# Learning Defects from Aircraft Non-Destructive Testing (NDT) Data

Navya Prakash

Matriculation number: 3042906

March 2, 2020

Master Thesis

**Computer Science**

Examiners:

Prof. Dr. Jens Lehmann
Dr. Alfons Schuster, DLR, Augsburg

Supervisors:

Dr. Hajira Jabeen
Dorothea Nieberl, DLR, Augsburg
Monika Mayer, DLR, Augsburg

Institut für Informatik III

Rheinische Friedrich-Wilhelms-Universität Bonn

# Declaration of Authorship

I, Navya Prakash, declare that this thesis, titled "Learning Defects from Aircraft Non-Destructive Testing (NDT) Data", and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. Except for such quotations, this thesis is entirely my own work. I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

# Acknowledgements

I would like to thank my examiners, *Prof. Dr. Jens Lehmann*, and *Dr. Alfons Schuster* for their guidance and support. I would extend my thanks to all my supervisors, *Dr. Hajira Jabeen*, *Dorothea Nieberl*, and *Monika Mayer* for their enormous encouragement and time spent with me during the entire thesis work.

Both Doro, and Monika could help me with my queries through valuable meetings and gave their best advises whenever I needed the most. Thank you. Hajira had constant interactions with me as well provided her valuable advices and supported all of my ideas throughout these months, even though it was a remote research work. Thank you.

At last but not the least, I would like to thank my family for being with me. Special thanks to *Charles Lennart Müller* for being there at all odd-times.

I would also like to thank *Premium Aerotec AG* for their permission to use the NDT data for this study.

# Acronyms and Glossary

- AI - Artificial Intelligence is a sub-area of computer science that deals with the automation of intelligent behavior and machine learning.
- FML - Fiber Metal Laminates is one of a class of metallic materials consisting of a laminate of several thin metal layers bonded with layers of composite material. This allows the material to behave much as a simple metal structure, but with considerable specific advantages regarding properties such as metal fatigue, impact, corrosion resistance, fire resistance, weight savings, and specialized strength properties.
- NDT - Non-destructive Testing is a testing and analysis technique used by industry to evaluate the properties of a material, component, structure or system for characteristic differences or welding defects and discontinuities without causing damage to the original part. NDT also known as non-destructive examination (NDE), non-destructive inspection (NDI) and non-destructive evaluation (NDE).
- CFRP - Carbon Fiber Reinforced Polymer or carbon fiber reinforced plastic, or carbon fiber reinforced thermoplastic (CFRP, CRP, CFRTP, or often simply carbon fiber, carbon composite, or even carbon), is an extremely strong and light fiber-reinforced plastic which contains carbon fibers. CFRPs can be expensive to produce but are commonly used wherever high strength-to-weight ratio and stiffness (rigidity) are required, such as aerospace, superstructure of ships, automotive, civil engineering, sports equipment, and an increasing number of consumer and technical applications.
- PoD - Probability of Detection curves have been produced for a range of Non-Destructive Testing (NDT) methods (e.g. ultrasound, radiography, eddy currents, magnetic particle inspection, liquid penetrants, visual).
- ML - Machine Learning is the scientific study of algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions, relying on patterns and inference instead.
- DNN - Deep Neural Network (also known as deep structured learning or differential programming or deep learning) is part of a broader family of machine learning methods based on artificial neural networks with representation learning.

- Pre-preg - is pre-impregnated composite fibers where a thermoset polymer matrix material, such as epoxy, or a thermoplastic resin is already present.
- QA - Quality Assurance is a way of preventing mistakes and defects in manufactured products and avoiding problems when delivering products or services to customers.

# Contents

# List of Figures

# List of Tables

**Abstract**

Industry 4.0 has made an impact on Aircraft production companies. This work is designed for the in-line quality assurance of Fibre Metal Laminates (FML) in the production of aircraft parts. The use of Artificial Intelligence in analyzing Non-Destructive Testing (NDT) Aircraft data is first of its kind. In our work, NDT ultrasonic scan data has been investigated to learn defect and good parts of the aircraft. It includes a traditional Machine Learning approach which involves different feature extraction techniques in combination with various classifiers. It is compared to the previous research work of Deep Learning Neural Network. Evaluation of the proposed approach is performed with benchmarking distinct feature extraction techniques in association of different classifiers and is promoted to research in Certification. Recent trials prove that the proposed approach can be adopted to help the Human examiner in the Industry.

**Keywords**: NDT data, Machine Learning, Image Feature Extraction, Classifiers, PoD.

# Chapter 1

# Introduction

The manufacturing of aircraft parts made of fiber-metal laminates (FML) has cascaded steps such as placement of aluminium, glass prepeg[1], adhesive, doublers and stringers plus vacuum bagging and curing in an autoclave. Quality control is performed first at the layup of the skin (without stringers) and its curing. Evaluation of the Quality Assessment is performed manually, to be more specific visually (Apmann et al., 2016).

As Industry 4.0 helps streamlining processes, increase quality and lower costs, the in-line quality system of the aircraft production decides to use automated production chain and effective quality assurance. By doing so, it can reduce production cost. The manufacturing process for FML consists of several individual steps. Instead of examining the quality of the component after each of the individual steps, any deviations should be discovered when they occur. This way, corrective action can be provided and intervened in the ongoing process in time. Thus avoid process deviations and costly repairs and waste is reduced. Preferably, defects can be detected as soon as they occur with inline Quality Assurance (QA) measures. However, there is also a final step with NDT. In addition to new concepts here the automated analysis of the collected data is in focus, in order to substantially reduce the previous efforts (Apmann et al., 2016).

The manual quality assurance process has inadequate process documents, low and highly variable quality of the manual testing process, the inspection process is the most time consuming part in the entire production process chain along with a lot of effort. The technological and economic requirements for structural components in the aerospace industry are very different to automotive[2]. Automated evaluation can be applied to many materials such as

[1] For definition and complete explanation, visit `https://www.fibreglast.com/product/about-prepregs/Learning_Center`.

[2] `https://www.sglcarbon.com/en/markets-solutions/applications/cfrp-lightweight-components-for-the-automotive-industry`

CFRP (Carbon Fiber Reinforced Polymer), FML, monolithic aluminium. To ensure sufficient component quality appropriate quality assurance measures are necessary. The quality control of a component is realized by means of a visual inspection by a worker and taking between 32% and 68% of the total production time (Apmann et al., 2016).

The technical basis for Industry 4.0 is intelligent, networked production systems. Intelligent systems are able to act with foresight and to make independent decisions on the basis of a wide range of information. Corresponding correlations of different information are often unknown or much too complex for classical rule-based software systems. The current trend towards artificial intelligence promises to remedy such situations. The advantage of artificial intelligence is that rules do not have to be programmed explicitly, but can be learned from existing data in a machine (Brehm and Lienhart, 2018).

The current algorithms of artificial intelligence originate from the research field of machine learning, which is already decades old. Machine learning is generally concerned with learning from data - something that is currently collected and stored on an unprecedented scale. Data-driven learning methods are predestined to recognize patterns in data and to recognize similar patterns in previously unseen data. This allows machine learning to evaluate unknown and new data. This assessment of a data situation is commonly referred to as prediction or inference. Because of these characteristics, mechanical learning methods are particularly suitable for tasks for which people need extensive experience. This experience can be transformed into a machine learning process from data (Brehm and Lienhart, 2018).

With the rise of big data, machine learning has become an important technique for solving problems in areas such as the following: Computerized financial services, for credit scoring, algorithmic trading and many more. Image processing and computer vision, for face detection, motion detection and object detection. Bio-informatics, for tumor detection, drug research and DNA sequencing. Energy production, for predicting electricity loads and prices. Automotive, aerospace and manufacturing, for predictive maintenance. Natural speech processing, for speech recognition applications. Machine learning algorithms find natural patterns in data that provide insight and help you make better decisions and forecasts. They are used on a daily basis to make important decisions regarding medical diagnoses, stock exchange trading, the prediction of current loads, etc. For example, media websites use machine learning to look through millions of options and recommend suitable songs or films. Retailers use it to gain insight into their customers' buying behavior.

In contradiction to the manual process, an automated quality assurance process will have a cascading procedure for minimization of the computation

time and use of Artificial Intelligence method such as Machine Learning to train sample error locations and error survey. This can save time and cost as well as add effectiveness and efficiency. The quality analysis process with Machine Learning (ML) could include pre-analysis of the parts without errors and potential flaw regions after getting the input from the sensors (which scans the produced aircraft part - giving image matrix), classify the parts accordingly into flaw and good parts and as a next step to measure the flaw area and store the results. There could be various flaw occurrences such as fold, twist, overlap, gap and foreign body (Apmann et al., 2016).

## 1.1 Motivation

Non-destructive testing of structural components are used in the production of aircraft parts. Apart from visual inspection during manufacturing ultrasonic scans of the final part are made. Since the examination of the scans regarding anomalies has to be performed manually and is very time-consuming and personnel-intensive. A fully automated detection and evaluation of discontinuities using Machine Learning methods is desirable (Schmidt et al., 2015). Its objective is to support the human inspector, not make him redundant. It can save time and lower the risk of missing defects. Our work aims to play a vital role in the automated in-line quality assurance for the production of FML using traditional Machine Learning. It can also pave the way for using this method on other materials like CFRP or aluminium. It includes feature extraction and classification (to detect flaw and good parts in the given data only and not to analyze the type of the flaws present - due to limited data). We have implemented various feature extraction techniques to analyze which suits best for the given data in combination with profuse classifiers. Most importantly it is structured as a binary classification because it involves only two classes to classify, either flaw or good part.

The objectives of this thesis includes understanding and preparing raw data given by the Industry and pre-processing the same to attain feasibility for input with ML algorithms. Selection of suitable feature extraction techniques is also an objective. Selection of qualified machine learning classifiers to achieve the best results. This work further has been researched for certification.

## 1.2 Thesis Structure

The rest of the thesis is structured as follows: In Chapter 2, we introduce diverse theories pertinent to our work such as description of NDT data, previous research work (deep learning approach), useful image processing and feature extraction techniques, importance of machine learning and its classifiers. Chapter 3 details the proposition of our work such as the composition of the custom dataset, proposed architecture and its model. We describe our experiments in Chapter 4, and discuss its results in Chapter 5. In the adjoining Chapter 6 we explain the entire process required to certify our approach as it is observed in the industry. At last we compare our work with the previous research methodology - deep learning model in Chapter 7 as well advocate possible future enhancements.

# Chapter 2

# Background

Before conferring our work, we describe some concepts that are necessary to interpret our work. In the following sections, we shall provide an overview of NDT data and its format, previous research work to compare to our work, image feature extraction methodologies and distinguished ML classifiers that are acceptable for our model.

## 2.1 Non-Destructive Testing Data

The raw dataset obtained from the Industry originates from NDT of the manufactured aircraft parts. Non-Destructive Testing (NDT) consists of a variety of non-invasive inspection techniques used to evaluate material properties, components or entire process units. The techniques can also be utilized to detect, characterize or measure the presence of damage mechanisms (e.g. corrosion or cracks). NDT is also commonly referred to as nondestructive examination (NDE), nondestructive evaluation (NDE), and nondestructive inspection (NDI). Many NDT techniques are capable of locating defects and determining the features of the defects such as size, shape and orientation. The purpose of NDT is to inspect a component in a safe, reliable and cost effective manner without causing damage to the equipment or shutting down plant operations. This is in contrast to destructive testing where the part being tested is damaged or destroyed during the inspection process [1]. There are two kinds of NDT testing, namely, conventional and advanced. Conventional NDT testing includes:

- Acoustic Emission Testing (AET)
- Infrared Testing (IR)

[1]Source: `https://inspectioneering.com/tag/nondestructive+testing`

- Leak Testing (LT)
- Liquid Penetrant Testing (PT)
- Electromagnetic Testing (ET)
- Magnetic Particle Testing (MPT)
- Radiographic Testing (RT)
- Film Radiography (FR)
- Ultrasonic Testing (UT)
- Straight Beam
- Vibration Analysis (VA)
- Visual Inspection (VI)

Advanced NDT testing includes (Alvarado et al., 2020):

- Electromagnetic Testing (ET)
- Laser Testing Methods (LM)
- Radiographic Testing (RT)
- Ultrasonic Testing (UT)

Our work utilizes Ultrasonic Testing data and on which the human examiner performs Visual Inspection and gives a report on defects found within the part. This is the basis for understanding and pre-processing the data. When planning an NDT inspection, there are four considerations one should account for (Apmann et al., 2016):

i. The type of damage mechanism to be inspected for

ii. The minimum detectable flaw size, shape, and orientation of the defect

iii. Where the defect is located (surface or internal)

iv. The sensitivities and limitations of the NDT method

All these considerations are handled in our work using Machine Learning to reduce the Visual Inspection time of the human examiner and lower his workload.

For our work, the raw data was given by Premium Aerotec AG Industry after NDT testing using UT. Visualization was done with ULTIS-TESTIA software which is a ultrasonic analyzing software developed by AIRBUS company and supported by TESTIA. The main benefits of ULTIS are: It spreads a common diagnosis tool even if using various ultrasonic instruments, it provides diagnosis tools which always comply with composite quality requirements (especially those from AIRBUS)[2], it automates the analysis process as much as possible in order to reduce human factor and saves time and it assesses the possibility to read data during decades. This software was able to open and visualize the data format used by the scanning system. By doing so ULTIS creates three types of files for each scan file, namely, .nkc - which has the original data stored, .nkd - which stores the defect information such as its 2D position (location) over the scan area, defect area covered, length and breadth of each and every defects present in that scan data file and .nkz stores any other information. The NKC file contains 1 C-scan[3] (C-scan refers to the image produced when the data collected from an ultrasonic inspection is plotted on a plan view of the component). The NKC format is composed of two blocks:

1. The first block, which represents the header of the file, has a length (in byte) defined by the data offset field. The header is written in ASCII format (indications and values).

2. The second block which contains the physical data (written in binary). The information from all .nkc, .nkd and .nkz files corresponding to each defect scan file is stored as an image (.jpg) using the smart plug-in that has been provided by the ULTIS software. The good parts of the data does not contain the .nkd file as there is no defect information. So, only the NKC and NKZ files corresponding to each good scan file is stored as an image. Using these image dataset, further research work continues. The next question that arises is how to analyze this dataset and preprocess it to feed into Machine Learning algorithm. By understanding the previous research work Brehm and Lienhart (2018) performed on the NDT data, we could obtain insights to perceive the dataset.

[2] `https://www.testia.com/wp-content/uploads/2015/12/tlgment-ultis.pdf`

[3] Definition according to BS EN 1330-4:2000: Image of the results of an ultrasonic examination showing a cross-section of the test object parallel to the scanning surface

## 2.2 Machine Learning

Machine learning is an application of artificial intelligence (AI) that provides systems with the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it to learn independently. The process of learning begins with observations or data, such as examples, direct experience or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. The primary aim is to allow the computers to learn automatically without human intervention or assistance and adjust actions accordingly[4].

There are many different types of Machine Learning systems so it can be useful to classify them in broad categories based on: whether or not they are trained with human supervision,

- Supervised
- Unsupervised
- Semisupervised
- Reinforcement Learning

whether or not they can learn incrementally on the fly,

- Online learning
- Batch learning

whether they work by simply comparing new data points to known data points or instead detect patterns in the training data and build a predictive model, much like scientists do[5],

- Instance-based learning
- Model-based learning

Supervised learning - monitored machine learning creates a model that provides forecasts based on evidence if there are also uncertainties. A supervised learning algorithm uses a known set of input data and known outputs for the data to train a model that produces well-founded predictions for the new input data. Use supervised learning if you have known data for

[4] `https://expertsystem.com/machine-learning-definition`

[5] Book: Hands-On Machine Learning with Scikit-Learn and TensorFlow

the expenses you want to predict. Supervised learning uses classification and regression techniques to develop predictive models. Supervised machine learning algorithms can apply what has been learned in the past to new data using labeled examples to predict future events. Starting from the analysis of a known training dataset, the learning algorithm produces an inferred function to make predictions about the output values. The system is able to provide targets for any new input after sufficient training. The learning algorithm can also compare its output with the correct, intended output and find errors in order to modify the model accordingly. In supervised learning, the training data you feed to the algorithm includes the desired solutions, called labels. A typical supervised learning task is classification. The spam filter is a good example of this: It is trained with many example emails along with their class (spam or ham) and it must learn how to classify new emails. Classification techniques predict discrete outcomes - for example, whether an email is real or spam or whether a tumor is cancerous or benign. Classification models classify input data into categories. Typical applications are medical imaging, speech recognition and credit scoring. Use the classification if your data can be tagged, divided into categories or divided into specific groups or classes. For example, handwriting recognition applications use classification to recognize letters and numbers. In image processing and computer vision, the techniques of unsupervised pattern recognition are used for object recognition and image segmentation. Common algorithms for performing the classification are:

- Support Vector Machine (SVM)
- Decision trees with boosting and bagging
- K - Nearest Neighbor method
- Naive Bayes classification
- Discriminant analysis
- Logistic regression
- Neural Networks

Another typical task is to predict a target numeric value, such as the price of a car, given a set of features (mileage, age, brand, etc.) called predictors. This sort of task is called regression. To train the system, you need to give it many examples of cars, including both their predictors and their labels (i.e., their prices). Note that some regression algorithms can be used for classification

as well and vice versa. For example, Logistic Regression is commonly used for classification, as it can output a value that corresponds to the probability of belonging to a given class (e.g., 20% chance of being spam).

Regression techniques predict continuous outputs - for example temperature changes or fluctuations in energy consumption. Typical applications are the prediction of current loads and algorithmic trading. Use regression techniques when working with a range of data or when the output is a real number, such as a temperature or the time it takes for a device to fail. Common regression algorithms are:

- Linear model
- Non-linear model
- Regularization
- Stepwise regression
- Decision trees with boosting and bagging
- Neural networks
- Adaptive neuro-fuzzy learning

In contrast, Unsupervised learning finds hidden patterns or internal structures in data. It is used to draw conclusions from data sets that consist of input data without classified outputs. Clustering is the most common unsupervised learning technique. It is used for exploratory data analysis to find hidden patterns or groupings in data. Applications for cluster analysis are, for example, gene sequence analysis, market research and object recognition. For example, if a mobile operator wants to optimize locations for transmission towers, they can use machine learning to estimate how many clusters of people will use these towers. A cell phone can only communicate with one mast at any one time. The team therefore uses clustering algorithms to find the best placement of transmission towers that optimize signal reception for groups (or clusters) of customers. Common algorithms for clustering are:

- K-Means and k-Medoids
- Hierarchical clustering
- Gaussian mixture models
- Hidden Markov models

- Self-organizing maps
- Fuzzy c-means clustering
- Subtractive clustering

Unsupervised machine learning algorithms are used when the information used to train is neither classified nor labeled. It studies how systems can infer a function to describe a hidden structure from unlabeled data. The system does not figure out the right output, but it explores the data and can draw inferences from datasets to describe hidden structures from unlabeled data. In unsupervised learning the training data is unlabeled. The system tries to learn without a teacher.

Semi-supervised machine learning algorithms fall somewhere in between supervised and unsupervised learning, since they use both labeled and unlabeled data for training – typically a small amount of labeled data and a large amount of unlabeled data. The systems that use this method are able to considerably improve learning accuracy. Usually, semi-supervised learning is chosen when the acquired labeled data requires skilled and relevant resources in order to train it or learn from it. Otherwise, acquiring unlabeled data generally does not require additional resources. Some algorithms can deal with partially labeled training data, usually a lot of unlabeled data and a little bit of labeled data. This is called semi-supervised learning. Reinforcement machine learning algorithms is a learning method that interacts with its environment by producing actions and discovers errors or rewards. Trial and error search and delayed reward are the most relevant characteristics of reinforcement learning. This method allows machines and software agents to automatically determine the ideal behavior within a specific context in order to maximize its performance. Simple reward feedback is required for the agent to learn which action is best; this is known as the reinforcement signal. Reinforcement Learning is a very different beast. The learning system, called an agent in this context, can observe the environment, select and perform actions, and get rewards in return (or penalties in the form of negative rewards). It must then learn by itself what is the best strategy, called a policy, to get the most reward over time. A policy defines what action the agent should choose when it is in a given situation. In batch learning, the system is incapable of learning incrementally: it must be trained using all the available data. This will generally take a lot of time and computing resources, so it is typically done offline. First the system is trained, and then it is launched into production and runs without learning anymore; it just applies what it has learned. This is called offline learning. In online learning, you train the

system incrementally by feeding it data instances sequentially, either individually or by small groups called mini-batches. Each learning step is fast and cheap, so the system can learn about new data on the fly, as it arrives.

In machine learning and statistics, classification is the problem of identifying to which set of categories (sub-populations) a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known. Examples are assigning a given email to the spam or non-spam class and assigning a diagnosis to a given patient based on observed characteristics of the patient (sex, blood pressure, presence or absence of certain symptoms). Classification is an example of pattern recognition. In the terminology of machine learning, classification is considered an instance of supervised learning, i.e., learning where a training set of correctly identified observations is available. The corresponding unsupervised procedure is known as clustering and involves grouping data into categories based on some measure of inherent similarity or distance. Classification can be thought of as two separate problems – binary classification and multiclass classification. In binary classification, a better understood task, only two classes are involved, whereas multiclass classification involves assigning an object to one of several classes. Since many classification methods have been developed specifically for binary classification, multiclass classification often requires the combined use of multiple binary classifiers. Most algorithms describe an individual instance whose category is to be predicted using a feature vector of individual, measurable properties of the instance. Each property is termed a feature, also known in statistics as an explanatory variable (or independent variable, although features may or may not be statistically independent). Features may variously be binary (e.g. on or off); categorical (e.g. A, B, AB or O, for blood type); ordinal (e.g. large, medium or small); integer-valued (e.g. the number of occurrences of a particular word in an email); or real-valued (e.g. a measurement of blood pressure). If the instance is an image, the feature values might correspond to the pixels of an image; if the instance is a piece of text, the feature values might be occurrence frequencies of different words. Some algorithms work only in terms of discrete data and require that real-valued or integer-valued data be discretized into groups (e.g. less than 5, between 5 and 10, or greater than 10).

The only way to know how well a model will generalize to new cases is to actually try it out on new cases. One way to do that is to put your model in production and monitor how well it performs. A better option is to split data into two sets: the training set and the test set. As these names imply, we train our model using the training set and we test it using the test set. The error rate on new cases is called the generalization error (or out-of-sample

error) and by evaluating your model on the test set, we get an estimation of this error. This value tells us how well our model will perform on instances it has never seen before. If the training error is low (i.e., your model makes few mistakes on the training set) but the generalization error is high, it means that our model is overfitting the training data. Classifier performance depends greatly on the characteristics of the data to be classified. There is no single classifier that works best on all given problems (a phenomenon that may be explained by the no-free lunch theorem[6]). Various empirical tests have been performed to compare classifier performance and to find the characteristics of data that determine classifier performance. Determining a suitable classifier for a given problem is however still more an art than a science. The measures precision and recall are popular metrics used to evaluate the quality of a classification system. More recently, receiver operating characteristic (ROC) curves have been used to evaluate the trade-off between true and false-positive rates of classification algorithms. As a performance metric, the uncertainty coefficient has the advantage over simple accuracy in that it is not affected by the relative sizes of the different classes. Further, it will not penalize an algorithm for simply rearranging the classes[7].

### SVM

In machine learning, support-vector machines (SVMs, also support-vector networks) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier (although methods such as Platt scaling[8] exist to use SVM in a probabilistic classification setting). An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap. New examples are then mapped into that same space and predicted to belong to a category based on the side of the gap on which they fall. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces. More formally, a support-vector machine constructs a hyperplane or set of hyperplanes in a high- or infinite-dimensional space, which can be used for classification, regression or other tasks like outliers detection. Intuitively,

[6] `https://en.wikipedia.org/wiki/No_free_lunch_theorem`

[7] `https://en.wikipedia.org/wiki/Statistical_classification`

[8] Refer: `https://en.wikipedia.org/wiki/Platt_scaling`

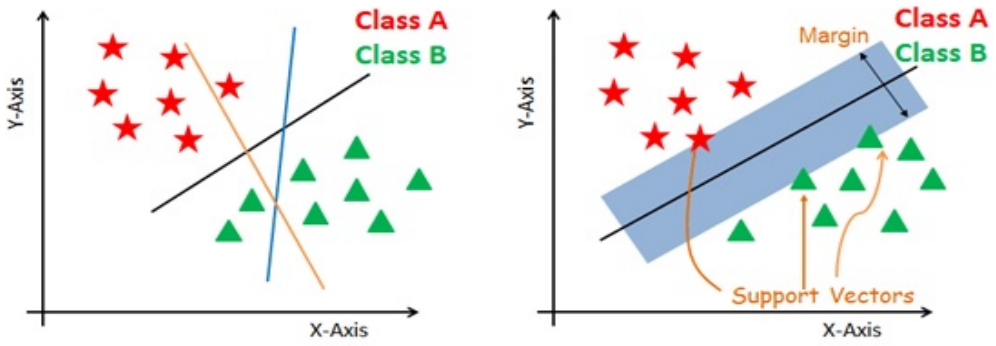



Figure 2.1: General SVM model (datacamp.com)

a good separation is achieved by the hyperplane that has the largest distance to the nearest training-data point of any class (so-called functional margin), since in general the larger the margin, the lower the generalization error of the classifier[9]. The main objective of the support vector machine algorithm is to find a hyperplane in an N-dimensional space (N — the number of features) that distinctly classifies the data points.

To separate the two classes of data points, there are many possible hyperplanes that could be chosen as shown in the Figure 2.1[10]. The objective is to find a plane that has the maximum margin, i.e the maximum distance between data points of both classes. Maximizing the margin distance provides some reinforcement so that future data points can be classified with more confidence. Hyperplanes are decision boundaries that help classify the data points. Data points falling on either side of the hyperplane can be attributed to different classes. Also, the dimension of the hyperplane depends upon the number of features. If the number of input features is 2, then the hyperplane is just a line. If the number of input features is 3, then the hyperplane becomes a two-dimensional plane. It becomes difficult to imagine when the number of features exceeds 3. Support vectors are data points that are closer to the hyperplane and influence the position and orientation of the hyperplane[11].

Whereas the original problem may be stated in a finite-dimensional space, it often happens that the sets to discriminate are not linearly separable in that space. For this reason, it was proposed that the original finite-dimensional space be mapped into a much higher-dimensional space, presumably making the separation easier in that space. To keep the computational load reason-

[9] https://en.wikipedia.org/wiki/Support-vector_machine

[10] https://www.datacamp.com/community/tutorials/svm-classification-scikit-learn-python

[11] https://towardsdatascience.com

able, the mappings used by SVM schemes are designed to ensure that dot products of pairs of input data vectors may be computed easily in terms of the variables in the original space, by defining them in terms of a kernel function $K(x, y)$ selected to suit the problem. The hyperplanes in the higher-dimensional space are defined as the set of points whose dot product with a vector in that space is constant, where such a set of vectors is an orthogonal (and thus minimal) set of vectors that defines a hyperplane. The vectors defining the hyperplanes can be chosen to be linear combinations with parameters $a_i$ of images of feature vectors $x_i$ that occur in the data base. With this choice of a hyperplane, the points $x$ in the feature space that are mapped into the hyperplane are defined by the relation $\sum_i a_i k(x_i, x) = constant$. Note that if $k(x, y)$ becomes small as $y$ grows further away from $x$, each term in the sum measures the degree of closeness of the test point $x$ to the corresponding data base point $x_i$. In this way, the sum of kernels above can be used to measure the relative nearness of each test point to the data points originating in one or the other of the sets to be discriminated. Note the fact that the set of points $x$ mapped into any hyperplane can be quite convoluted as a result, allowing much more complex discrimination between sets that are not convex at all in the original space. SVMs can be used to solve various problems. SVMs are helpful in text and hypertext categorization, as their application can significantly reduce the need for labeled training instances in both the standard inductive and transductive settings. Some methods for shallow semantic parsing are based on support vector machines. Classification of images can also be performed using SVMs. Experimental results show that SVMs achieve significantly higher search accuracy than traditional query refinement schemes after just three to four rounds of relevance feedback. This is also true for image segmentation systems, including those using a modified version SVM that uses the privileged approach as suggested by Vapnik. Hand-written characters can be recognized using SVM. The SVM algorithm has been widely applied in the biological and other sciences. They have been used to classify proteins with up to 90% of the compounds classified correctly. Permutation tests based on SVM weights have been suggested as a mechanism for interpretation of SVM models. Support-vector machine weights have also been used to interpret SVM models in the past. Posthoc interpretation of support-vector machine models in order to identify features used by the model to make predictions is a relatively new area of research with special significance in the biological sciences[12].

[12] `https://en.wikipedia.org/wiki/Support-vector_machine`

### Decision Trees

A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs and utility. It is one way to display an algorithm that only contains conditional control statements. Decision trees are commonly used in operations research, specifically in decision analysis, to help identify a strategy most likely to reach a goal, but are also a popular tool in machine learning. A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute (e.g. whether a coin flip comes up heads or tails), each branch represents the outcome of the test and each leaf node represents a class label (decision taken after computing all attributes). The paths from root to leaf represent classification rules. In decision analysis, a decision tree and the closely related influence diagram are used as a visual and analytical decision support tool, where the expected values (or expected utility) of competing alternatives are calculated. A decision tree consists of three types of nodes: decision nodes – typically represented by squares, chance nodes – typically represented by circles and end nodes – typically represented by triangles as shown in Figure 2.2[13]. They are commonly used in operations research and operations management. If, in practice, decisions have to be taken online with no recall under incomplete knowledge, a decision tree should be paralleled by a probability model as a best choice model or online selection model algorithm. Another use of decision trees is as a descriptive means for calculating conditional probabilities[14]. Decision trees, influence diagrams, utility functions and other decision analysis tools and methods are taught to undergraduate students in schools of business, health economics, and public health and are examples of operations research or management science methods. The decision tree can be linearized into decision rules, where the outcome is the contents of the leaf node, and the conditions along the path form a conjunction in the if clause. In general, the rules have the form: if condition1 and condition2 and condition3 then outcome.

Decision rules can be generated by constructing association rules with the target variable on the right. They can also denote temporal or causal relations. Decision trees have some advantages such as: they are simple to understand and interpret, people are able to understand decision tree models after a brief explanation, have value even with little hard data and important insights can be generated based on experts describing a situation

[13] https://www.lucidchart.com/pages/decision-tree

[14] https://ocw.mit.edu/courses/health-sciences-and-technology/hst-951j-medical-decision-support-fall-2005/lecture-notes/hst951_2.pdf

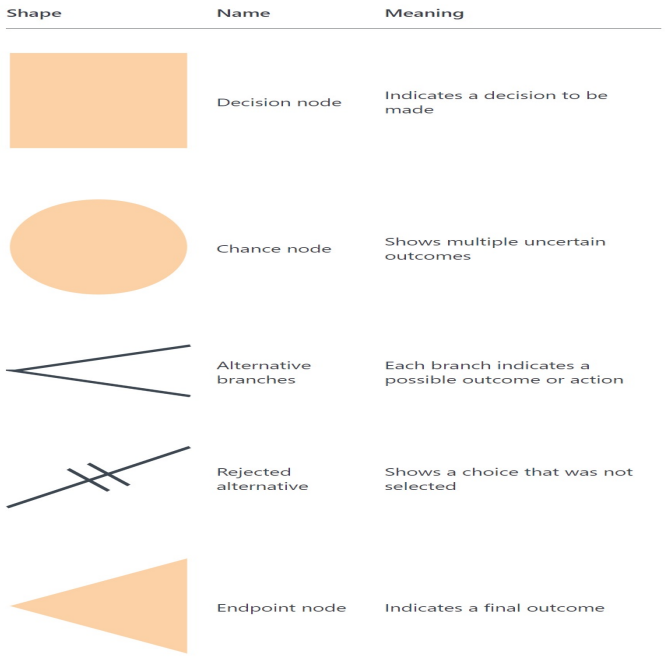



Figure 2.2: Decision Tree symbols (lucidchart.com)

(its alternatives, probabilities and costs) and their preferences for outcomes. They help determine worst, best and expected values for different scenarios, use a white box model. If a given result is provided by a model, it can be combined with other decision techniques. To describe the disadvantages of decision trees, they are unstable, meaning that a small change in the data can lead to a large change in the structure of the optimal decision tree. Many other predictors perform better with similar data. This can be remedied by replacing a single decision tree with a random forest of decision trees, but a random forest is not as easy to interpret as a single decision tree. For data including categorical variables with different number of levels, information gain in decision trees is biased in favor of those attributes with more levels. Calculations can get very complex, particularly if many values are uncertain and/or if many outcomes are linked[15].

Tree models where the target variable can take a discrete set of values are called classification trees; in these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels. Decision trees where the target variable can take continuous values (typically real numbers) are called regression trees. A decision tree

[15] `https://en.wikipedia.org/wiki/Decision_tree`

is a simple representation for classifying examples. Assume that all of the input features have finite discrete domains and there is a single target feature called the *classification*. Each element of the domain of the classification is called a class. A decision tree or a classification tree is a tree in which each internal (non-leaf) node is labeled with an input feature. The arcs coming from a node labeled with an input feature are labeled with each of the possible values of the target or output feature or the arc leads to a subordinate decision node on a different input feature. Each leaf of the tree is labeled with a class or a probability distribution over the classes, signifying that the data set has been classified by the tree into either a specific class or into a particular probability distribution (which, if the decision tree is well-constructed, is skewed towards certain subsets of classes). A tree is built by splitting the source set, constituting the root node of the tree, into subsets - which constitute the successor children. The splitting is based on a set of splitting rules based on classification features. This process is repeated on each derived subset in a recursive manner called recursive partitioning. The recursion is completed when the subset at a node has all the same values of the target variable or when splitting no longer adds value to the predictions. This process of top-down induction of decision trees (TDIDT) is an example of a greedy algorithm and it is by far the most common strategy for learning decision trees from data. Classification tree analysis is when the predicted outcome is the class (discrete) to which the data belongs. Regression tree analysis is when the predicted outcome can be considered a real number (e.g. the price of a house or a patient's length of stay in a hospital). The term Classification And Regression Tree (CART) analysis is an umbrella term used to refer to both of the procedures. Some techniques, often called ensemble methods, construct more than one decision tree: Boosted trees Incrementally building an ensemble by training each new instance to emphasize the training instances previously mismodeled. A typical example is AdaBoost. These can be used for regression-type and classification-type problems. Secondly, Bootstrap aggregated (or bagged) decision trees, an early ensemble method, builds multiple decision trees by repeatedly resampling training data with replacement and voting the trees for a consensus prediction. Next, a random forest classifier is a specific type of bootstrap aggregating and lastly, Rotation forest – in which every decision tree is trained by first applying principal component analysis (PCA) on a random subset of the input features[16].

[16] `https://en.wikipedia.org/wiki/Decision_tree_learning`

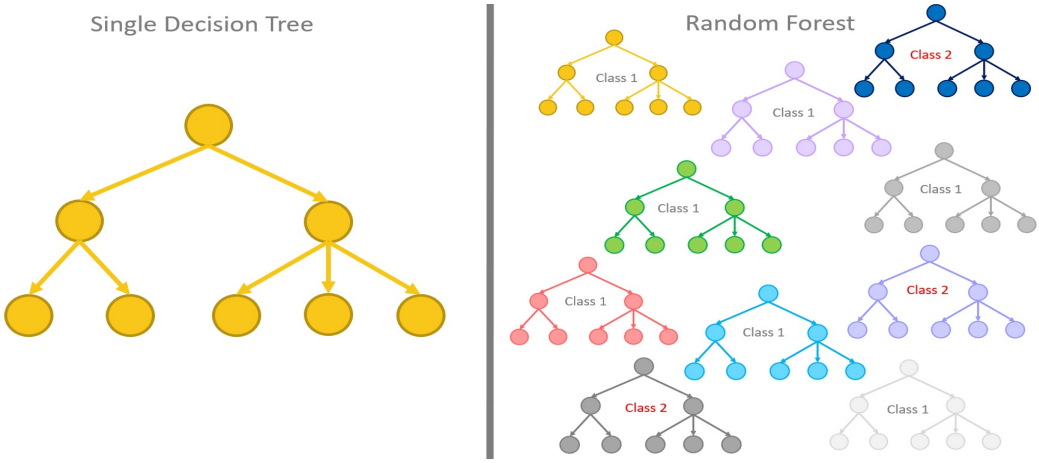



Figure 2.3: General Decision Tree and Random Forest structure (towards-datascience.com)

### Random Forest

Random forests or random decision forests are an ensemble learning method as shown in the Figure 2.3[17] for classification, regression and other tasks that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set. Decision trees are a popular method for various machine learning tasks. Tree learning comes closest to meeting the requirements for serving as an off-the-shelf procedure for data mining because it is invariant under scaling and various other transformations of feature values, is robust to inclusion of irrelevant features and produces inspectable models. However, they are seldom accurate. In particular, trees that are grown very deep tend to learn highly irregular patterns: they overfit their training sets, i.e. have low bias, but very high variance. Random forests are a way of averaging multiple deep decision trees, trained on different parts of the same training set, with the goal of reducing the variance. This comes at the expense of a small increase in the bias and some loss of interpretability, but generally greatly boosts the performance in the final model[18].

[17] https://towardsdatascience.com/from-a-single-decision-tree-to-a-random-forest-b9523be65147
[18] https://en.wikipedia.org/wiki/Random_forest

**k - NN**

The k-nearest neighbors algorithm (k-NN)[19] is a non-parametric method used for classification and regression. In both cases, the input consists of the k closest training examples in the feature space. The output depends on whether k-NN is used for classification or regression:

- In k-NN classification, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). If $k = 1$, then the object is simply assigned to the class of that single nearest neighbor.

- In k-NN regression, the output is the property value for the object. This value is the average of the values of k nearest neighbors.

k-NN is a type of instance-based learning or lazy learning, where the function is only approximated locally and all computation is deferred until function evaluation. Both for classification and regression, a useful technique can be to assign weights to the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. For example, a common weighting scheme consists in giving each neighbor a weight of 1/d, where d is the distance to the neighbor. The neighbors are taken from a set of objects for which the class (for k-NN classification) or the object property value (for k-NN regression) is known. This can be thought of as the training set for the algorithm, though no explicit training step is required. A peculiarity of the k-NN algorithm is that it is sensitive to the local structure of the data.

The training examples are vectors in a multidimensional feature space, each with a class label. The training phase of the algorithm consists only of storing the feature vectors and class labels of the training samples. In the classification phase, k is a user-defined constant and an unlabeled vector (a query or test point) is classified by assigning the label which is most frequent among the k training samples nearest to that query point. A commonly used distance metric for continuous variables is Euclidean distance. For discrete variables, such as for text classification, another metric can be used, such as the overlap metric (or Hamming distance). In the context of gene expression microarray data, for example, k-NN has been employed with correlation coefficients, such as Pearson and Spearman, as a metric. Often, the classification accuracy of k-NN can be improved significantly if the distance metric is

[19] `https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm`

learned with specialized algorithms such as Large Margin Nearest Neighbor or Neighbourhood components analysis.

A drawback of the basic majority voting classification occurs when the class distribution is skewed. That is, examples of a more frequent class tend to dominate the prediction of the new example, because they tend to be common among the k nearest neighbors due to their large number. One way to overcome this problem is to weight the classification, taking into account the distance from the test point to each of its k nearest neighbors. The class (or value, in regression problems) of each of the k nearest points is multiplied by a weight proportional to the inverse of the distance from that point to the test point. Another way to overcome skew is by abstraction in data representation. For example, in a self-organizing map (SOM), each node is a representative (a center) of a cluster of similar points, regardless of their density in the original training data. K-NN can then be applied to the SOM.

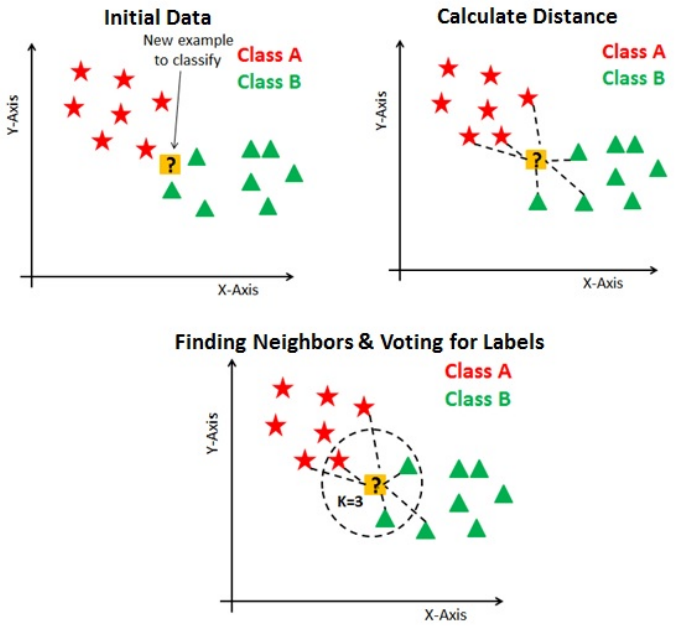



Figure 2.4: K-NN (datacamp.com)

The best choice of $k$ depends upon the data as shown in the Figure 2.4[20]; generally, larger values of $k$ reduces effect of the noise on the classification, but make boundaries between classes less distinct. A good k can be selected by various heuristic techniques. The special case where the class is predicted

[20] https://www.datacamp.com/community/tutorials/k-nearest-neighbor-classification-scikit-learn

to be the class of the closest training sample (i.e. when $k = 1$) is called the nearest neighbor algorithm.

The accuracy of the k-NN algorithm can be severely degraded by the presence of noisy or irrelevant features or if the feature scales are not consistent with their importance. Much research effort has been put into selecting or scaling features to improve classification. A particularly popular approach is the use of evolutionary algorithms to optimize feature scaling. Another popular approach is to scale features by the mutual information of the training data with the training classes. In binary (two class) classification problems, it is helpful to choose k to be an odd number as this avoids tied votes. One popular way of choosing the empirically optimal k in this setting is via bootstrap method. The K-nearest neighbor classification performance can often be significantly improved through (supervised) metric learning. Popular algorithms are neighbourhood components analysis and large margin nearest neighbor. Supervised metric learning algorithms use the label information to learn a new metric or pseudo-metric.

### Naive Bayes

Naive Bayes classifiers[21] are a family of simple probabilistic classifiers based on applying Bayes' theorem with strong (naïve) independence assumptions between the features. They are among the simplest Bayesian network models. Naive Bayes has been studied extensively since the 1960s. It was introduced (though not under that name) into the text retrieval community in the early 1960s and remains a popular (baseline) method for text categorization, the problem of judging documents as belonging to one category or the other (such as spam or legitimate, sports or politics, etc.) with word frequencies as the features. With appropriate pre-processing, it is competitive in this domain with more advanced methods including support vector machines. It also finds application in automatic medical diagnosis. Naive Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. Maximum-likelihood training can be done by evaluating a closed-form expression, which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers. In the statistics and computer science literature, naive Bayes models are known under a variety of names, including simple Bayes and independence Bayes.

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature val-

[21] `https://en.wikipedia.org/wiki/Naive_Bayes_classifier`

ues, where the class labels are drawn from some finite set. There is not a single algorithm for training such classifiers, but a family of algorithms based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable. For example, a fruit may be considered to be an apple if it is red, round and about 10 cm in diameter. A naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible correlations between the color, roundness and diameter features. For some types of probability models, naive Bayes classifiers can be trained very efficiently in a supervised learning setting. In many practical applications, parameter estimation for naive Bayes models uses the method of maximum likelihood; in other words, one can work with the naive Bayes model without accepting Bayesian probability or using any Bayesian methods. Despite their naive design and apparently oversimplified assumptions, naive Bayes classifiers have worked quite well in many complex real-world situations. In 2004, an analysis of the Bayesian classification problem showed that there are sound theoretical reasons for the apparently implausible efficacy of naive Bayes classifiers. Still, a comprehensive comparison with other classification algorithms in 2006 showed that Bayes classification is outperformed by other approaches, such as boosted trees or random forests. An advantage of naive Bayes is that it only requires a small number of training data to estimate the parameters necessary for classification.

## 2.3 Feature Extraction Techniques

Image Processing involved feature extraction techniques are discussed.

### Local Binary Pattern

Local Binary Patterns (LBP)[22] is a type of visual descriptor used for classification in computer vision. LBP is the particular case of the Texture Spectrum model proposed in 1990. LBP was first described in 1994, it has since been found to be a powerful feature for texture classification. It has further been determined that when LBP is combined with the Histogram of Oriented Gradients (HOG) descriptor, it improves the detection performance considerably on some datasets. The LBP feature vector, in its simplest form as shown in Figure 2.5, is created in the following manner:

- Divide the examined window into cells (e.g. 16x16 pixels for each cell).

[22] `https://en.wikipedia.org/wiki/Local_binary_patterns`

- For each pixel in a cell, compare the pixel to each of its 8 neighbors (on its left-top, left-middle, left-bottom, right-top, etc.). Follow the pixels along a circle, i.e. clockwise or counter-clockwise.
- Where the center pixel's value is greater than the neighbor's value, write 0. Otherwise, write 1. This gives an 8-digit binary number (which is usually converted to decimal for convenience).
- Compute the histogram, over the cell, of the frequency of each number occurring (i.e., each combination of which pixels are smaller and which are greater than the center). This histogram can be seen as a 256-dimensional feature vector.
- Optionally normalize the histogram.
- Concatenate (normalized) histograms of all cells. This gives a feature vector for the entire window.

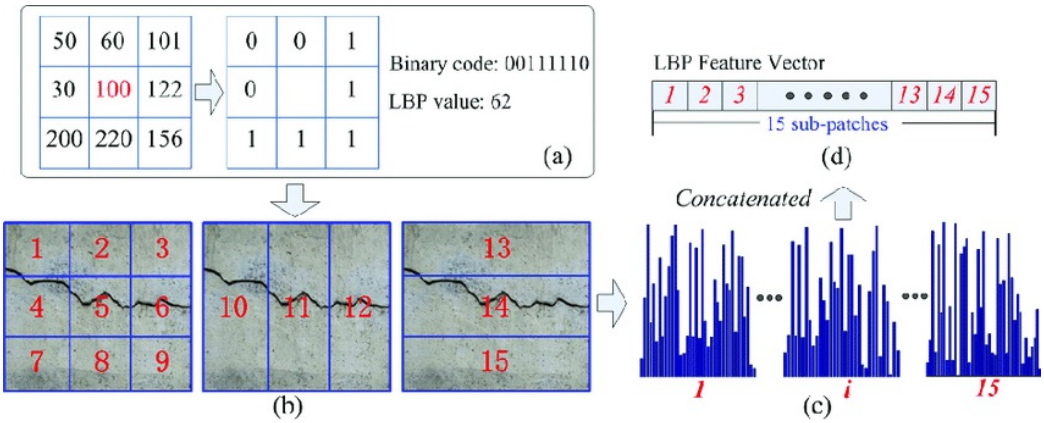



Figure 2.5: LBP feature extraction (Wang et al. (2018))

The feature vector can now be processed using the Support vector machine, extreme learning machines, or some other machine learning algorithm to classify images. Such classifiers can be used for face recognition or texture analysis. A useful extension to the original operator is the so-called uniform pattern, which can be used to reduce the length of the feature vector and implement a simple rotation invariant descriptor. This idea is motivated by the fact that some binary patterns occur more commonly in texture images than others. A local binary pattern is called uniform if the binary pattern contains at most two 0-1 or 1-0 transitions.

### Maximally Stable Extremal Regions

Maximally Stable Extremal Regions (MSER)[23] are used as a method of blob detection in images as shown in Figure 2.6. This technique was proposed to find correspondences between image elements from two images with different viewpoints. This method of extracting a comprehensive number of corresponding image elements contributes to the wide-baseline matching, and it has led to better stereo matching and object recognition algorithms.The MSER algorithm has been adapted to colour images, by replacing thresholding of the intensity function with agglomerative clustering, based on colour gradients. The MSER algorithm can be used to detect regions based on color as opposed to intensity. This is done by creating an intensity function for red, green, and blue in the HSV color space. The MSER algorithm is then run five times; over the three color pseudo-intensities and then over the grey scale intensities using the standard MSER+ and MSER- functions. The MSER algorithm can be used to track colour objects, by performing MSER detection on the Mahalanobis distance to a colour distribution. By detecting MSERs in multiple resolutions, robustness to blur, and scale change can be improved.

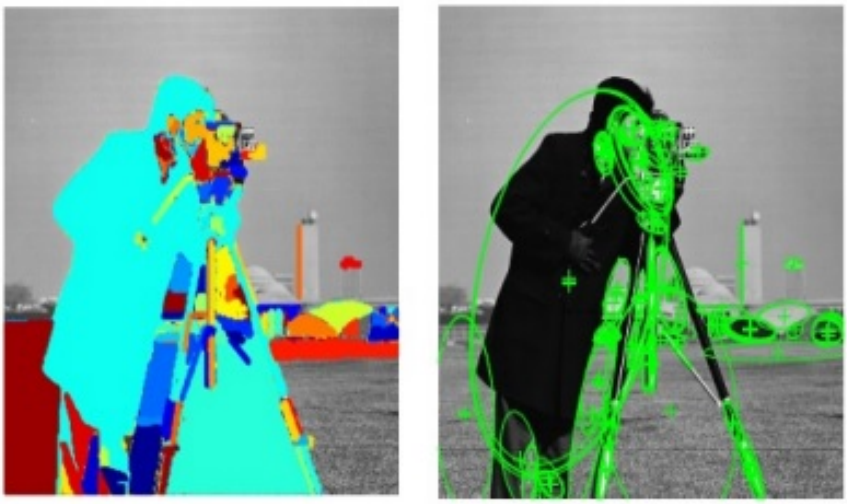


Figure 2.6: MSER features with strong regions (mathworks.com)

The Figure 2.6[24] shows the MSER features from an image, it shows the extracted MSER regions of an input image also the strongest regions with the ellipses and centroids that fit into the MSER regions.

[23] https://en.wikipedia.org/wiki/Maximally_stable_extremal_regions
[24] https://de.mathworks.com/help/vision/ref/detectmserfeatures.html

### KAZE

KAZE is a novel multiscale 2D feature detection and description algorithm in nonlinear scale spaces. It exploits non-linear scale space through non-linear diffusion filtering. This makes blurring in images locally adaptive to feature-points, thus reducing noise and simultaneously retaining the boundaries of regions in subject images. KAZE detector is based on scale normalized determinant of Hessian Matrix which is computed at multiple scale levels. The maxima of detector response are picked up as feature-points using a moving window. Feature description introduces the property of rotation invariance by finding dominant orientation in a circular neighborhood around each detected feature. KAZE features are invariant to rotation, scale, limited affine and have more distinctiveness at varying scales with the cost of moderate increase in computational time Tareen and Saleem (2018).

### Speeded Up Robust Feature

Speeded Up Robust Features (SURF)[25] is a patented local feature detector and descriptor. It can be used for tasks such as object recognition, image registration, classification, or 3D reconstruction. It is partly inspired by the Scale-Invariant Feature Transform (SIFT) descriptor. The standard version of SURF is several times faster than SIFT and claimed by its authors to be more robust against different image transformations than SIFT. To detect interest points, SURF uses an integer approximation of the determinant of Hessian blob detector, which can be computed with 3 integer operations using a pre-computed integral image. Its feature descriptor is based on the sum of the Haar wavelet response around the point of interest. These can also be computed with the aid of the integral image. SURF descriptors have been used to locate and recognize objects, people or faces, to reconstruct 3D scenes, to track objects and to extract points of interest. The image is transformed into coordinates, using the multi-resolution pyramid technique, to copy the original image with Pyramidal Gaussian or Laplacian Pyramid shape to obtain an image with the same size, but with reduced bandwidth. This achieves a special blurring effect on the original image, called Scale-Space and ensures that the points of interest are scale invariant.

[25] `https://en.wikipedia.org/wiki/Speeded_up_robust_features`

### Histogram of Oriented Gradients

Histogram of Oriented Gradients (HOG)[26] is a feature descriptor used in computer vision and image processing for the purpose of object detection. The technique counts occurrences of gradient orientation in localized portions of an image. This method is similar to that of edge orientation histograms, scale-invariant feature transform descriptors, and shape contexts, but differs in that it is computed on a dense grid of uniformly spaced cells and uses overlapping local contrast normalization for improved accuracy. The essential thought behind the histogram of oriented gradients descriptor is that local object appearance and shape within an image can be described by the distribution of intensity gradients or edge directions. The image is divided into small connected regions called cells, and for the pixels within each cell, a histogram of gradient directions is compiled. The descriptor is the concatenation of these histograms. For improved accuracy, the local histograms can be contrast-normalized by calculating a measure of the intensity across a larger region of the image, called a block, and then using this value to normalize all cells within the block. This normalization results in better invariance to changes in illumination and shadowing. The HOG descriptor has a few key advantages over other descriptors. Since it operates on local cells, it is invariant to geometric and photometric transformations, except for object orientation.

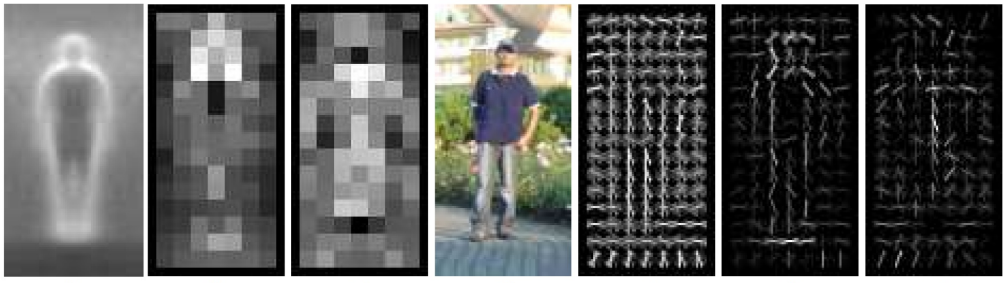



Figure 2.7: HOG feature extraction (Navneet Dalal (2005))

The Figure 2.7 shows HOG detectors cue mainly on silhouette contours (especially the head, shoulders and feet). The most active blocks are centred on the image background just outside the contour. Figure 2.7(a): The average gradient image over the training examples. Figure 2.7(b): Each pixel shows the maximum positive SVM weight in the block centred on the pixel. Figure 2.7(c): Likewise for the negative SVM weights. Figure 2.7(d): A test

[26] https://en.wikipedia.org/wiki/Histogram$_o f_o riented_g radients$

image. Figure 2.7(e): It's computed R-HOG descriptor. Figure 2.7(f,g): The R-HOG descriptor weighted by respectively the positive and the negative SVM weights Navneet Dalal (2005).

## 2.4 Related Work

Over a period of time, Machine Learning algorithms have been widely used to solve various problems. ML in NDT industry has not been widely used yet, but it is desired to introduce AI in the future. We have grasped a lot of knowledge from the previous research work of various authors in the field of NDT alike, defect detection on distinguished dataset, use of appropriate ML algorithms for NDT types of data. A few research works we would like to discuss which advised to form our model. In the research work of Zhang et al. (2005), the data set consists of ultrasonic C-Scan images of IC packaging, they use SVM with Mumford-Shah[27] model for image segmentation and feature extraction to detect defects. An empirical comparison between ten supervised learning methods: SVMs, neural nets, logistic regression, naive bayes, memory-based learning, random forests, decision trees, bagged trees, boosted trees, and boosted stumps are well explained for various datasets in Caruana and Niculescu-Mizil (2006), which allows us to decide which supervised learning algorithm would fit best for our dataset. Another such research work of Bernieri et al. (2006) leads to a measurement station for Non-Destructive Testing on conductive materials based on Eddy Current[28] approach. It is characterized by an integration between a suitable measurement hardware with a powerful elaboration procedure using a Support Vector Machine (SVM) based algorithm. This integration performs the defect detection, location and characterization in terms of crack length, height and depth in the specimen with noticeable accuracy. In Bernieri et al. (2008) a method is proposed for defect characterization in NDT of conductive materials based on the Eddy Current Testing (ECT)[29]. It is based on an NDT measurement station and a suitable processing procedure using machine learning systems. The comparison between two machine learning systems [an Artificial Neural Network (ANN) and a SVM for Regression (SVR)], both in a simulated environment and in real cases, proved that the SVR is better for solving crack characterization problems in ECT. The work of Wei and Cheng-Tong (2009) explains us a real-time processing and SVM prediction algorithm, they have

[27]For more details of this technique refer: `http://www.its.caltech.edu/~matilde/MumfordShahRecentSurvey.pdf`

[28]For detail description, refer García-Martín et al. (2011)

[29]For detail description, refer García-Martín et al. (2011)

developed a software to process and display the classification and grade of the rail flaws. D.Benitez et al. (2009) explains defect characterization in infrared (IR) NDT data with Machine Learning. They have designed a reference-free thermal contrast by using the thermal quadrupoles [30] theory and evaluate the limit of defect detection in composite samples by using dynamic principal components analysis (DPCA) and k-nearest neighbor algorithm. They also use the radial basis functions (RBF) networks and support vector machines (SVM) for validating the detection and quantification of defect depth in composite material samples affected by non-uniform heating and with complex shapes.

The Khodayari-Rostamabad et al. (2009) research explains the analysis of magnetic flux leakage images in pipeline inspection using k-nearest neighbour, SVR and managed to obtained promising results. In Shumin et al. (2011) work, fabric defect detection scheme based on HOG and SVM is explained. Firstly, each block-based feature of the image is encoded using the histograms of orientated gradients (HOG), which are insensitive to various lighting and noises. Then, a powerful feature selection algorithm, AdaBoost[31], is performed to automatically select a small set of discriminative HOG features in order to achieve robust detection results. In the end, support vector machine (SVM) is used to classify the fabric defects. The combination of HOG with SVM has produced promising results. The Saechai et al. (2012) describes about a test system for structure inspection and evaluation of cement-based products by applying ultrasonic test with support vector machine (SVM) classifier. Well, in D'Angelo and Rampone (2013) work classifies the aerospace structure defects detected by Eddy current non-destructive testing. This method is based on the assumption that the defect is bound to the reaction of the probe coil impedance during the test. Impedance plane analysis is used to extract a feature vector from the shape of the coil impedance in the complex plane, through the use of some geometric parameters. Shape recognition is tested with three different machine-learning based classifiers: decision trees, neural networks and Naive Bayes. The performance of this detection system are measured in terms of accuracy, sensitivity, specificity, precision and Matthews correlation coefficient[32].

Moving further with our literature survey, we have got insights from Medjahed (2015) as they proposed a comparison protocol of several feature extraction techniques under different classifiers, as well evaluate the performance of feature extraction techniques in the context of image classification tested on

[30]For more explaination refer: Maillet et al. (2000)

[31]For detailed explaination refer: `https://cseweb.ucsd.edu/~yfreund/papers/IntroToBoosting.pdf`

[32]To understand it better, refer Boughorbel et al. (2017)

both binary and multi-class classifications. The analyses of performance were conducted in terms of: classification accuracy rate, recall, precision, F-score or F-measure or F1. This paper gave us a thorough survey what suits best for our problem. Further major remarks made by A.Sumesh et al. (2015) was useful for our work, as it explains about an experimental set up that established to carry out Shielded Metal Arc Welding (SMAW)[33] of Carbon Steel plates with an objective of correlating arc sound with good weld and weld with lack of fusion and burn through. Arc sound signal was recorded while welding and the raw data of sound signal was generated. The statistical features extracted from the raw data were given as an input to the classifier for classifying the features as good weld and the weld with burn through and lack of fusion. Two different classifier algorithms, J48[34] and Random forest were used for classification and the results were compared. In Malekzadeh et al. (2016), an automatic image-based aircraft defect detection using Deep Neural Networks (DNNs), with evaluation of state-of-the-art feature descriptors and show that the best performance is achieved by vgg-f DNN as feature extractor with a linear SVM classifier. To reduce the processing time, SURF key point detector is used to identify defect patch candidates. Research of Huang et al. (2017) have designed a complete Mobilephone Panel Surface Defects Detection (MPSDD) framework based on both Machine Vision and Machine Learning. Local Binary Patterns (LBP) and Histogram of Oriented Gradients (HOG) are used as feature extraction techniques in combination of Naive Bayes and SVM. Results show that the combination of HOG with SVM have achieved high accuracy for classification.

A competitive research work is Brehm and Lienhart (2018) which is designed to detect anomalies in ultrasonic images of fiber-metal laminate skin fields. This research work uses the same dataset as of our problem. It uses Deep Neural Network approach to classify defect and good aircraft parts. The dataset is formed by visualizing the ultrasonic scans as gray-level images (converted .nkc (Sec. 2.1) and .nkd files). Atrous Spatial Pyramid Pooling (ASPP)[35] is used as it allows different large catchment areas for the final pixel-by-pixel classification. This system achieves high exclusion rates of up to 97.36% on the test data. the performance of this system deviates from the performance of the test set due to the very small amount of test data used in productive use. Deep neural networks can be easily fooled with less meaningful data as explained in the research of Nguyen et al. (2015).

A great support to our work is based on the Apmann et al. (2016) and

[33]For explaination, refer: https://www.sciencedirect.com/topics/materials-science/shielded-metal-arc-welding

[34]Refer: https://www.sciencedirect.com/science/article/pii/S1877050918309207

[35]For more information, refer: https://www.analyticsvidhya.com/blog/2019/02/tutorial-semantic-segmentation-google-deeplab/

Schmidt et al. (2015). Even though they experiment on different type of dataset, we could extract an adequate amount of knowledge for our model design.

## 2.5 Conclusion

In this chapter, we have explained the basics of Non-Destructive Testing (NDT) data and its format in (Sec. 2.1). Introducing Machine learning and its types in (Sec. 2.2), as well as Image Processing for feature extraction and various techniques involved in (Sec. 2.3) that makes it possible to understand our approach and results.

# Chapter 3

# Design

In this chapter, we discuss our proposed design for the ML model from the knowledge we have obtained from previous research works, discussed in Chapter 2.4.

## 3.1 Motivation

In order to classify good and flaw aircraft parts, we use traditional ML methods as described in the Chapter 2.2. ML methodologies necessarily have a training set and a test set. Feature extraction is performed on both training and test sets. For the classification, feature vectors from the training set are used to train a classification model and tested against the test set feature vectors.

## 3.2 Architecture

The selection of the ML algorithm to be used is a challenge. Choosing the right algorithm can seem difficult because there are dozens of supervised and unsupervised ML algorithms and each uses a different approach to learning. There is no best method and none that is always suitable. The selection of the right algorithm consists in part of trial and error - even very experienced data scientists can only speculate whether an algorithm is suitable without having tried it. However, choosing an algorithm also depends on the amount and type of data to work with, the insights you want to get from the data, and how you plan to use those insights. Highly flexible models tend to overfit data by modeling minor variations that could be noise. Simple models are easier to interpret, but might have lower accuracy. Therefore, choosing the right algorithm requires trading off one benefit against another, including

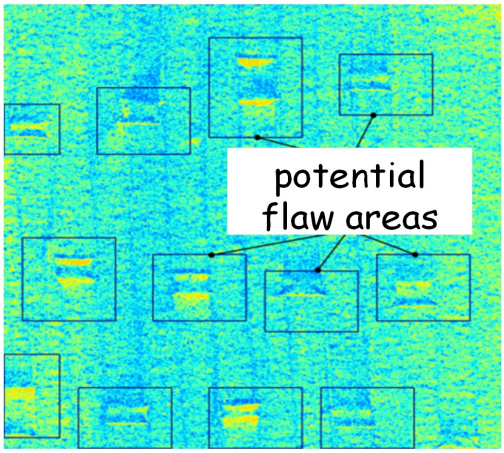



Figure 3.1: Example of a flaw image with potential flaw areas (Apmann et al. (2016))

model speed, accuracy, and complexity. Trial and error is at the core of ML — if one approach or algorithm does not work, we try another.

The architecture constitutes two parts: training and test sets. The training set has both good and flaw image dataset, feature extraction with various techniques as explained in Chapter 2.3 is performed on this set in combination with various ML classifiers as explained in Chapter 2.2. The test set contains both good and majority flaw image dataset, feature extraction technique is the same as used in the training set and respectively trained ML classifiers are validated. An algorithm for our research work proposed is as follows and these steps are followed for each feature extraction technique in combination with ML classifiers:

1. Input images (converted from raw data (Chapter 2.1) as shown in Figure 3.1) – True color or RGB images.

2. Extraction of image features – Local Binary Pattern (LBP), Maximally Stable Extremal Regions (MSER), KAZE, Speeded Up Robust Features (SURF), Histogram of Oriented Gradients (HOG) as explained in Chapter 2.3.

3. Train ML classifiers – k-Nearest Neighbour, Naive Bayes, SVM, Random Forest as explained in Chapter 2.2.

4. Predict using test data set over trained ML classifiers – check for best results.

5. Research towards certification.

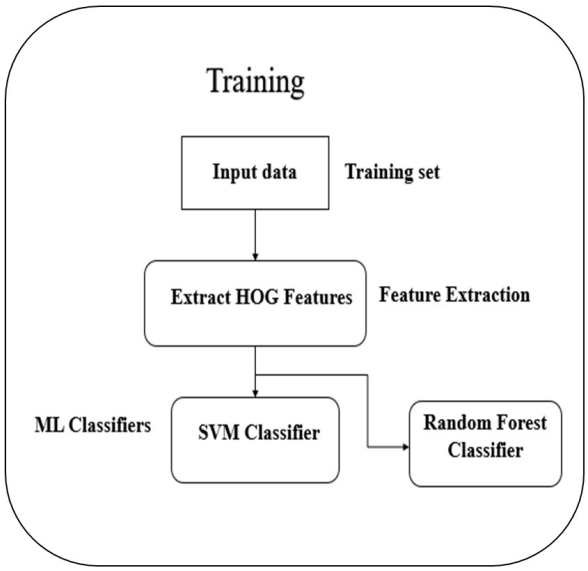



Figure 3.2: Training Methodology

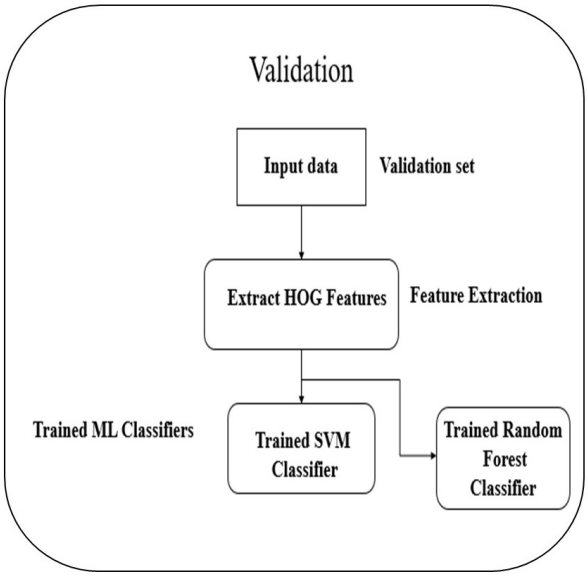



Figure 3.3: Validation Methodology

## 3.3 Model

Our proposed model uses various feature extraction techniques as explained in Chapter 2.3 with distinguished ML classifiers as discussed in Chapter 2.2 in order to obtain best results for the given dataset (Chapter 2.1). An example of our designed model for training is shown in Figure 3.2 and for validation is shown in Figure 3.3. In the feature extraction block, various feature extraction techniques can be used such as HOG, LBP as discussed in Chapter 2.3, likewise in the ML classifiers block SVM, Random Forest or k-Nearest Neighbour as explained in Chapter 2.2 can be used.

One vital condition is that the validation image dataset has to undergo the same feature extraction technique that was used in the training image dataset as well the same trained ML classifier has to be validated with the validation set. The use of different feature extraction techniques versus different ML classifiers, in short, benchmarking gives us a broad chance to choose which combination is best suited of our problem.

## 3.4 Conclusion

In this chapter we outline the architecture of our proposed model in Chapter 3.2 and its design in Chapter 3.3 with the knowledge that we acquired from Chapter 2.4. In the next chapter, we will discuss about the implementation of our proposed designed model.

# Chapter 4

# Experiments

In this chapter, we would like to discuss the possible experiments that we conducted using our proposed design (Chapter 3). The outcome of our experiments will be discussed in the next chapter, Chapter 5.

## 4.1 Motivation

In this chapter, we discuss the implementation procedure for our proposed model design. Our main goal is to obtain the best results for our problem. We analyze which model fits best to our image dataset and use this model for further research in certification process.

## 4.2 Dataset

The dataset is formed by images as described in Chapter 2.1. The datasets are divided into two parts: training dataset and testing dataset. We have used the same training set and test set (hereby referred to as: new dataset) as used in Brehm and Lienhart (2018) as our work aims to compare their methodology with our proposed work. We have tried to use 80% out of the entire dataset for training and 20% for validation, experiments prove that no significant changes could be recorded when in comparison to this dataset formed.

As described in Chapter 2.1, the raw dataset was converted to an image dataset. The image file format used for our experiments are *jpeg* (Joint Photographic Experts Group)[1]. Another possible image dataformat that

[1] `https://en.wikipedia.org/wiki/JPEG`

could be used is *bmp* (BitMaP)[2] image file format. We have conducted experiments to check the data loss incurred due to the use of *jpeg* image file format and results prove that in worst case only 1.5% of data loss occurred, this data loss had no influence on the ML classifier's accuracy rate.

The training dataset for our proposed method constitutes of 189 defect images (hereby referred as: positive image dataset) and 231 good images (hereby referred as: negative image dataset). The validation set consists of 15 images out of which major images are from the positive image dataset.

## 4.3 Experimental setup

We have used MATLAB R 2019a for all our experiments.

**Labelling data - Image Labeler App**

The labelling of the data was conducted by Image Labeler App of MATLAB R 2019a. The Image Labeler app enables us to label ground truth data in a collection of images. Using the app, we can:

i. Define rectangular regions of interest (ROI) labels, polyline ROI labels, pixel ROI labels, and scene labels. Use these labels to interactively label your ground truth data.

ii. Use built-in detection or tracking algorithms to label your ground truth data.

iii. Write, import, and use your own custom automation algorithm to automatically label ground truth. See Create Automation Algorithm for Labeling.

iv. Evaluate the performance of your label automation algorithms using a visual summary.

v. Export the labeled ground truth as a groundTruth object. You can use this object for system verification or for training an object detector or semantic segmentation network.

The Image Labeler app supports all image file formats supported by imread(). The Image Labeler app provides an easy way to mark rectangular region of interest (ROI) labels, polyline ROI labels, pixel ROI labels and

[2] `https://en.wikipedia.org/wiki/BMP_file_format`

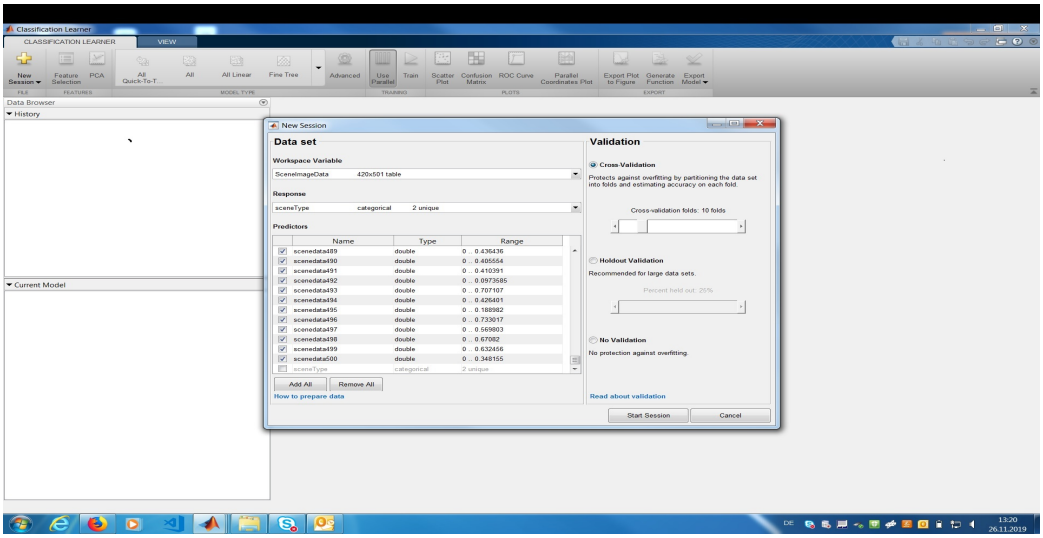



Figure 4.1: Classification Learner App

scene labels in a video or image sequence. This app shows us how to:

i. Manually label an image frame from an image collection.

ii. Automatically label across image frames using an automation algorithm.

iii. Export the labeled ground truth data.

An ROI label corresponds to either a rectangular, polyline, or pixel region of interest. These labels contain two components: the label name, such as "defect" and the region we create. A Scene label describes the nature of a scene, such as *defect part*, we can associate this label with a frame.

Positive (defect) labels were pre-defined using ULTIS app as describe in Chapter 2.1. The location of the defect and its area was defined by the Industry Examination Report and was labeled accordingly. For the negative (good) labels the entire image was considered as a label.

### Machine Learning Models - Classification Learner App

We have also used the Classification Learner App as shown in Figure 4.1 for training ML classifiers. Use the Classification Learner app to train models to classify data using supervised machine learning. The app lets us explore supervised machine learning interactively using various classifiers:

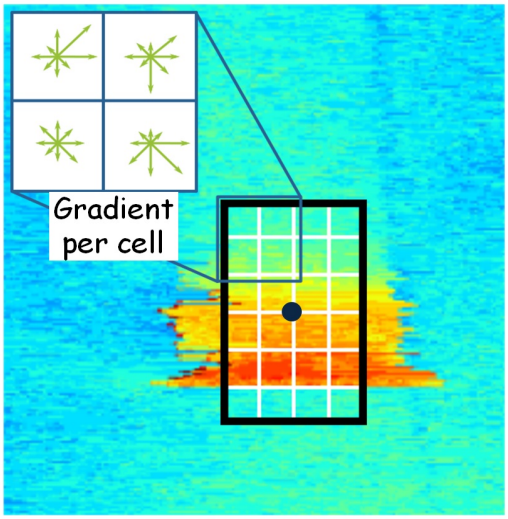



Figure 4.2: HOG from sample dataset (Apmann et al. (2016))

i. Automatically trains a selection of models and helps you choose the best model. Model types include decision trees, discriminant analysis, support vector machines, logistic regression, nearest neighbors, naive Bayes and ensemble classification.

ii. Explores our data, specifies validation schemes, selects features, and visualizes results. By default, the app protects against overfitting by applying cross-validation. Alternatively, we can select holdout validation. Validation results help us choose the best model for our data. Plots and performance measures reflect the validated model results.

iii. Exports models to the workspace to make predictions with new data. The app always trains a model on full data in addition to a model with the specified validation scheme and the full model is the model we export.

iv. Generates MATLAB code from the app to create scripts, trains with new data, works with huge data sets, or modifies the code for further analysis.

Training a model in Classification Learner consists of two parts:

1. Validated Model: Train a model with a validation scheme. By default, the app protects against overfitting by applying cross-validation. Alternatively, you can choose holdout validation. The validated model is visible in the app.

2. Full Model: Train a model on full data without validation. The app trains this model simultaneously with the validated model. However, the model trained on full data is not visible in the app. When you choose a classifier to export to the workspace, Classification Learner exports the full model.

The app displays the results of the validated model. Diagnostic measures, such as model accuracy and plots, such as a scatter plot or the confusion matrix chart, reflect the validated model results. You can automatically train a selection of or all classifiers, compare validation results and choose the best model, that works for your classification problem. When you choose a model to export to the workspace, Classification Learner exports the full model. Because Classification Learner creates a model object of the full model during training, you experience no lag time when you export the model. We can use the exported model to make predictions on new data.

The HOG feature extraction is performed on all training data and validation set. Random feature selection was experimented to check the performance measures of the training models, but all features from the vectors of the training set were considered. One such example is shown in Figure 4.2. The calculated positive (defect data parts) and negative (good data parts) feature vectors from the training set are used to train the ML models. The feature vectors from the validation set are used to test on the trained ML models and their performance is measured as explained in the next Chapter 5.

## 4.4 Conclusion

In this Chapter we have discussed the experimental method that was conducted to obtain best results and these will be discussed in the next Chapter 5.

# Chapter 5

# Results

In this chapter we would like to discuss the results that were obtained from our experiments as explained in Chapter 4. Image feature extraction is performed on the input true-color dataset. Each image has 34596 HOG features, in total ($420 \times 34596$) feature vectors were used for training both SVM and Random Forest and also other ML algorithms. The testing HOG feature set had ($15 \times 34596$) HOG feature vectors. For SURF feature extraction, bag-of-features[1] was used and from all 420 images 1209367 feature vectors were extracted and used for ML training. Comparing the number of feature vectors, HOG produces more feature vectors for ML training. We need more feature vectors for training to obtain better accuracy of classification. The best result has been considered for the next research topic, certification, discussed in Chapter 6.

## 5.1 Evaluation Metrics

First, the different evaluation metrics are introduced, that are used to prove which of our proposed methodology is best suited for real-time in Industry. Common Evaluation metrics for Machine Learning models are:

- confusion matrix
- accuracy
- precision and recall
- F-score
- ROC and AUC

[1]For more refer: `https://en.wikipedia.org/wiki/Bag-of-words_model_in_computer_vision`

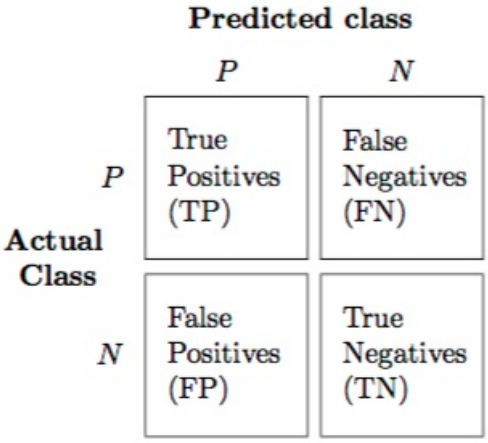



Figure 5.1: Confusion Matrix

### Confusion Matrix

A confusion matrix as shown in Figure 5.1 is a technique for summarizing the performance of a classification algorithm. The number of correct and incorrect predictions are summarized with count values and broken down by each class. This is the key to the confusion matrix. The confusion matrix shows the ways in which our classification model is confused when it makes predictions[2].

To define all the terms in the Figure 5.1: Positive (P): Observation is positive, Negative (N): Observation is not positive, True Positive (TP): Observation is positive, and is predicted to be positive. False Negative (FN): Observation is positive, but is predicted negative. True Negative (TN): Observation is negative, and is predicted to be negative. False Positive (FP): Observation is negative, but is predicted positive.

Using the confusion matrix, accuracy, precision, recall and F-score can be calculated.

### Accuracy

Classification Accuracy (accuracy) is a method for measuring a classification model's performance. It is typically expressed as a percentage. Accuracy is the count of predictions where the predicted value is equal to the true value. It is binary (true/false) for a particular sample. Accuracy is often graphed and monitored during the training phase though the value is often associated with the overall or final model accuracy.

[2]://machinelearningmastery.com/confusion-matrix-machine-learning/

$$Accuracy = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \tag{5.1}$$

$$Accuracy = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{5.2}$$

**Precision**

To get the value of precision, divide the total number of correctly classified positive examples by the total number of predicted positive examples. High Precision indicates an example labeled as positive is indeed positive (small number of FP).

$$Precision = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{5.3}$$

**Recall**

Recall can be defined as the ratio of the total number of correctly classified positive examples divide to the total number of positive examples. High Recall indicates the class is correctly recognized (small number of FN). High recall, low precision: This means that most of the positive examples are correctly recognized (low FN) but there are a lot of false positives. Low recall, high precision: This shows that, there is a lot of positive examples missing (high FN) but those predicted as positive are indeed positive (low FP).

$$Recall = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{5.4}$$

**F-score**

A measure that combines precision and recall. It is the harmonic mean of precision and recall, traditionally called as F-measure or balanced F-score. This measure is approximately the average of the two when they are close and is more generally the harmonic mean, which, for the case of two numbers, coincides with the square of the geometric mean divided by the arithmetic mean.

There are several reasons that the F-score can be criticized in particular circumstances due to its bias as an evaluation metric. This is also known as the $F_1$ measure, because recall and precision are evenly weighted.

$$F = 2 \cdot \frac{\text{precision . recall}}{\text{precision} + \text{recall}} \tag{5.5}$$

### ROC and AUC

An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters: True Positive Rate (recall) and False Positive Rate (FP/(FP + TN)). An ROC curve plots TPR vs. FPR at different classification thresholds. Lowering the classification threshold classifies more items as positive, thus increasing both False Positives and True Positives. The Figure 5.2[3] shows a typical ROC curve.

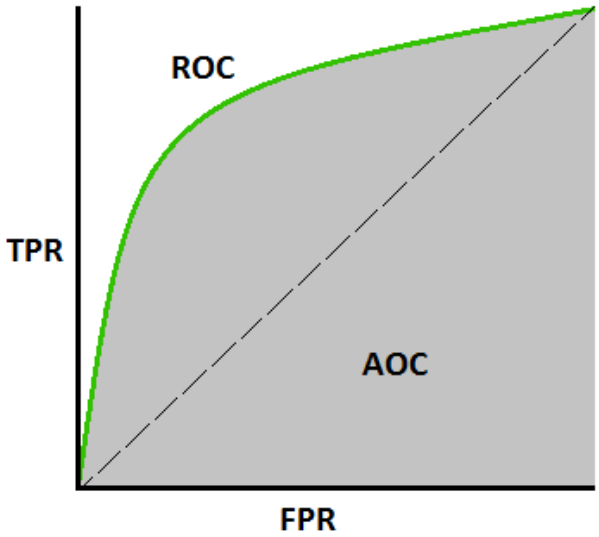



Figure 5.2: ROC-AUC curve (towardsdatascience.com)

AUC stands for "Area under the ROC Curve." AUC measures the entire two-dimensional area underneath the entire ROC curve. It provides an aggregate measure of performance across all possible classification thresholds. One way of interpreting AUC is as the probability that the model ranks a random positive example more highly than a random negative example. AUC ranges in value from 0 to 1. A model whose predictions are 100% wrong has an AUC of 0.0; one whose predictions are 100% correct has an AUC of 1.0. AUC is desirable for the following two reasons: AUC is scale-invariant. It measures how well predictions are ranked, rather than their absolute values. AUC is classification-threshold-invariant. It measures the quality of the model's predictions irrespective of what classification threshold is chosen.

[3] https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5

## 5.2 Evaluation Results

The evaluations of the ML models with corresponding feature extraction techniques are discussed in this section. We begin with the basics of evaluation criterias to understand produced results of our work, which is discussed later.

### Statistical Model Validation

In statistics, model validation is the task of confirming that the outputs of a statistical model are acceptable with respect to the real data-generating process. In other words, model validation is the task of confirming that the outputs of a statistical model have enough fidelity to the outputs of the data-generating process that the objectives of the investigation can be achieved. Model validation can be based on two types of data: data that was used in the construction of the model and data that was not used in the construction. Validation based on the first type usually involves analyzing the goodness of fit of the model or analyzing whether the residuals seem to be random (i.e. residual diagnostics).

Validation based on the second type usually involves analyzing whether the model's predictive performance deteriorates non-negligibly when applied to pertinent new data. Validation based on only the first type (data that was used in the construction of the model) is often inadequate. When doing a validation, there are three notable causes of potential difficulty[4]:

- Lack of data
- Lack of control of the input variables
- Uncertainty about the underlying probability distributions and correlations

### Overfitting and Underfitting

In statistics, overfitting is the production of an analysis that corresponds too closely or exactly to a particular set of data, and may therefore fail to fit additional data or predict future observations reliably. An overfitted model is a statistical model that contains more parameters than can be justified by the data. The essence of overfitting is to have unknowingly extracted some of the residual variation (i.e. the noise) as if that variation represented underlying model structure.

[4] `https://en.wikipedia.org/wiki/Statistical_model_validation`

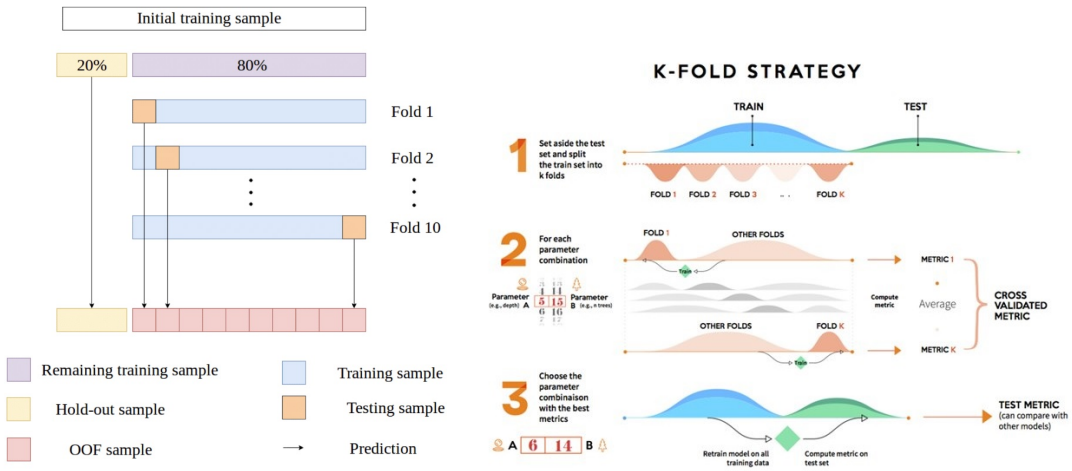
Underfitting occurs when a statistical model cannot adequately capture the underlying structure of the data. An underfitted model is a model where some parameters or terms that would appear in a correctly specified model are missing. Underfitting would occur, for example, when fitting a linear model to non-linear data. Such a model will tend to have poor predictive performance.

Overfitting and underfitting can occur in machine learning, in particular. In machine learning, the phenomena are sometimes called **overtraining** and **undertraining**. The possibility of overfitting exists because the criterion used for selecting the model is not the same as the criterion used to judge the suitability of a model. For example, a model might be selected by maximizing its performance on some set of training data, and yet its suitability might be determined by its ability to perform well on unseen data; then overfitting occurs when a model begins to memorize training data rather than learning to generalize from a trend. Overfitting is especially likely in cases where learning was performed too long or where training examples are rare, causing the learner to adjust to very specific random features of the training data, that have no causal relation to the target function. In this process of overfitting, the performance on the training examples still increases while the performance on unseen data becomes worse. Generally, a learning algorithm is said to overfit relative to a simpler one if it is more accurate in fitting known data (hindsight) but less accurate in predicting new data (foresight). One can intuitively understand overfitting from the fact that information from all past experience can be divided into two groups: information that is relevant for the future and irrelevant information (noise). Everything else being equal, the more difficult a criterion is to predict (i.e., the higher its uncertainty), the more noise exists in past information that needs to be ignored. The problem is determining which part to ignore. A learning algorithm that can reduce the chance of fitting noise is called robust. Underfitting occurs when a statistical model or machine learning algorithm cannot adequately capture the underlying structure of the data. It occurs when the model or algorithm does not fit the data enough. Underfitting occurs if the model or algorithm shows low variance but high bias (to contrast the opposite, overfitting from high variance and low bias). It is often a result of an excessively simple model[5]. To avoid overfitting and underfitting, Cross-validation and Holdout validation techniques are used.

[5] `https://en.wikipedia.org/wiki/Overfitting`

### Cross-validation

Cross-validation[6], sometimes called rotation estimation or out-of-sample testing, is a similar model in validation techniques for assessing how the results of a statistical analysis will generalize to an independent data set. It is mainly used in settings where the goal is prediction, and one wants to estimate how accurately a predictive model will perform in practice. In a prediction problem, a model is usually given a dataset of known data on which training is run (training dataset), and a dataset of unknown data (or first seen data) against which the model is tested (called the validation dataset or testing set). The goal of cross-validation is to test the model's ability to predict new data that was not used in estimating it, in order to flag problems like overfitting or selection bias and to give an insight on how the model will generalize to an independent dataset (i.e., an unknown dataset, for instance from a real problem).




(a) Cross-Validation technique (Khramtsov et al. (2019))

(b) k-fold Cross-Validation (kdnuggets.com)

Figure 5.3: Cross-Validation

One round of cross-validation involves partitioning a sample of data into complementary subsets as shown in Figure 5.3a, performing the analysis on one subset (called the training set) and validating the analysis on the other subset (called the validation set or testing set). To reduce variability, in most methods multiple rounds of cross-validation are performed using different partitions and the validation results are combined (e.g. averaged) over the

[6] `https://en.wikipedia.org/wiki/Cross-validation_(statistics)`

rounds to give an estimate of the model's predictive performance.

In summary, cross-validation combines (averages) measures of fitness in prediction to derive a more accurate estimate of model prediction performance [7].

Two types of cross-validation can be distinguished:

- **Exhaustive cross-validation** - method which learns and tests on all possible way to divide the original sample into a training and a validation set. They are of two types:

  i. ***Leave-p-out cross-validation*** - LpOCV, involves using $p$ observations as the validation set and the remaining observations as the training set. This is repeated on all ways to cut the original sample on a validation set of p observations and a training set.

  ii. ***Leave-one-out cross-validation*** - LOOCV, is a particular case of leave-p-out cross-validation with $p = 1$.

- **Non-exhaustive cross-validation** - do not compute all ways of splitting the original sample. They are of 3 types:

  i. ***k-fold cross-validation*** - as shown in Figure 5.3b[8] the original sample is randomly partitioned into $k$ equal sized subsamples. Of the $k$ subsamples, a single subsample is retained as the validation data for testing the model and the remaining $k1$ subsamples are used as training data. The cross-validation process is then repeated $k$ times, with each of the $k$ subsamples used exactly once as the validation data. The $k$ results can then be averaged to produce a single estimation. The advantage of this method over repeated random sub-sampling is that all observations are used for both training and validation, and each observation is used for validation exactly once. 10-fold cross-validation is commonly used but in general $k$ remains an unfixed parameter. In stratified k-fold cross-validation, the partitions are selected so that the mean response value is approximately equal in all the partitions. In the case of binary classification, this means that each partition contains roughly the same proportions of the two types of class labels. In repeated cross-validation the data is randomly split into k partitions several times. The performance of the model can thereby be averaged over several runs, but this is rarely desirable in practice.

[7] `https://en.wikipedia.org/wiki/Cross-validation_(statistics)`

[8] `https://www.kdnuggets.com/2017/08/dataiku-predictive-model-holdout-cross-validation.html`

ii. ***Holdout method*** - as shown in Figure 5.4[9] randomly assign data points to two sets $d_0$ and $d_1$, usually called the training set and the test set, respectively. The size of each of the sets is arbitrary although typically the test set is smaller than the training set. Then train (build a model) on $d_0$ and test (evaluate its performance) on $d_1$. In typical cross-validation, results of multiple runs of model-testing are averaged together; in contrast, the holdout method, in isolation, involves a single run. It should be used with caution because without such averaging of multiple runs, one may achieve highly misleading results. One's indicator of predictive accuracy ($F*$) will tend to be unstable since it will not be smoothed out by multiple iterations. Similarly, indicators of the specific role played by various predictor variables (e.g., values of regression coefficients) will tend to be unstable.While the holdout method can be framed as the simplest kind of cross-validation, many sources instead classify holdout as a type of simple validation, rather than a simple or degenerate form of cross-validation.

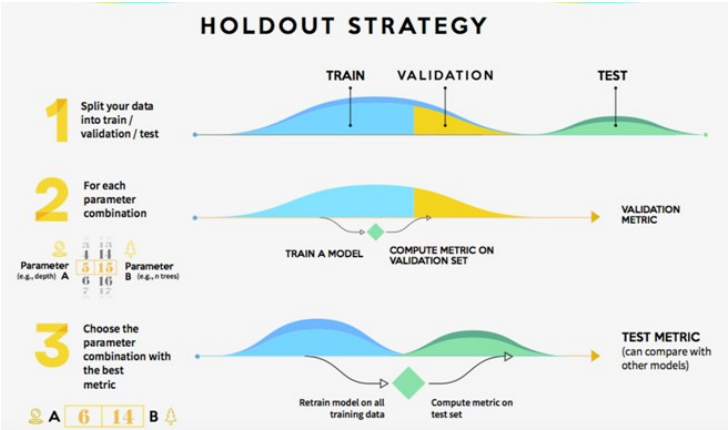



Figure 5.4: Holdout Cross-Validation (kdnuggets.com)

iii. ***Repeated random sub-sampling validation*** - also called as Monte Carlo cross-validation, creates multiple random splits of the dataset into training and validation data. For each such split, the model is fit to the training data and predictive accuracy is assessed using the validation data. The results are then averaged over the splits. The advantage of this method (over k-fold cross validation) is that the proportion of the training/validation

[9] `https://www.kdnuggets.com/2017/08/dataiku-predictive-model-holdout-cross-validation.html`

split is not dependent on the number of iterations (i.e., the number of partitions). The disadvantage of this method is that some observations may never be selected in the validation subsample, whereas others may be selected more than once. In other words, validation subsets may overlap. This method also exhibits Monte Carlo variation, meaning that the results will vary if the analysis is repeated with different random splits. As the number of random splits approaches infinity, the result of repeated random sub-sampling validation tends towards that of leave-p-out cross-validation. In a stratified variant of this approach, the random samples are generated in such a way that the mean response value (i.e. the dependent variable in the regression) is equal in the training and testing sets. This is particularly useful if the responses are dichotomous with an unbalanced representation of the two response values in the data.

**Evaluation Results - Proposed Work**

We would like to thank *Premium Aerotec AG* for their permission to use the NDT data for this study.

We have used the k-fold Cross Validation method (refer Figure 5.3) to check the accuracy of the ML algorithms in combination with different feature extraction techniques and the results are discussed.

SVM gains accuracy of 66.4% when trained in combination with LBP as shown in Table 5.1, following Random Forest with 64.3% accuracy, Naive Bayes is not suitable.

| **Feature Extraction: LBP** | |
|---|---|
| **Classifiers** | **Accuracy (%)** |
| SVM | 66.4 |
| Random Forest | 64.3 |
| k-NN | 59.3 |
| Naive Bayes | 55 |

Table 5.1: Accuracy Chart using LBP

SVM gains best accuracy with MSER feature extraction with 92.6%, the Random forest is close to SVM with 91.9% accuracy as shown in Table 5.2. K-NN performs nearly as good as SVM with 92.40% of accuracy and Naive

Bayes have improved its accuracy level when compared to training with LBP method.

| **Feature Extraction: MSER** | |
|---|---|
| **Classifiers** | **Accuracy (%)** |
| SVM | 92.6 |
| Random Forest | 91.9 |
| k-NN | 92.40 |
| Naive Bayes | 57.6 |

Table 5.2: Accuracy Chart using MSER

KAZE feature extraction in combination with Naive Bayes achieves more than any other ML algorithms gaining accuracy of 94.30%. Random Forest achieves second best accuracy of 91% when compared to SVM having the third highest accuracy of 90.7% as shown in Table 5.3. K-NN performs lower than with MSER method.

| **Feature Extraction: KAZE** | |
|---|---|
| **Classifiers** | **Accuracy (%)** |
| SVM | 90.7 |
| Random Forest | 91 |
| k-NN | 80.5 |
| Naive Bayes | 94.30 |

Table 5.3: Accuracy Chart using KAZE

Random Forest has recorded the highest accuracy of 97.9% with SURF feature extraction extraction and SVM has the second highest accuracy of 96.9% as shown in Table 5.4. K-NN performs best with SURF with 95.2% accuracy. Naive Bayes is not suitable as the accuracy rate has been dropped when compared with KAZE method.

SVM records the highest and best accuracy with HOG feature extraction with 99.03% and Random Forest is the second highest accuracy of 97.90% as shown in Table 5.5. K-NN has downfall in its accuracy to 90% when compared to the SURF method, Naive Bayes is still not suitable.

| **Feature Extraction: SURF** | |
|---|---|
| **Classifiers** | **Accuracy (%)** |
| SVM | 96.9 |
| Random Forest | 97.9 |
| k-NN | 95.2 |
| Naive Bayes | 59.5 |

Table 5.4: Accuracy Chart using SURF

| **Feature Extraction: HOG** | |
|---|---|
| **Classifiers** | **Accuracy (%)** |
| SVM | 99.03 |
| Random Forest | 92.14 |
| k-NN | 90 |
| Naive Bayes | 56 |

Table 5.5: Accuracy Chart using HOG

| **Classifiers** | **Feature Extraction** | **Accuracy (%)** |
|---|---|---|
| SVM | HOG | 99.03 |
| Random Forest | SURF | 97.90 |
| Naive Bayes | KAZE | 94.30 |
| k-NN | MSER | 92.40 |
| Random Forest | HOG | 92.14 |
| SVM | LBP | 66.4 |

Table 5.6: Accuracy Chart - consolidated

The best of all Machine Learning with different feature extraction techniques is consolidated in Table 5.6. We have compared SVM+HOG with Random-Forest+SURF in terms of accuracy, SVM has gained more than Random Forest, the recall of SVM is more 0.99 and precision 0.98 is also more than Random Forest. The final deciding factor is the F1-score, SVM's

F1-score 0.98 is more than Random Forest as shown in Table 5.7.

| **Classifiers** | **Accuracy (%)** | **Recall** | **Precision** | **F1-score** |
|---|---|---|---|---|
| SVM (HOG) | 99.05 | 0.9919 | 0.9880 | 0.984 |
| RF (SURF) | 97.90 | 0.9839 | 0.97 | 0.97 |

Table 5.7: Evaluation Chart

Confusion Matrix results for our problem, the Figure 5.5 shows the confusion matrix obtained from SVM with HOG, the true positive rate of 98% is achieved and only 2% of false negative rate.

The Figure 5.6 shows the confusion matrix obtained from Random Forest with HOG, the true positive rate of 87% is achieved and 13% of false negative rate. Our main aim was to reduce the false negatives, in order to do so, we have to pay for the false positives. The recall we expected was $recall = 1$, which denotes no false negatives. Missing a defect was the worst case in our problem.

The ROC-AUC curve results for our problem, the Figure 5.7 shows that ROC-AUC curve, $AUC = 1.0$, which denotes the best classification. The Figure 5.8 shows that ROC-AUC curve, $AUC = 0.92$, which denotes a better classification, but not the best as SVM's AUC.

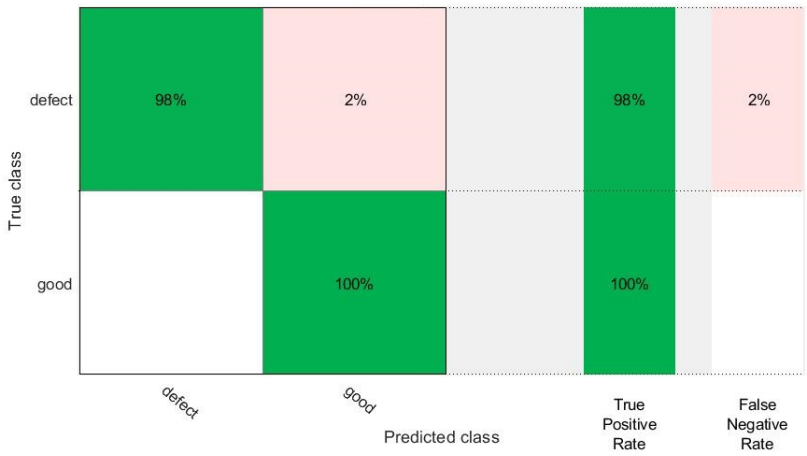



Figure 5.5: SVM Confusion Matrix

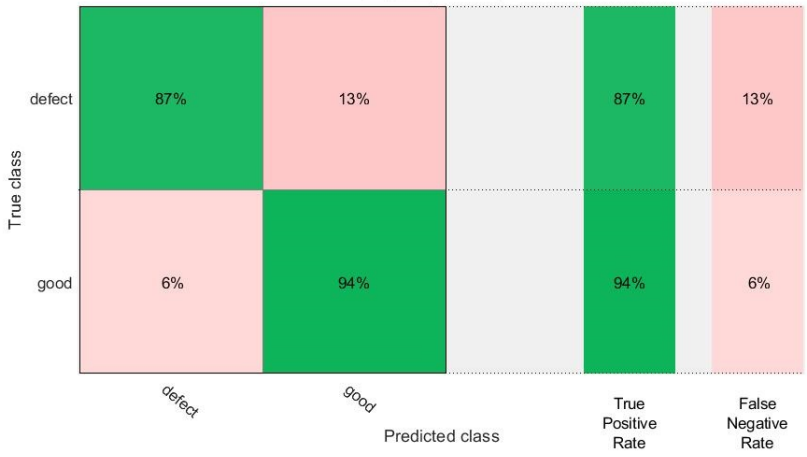



Figure 5.6: Random Forest Confusion Matrix

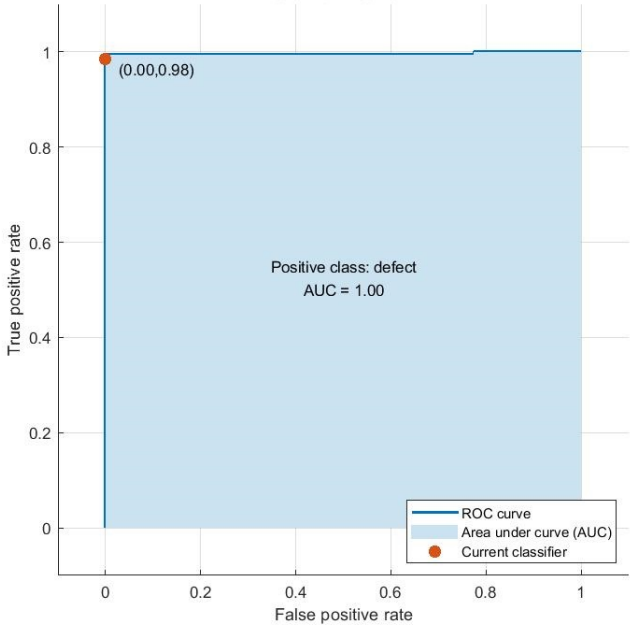



Figure 5.7: SVM ROC-AUC

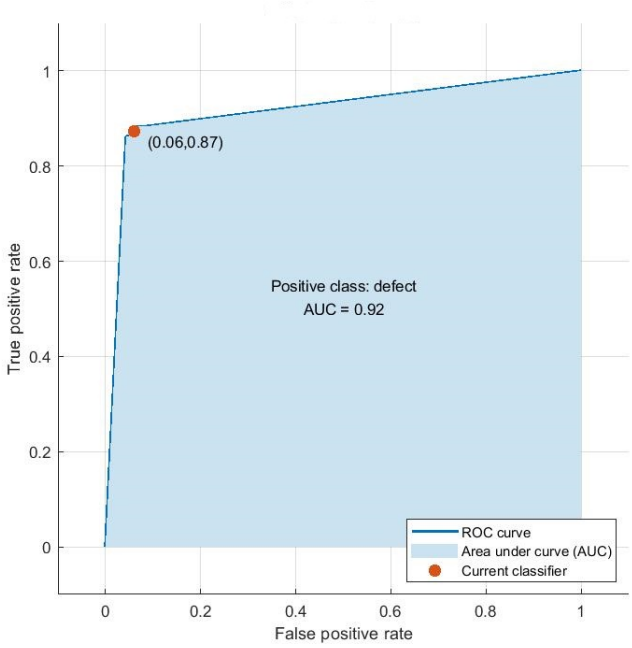



Figure 5.8: Random Forest ROC-AUC

## 5.3 Conclusion

To conclude, SVM+HOG gives the best results either in terms of accuracy or F1-score. Random Forest+HOG results does not seem to be so promising. But the combination of Random Forest+SURF would work better than Random Forest+HOG. SVM works good for binary class problems, whereas Random Forest works even for multi-class problems. Random Forest can estimate the missing defects. SVM and Random Forest both have their advantages and disadvantages. We leave the decision of deciding which combination of ML algorithm and feature extraction would best suit, to the Industry. The industry would select according to its needs.

# Chapter 6

# Certification

Certification is a regulatory obligation, with all aircraft, their engines and propellers certifiable. The "Type Certificate" – issued to signify the airworthiness of an aircraft manufacturing design – is followed by the "Airworthiness Certificate," which authorises aircraft operations in a certain country or region[1]. The certification process covers the complete development process of a new aircraft. It includes various phases:

i. Detailed design review

ii. Test review and participation in laboratory

iii. Test review and participation in flight (designed to take into account modifications in light of the results)

iv. Aircraft operators (closely involved in design definition, development and service introduction)

The certification process is about automatic error detection (as it is intended for ultrasonic testing), then the entire testing using ML classifiers must meet the same criteria as an entire "human" test. By total test, we mean the two parts acquisition and evaluation. Acquisition is the bare scan in the squirter system or X-ray system. Evaluation is the human search in the scan image and then the mechanical finding in the scan data. The qualification philosophy is based on the PoD concept (Probability of Detection), to answer: How reliably can I find defect sizes?

In practice, PoD is usually translated into the question: Can we reliably find a given defect size (minimum size to be detected)? The minimum size,

[1] `https://www.airbus.com/aircraft/how-is-an-aircraft-built/test-programme-and-certification.html`

however, already contains the PoD knowledge. This is then implemented using the 29/29 method, for example. We get 29 errors (in the minimum size to be detected) and the method has to detect all 29 errors reliably (without missing a single one). Thus, the PoD criterion is automatically fulfilled in everyday practice (a90/95, i.e. 0.9 at 95%) without having to deal with the PoD concept. The disadvantage of 29/29: we have fulfilled the PoD requirement, but we do not know the true potential of the test method[2].

## 6.1 Motivation

The certification process is mainly to avoid the risks and challenges such as:

i. The software is a black box

ii. Training of the algorithm is crucial and requires a lot of experience for underfitting, overfitting

iii. Reliability - in terms of new types of defects, new parts and different testing methods

Machine Learning algorithms are **black-box** other than for Random Forest, a sample of it from our work as shown in Figure 6.1. But it would be difficult for the human examiner to understand the branching values of the decision trees, and also it would get complicated when there are more decision trees involved in the Random Forest.

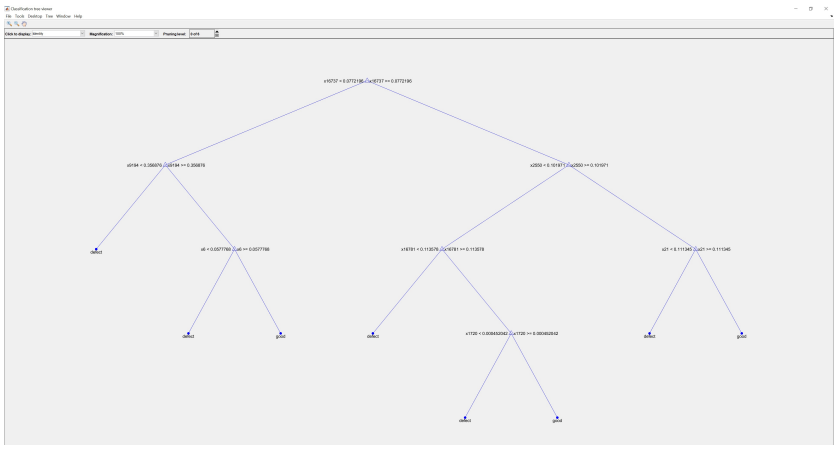


Figure 6.1: Random Forest Tree View

[2]Input from Industry Experts

The possible road for certification are:

i. Evaluation of NDT by human test engineer and algorithm

ii. Feedback of human engineer regarding quality of algorithm generated evaluation leads to further training of algorithm – repeat it often

iii. Reliability - in terms of new types of defects, new parts and different testing methods

The final conclusion, is to use the statistical validation - use of the PoD curve to maintain the industry standards.

## 6.2 PoD Curve

After understanding the work of Georgiou (2006), Non-destructive Testing (NDT) reliability may be defined as the probability of detecting a crack in a given size group under the inspection conditions and procedures specified. There are of course other similar definitions, but the underlying statistical parameter is the PoD, which has become the accepted formal measure of quantifying NDT reliability. The PoD is usually expressed as a function of flaw size (i.e. length or depth), although in reality it is a function of many other physical and operational parameters, such as, the material, the geometry, the flaw type, the NDT method, the testing conditions and the NDT personnel (e.g. their certification, education and experience). Repeat inspections of the same flaw size or the same flaw type will not necessarily result in consistent hit or miss indications. Hence there is a spread of detection results for each flaw size and flaw type and this is precisely why the detection capability is expressed in statistical terms such as the PoD.

PoD functions, for describing the reliability of an NDT method or technique have been the subject of many studies and have undergone considerable development since the late 1960's and early 1970's, where most of the pioneering work was carried out in the aerospace industry. In order to ensure the structural integrity of critical components it was becoming more evident that instead of asking the question '... what is the smallest flaw that can be detected by an NDT method?' it was more appropriate, from a fracture mechanics point of view, to ask '... what is the largest flaw that can be missed?' To elaborate on this point here, ultrasonic inspection data has been re-plotted from the 'Non-destructive Testing Information Analysis Centre' (NTIAC) capabilities data book.

Early on in the mid-1970's, a constant PoD for all flaw types of a given size was proposed and Binomial distribution methods were used to estimate this

probability, along with an associated error or lower confidence limit. Whilst good PoD estimates could be obtained for a single flaw size, very large sample sizes were required to obtain good estimates of the 'lower confidence limit'. Clearly assumption about a constant PoD for flaws of a given size, whilst making the probability calculations easier, was too simplistic as different detection percentages were being recorded for the same flaw size. In cases where there was an absence of large sample sizes, various grouping schemes were introduced to analyse the data, but in these cases estimates for the lower confidence limit were no longer valid. In the early to the mid-1980s, the approach was to assume a more general model for the PoD vs. flaw size 'a'. Various analyses of data from reliability experiments on NDT methods indicated that the PoD (a) function could be modelled closely by either the cumulative 'log-normal' distribution or the 'log-logistic' (or 'log-odds') distribution. The statistical parameters (e.g. mean, median and standard deviation) associated with the PoD (a) functions can be estimated using standard statistical methods like 'maximum likelihood methods'.

The 'Recommended Practice', which was originally prepared for the aircraft industry, provides comprehensive information on the experimental sequence of events for generating data to produce PoD curves and to 'certify' (i.e. validate) an NDT method or procedure. The sequence of events can be broadly summarised as follows:

i. Manufacture or procure flaw specimens with the required large number of relevant flaw sizes and flaw types

ii. Inspect the flaw specimens with the appropriate NDT method

iii. Record the results as a function of flaw size

iv. Plot the PoD curve as a function of flaw size

However, before the manufacture or procurement of flaw specimens, it is necessary to make the following crucial decisions:

i. What flaw parameter size will be used (e.g. flaw length or flaw depth)?

ii. What overall flaw size range is to be investigated (e.g. 1mm to 9mm)?

iii. How many intervals are required within the flaw size range to be investigated (e.g. if 6 intervals are selected for a 1mm to 9mm flaw size range, this implies a flaw width interval of 1.5mm)?

The recommended practice also provides critical information on the necessary flaw sample size for each flaw width interval in order to demonstrate

the desired PoD, along with an appropriate lower confidence limit, has been achieved. Usually it is not known before hand, how large a flaw has to be before the desired PoD is satisfied and this can present problems in knowing the most appropriate flaw size range to select. Following the above experimental approach should lead to the largest flaw that can be detected with the desired PoD and confidence limit. It is important to appreciate that in selecting the sample size there are two distinct issues that have to be addressed. First, there is the issue of the sample size being large enough to achieve the desired PoD and confidence limit combination. Second, the sample size has to be large enough to be able to compute the statistical parameters associated with the PoD curve that best fits the data. It is believed that this distinction is not always made clear in the open literature. It may be of course, that the sample size required to achieve the desired PoD/confidence limit combination is always sufficiently large to compute the statistical parameters for the PoD curve accurately enough.

In NDT reliability methods, there are two related probabilistic methods for analysing reliability data and producing PoD curves as functions of the flaw size 'a'. Originally, NDT results were only recorded in terms of whether the flaw was detected or not. This type of data is called 'hit/miss' data and it is discrete data. This way of recording data is still appropriate for some NDT methods (e.g. penetrant testing or magnetic particle testing). However, in many NDT systems there is more information in the NDT response (e.g. peak voltage in eddy current NDT, the signal amplitude in ultrasonic NDT, the light intensity in fluorescent penetrant NDT). Since the NDT signal response can be interpreted as the perceived flaw size, the data is sometimes called â data (i.e. 'a hat data') or signal response data and it is continuous data. Each type of data (i.e. hit/miss or signal response) is usually analysed using a different probabilistic model to produce the PoD (a) function.

For hit/miss data a number of different statistical distributions were originally considered for the best fit. It was found that the log-logistic distribution was the most acceptable and the PoD (a) function. For signal response data, much more information is supplied in the signal for analysis than is in the hit/miss data. In fact, as will be shown below, the PoD (a) function is derived from the correlation of â vs. a data.

For the hit/miss data, there is a flaw size range (i.e. asmallest, alargest) in which there is a definite uncertainty whether the inspection system will detect the flaw or not. On the other hand, if the flaw size $a < asmallest$ the inspection system would be expected to miss the flaw. Similarly if $a > alargest$ the inspection system would be expected to detect the flaw. So having a large number of very small or very large flaws will not provide much information on the PoD (a) function that will fit the data. To maximise the information

required for estimating the PoD (a) function (i.e. the parameters) it is recommended that the flaw sizes be uniformly distributed between the minimum and maximum flaw size of interest. A minimum of 60 flaws is recommended for hit/miss data.

In practice, a PoD and lower confidence limit combination that is often quoted is 90% and 95% respectively (sometimes written 90-95). For the hit/miss NDT data discussed in the recommended practice (1), it is necessary to have a minimum sample of 29 flaws in each flaw width interval. This could be interpreted as 29 flaw specimens with one flaw in each specimen. This means that with 6 flaw width intervals, a minimum of 174 flaw specimens would be necessary (i.e. 174 flaws spread across the overall flaw range). So when published articles on this theme often refer to the 'considerable' cost associated with producing PoD curves experimentally, it is often understated. In addition, it is necessary to have the same number of 'control' specimens (specimens having no flaws) as flaw specimens, which are randomly mixed in with the flaw specimens before all the specimens are inspected. With such a large number of flaws, the requirement to compute the PoD (a) function parameters, is easily satisfied. In order to achieve the 90% PoD with a 95% lower confidence limit for any flaw width interval, it is necessary to detect all the 29 flaws in that flaw width interval. For each flaw that is not detected in any particular flaw width interval, the recommended practice provides tables of how many flaws in total need to be detected to achieve certification. There are also 'maximum probability' tables which indicate the probability of achieving certification after failing to achieve it at the first attempt or the second attempt and so on. Following any failure to certify, the decision as to whether it is economically viable to continue has to be considered very carefully and the maximum probability tables in the recommended practice are provided as assistance.

PoD curves provide reference to results that have been obtained for particular flaws using specific NDT procedures. However, it is important to appreciate that in using particular PoD curves for different applications that some validation of the NDT procedures is carried out. The POD curves provide important results for quantifying the performance capability of NDT procedures as well as the operators and could be used as a basis for:

- Establishing design acceptance requirements
- NDT procedure qualification and acceptance
- Qualification of personnel performance
- Comparing the performance capabilities of NDT procedures

- Selecting an applicable NDT procedure
- Quantifying improvements in NDT procedures
- Developing repeatable NDT data for fracture mechanics

Whilst the approaches to determine the PoD (a) function for hit/miss data and the signal response data are quite different, the log-odds and cumulative log-normal distribution functions are very similar for the same statistical parameters. On occasions the behaviour of the PoD data may appear illogical and the PoD (a) function selected (e.g. log-odds or cumulative log-normal) may not fit the data. However, it is useful to carry out some quick checks to see if there is something specific about the data in order to decide what action to take. In the case of hit/miss data it has been observed that the PoD (a) function can sometimes decrease with flaw size (i.e. large flaws are missed more than small ones). This is usually because the NDT experiment was poorly designed and it would require a repeat some of the trials with better designed NDT procedures. Regions of flaw hits and flaw misses should not be distinct, there has to be a good overlap, otherwise the analysis that fits the log-odds model will not produce a valid solution. This usually means more data is required in the region between asmallest and alargest. It is also possible to produce what appears to be an acceptable PoD (a) function that fits the data well, but the confidence limit decreases with increasing flaw size. This is usually evidence that the log-odds model is not a good fit. Confidence limit is inversely proportional to flaw size.

### PoD curve - Proposed Work

For our problem, we use the best result obtained that is the combination of HOG with SVM with highest accuracy as discussed in the Chapter 5 to obtain the PoD curve. We form a test dataset with the given flaws (defect dataset) and some flaws were artificially created to generate more data. Techniques such as image rotation, skewing and mirroring were used to create the artificial flaws from each original flaw. The original number of flaw dataset we used is explained in the Figure 6.2, Brehm and Lienhart (2018).

In Figure 6.2, X-axis in the graph is the flaw size that was calculated by the square root of the area in pixels and the Y-axis is the frequency of the flaws of corresponding size. The smallest flaw size in the positive image dataset was 6 pixels and the largest we recorded was 383 pixels. For the purpose of the PoD curve, we created 29 artificial flaws in each flaw width interval and tested against the best trained classifier model (HOG + SVM) and noted the prediction accuracy rate to plot the PoD curve. The flaw width

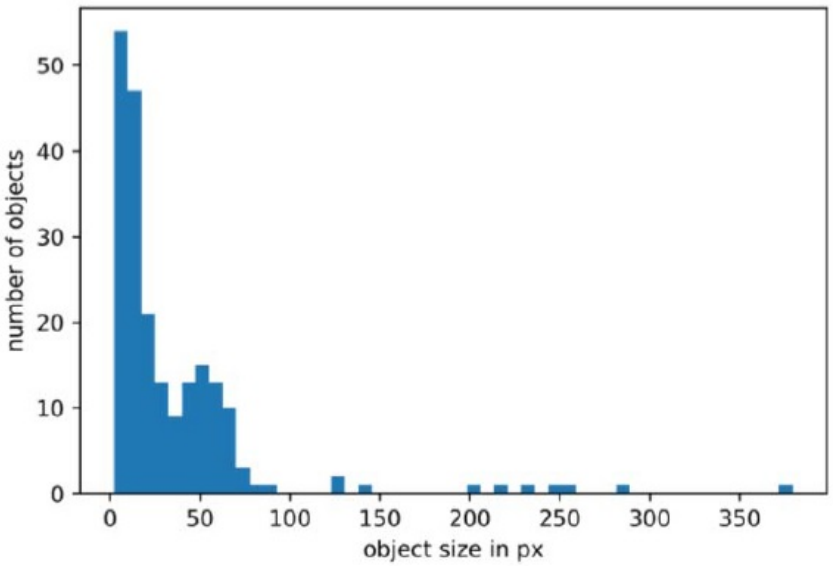



Figure 6.2: Flaw size v/s number of flaws

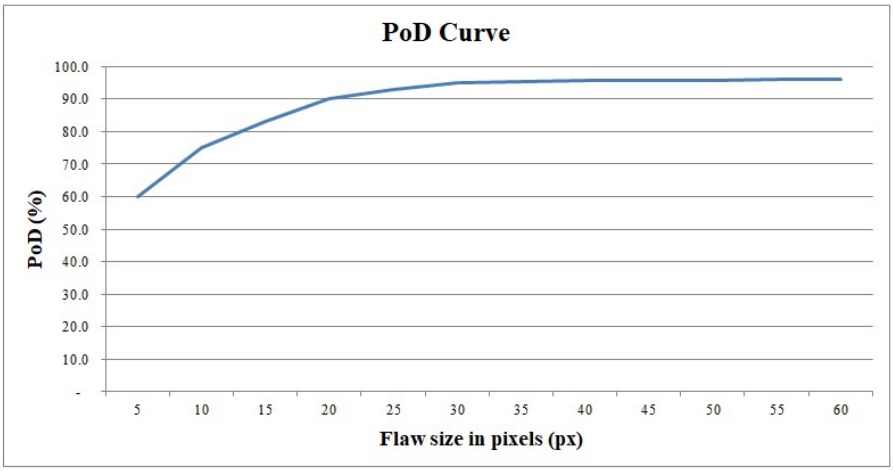



Figure 6.3: PoD Curve

interval we chose was 5 as shown in the Figure 6.3. From the Figure 6.3, we observe that even the smallest defect could be identified as true positive. We could achieve PoD of 90% with the lower confidence of 95%. This could be further improved with more availability of the positive dataset.

## 6.3 Conclusion

In this chapter, we have explained the certification process in Industry standards. If our proposed algorithm is executed in parallel to a human examiner in the Industry with a quality feedback input system and our proposed algorithm could obtain a certificate from the higher authorities in the Industry.

# Chapter 7

# Comparison and Future Work

In this Chapter we compare our work with the related work of Brehm and Lienhart (2018).

### Comparision

The related work used Deep Neural Network, we have used Machine Learning algorithms such as SVM, Random Forest, KNN and Naive Bayes. We have used different feature extraction techniques such as LBP, MSER, KAZE, SURF and HOG. The previous work used grayscale image dataset as input, but we used truecolor image dataset, we have overcome the data loss during conversion from truecolor to grayscale. Random Forest breaks-the-ice of being a black-box. We have tried statistical validation for the certification process using the best of our results. Time consumption for our method was less, as we chose MATLAB R 2019.

To conclude, our work is first-of-its-kind in aircraft NDT inline QA. We have used a traditional Machine Learning approach. The best combination is SVM+HOG which tried to achieve a recall almost equal to 1. The second best combination is Random-Forest+SURF which also performs well with a high accuracy rate. We let the Industry decide which combination of ML+feature-extraction they would like to use in real-time to help the Human examiner. Our method checks the defects pixel-by-pixel, which is not performed by a Human examiner. Our method will gain as much experience as the Human examiner when run in parallel to him and gains insight from him as a feedback.

### Future Work

In future, more training data could be made available. More information about the flaw types would be helpful to train the ML models in a better way. Meaningful data has to be provided to make the PoD curve better. Our work can be improvised to receive feedback from the Human examiner regarding its results, this leads to Reinforcement Learning. Even though Random Forest tries to break the black-box as shown in Figure 6.1, it is very difficult to understand the decision tree branches and it can get complicated when more decision trees are included in the Random Forest. In order to overcome this, our work could be further developed to be an explainable AI (XAI)[1] model to make the ML **black-box** to a ***white-box*** for the direct use in the Industry. It could be further developed to be a digital twin with XAI and Reinforcement Learning. PoD curve could also be derived from the Deep Neural Network from the previous work of Brehm and Lienhart (2018).

[1]Further explanation refer: `https://en.wikipedia.org/wiki/Explainable_artificial_intelligence`

# Bibliography

Alvarado, T., Schmoyer, N., and Wooten, J. (2020). Inspectioneering.

Apmann, H., Mayer, M., and Fortkamp, K. (2016). A method of in-line quality assurance and non-destructive testing within the production line of fiber-metal laminates.

A.Sumesh, K.Rameshkumar, and K.Mohandas (2015). Use of machine learning algorithms for weld quality monitoring using acoustic signature.

Bernieri, A., Ferrigno, L., Laracca, M., and Molinara, M. (2006). An svm approach to crack shape reconstruction in eddy current testings.

Bernieri, A., Ferrigno, L., Laracca, M., and Molinara, M. (2008). Crack shape reconstruction in eddy current testing using machine learning systems for regression.

Boughorbel, S., Jarray, F., and El-Anbari, M. (2017). Optimal classifier for imbalanced data using matthews correlation coefficient metric. *PLOS ONE*, 12:e0177678.

Brehm, S. and Lienhart, R. (2018). Final report: Detection of anomalies in ultrasonic images of fiber-metal-laminate skin fields.

Caruana, R. and Niculescu-Mizil, A. (2006). An empirical comparison of supervised learning algorithms.

D.Benitez, H., Loaiza, H., and Caicedo, E. (2009). Defect characterization in infrared non-destructive testing with learning machines.

D'Angelo, G. and Rampone, S. (2013). Shape-based defect classification for non destructive testing.

García-Martín, J., Gomez-Gil, J., and Vázquez-Sánchez, E. (2011). Non-destructive techniques based on eddy current testing. *Sensors (Basel, Switzerland)*, 11:2525–65.

Georgiou, G. A. (2006). Probability of detection (pod) curves derivation, applications and limitations.

Huang, H., Hu, C., and Wang, T. (2017). Surface defects detection for mobilephone panel workpieces based on machine vision and machine learning.

Khodayari-Rostamabad, A., Reilly, J. P., and Nikolova, N. K. (2009). Machine learning techniques for the analysis of magnetic flux leakage images in pipeline inspection.

Khramtsov, V., Sergeyev, A., Spiniello, C., Tortora, C., Napolitano, N., Agnello, A., Getman, F., Jong, J., Kuijken, K., Radovich, M., Shan, H., and Shulga, V. (2019). Kids-squad ii: Machine learning selection of bright extragalactic objects to search for new gravitationally lensed quasars.

Maillet, D., André, S., Batsale, J.-C., Degiovanni, A., and Moyne, C. (2000). *Thermal Quadrupoles,Solving the heat equation through integral transforms*.

Malekzadeh, T., Abdollahzadeh, M., and Nejati, H. (2016). Aircraft fuselage defect detection using deep neural networks.

Medjahed, S. A. (2015). A comparative study of feature extraction methods in images classification.

Navneet Dalal, B. T. (2005). Histograms of oriented gradients for human detection.

Nguyen, A., Yosinski, J., and Clune, J. (2015). Deep neural networks are easily fooled: High confidence predictions for unrecognizable images.

Saechai, S., Kongprawechnon, W., and Sahamitmongkol, R. (2012). Test system for defect detection in construction materials with ultrasonic waves by support vector machine and neural network.

Schmidt, T., Mayer, M., and Lienhart, R. (2015). Pilot study automated evaluation of ndt data.

Shumin, D., Zhoufeng, L., and Chunlei, L. (2011). Adaboost learning for fabric defect detection based on hog and svm.

Tareen, S. A. K. and Saleem, Z. (2018). A comparative analysis of sift, surf, kaze, akaze, orb, and brisk.

Wang, B., Zhao, W., Gao, P., Zhang, Y., and Wang, Z. (2018). Crack damage detection method via multiple visual features and efficient multi-task learning model. *Sensors*, 18:1796.

Wei, H. and Cheng-Tong, L. (2009). Automatic real-time svm-ased ultrasonic rail flaw detection and classification system.

Zhang, Y., Guo, N., Du, H., and Li, W. (2005). Automated defect recognition of c-sam images in ic packaging using support vector machines. *The International Journal of Advanced Manufacturing Technology*, 25:1191–1196.