection". In: *IEEE Access* 8, pp. 35583–35591. DOI: 10.1109/ACCESS.2020.2975030.

Lin, Chih-Hsueh et al. (2021). "Alloy Cast Product Defect Detection Based on Object Detection". In: *2021 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)*, pp. 1–2. DOI: 10.1109/ISPACS51563.2021.9651119.

Liu, Kun et al. (2020). "A New Self-Reference Image Decomposition Algorithm for Strip Steel Surface Defect Detection". In: *IEEE Transactions on Instrumentation and Measurement* 69.7, pp. 4732–4741. DOI: 10.1109/TIM.2019.2952706.

Lyra, Maria, Agapi Ploussi, and Antonios Georgantzoglou (Oct. 2011). "MATLAB as a Tool in Nuclear Medicine Image Processing". In: ISBN: 978-953-307-907-3. DOI: 10.5772/19999.

Martínez-Plumed, Fernando et al. (2021). "CRISP-DM Twenty Years Later: From Data Mining Processes to Data Science Trajectories". In: *IEEE Transactions on Knowledge and Data Engineering* 33.8, pp. 3048–3061. DOI: 10.1109/TKDE.2019.2962680.

Phung and Rhee (Oct. 2019). "A High-Accuracy Model Average Ensemble of Convolutional Neural Networks for Classification of Cloud Image Patches on Small Datasets". In: *Applied Sciences* 9, p. 4500. DOI: 10.3390/app9214500.

Schröer, Christoph, Felix Kruse, and Jorge Marx Gómez (2021). "A systematic literature review on applying CRISP-DM process model". In: *Procedia Computer Science* 181, pp. 526–534.

Zhang, Xiao-Guang, Jian-Jian Xu, and Guang-Ying Ge (2004). "Defects recognition on X-ray images for weld inspection using SVM". In: *Proceedings of 2004 International Conference on Machine Learning and Cybernetics (IEEE Cat. No.04EX826)*. Vol. 6, 3721–3725 vol.6. DOI: 10.1109/ICMLC.2004.1380463.

Zhou, Awei et al. (2020). "Defect Inspection Algorithm of Metal Surface Based on Machine Vision". In: *2020 12th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA)*, pp. 45–49. DOI: 10.1109/ICMTMA50254.2020.00017.

# INDEX