GAS- UND
WÄRMETECHNISCHE
ANLAGEN

TECHNISCHE UNIVERSITÄT
BERGAKADEMIE FREIBERG
Die Ressourcenuniversität. Seit 1765.

THEMA DER STUDIENARBEIT

Nutzung von Methoden des maschinellen Lernens für den Nachweis von Lackschichten und –resten auf Beton- und Metalloberflächen

Use of machine learning methods for the detection of paint layers and residues on concrete and metal surfaces

Bearbeiter:	Yasa, Viswambhar Studiengang: CMS
Prüfer:	Prof. Dr.-Ing. H. Krause
Betreuer:	Dipl.-Ing. A. Pestel
Übergabetermin des Themas:	01.11.2021
Abgabetermin der Arbeit:	04.04.2022

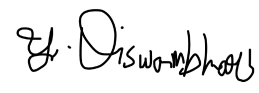
Prof. Dr.-Ing. H. Krause

Declaration

I, Viswambhar Yasa, here by declare that the student work presented is my own and has been generated based on my own original research.

Further, I also confirm that :

- Wherever, I have consulted the published work of others, this is always clearly attributed. This assurance also refers to the pictorial representations;
- Where, I have quoted from the work of others, the source is always given. With the exception of such quotations, this work is entirely my own work;
- 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;



Place and date: Freiberg, 04-04-2022 Viswambhar Reddy Yasa

Acknowledgements

Foremost, I would like to express my sincere gratitude to my supervisor, **M.Sc Andreas Pestel** for motivating me to choose this path, supporting my work, and always keeping a high interest in it. His friendly guidance and expert advice have been invaluable throughout all stages of work.

I would also like to thank my parents and friends for always believing and encouraging me to follow my dreams and for the patience, moral support, and faith that they have always shown.

-Viswambhar Yasa

Abstract

Project LaDECO is a quantification of laser-based decontamination technology for use in nuclear dismantling. The project is supported by the FORKA funding concept of the Federal Ministry of Education and Research (BMBF). As part of the research project, epoxy resin-based coatings are being removed from concrete surfaces with the aid of a laser. To ensure that the paint is completely removed, the concrete surface is then checked for paint residues using active thermography. We are looking to optimize this method using machine learning models for the detection and thickness estimation of paint layers and residues on concrete and metal surfaces.

The student work begins with an extensive literature review. For this purpose, scientific articles are searched, analyzed, and summarized, which deal with the mentioned topic. A python API is built which extract different thermographic video format without the need for any external software components/interfaces. A machine learning API consists of object segmentation and thickness evaluation modules which consist of various deep learning models. The performance of the various models is compared and experimented with different datasets and parameters for optimization. The thickness of the coating is evaluated using time-series analysis for different contrast functions. After the segmentation, and evaluation API is built to perform a thermographic analysis of the identified features.

The following subtasks were also achieved:

- Optimization of the existing software to handle a large number of samples using big-data.
- Graphical User interface for data handling process.

Contents

List of Acronyms

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Kaalo asmi loka kshaya
kritpraviddho

Lord Krishna in Bhagavad Gita

"Now I am Death, the Destroyer of Worlds" were the words used by Dr.Robert Oppenheimer, quoted from Bhagavad Gita, after the destruction caused by nuclear fission in World War II (WWII). Nuclear fission is the major source of energy generation in the world, but handling nuclear waste and guarding it for thousands of years to avoid potential breakout is the price future generations have to pay. The Chernobyl nuclear power plant mishap and Fukushima Daiichi nuclear disaster, which led to immense radioactive contamination of the surrounding environment, caused rejigged views about the future of nuclear fission in the energy sector.

Initially, the German government wanted to phase out nuclear power by 2036. Following the Fukushima Daiichi nuclear disaster in 2011, plans for an early phase-out of the nuclear power plant by 2022 were put into motion. Thus decommissioning all 23 nuclear power plants in Germany by 2023 [21]. As the nuclear power plant ages, there is an imminent need to use efficient practices which include complex and large-scale decommissioning processes.

This rapid and ex-ante uncoordinated phase-out of nuclear plants requires planning of a safe decommissioning processes, thus expediting many technological advancements.

1.2 Motivation

As the physical dismantling process moves from the outside of the reactor building towards the reactor vessel [21]. Radioactive decontamination, chemical-toxic decontamination is carried out in the reactor building. The majority of the dismantling process includes working with concrete and metal. Most of the surfaces are coated with epoxy resin-based paints containing Polychlorinated Biphenyls (PCB) ($C_{12}H_{10-x}Cl_x$). PCB was extensively used in the 1950s and 60s as affective retardants in protective coatings because of their long durability, excellent chemical stability, and high ash point. These chemicals were banned for commercial use in 1970, as they are carcinogenic and neurotoxic agent which poses a high rise to humans and nature. Due to the exposure of the coating to radioactivity, if not handled properly many also radioactive contamination before disposing of in a landfill [1].

Currently, the processes to remove coating are done manually using millers cutting type tools. These forms of coating removal generate a lot of radioactive and carcinogenic dust, thus creating a hazardous work environment. Laser-based coating removal provides a possible solution to the above problem by creating more favourable working conditions [1].

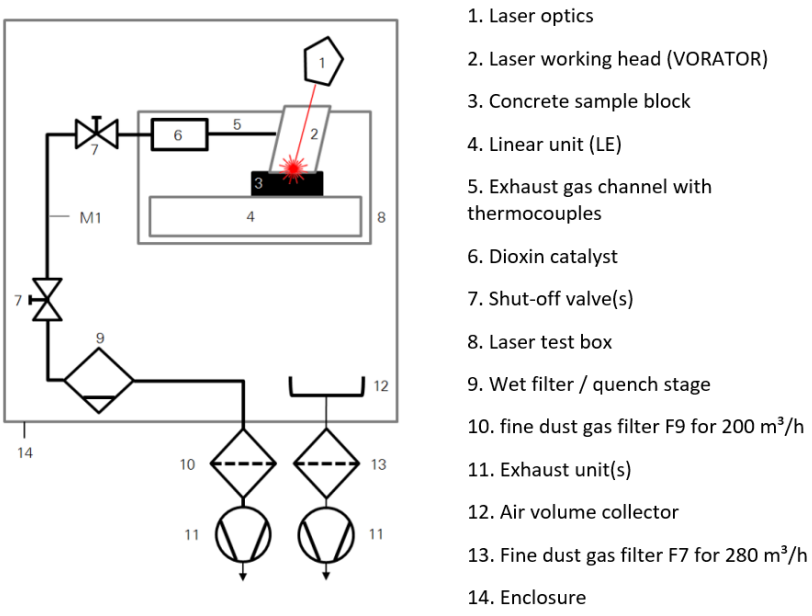


Figure 1: Illustration of laser based coating removal experiment [1]

As indicated in fig.1, the set-up would require a high-power diode laser which can generate lasers. These high power lasers can thermally destroy all chemical

compounds and the subsequent system can filter the fumes from any radioactive dust [1].

Due to the high energy of these lasers, we have to be calibrated them accurately for optimal removal of the coating. A subsequent system that works on thermographic principles and machine learning algorithms ensure that the paint is completely removed and the surface is checked for any paint residues, based on which the laser can be recalibrated.

Chapter 2

Thermography

2.1 Introduction

Temperature is the most measured physical property. It is used to describe the average kinetic energy of the molecules and atoms that make up a substance. Temperature provides information about the internal energy of an object, thus measurement, monitoring and control are crucial in most industrial processes. Temperature sensors based on infra-red thermography have many advantages over other types of sensors. Infra-red sensors are non-contact, thus they do not intrude upon the measurement. Moreover, they can measure the temperature of extremely hot objects. These sensors are also very fast, producing temperature readings in microseconds [3].

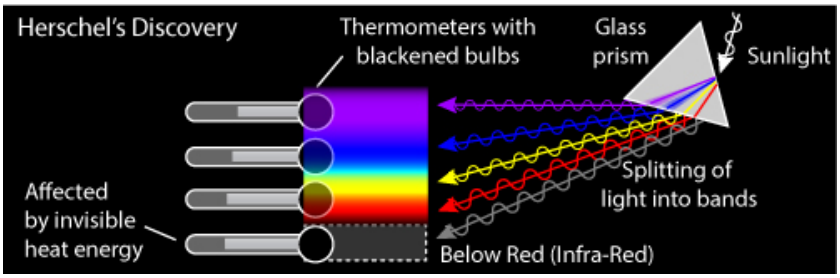


Figure 2: Illustration of William Hershel's light scattering experiment [2]

William Herschel, the astronomer, in the early 19^{th} century identified infra-red spectrum while conducting an experiment of light scatter using prism. During the experiment, he measured the temperatures of different wavelengths and identified a new spectrum of invisible light beyond the red band termed them as **calorific rays**. In late 19^{th} century, people started to identify the significance of infra-red

spectrum and it's applications in diverse industry verticals after the discovery of seebeck effect and thermocouple [2].

Thermocouple is an electric device which convert thermal energy into electrical signal based on See-beck effect. See-beck effects states that when two dissimilar electrical conduction metal or semi-conductor connected are place at different temperature, a electro-motive force (emf) is generated. When multiple thermocouple is joined in series, thermopile can be developed. These thermopile are playing a very import role in modern thermal sensor which convert the heat energy from surface into pixel format electrical signals [2].

During the world wars, there was a significant developement in thermography due to it's military application. For example, identification of enemy war crafts, tanks and developing self-guiding infra-red missiles which follow the targets visualheat signature. Due to the advances in computer technology and the mass production of thermal cameras, the thermography is use in everyday application from night vision goggle to IR photography of distant galaxies. The main application of thermography is in medical for screening and detection of cancer cells and in construction industry. Now autonomous driving and non-destructive testing are field, where there are advancement using thermography. The work presented here will refer specifically to the latter sub-area [22].

2.2 Principles of Infrared Thermography

Infra-red (IR) thermal imaging, also often called thermography. Any object with temperature above absolute temperature (0 K) emits electromagnetic radiation which are not visible to human eye. The radiation is characterised into two features; wavelength (λ) and intensity (Q). These parameters can be used to measure the temperature of the surface without the need of physical contact only simple using physical laws [23].

The focus here is on electromagnetic radiation from the invisible infra-red range, which is in the wavelength range between visible light ($\lambda = 380 \ldots 780$ nm) and Microwave radiation ($\lambda = 1$ mm $\ldots$ 1 m) is located. Infra-red radiation can be divided into near infrared (NIR, $\lambda = 780$ nm $\ldots$ 3 μm), mid-infrared (MIR, $\lambda =$ 3 μm $\ldots$ 50 μm), and far infra-red (FIR, $\lambda = 50$ μm $\ldots$ 1 mm), as indicated in fig.3.

Temperature measurement by IR cameras utilizes radiation transfer. The radiation energy striking the body can dissipated in the form of absorption, reflection and transmission. As shown in equ.2.2.1, α_λ, ρ_λ, τ_λ are the spectral parameters which are the ratio of spectral radiation of absorption, reflection and transmis-

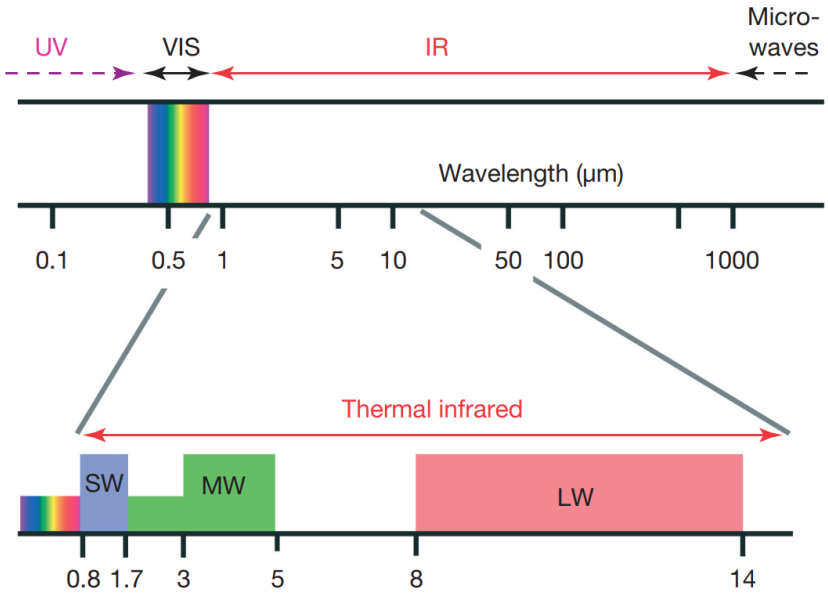


Figure 3: Illustration of energy spectrum with intensity and wavelength scale [3].

sion of the object respectively, can be used to describe the radiation phenomenon. These three parameters are wavelength dependent, whose sum should be one. A black body would absorbed all the heat energy (α_λ=1) [3].

$$\alpha_\lambda + \rho_\lambda + \tau_\lambda = 1 \tag{2.2.1}$$

The electromagnetic radiation ($W_{\lambda b}$) for a black body can be calculated using Plank's law. The result of Plank's law is the power emitted per unit area per unit wavelength, which is a function of λ and T [23].

$$W_{\lambda b} = \frac{C_1 \lambda^{-5}}{e^{\frac{C_2}{\lambda T}} - 1} \tag{2.2.2}$$

The emitted radiation is a function of the temperature of the material, the higher the temperature, the greater the intensity of infra-red energy emitted. The wavelength at which the maximal intensity of radiation is emitted depends on the surface temperature: the higher the temperature, the shorter the wavelength at which most of the radiation is emitted. This relationship is described by Weins displacement law as shown in equ.2.2.3 which is the derivative of plank's law with respect to λ [23].

$$\lambda_{\max} = 0.0029/\mathrm{T} \tag{2.2.3}$$

The total hemispherical radiation intensity of an object can be obtained by integration plank's law equ.2.2.2 for λ from (0 to ∞) to obtain Stefan-Boltzmann equ. 2.2.4, where σ is a constant [23].

$$W_b = \sigma \cdot T^4 \tag{2.2.4}$$

2.2.1 Emissivity

The emissivity of a body is defined formally for a wavelength λ using equ.2.2.3, as the ratio of the radiant energy emitted by the body to the radiation that would be emitted by a black body at the same temperature [3].

$$\varepsilon_\lambda = \frac{W_\lambda}{W_{\lambda b}} \tag{2.2.5}$$

Emissivity of an object is a measurement of its ability to emit thermal energy. For a black body which absorbs all the radiant energy should re-emit all the radiation. Therefore, the absorptivity in a black body is equal to emissivity, which is one. In general, according to Kirchhoff's law, the emissivity and absorptivity of any material are equal at any specified temperature and wavelength as shown in equ.2.2.6 [3].

$$\varepsilon_\lambda = \alpha_\lambda \tag{2.2.6}$$

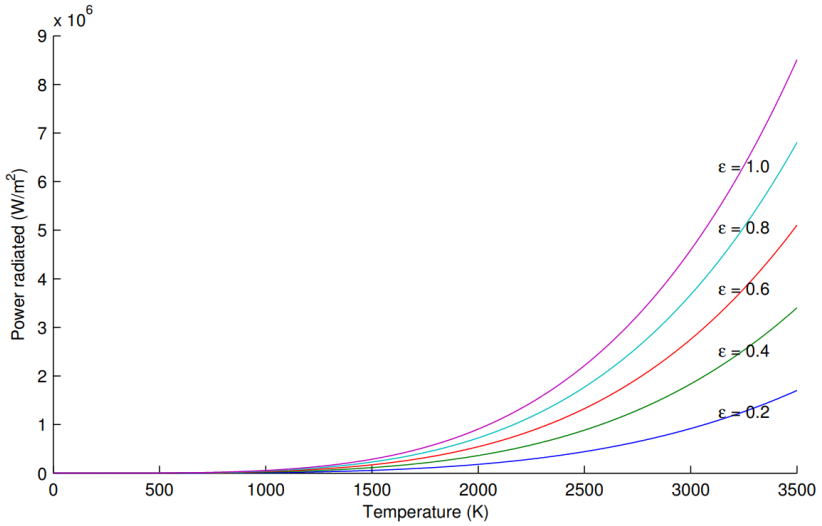


Figure 4: Illustration of power radiance over temperature scale for different material with different emissivity [3]

Emissivity is a materials property of the emitting object which influence thermal cameras. In addition to that, other factor like temperature, reflective surface, surrounding conditions of natural environment can affect thermography [3].

2.2.2 Atmospheric Window

In 1840, John Herschel, the son of William Herschel, discovered atmospheric windows, which are of great importance for the technical understanding of infra-red sensors. For temperature measurement using infra-red radiation, we convert the energy detected into electric signal and then into temperature. To measure, the temperature of the surface accurately, the radiation from other source needed to removed (mainly surrounding atmosphere or air). This process is called as compensation [2].

The total radiation can be divided as emission of target object(E_{obj}) ,emission of the reflected object (E_{refl}) and emission from the atmosphere (E_{atm}) [3].

$$W_{tot} = E_{obj} + E_{refl} + E_{atm} \tag{2.2.7}$$

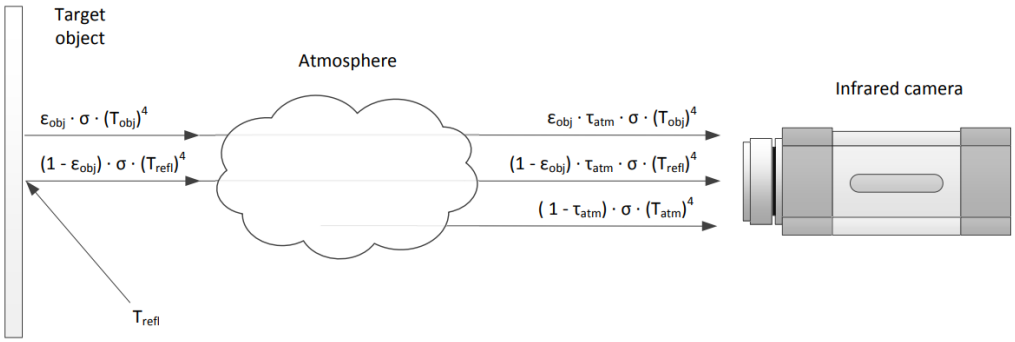


Figure 5: Illustration of Atmospheric window schematic diagram [3]

.

The energy from the target object need to pass through atmosphere. Thus the emission of the target object is a function of transmittance of the atmosphere (τ_{atm}) and radiance at the target obtained using Stefan-Boltzmann equation (equ.2.2.4) [3].

$$E_{obj} = \tau_{atm} \cdot W_{obj} = \varepsilon_{obj} \cdot \tau_{atm} \cdot \sigma \cdot (T_{obj})^4 \tag{2.2.8}$$

The energy reflected into surrounding atmosphere is function of the radiance of atmosphere and emittance of object into surrounding atmosphere [3].

$$E_{refl} = \rho_{obj} \cdot \tau_{atm} \cdot \sigma \cdot (T_{refl})^4 = (1 - \varepsilon_{obj}) \cdot \tau_{atm} \cdot \sigma \cdot (T_{refl})^4 \tag{2.2.9}$$

The energy of the surrounding atmosphere is function of (1 -τ_{atm}) is known as the emittance of the atmosphere.

$$E_{atm} = \varepsilon_{atm} \cdot \sigma \cdot (T_{atm})^4 = (1 - \tau_{atm}) \cdot \sigma \cdot (T_{atm})^4 \tag{2.2.10}$$

The total radiation from the body is given as

$$W_{tot} = \varepsilon_{obj} \cdot \tau_{atm} \cdot \sigma \cdot (T_{obj})^4 + (1 - \varepsilon_{obj}) \cdot \tau_{atm} \cdot \sigma \cdot (T_{refl})^4 + (1 - \tau_{atm}) \cdot \sigma \cdot (T_{atm})^4 \tag{2.2.11}$$

The temperature of the object is calculated from Equ.[2.2.11]

$$T_{obj} = \sqrt[4]{\frac{W_{tot} - (1 - \varepsilon_{obj}) \cdot \tau_{atm} \cdot \sigma \cdot (T_{refl})^4 - (1 - \tau_{atm}) \cdot \sigma \cdot (T_{atm})^4}{\varepsilon_{obj} \cdot \tau_{atm} \cdot \sigma}} \tag{2.2.12}$$

The transmittance of the atmosphere (τ_{atm}) is generally estimated using the distance from the object to the camera and the relative humidity which is very close to one. The temperature of the atmosphere is calculated using a common thermometer. Thus, the emittance (ε_{atm}) of the atmosphere is very close to zero (1 - τ_{atm}). The emissivity of the object (ε_{obj}) and the reflected temperature (T_{refl}) have a very high influence on the temperature measurement and must be measured very accurately [3].

2.3 Sensors and Digitalization

Digitalization of thermography started after the use of semi-conductor based sensor which convert infra-red radiation into electrical signal. The sensor generates voltage difference proportional to radiation. Most of the modern day thermography cameras are made using InSb, InGaAs, PtSi, HgCdTe technology.[3]

There are two types of infra-red cameras

- Single Sensor.
- Focal Plane Array (FPA).

2.3.1 Single Sensor

In the late 1950s, Texas Instruments developed single-element detectors. A single sensor consist of color filter on a rotating mirror. The rotating mirror measures the radiation along the horizontal axis generating linear fringe known as line-scan. The line-scan are repeated for the measurement of the object to create a temperature-time discrete value. Due to it's sensitive military application, the technology was not available for commercial use till 1970s. These detectors laid the foundation for the development of modern thermal imaging cameras [24].

2.3.2 Focal Plane Array (FPA)

Focal plane array is currently used for digital photography and video recording. FPA convert infrared radiation into an electrical signal which can be processed and stored. The architecture of FPA is indicated in fig.6 which consist of million of photoelectrons transistors in a 30 mm pitch pixel integrated with a capacitor to store the the voltage. After the thermogram is collected, the resultant charge-based image is convert from analog-to-digital format using a sequence of analog bus which process the data by row-by-row or column-by-column.

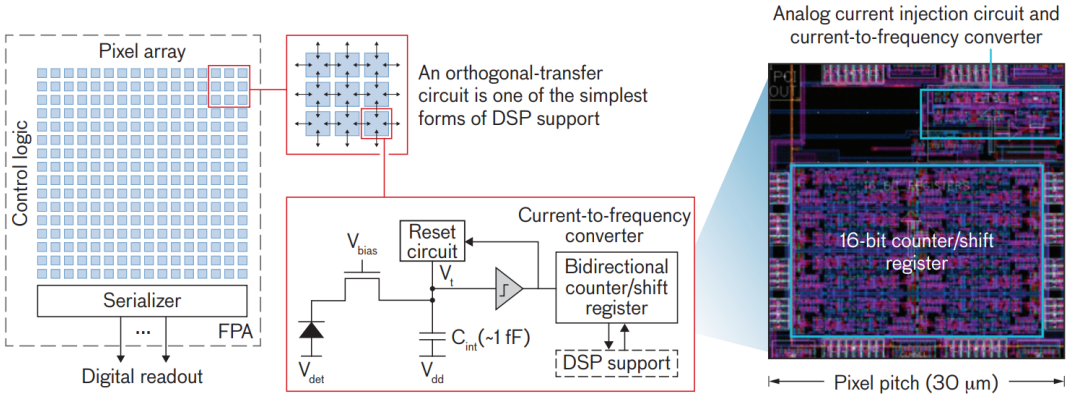


Figure 6: Illustration of focal plane array (FPA) architecture with simplified unit-cell circuit diagram and unit-cell layout [4]

The thermogram can be described as a function of f(x,y). The function is the amplitude of the signal for scalar co-ordinate values, which corresponds to the intensity of the radiations. The f(x,y) is a continuous function in space and amplitude. The analog bus can converts the function f(x,y) into a digital format.

Sampling

Digitization of image space signifies the resolution of the infrared camera which depends on the number of photoelectrons transistors on FPA. The resolution consists of M x N (rows x columns) transistors and typically range from 120 x 140 to 1280 x 1024 [4].

Quantization

Digitization of amplitude which defined the intensity of radiation along co-ordinates space are represented in the form of bits. A typical value is 16 bits. Quantization causes the change in the sensitivity due to noise-equivalent temperature difference [4].

Temporal sampling

Digitization of time are the number of image or frame per seconds which can be processed. Typical frame rate are 30 Hz and 60 Hz which depend on the camera type. Current Analog-to-Digital converters (ADC) can process data at 20 mega-pixels per second. Thus, a 1 mega-pixel data can be read at 120 Hz video frame rate [4].

2.4 Classification of Infrared Thermography

Thermography is mainly divided into two types, i.e.

- Passive thermography.
- Active thermography.

2.4.1 Passive Thermography

Passive thermography uses natural heat of the object to generate a thermal gradient based on which the defect area is identified due to it's thermal contract with the surrounding surface. However, the quality of an evaluation depends on number of factor like humidity, weather etc.

As passive thermography doesn't require any external heat, it can be used for large scale inspection. Passive thermography can be used for monitoring industrial

production processes, for detecting processes, the detection of thermal bridges and leaks in the construction industry, and even the electrical connections. However, it cannot be used for detecting complex defects like anomalies on surface with multiple coating .

2.4.2 Active Thermography

The basic difference between the passive and active thermography is the absence of any external heat stimulus. Active thermography requires a heat source to excite the object to generate a consistent thermal gradient of the surface. Quantitative analysis like defect length, width and depth can be accessed by controlling the amount of heat projected onto the target object. Due to the need of heat source, active thermography has spacial constrains.

Active thermography can be classified based on excitation methods.

- Lock-in thermography - The heat source supply energy in the form modulated sinusoidal wave energy from which the object phase and amplitude are analysed.
- Pulsed thermography - The heat source provides a short pulse of energy for few milliseconds using which the temperature curve from the object are analysed.
- Heat-Flow thermography - A type of pulsed thermography in which the heat source provide energy for few seconds (used for thermographic analysis on thermal retardant materials)

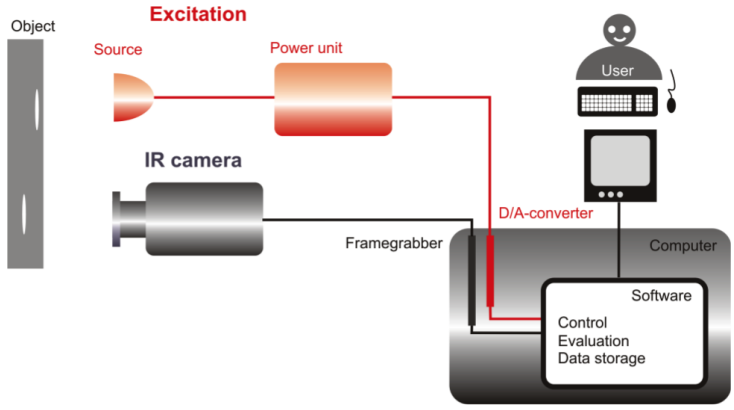


Figure 7: Illustration of active thermography experiment [5]

Excitation Sources

In active thermography, there is need of a excitation source. The following tools are used as excitation sources.

- Thermal blankets - They provide precise heating for medium and larger object and can be programmed to change the heating range. But data can be acquired during cool-down analysis as blanket covers the object during heating phase.
- Heat gun - It's a simple, low cost and heats the object by blowing hot air toward the point of interest. It can be used for small or medium sized objects.
- Flash lamp - It generates a short,flash of high intensity heat energy which spans for few milliseconds.
- Halogen lamp - It generates high intensity heat energy for a long period of time.

Figure 8: Heat Gun [4]

Figure 9: Thermal Blanket [4]

Figure 10: Flash Lamp [4]

Figure 11: Halogen Lamp [4]

Chapter 3

Machine Learning

3.1 Introduction

> Machine learning discovers rules to
> execute a data-processing task,
> given examples of what's expected.
>
> *Francois Chollet*

Turing machine developed by Alan Turing was the first computational device used to decode enigma codes. This machine can be called as first machine learning model as the parameters were adjusted until the machine was able to predict the output precisely. The term machine learning was coined by Arthur Samuel and defined it as -"**Field of study that gives computers the capability to learn without being explicitly programmed**".

Machine learning (ML) can be understood as an automation process were the machine try to learn the process and correct itself based on it's experiences without any assistance from human. With advancement in computational power, the application of machine learning and deep learning techniques in fields like science, technology, finance, etc. is predominant. This caused a change in the ideology from "**Think and Store Data**" to "**Store Data and Think**" [25].

Traditional Programming: Data (Inputs) + Program (algorithms)= output.

Machine Learning: Data (Inputs) +Output = Learns logic within the Program (algorithms).

Mostly, Machine learning is a advance curve fitting tool, i.e We try to generate

a curve which could fit the known data points using sophisticated algorithms or methods.

3.1.1 Classification of Machine Learning models

The classification of ML models depends on the type output provided to the system.

- Supervised Learning : The model learns from the inputs and associated outputs which are numeric values divided into multiple classes. Supervised learning predicts whether the data lies within the classes or not.
- Unsupervised Learning : The algorithms in this model are provided with input data with no labelled target or outputs. The algorithms tries to predict the pattern in the data and create new classes or new series.
- Reinforcement Learning : In this method, the algorithms have the ability to take decision based on the output and correct itself to achieve maximum reward. Generally used in Autonomous driving systems.

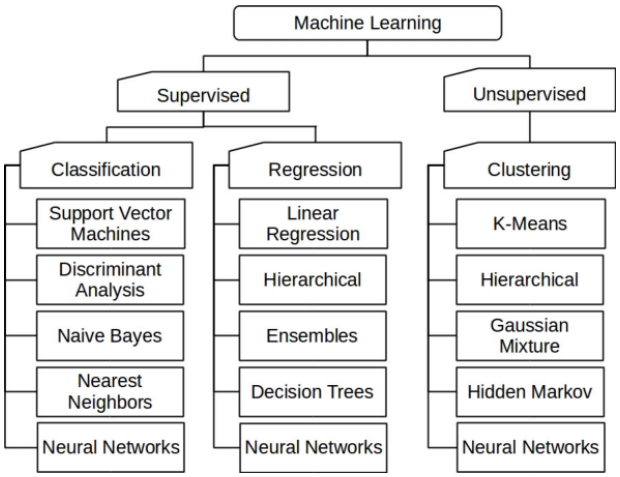


Figure 12: Illustration of different Machine learning methods and models [6]

Due to large pool of machine learning algorithms. Firstly, the selection of models depends on the complexity, type of dataset and size of dataset. Neural Network (NN) have the ability to solve huge and complex problem for varied datasets.

3.2 Deep Learning

Deep learning is a sub-branch of machine learning in which the model is inspired by the structure of brain or we are trying to create an artificial brain. By mimicking the human brain, we can train the machine to process, think and work as a human.

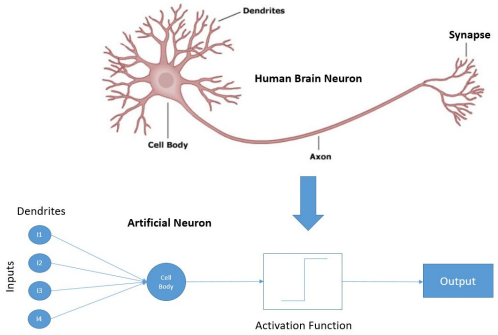


Figure 13: Illustration of single brain cell (Neuron)[7]

By understanding the functioning of human brain, we can model neural network efficiently. A brain consist of brain cells known as neurons which transfer data as electrical impulse. A neuron consist of dendrites which receive impulses from other neurons. The body of neuron is axons which fire the impulses to other neurons.

3.2.1 Artificial Neural Network (ANN)

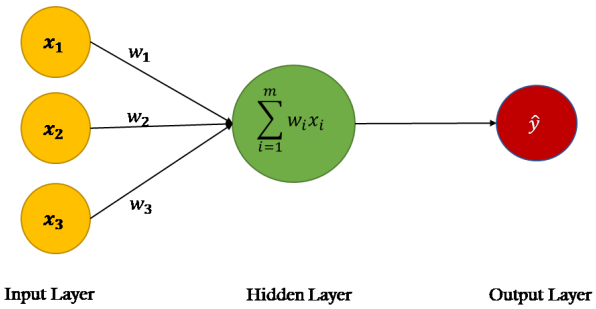


Figure 14: Illustration of single neuron network [8]

Artificial Neural Network (ANN) is an electronic model build based on the neural structure of the brain. Brains store data in form of pattern. ANN is trained to identify those pattern and learn until it is able to predict those patterns. ANN

can be built with modern architecture which can be parallelized, thus a deep ANN can be built.

A single layer perceptron consist of input layer , hidden layer and output layer with no feedback from any other perceptron. From fig 14, we can understand the feed forward architecture of the network were each input x_i has a weight w_i and bias b_i are the parameter of the network which can be adjusted to change the output value.

Hidden layer consist of perceptrons which is $\sum_{i=1}^{m} w_i x_i + b_i$ after which activation function g(:) applied. ANN is built using McCulloch-Pitts neurons. In McCulloch-Pitts neurons based on the threshold value the neuron is either active or inactive. If active, information is passed to the next layer else not. This can be achieved by using specific activation function.[26]

Activation Functions

Activation function manipulates the incoming data in hidden layer before sending it to another layer. Non-linear activation function improve the performance of the neural network. Activation Function in the output layer depends on the type of problem.

Simple classification : Logistic activation (Sigmoid)

Multi classification : SoftMax activation (ReLU)

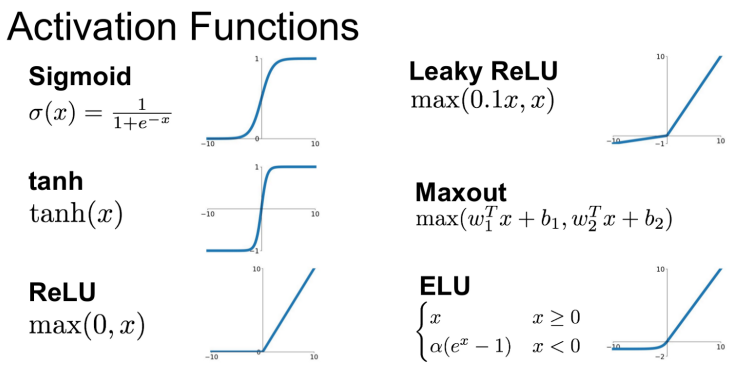


Figure 15: Illustration of different activation functions [9]

Some typical activation functions used in Neural Networks are given in the fig along with corresponding formulae.

3.2.2 Working Principle

Training of neural network depend on two processes feed-forward propagation and feed- backward propagation.

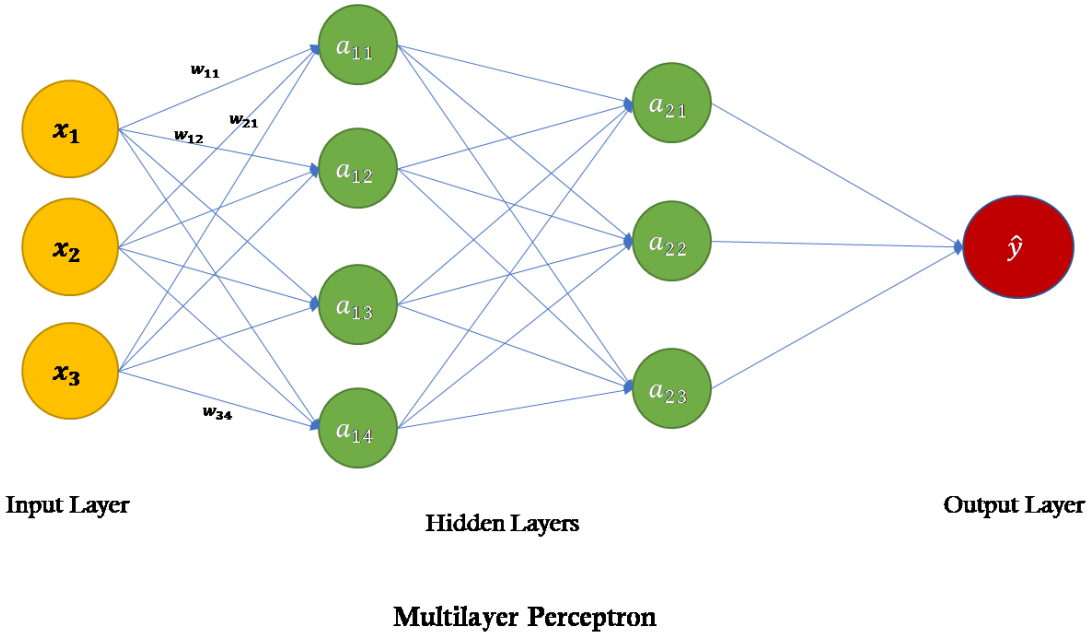


Figure 16: Illustration of multi layer perceptron [8]

Loss Function A loss function also known as cost function. It measures the compatibility between predicted output and actual output.A type of loss function is a hyper-parameter determined based on type of problem. General mathematical formulation is as follows, Let the training data (known outputs for inputs) be given as input-output pairs as :

$$\{(x_1, y_1), \ldots, (x_N, y_N)\} \subseteq X \times y \tag{3.2.1}$$

Where X is the input sample space and Y is the output sample space. A function which maps from X to Y and gives predictions($\hat{y}$) according to training data, i.e

$$f : x \mapsto \hat{y} \tag{3.2.2}$$

The function f estimation is called "Forward propagation" and computing the loss and gradients is referred as "Backward propagation" As a performance measure(to see whether f is doing a good job) loss function is defined as

$$L : Y \times y \to R \tag{3.2.3}$$

Here L measures how close actual y is to the predicted output $\hat{y}$

1. Squared loss :
$$L(y, \hat{y}) = |y - \hat{y}|^2 \text{ for } y = R \tag{3.2.4}$$
2. Zero one loss :
$$L(y, \hat{y}) = 1_y(\hat{y}) \text{ for arbitrary } y \tag{3.2.5}$$
3. Cross-entropy loss :
$$L(y, \hat{y}) = -(y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})) \text{ for } y = [0, 1] \tag{3.2.6}$$

If $y = 1$:the Loss function, $Ł(\hat{y}, y) = -\log \hat{y} \leftarrow$ will push the parameter $\log \hat{y}$ large, want $\hat{y}$ large.

If $y = 0$: the Loss function, $Ł(\hat{y}, y) = -\log(1 - \hat{y}) \leftarrow$ will push the parameter $1 - \hat{y}$ large, want $\hat{y}$ small.

Feed-Forward Propagation During training, initially all weight w_i and bias b_i are randomly initialized. The inputs x_1,x_2,x_3 propagate from layer-by-layer until the output is obtained. The main intent of forward propagation is to check the response of the network for the current set of values based on which the loss is calculated [27].

$$\hat{y} = \sigma\left(w^T x + b\right), \text{ where } \sigma(z) = \frac{1}{1 + e^{-z}} \tag{3.2.7}$$

The cost function is the derivative of the loss function equ.3.2.6]

$$J(\omega, b) = \frac{1}{m} \sum_{i=1}^{m} L\left(\hat{y}^{(i)}, y^{(i)}\right) = -\frac{1}{m} \sum_{i=[}^{n} \left[y^{(1)} \log \hat{y}^{(i)} + \left(1 - y^{(1)}\right) \log(1 - \hat{y}(i)]\right] \tag{3.2.8}$$

$$\left.\begin{array}{l} x_1 \\ w_1 \\ w_2 \\ \mathbf{b} \end{array}\right] \longrightarrow \underbrace{z = w_1 x_1 + w_2 x_2 + b} \longrightarrow a = \sigma(z) \longrightarrow \mathcal{L}(\mathrm{a}, y) \tag{3.2.9}$$

$$\begin{aligned} z^{[l]} &= w^{[l]} A^{[l-1]} + b^{[l]} \\ A^{[l]} &= \sigma^{[l]}\left(z^{[l]}\right) \end{aligned} \tag{3.2.10}$$

where $A^{[l-1]}$ is the parameters from the previous layer. For first hidden layer, the parameters are the inputs.

Feed-Backward Propagation Back propagation algorithm is known as generalized delta rule. Based on the loss calculated after forward propagation, the gradients are calculated using which the weights and bias are updated based on the chain rule. The chain rule in calculus states that:

$$\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx} \tag{3.2.11}$$

The threshold of the neurons are adjusted automatically in each iteration, this process is repeated until the sum of the errors produced by the network is minimal. The gradient of loss function wrt weights is given as,

$$\begin{aligned} dz^{[l]} &= dA^{[l]} - Y \\ A^{[l]} &= g^{[l]}\left(z^{[l]}\right) = Y \\ d\omega^{(l)} &= \frac{1}{m} dz^{[l]} \cdot A^{[l-1]^T} \\ db^{[l]} &= \frac{1}{m}\left(dz^{[l]},\ \text{axis } = 1,\ \text{keepdims } =\ \text{True }\right) \\ dz^{[l-1]} &= w^{[l]^T} dz^{[l]} * g^{[l-1]'}\left(z^{[l-1]}\right) \end{aligned} \tag{3.2.12}$$

We need a gradient descent method to update parameter to minimize the loss function equ.3.2.6.

Optimizer An optimization algorithm which is used to iteratively update the trainable parameters to minimize the loss function. Hyper-parameters of gradient descent is learning rate (α) with the step size for the steepest descent to take place.

$$\begin{aligned} w &:= w - \alpha * \frac{\partial L}{\partial w} \\ b &:= b - \alpha * \frac{\partial L}{\partial w} \end{aligned} \tag{3.2.13}$$

Types of optimizers:

1. Batch Gradient Descent
2. Stochastic Gradient Descent
3. Stochastic Mini-Batch Gradient Descent

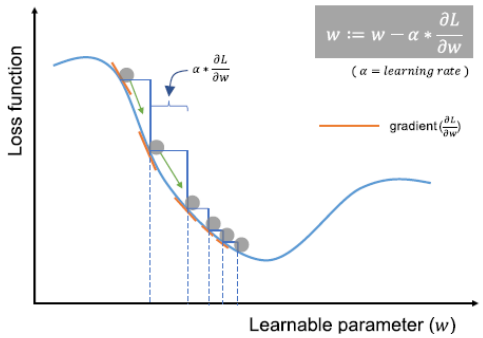


Figure 17: Gradient descent to minimize a function [10]

3.2.3 Convolution Neural Network (CNN)

Multi layer perceptron or ANN cannot be used to identify complex pattern or solve complex data like images without generating huge number of trainable parameter. The revolution in machine learning started after the used of CNN. CNN is used for processing data that is in grid format, such as images, videos. Most of the computer vision tasks use CNN. CNN adaptively learns spatial features from low (edges) to high (complex) patterns as the number of CNN layers increase.

CNN consists of three layers

1. Convolution layer
2. Pooling layer
3. Fully connected layer

Convolution layer

Convolution layer performs convolution operation. Convolution is mathematical operation in which summation of element-wise product between kernel and input tensor to obtain an output value with corresponding position in output tensor as shown in equ.3.2.14. The convolution operation extracts features like edges, pattern which could be identified for pattern.

Convolution operation consist of two parameters like stride and padding. Stride is the number of pixels which have to be skipped during convolution. Padding is adding zeros to the boundary of the images to increase the weight age of edge pixels. The hyper-parameters of convolution layer are size of the kernel and number of kernels.

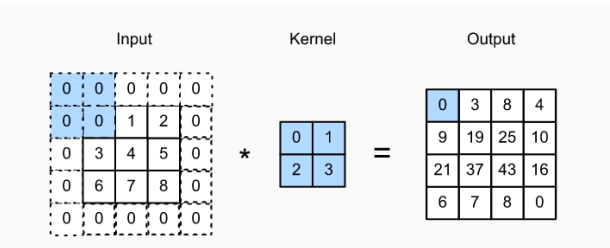


Figure 18: Convolution operation using a 2x2 kernel with 1x1 padding [10]

$$f(x) * g(x) = \int_{-\infty}^{\infty} f(\tau) \cdot g(x - \tau) d\tau \tag{3.2.14}$$

Pooling Layer

A pooling layer is used to reduce the in-plane dimensionality of the feature map in order to decrease the effect on distortions and decreasing the number of trainable parameters. Pooling layer has no trainable parameters. There are two type of pooling layer.

1. Max pooling
2. Global average pooling

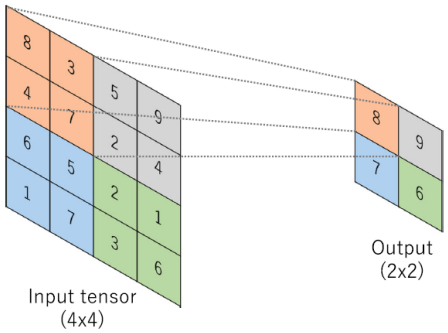


Figure 19: Illustration of max-pooling operation.[11]

A max value or a average of all the values are taken based on the kernel size to reduce the size of the tensor. Hyper-parameters of pooling layer are kernel size, stride, padding similar to convolution operation.

Fully Connected Layer

After the convolution and pooling operation, output feature map are flattened into 1-D array and connected to one or more hidden layers with independent weight and bias for each unit which are followed by non-linear function. Hyper-parameters of fully connected layer are number of hidden layer and number of units in each hidden layer.

This layer enables use to create output layer of our choice.

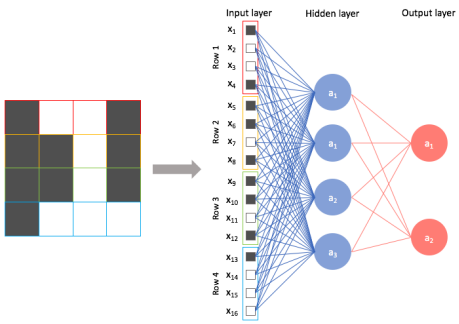


Figure 20: Illustration of fully connected layer[11]

Working Principle

Similar to ANN, CNN consist of feed
forward and backward propagation but the parameter in CNN are the weight and biases assigned to the kernels as shown in fig.[21] For example, the first feature map consist of 32, 5 x 5 kernels which on whole are 800 weights w_i and 800 bias b_i.

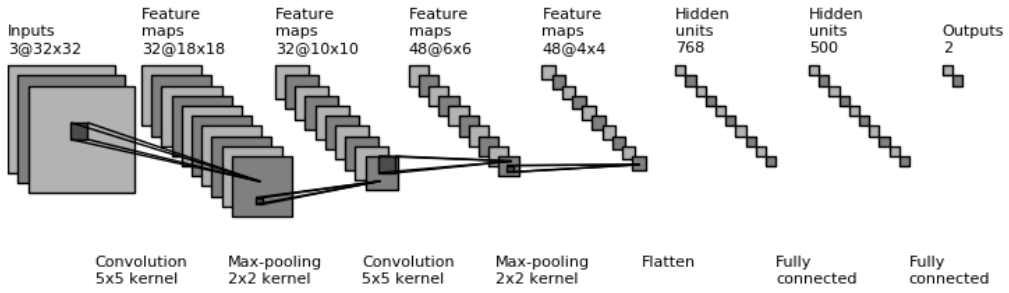


Figure 21: Illustration of CNN network with feature size of each convolution block [12]

3.2.4 Recurrent Neural Networks (RNN)

A recurrent neural network (RNN) is used for sequential data or time depended data. Unlike ANN or CNN, RNN have the ability to store information from previous time steps which can be passed to the next time step. In traditional NN, the inputs and outputs are independent of each other. In RNN, the output depends on the prior elements. Different type of RNN are indicated in fig.22

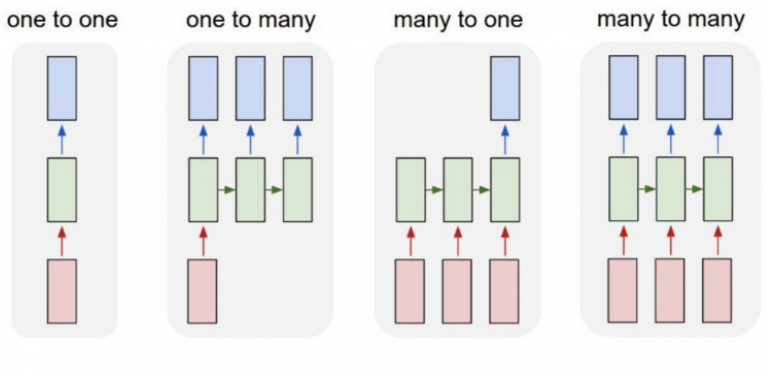


Figure 22: Illustration of different types of RNN [11]

Working Principle

RNN is a recursive technique which require a huge computational effort.

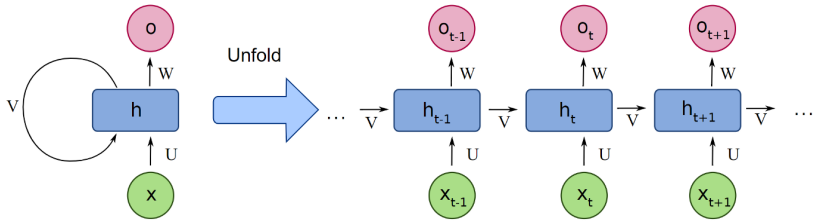


Figure 23: Illustration of a basic RNN unit [13]

x: The input,

O: The output at time step,

y: Final output of RNN

h: The main block of the RNN. It contains the weights and the activation functions of the network,

V: Represents the weight which are carried to net time step

Feed-Forward Propagation In feed-forward propagation, the information moves in forward direction and never touch the nodes twice. The input to RNN is the present data and recent past data. In feed forward, weights are assigned to the present inputs and previous state inputs based on which the output is effected. Initially the weight are random Gaussian distribution. The weight are update during back-propagation.

$$\begin{aligned} \boldsymbol{a}^{(t)} &= \boldsymbol{b} + \boldsymbol{W}\boldsymbol{h}^{(t-1)} + \boldsymbol{U}\boldsymbol{x}^{(t)} \\ \boldsymbol{h}^{(t)} &= \tanh\left(\boldsymbol{a}^{(t)}\right) \\ \boldsymbol{o}^{(t)} &= \boldsymbol{c} + \boldsymbol{V}\boldsymbol{h}^{(t)} \\ \hat{\boldsymbol{y}}^{(t)} &= \text{softmax}\left(\boldsymbol{o}^{(t)}\right) \end{aligned} \tag{3.2.15}$$

Feed-Backward propagation Also known as back propagation through time. After the calculation of loss function by using chain rules the gradients of the respective variable are update using gradient descent method.

The drawback of RNN are it can not store memory for higher hidden units

and it's inability to be parallelized which needs huge computational effort also the neural network runs recursively which creates a huge overhead.

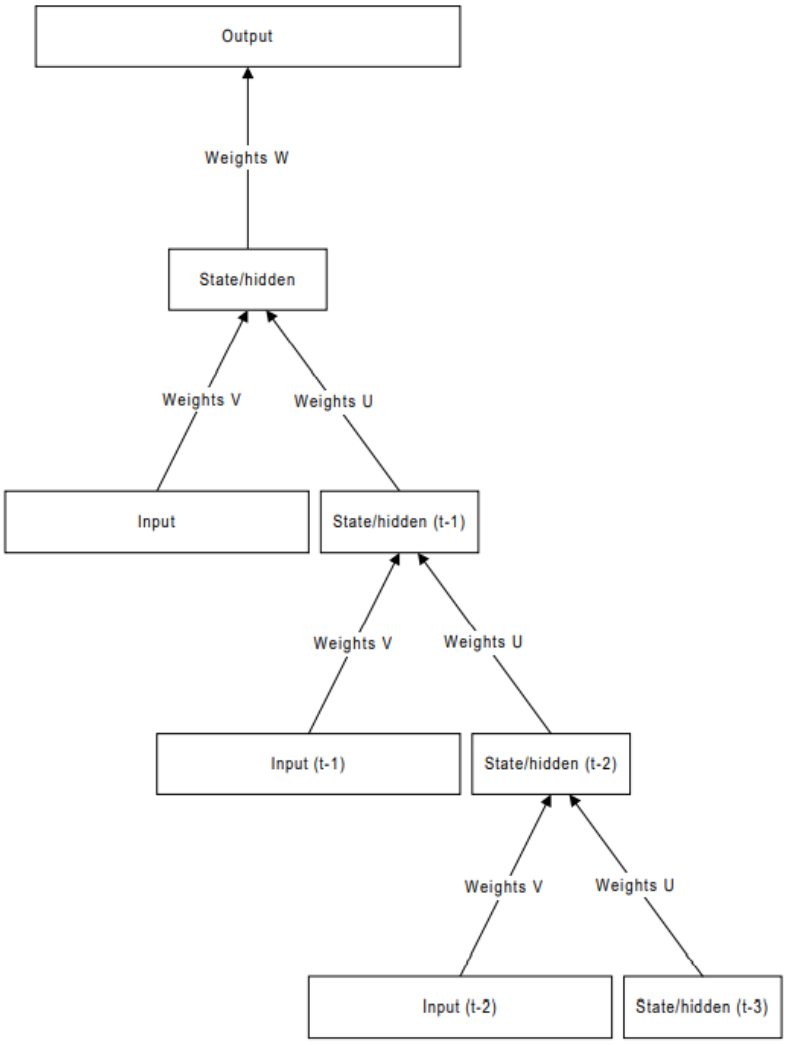


Figure 24: Back Propagation Through Time (BPTT) for 3 time steps [14]

The fig.24 shown the flow of data during back propagation

Chapter 4

Software Tools

In this chapter, the tools and software's used for this project will be discussed in detail.

4.1 Python

Python is a high-level programming language which supports object oriented approach. Python was developed by Guido van Rossum in 1989. For our project, we have used python version 3.8.3. Python has wide range of libraries which optimize the performance of code. The following libraries/packages were used during the development of the software.

4.1.1 Numpy

Numpy is the most popular library on python. It is used to perform high-level mathematical operation on multi-dimensional large arrays or matrices. Numpy was created by Jim Hugunins. Numpy is written in C and works using Cpython reference framework. During compilation, the python code is converted in a low-level structure which runs on C.

4.1.2 Matplotlib

Matplotlib is a visualization library which is an extension to numpy. It was developed by John D. Hunter. It is built in python and uses the graphics memory to

render plot for visualization. It supports object oriented programming and can be integrated with GUI tools like tkinter,Qt, etc.

4.1.3 Pandas

Pandas is a data-frame library which is used for the manipulation of data. Pandas can load data from various format like json, csv, asci in data frame format and perform cleaning, linear regression, shifting, lagging, pivoting. The pandas is integrated with matplotlib which enables efficient data transformations and visualization. Pandas is used to perform exploratory data analysis (EDA.

4.1.4 Sci-kit learn

Scikit learn is a machine learning library. We can perform classification, regression and clustering operation. It consist of random forest, K-means clustering, support vector machines (SVM), principle component analysis (PCA). The Scikit learn has other functionality like splitting data for training and testing, normalization and scaling of data, other image modification operation. It is built on top of Scipy and worked Cpython reference framework.

4.1.5 OpenCV

Opencv is an open-source software developed by intel used for computer vision and image manipulation. It has wide range of functionality to perform operation on images or videos. It is used for object detection,segmentation and image feature extraction. We can perform logical gate operation on images to extract important features. The software is written in C++. It can interact with the GPU and work on them to increase the performance. It also contains statistical machine learning algorithms like support vector machines, random forest, gradient boasting, etc.

4.1.6 Tkinter

Tkinter is a python GUI library. It is build in tk interface. Tkninter binds python with tcl interpreter. It can be used to make desktop application with python as back-end framework.

4.1.7 Pytest

Pytest is used to write and execute test code. It is mainly used for API testing. We can write simple to complex test cases to test the API, databases, UI, etc. Pytest can be parallelized reducing the execution time of test cases.

4.1.8 Other Libraries

- CSV,JSON,h5py are libraries used for data manipulation of respective file format.
- Time module is inbuilt python package used time analysis.
- OS module is used for Unix python package used for bash or shell scripting.

4.1.9 labelme

Labelme is a graphical image annotation tool inspired by http://labelme.csail.mit.edu. It is written in Python and uses Qt for its graphical interface. It can be used to create segmentation, bounding box or primitive shape like polygon,rectangles, circles or polyline annotations which can be exported in various formats.

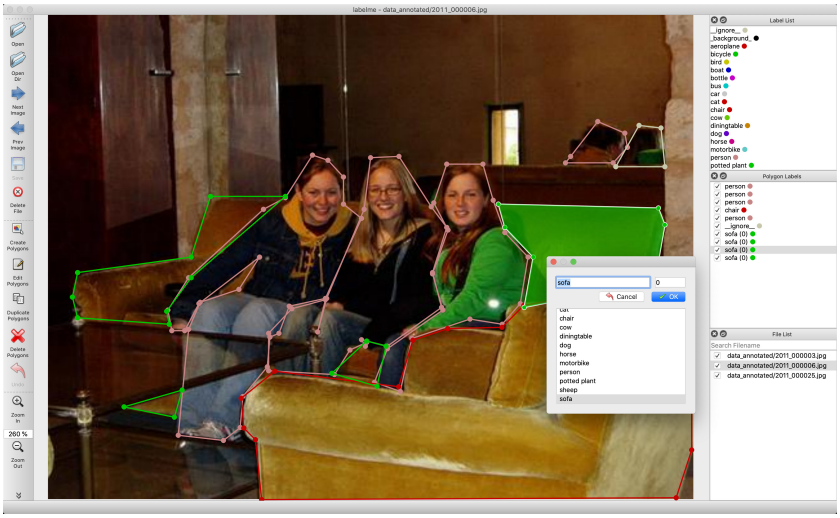


Figure 25: labelme tool interface.

4.2 TensorFlow

Tensorflow is an open source fast numerical computing framework owned by Google. Tensorflow is built on python,C++ and CUDA. Tensorflow is a flexible framework used for wide range of application and platforms. It can run on multiple CPU, GPU and TPU which can be used for research as well as for production.

Tensorflow library can be used to create deep learning models. Tensorflow has other module like auto-differentiation, optimizers, losses, metric which are required to update parameters during back-propagation.

4.2.1 Keras

Keras is a python library for deep learning which run on top of tensorflow. It can seamlessly integrate on the CPU and GPU increases the performance of the model.It is released with MIT licence. Keras is used for this project work because of its ease of readability, less complexity and it's modularity. The framework is design to have get better visualization and understanding of deep learning algorithms.

A deep learning model using keras:

- **Model creation**: Based on the architecture of the model, the sequential or functional API can be used to built a model.
- **Compilation**: After the model is generated, NN parameters like loss function and optimizers are specified along with their respective hyper-parameter during compilation process. compile() creates a executable code(gradients of all trainable parameters) which can be trained.
- **Training**:fit() is used to train the model. We can extract important training information like loss and accuracy at each epoch which are used for optimizing the hyper-parameter to improve the model accuracy.
- **Evaluation and prediction**: After training the model on training data. evaluate() is used to check the performs on test data through loss and accuracy. predict() is used predict the output based on the input data.

4.2.2 Sequential and Functional API

Sequential API Sequential API is easy to use and understand. It has a minimalist architecture. It allows the creation of model from top-to-bottom approach

by building layer-by-layer. This API is simple to use but has limitation and rigidity issues. The sequential API allows you to create models layer-by-layer for most problems. It is limited in that it does not allow you to create models that share layers or have multiple inputs or outputs.

```
seq_model = Sequential()
seq_model.add(GRU(32, input_shape=(100,1)))
seq_model.add(Dense(256))
seq_model.add(Dense(128))
seq_model.summary()
```

Listing 4.1: Sequential API

Functional API Functional API is more flexible and can handle non-linear topology, multiple input and output,shared layers. Functional API create a directed acyclic graph (DAG) which uses computational graphs to link the layer.

```
input1 = Input(shape=(100,1))
layer1 = LSTM(32, input_shape=(100,1))(input1)
layer2 = Dense(256)(layer1)
output1 = Dense(128)(layer2)
output2 = Dense(1)(layer2)
func_model = Model(inputs=input1, outputs=[output1, output2])
func_model.summary()
```

Listing 4.2: Functional API

4.2.3 Google Colaboratory (Colab)

Colab is a cloud hoster jupyter notebook service which can run on any browser without any setup. Colab allows for writing and execution of python code especially for machine learning, data analysis. It can be linked to Github and Google Drive for storing and accessing of files and data. Colab provides a free of charge GPU and TPU resources. The GPU resources are shown in fig.26.

4.3 Git

Git is a version control system which enables code deployment for tracking, continuous integration and deployment. Git support non-linear and distributed development along with cryptographic authentication of history for roll-backs or addition to code.

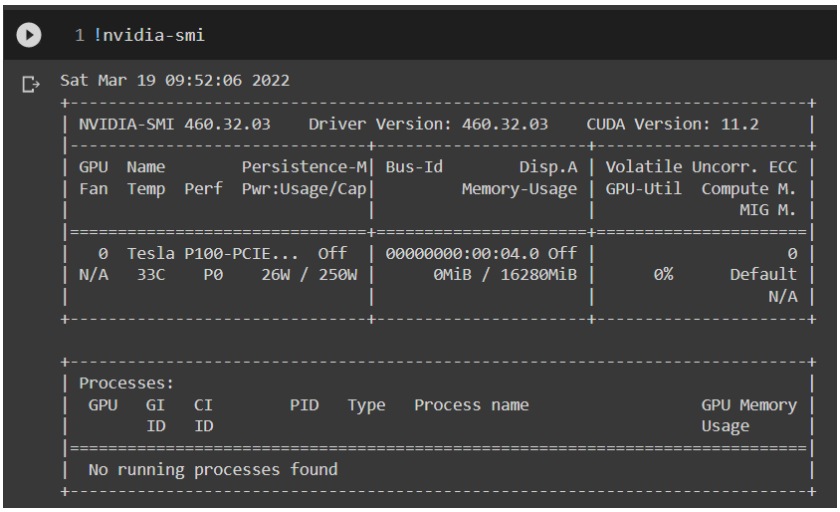


Figure 26: GPU resources on Google Colab .

For this project, we have use Gitlab. GitLab is open-source devOps tool used for development, security and operation of a software. Gitlab was developed by Ukrainian developer Dmitriy Zaporozhets and Dutch developer Sytse Sijbrandij. It is created using Ruby programming.

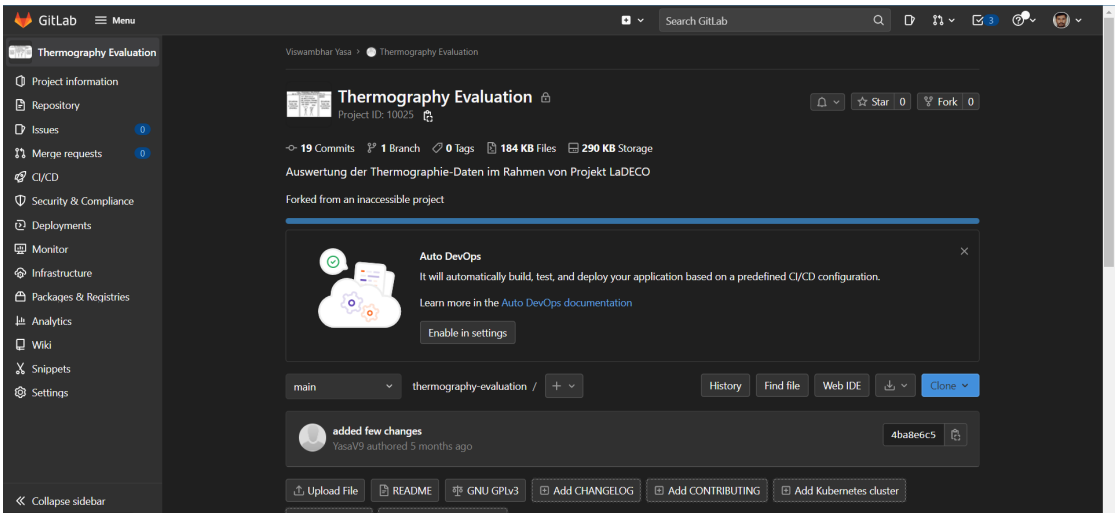


Figure 27: LaDECO gitlab repository.

Chapter 5

Methodology

5.1 Thermography Data Acquisition

5.1.1 Experiment Setup

The acquisition of thermogram is done using a special set-up as shown in fig.28.

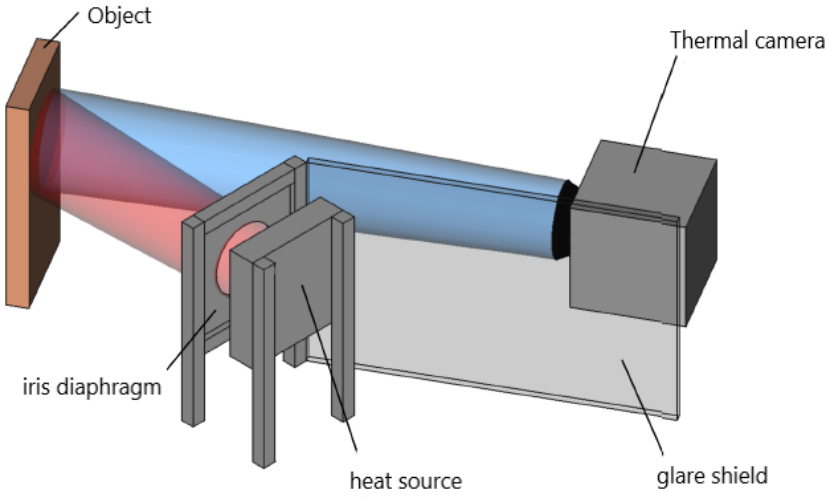


Figure 28: Illustration of Experimental setup [15]

The rig consist of a thermal cameras holder on a movable table with roller which are parallel to the object. The whole set-up is encased by a black lightweight foam board to protect the experiment from background radiation.

The right side of the infra-red cameras consist of housing which holds heat source

with a diaphragm controlled by compressed air. The object holder can be rotated at any angle. The camera and object are at angle w.r.t each other.

The following devices were used for experimentation:

Infra-red camera:

- VarioTHERM
- Optris PI 640

Radiation sources:

- Halogen lamp (1000 W)
- Infra-red spotlight (2500 W)
- Flash lamp (400 W)

5.1.2 Experiments

The test object is shown in fig.32. The object consist of coating which is partially and completely removed by laser. Some part of the substrate are burnt and turned into glass due to high temperature from laser.

Table 1: List of experiments for coating segmentation

Experiment number	Camera	Radiation source	Angle of incidence
1	VarioTHERM	Halogen lamp	0°
2	VarioTHERM	Halogen lamp	22.5°
3	VarioTHERM	Halogen lamp	45°
4	VarioTHERM	Infrared-spotlight	0°
5	VarioTHERM	Infrared-spotlight	22.5°
6	VarioTHERM	Infrared-spotlight	45°
7	VarioTherm	Flash lamp	0°

The experiment is conducted by changing the angle of incident of heat source on object. A 0°,22.5°,45° of incident is used to check the change in the thermography as shown in fig.29 and fig.30

The experiments are carried out for coating of different thickness as shown in table.[2]

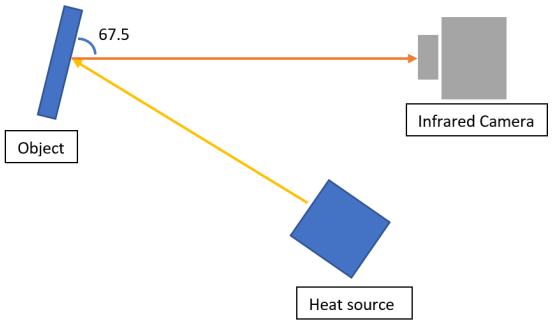


Figure 29: Angle of incidences 22.5°

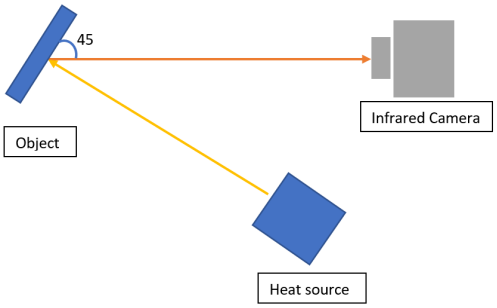


Figure 30: Angle of incidences 45°

Table 2: List of experiment on coating with different thickness and colour

Probennr.	Bezeichnung	1. Schicht	2. Schicht	3. Schicht	Farbe
1	5.1	Gehopon Haftbrücke EW 38-505	-	-	Grau
2	5.1	Gehopon Haftbrücke EW 38-505	-	-	Grau
3	5.1	Gehopon Haftbrücke EW 38-505	-	-	Grau
4	5.2	Gehopon Haftbrücke EW 38-505	Gehopon EW 10-S1013 40µm	-	Weiß
5	5.3	Gehopon Haftbrücke EW 38-505	Gehopon EW 10-S1013 40µm	Gehopon EW 10-S1013 40µm	Weiß
6	5.3	Gehopon Haftbrücke EW 38-505	Gehopon EW 10-S1013 40µm	Gehopon EW 10-S1013 40µm	Weiß
7	7.1	Gehopon Haftbrücke EW 38-505	-	-	Grau
8	7.2	Gehopon Haftbrücke EW 38-505	Gehopon EW 10-S1013	-	Weiß
9	7.3	Gehopon Haftbrücke EW 38-505	Gehopon EW 10-S1013 40µm	Gehopon EW 10-S1013 40µm	Weiß
10	7.4	Gehopon Haftbrücke EW 38-505	Gehopon EW 10-S1013 „dünn“	-	Weiß
11	7.5	Gehopon Haftbrücke EW 38-505	Gehopon EW 10-S1013 20µm	-	Weiß
12	7.6	Gehopon Haftbrücke EW 38-505	Gehopon EW 10-S1013 60µm	-	Weiß
13	7.7	Gehopon Haftbrücke EW 38-505	Gehopon EW 10-S1013 80µm	-	Weiß
14	8.1	Gehopon EW 10-S103 mit Pinsel	-	-	Weiß
15	8.2	Gehopon Haftbrücke EW 38-505	Gehopon EW 10-S1013 40µm	Gehopon EW 10-S1021 40µm	Gelb
16	8.3	Gehopon Haftbrücke EW 38-505	Gehopon EW 10-S1013 40µm	Gehopon EW 10-S3003 40µm	Rot
17	8.4	Gehopon Haftbrücke EW 38-505	Gehopon EW 10-S1013 40µm	Gehopon EW 10-S5010 40µm	Blau
18	8.5	Gehopon Haftbrücke EW 38-505	Gehopon EW 10-S1013 40µm	Gehopon EW 10-S6018 40µm	Grün
19	8.6	Gehopon Haftbrücke EW 38-505	Gehopon EW 10-S1013 40µm	Gehopon EW 10-S9005 40µm	Schwarz

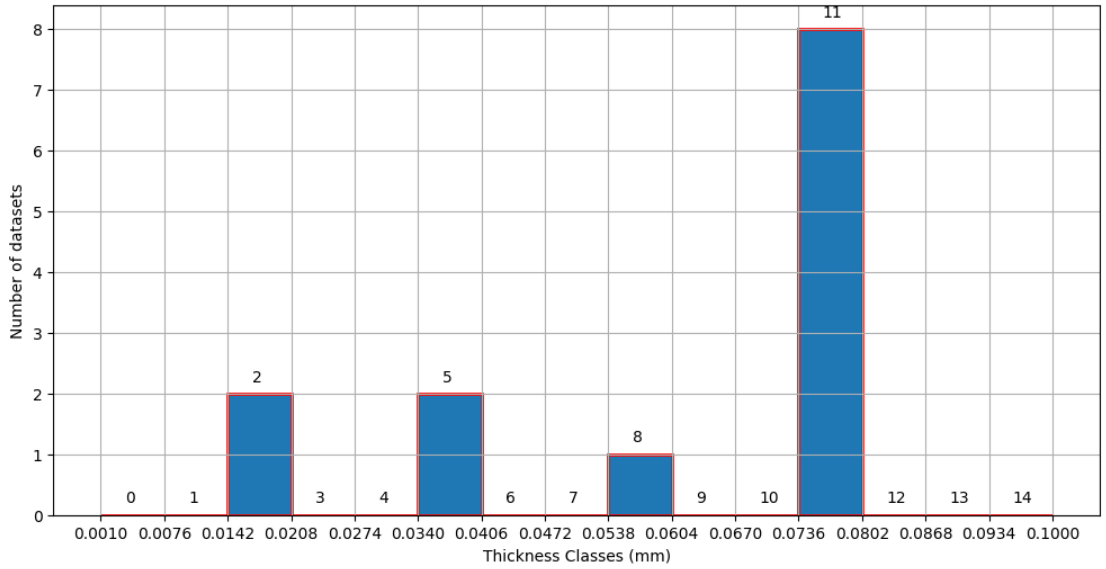


Figure 31: Coating thickness classification.

The above fig.31 is a histogram consist of different coating thickness's and which are classified into classes ranging from 0 to 15(number of bins). The size of the class depends on the number the bins, the larger the size of class lower the number of bins. An example,if the coating is 0.02, then it belong to class 1.

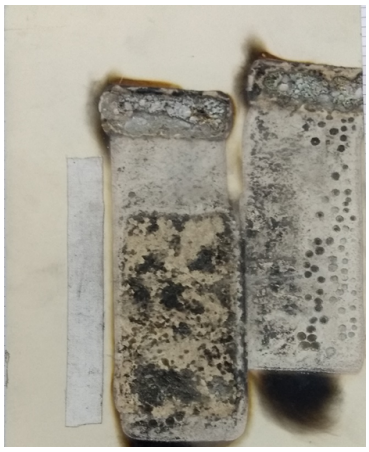


Figure 32: Test sample for object segmentation

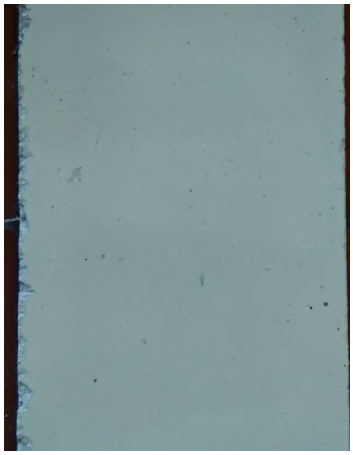


Figure 33: Test sample for coating thickness estimation

5.2 Thermal Sensors

5.2.1 VarioTHERM

VarioTHERM camera heads consist of a mid-wave infrared sensor which provides precise, real-time thermal data for scientific and industrial applications. A detailed technical specification of the camera are provided in tbl.3

Type of detector, image format	Cooled PtSi, 256 x 256 pixels
Spectral sensitivity	1.8 µm… 5 µm
Temperature measurement range	-25 °C ... +1200 °C, optionally up to 2500 °C -13 °F … +2192 °F, optionally up to 4530 °F
Thermal resolution (at 30° object temperature)	<100 mK (< 20 mK in Sensitivity Improvement mode)
Measurement accuracy	±2 K, ±2%
Emissivity	Adjustable from 0.1 to 1.0 in steps of 0.01
FOV/min. object distance/geometrical resolution	14° x 14° / 1.0 mrad / 0.4 m (standard lens)
IR Image rate	50 Hz
Dynamic range	16 bit
Image storage	CF-card, FireWire (IEEE 1394)
Power supply	12 V external
Interfaces	FireWire (IEEE-1394): Real-time data transfer & remote control, RS 232: Remote control S-VHS, C-Video (PAL): External display
Protection	IP 65, IEC 529
Operating temperature range Storage temperature range	-15°C...+50°C / +14 °F … +122 °F -40°C...+70°C / -13 °F … +149 °F
Functions	Up to 10 measurement areas, zoom, up to 10 isotherms, freeze mode, adjustable emissivity, online-averaging, image filter, autosave, autorange, autoimage
Dimensions (LxWxH)	194 mm x 110 mm x 126 mm
Weight	2.6 kg
Additional lenses	Wide-angle 1.4/25 mm 28° x 28° 2.0 mrad 0.2 m (min. focus) Telephoto 1 1.3/50 mm 7° x 7° 0.5 mrad 1 m (min. focus) Telephoto 2 1.3/100 mm 3.5° x 3.5° 0.25 mrad 1.5 m (min. focus) Close-up 1 1.0x 6 x 6 mm² 25 µm 230 mm*) Close-up 2 2.5x 2.6 x 2.6 mm² 10 µm 21 mm*) *) working distance

Table 3: VarioTHERM Technical Specifications

VarioTherm infrared camera systems allow for a precise theromograms even at high speeds. It allows for snapshot and fast data acquisition by handling large amount of data. The data is stored in binary radioactive file formate (.irb) which can only be analysed using IRBIS ®3 professional.

5.2.2 IRBIS ®3 professional

IRBIS ®3 professional is active thermography software tool provide by infraTech inc. to analysis thermograms. IRBIS convert the .irb file format into a sequence of data which can be extracted in the form of images, csv and ascii files. IRBIS also provides you with the option to perform thermal analysis through profiling, histograms and time sequence graphs.

The main disadvantage of using IRBIS 3 is the rigidity of the software and the requirement of a valid licence to use the software.

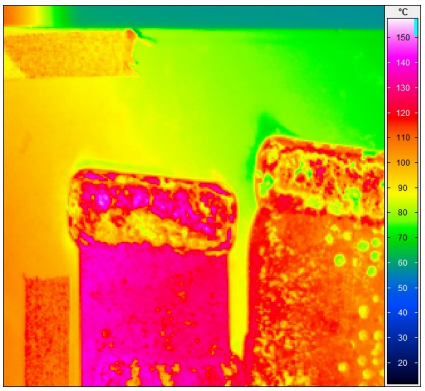


Figure 34: Heat map extracted from IRBIS3

5.2.3 VarioTHERM Thermogram Extraction

To build a thermal analysis system, the dependence on other softwares needs to be minimizes. Thus, a python API is built to extract data from .irb file format.

The data from the thermal cameras is in raw format which are not human-interpretable. Therefore a post processing is required to convert the data for analysis. A colour palettes is also applied to get a heat-map of the object in real-time as shown in fig.35.

VarioTherm class performs the extraction of thermograms from .irb videos.

```
import numpy as np
import matplotlib.pyplot as plt
import struct
import os

class VarioTherm():
    def __init__(self) -> None:
        self.thermo_video_data = None
        self.file_extension = None
        self.file_type = None
        self.file_type_extension = None
        self.sequence_step_data = {}
        self.max_length = None
        self.m_index = 0
```

```
15          self.sequence_offset = 0
16          self.sequence_count = 0
17          self.sequence_count_max = 0
18          self.no_time_seq = 0
19          self.image_data = None
20          self.image_length = 0
21          self.image_index = 0
22          self.image_width = 0
23          self.image_height = 0
24          pass
```

Listing 5.1: VarioTherm class with initial parameters

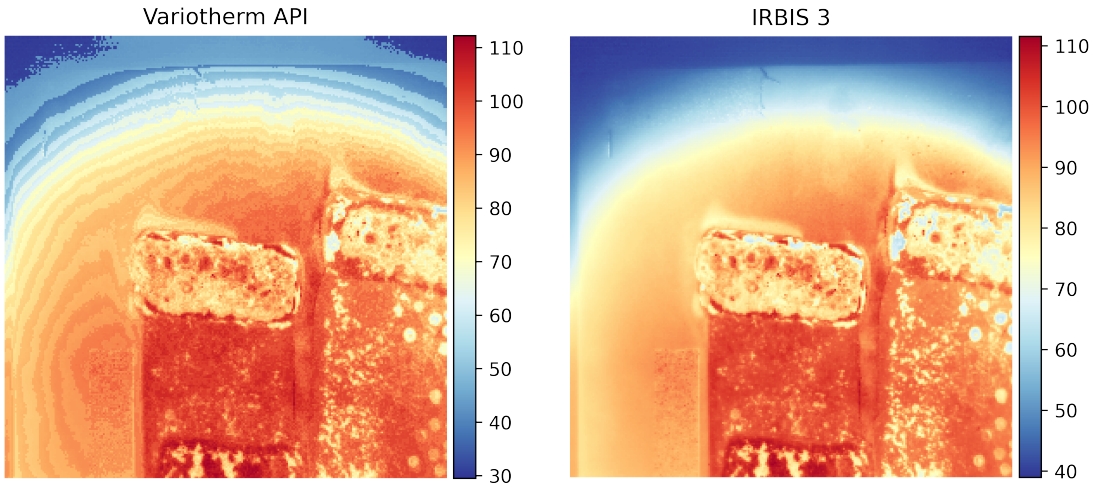


Figure 35: Python API vs IRBIS thermogram with colormap.

Method built in Variotherm class and their functionality

- *data_file_reading* : Reads the .irb video format and converts it into binary data-type.
- *sequence_block_data* : Identifies the images sequence by extracting the offset and length of the thermogram
- *video_info_extraction* : Extracts the video information like device serial number, max and min calibration of the camera and video format.
- *image_extraction* : Converts the binary data of thermogram for each time step . Other information like image shape,time steps, frames, emissivity are extracted.

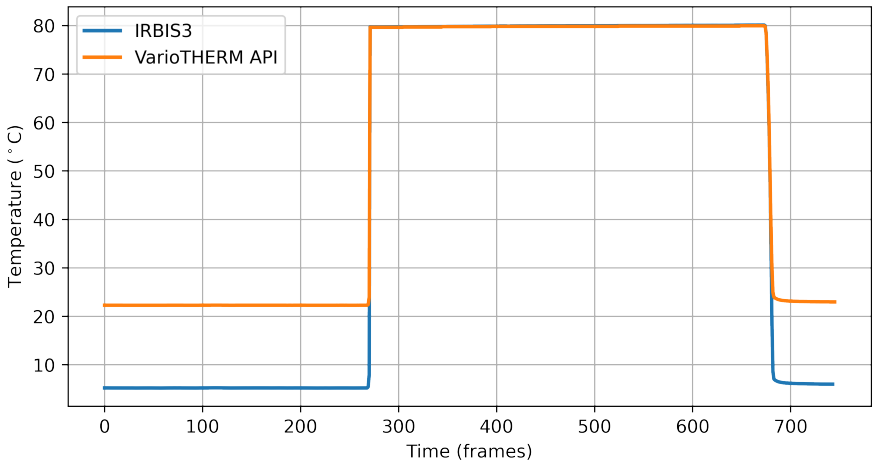


Figure 36: Average temperature of the thermogram for Python API and IRBIS.

- *image_read_byte* : This method takes the raw byte image input and based on the data-type converts it into 'int','float' or 'string'.
- *read_byte* : This method takes the raw byte input and based on the data-type converts it into 'int','float' or 'string'.
- *temperature_interpolation* : After extraction and conversion of image sequence, we obtain two temperature values for each pixel which need to be interpolated to obtain the temperatures in Kelvin.

```
1
2  def temperature_interpolation(self, index, interpolation_temp
       ):
3          no_pixcels = self.image_height * self.image_width
4          temperature_data = []
5          f = 0
6          self.image_index = index
7          for i in range(no_pixcels):
8              pixcel_1 = self.image_read_byte(self.image_index,
       length=1, type='int', byteorder='big')
9              pixcel_2 = self.image_read_byte(self.image_index,
       length=1, type='int', byteorder='big')
10             f = pixcel_1 * (1.0 / 256.0)
11             pixcel_temperature = interpolation_temp[pixcel_2
       +
12                                                     1] * f +
       interpolation_temp[pixcel_2] * (1.0 - f)
13             if pixcel_temperature < 0:
14                 pixcel_temperature = 255.0
15             temperature_data.append(pixcel_temperature)
```

```
16          return np.array(temperature_data)
```

Listing 5.2: VarioTherm interpolation

- *image_sequence_extractor* : This method calls the above methods iteratively and stores the data in an array.

As shown in fig.[36]. The difference in the average temperature can be due to temperature interpolation. $pixcel\_temperature = interpolation\_temp[pixcel\_2 + 1]$ * f + $interpolation\_temp[pixcel\_2]$ * (1.0 - f) where f = $pixcel\_1$ *(1.0 / 256.0) interpolation is used for python API.

5.2.4 Optris

Optris pi 640i is infrared camera which produces thermography in VGA resolution. The cameras are manufactured by Optris infra-red sensing. A detail technical specification are provide in tbl.4.

Table 4: Optris pi 640i Technical Specifications

Technical specifications	
Optical resolution	640 x 480 pixels
Detector	FPA, uncooled (17 µm x 17 µm)
Spectral range	8 – 14 µm
Temperature ranges	–20 ... 100 °C, 0 ... 250 °C, (20) 150 ... 900 °C[1)] optional temperature range: 200 ... 1500 °C
Frame rate	32 Hz / 125 Hz @ 640 x 120 pixels
Optics (FOV)	15° x 11° FOV / f = 41.5 mm or 33° x 25° FOV / f = 18.7 mm or 60° x 45° FOV / f = 10.5 mm or 90° x 66° FOV / f = 7.7 mm
Thermal sensitivity (NETD)	40 mK
Accuracy	±2 °C or ±2 %, whichever is greater
PC interface	USB 2.0 / optional USB GigE (PoE) interface
Process interface (PIF), standard	0 – 10 V input, digital input (max. 24 V), 0 – 10 V output
Process interface (PIF), industrial	2x 0 – 10 V input, digital input (max. 24 V), 3x 0/4 – 20 mA outputs, 3x relay (0 – 30 V/ 400 mA), fail-safe relay
Cable length (USB)	1 m (standard), 5 m, 10 m, 20 m 5 m and 10 m also as high temperature USB cable (180 or 250 °C)
Ambient temperature	0 ... 50 °C
Storage temperature	– 40 ... 85 °C
Relative humidity	20 – 80 %, non-condensing
Enclosure (size / rating)	46 x 56 x 76 – 100 mm (depending on lens + focus position) / IP 67 (NEMA)
Weight	269 - 340 g (depending on lens)
Shock / Vibration[2)]	IEC 60068-2-27 (25G and 50G) / IEC 60068-2-6 (sinus shaped), IEC 60068-2-64 (broadband noise)
Tripod mount	¼-20 UNC
Power supply	via USB

Optris infrared cameras provides .ravi files which are used for thermography analysis. ravi file are radioactive audio video info files. To analyis them we use optris PIX connect software provided by Optris infrared sensing limited.

5.2.5 Optris PIX Connect

Optris PIX connect is an active thermography analysis software use foe optris cameras. The software can extract temperatures from .ravi file and perform thermal profile, heat map of the object. The thermography can be extracted as csv files or as tiff images for further analysis.

The main disadvantage of Optris is the software need a valid licence and the extraction time is slow with huge computational effort. The size of .ravi file are also quite huge.

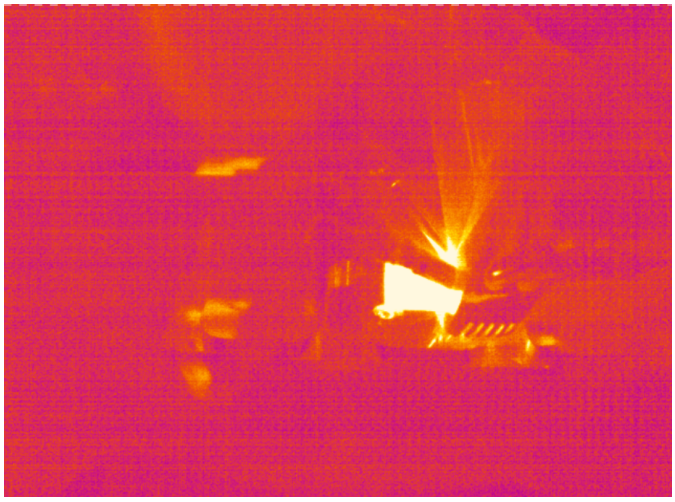


Figure 37: Heat map extracted from Optris PIX connect

5.2.6 Optris Thermogram Extraction

Extraction of Optris file (.ravi files) require the convertion of .ravi to .yuvi file which can be extracted as binary file which contains temperature information of each pixel in raw format. But the .ravi are modified with respect to vendor and thus a universal api cannot be created. For Optris Cameras an SDK is provided by Optris infrared sensing limited for extraction of data in C++.

```
class Optris():
    def __init__(self, root_directory, video_file_name) -> None:
        self.Root_directory = root_directory
        self.Ravi_file_name = video_file_name
        pass

    def create_directory(self, name, cd_disp=False) -> None:
```

```
8         Utilities().create_directory(self.Root_directory, name,
  cd_disp)
9
10     pass
11
12     def video_simulation(self, fps=30, vs_disp=False):
13         video_file_path = os.path.join(self.Root_directory, self.
  Ravi_file_name)
14         ravi_video_data = cv.VideoCapture(video_file_path)
15         ravi_video_data.set(cv.CAP_PROP_FORMAT, -1)
16         ravi_video_data.set(cv.CAP_PROP_FPS, fps)
17         if not ravi_video_data.isOpened():
18             raise Exception('Error while loading the {} video file
  '.format(self.Ravi_file_name))
19         width = int(ravi_video_data.get(cv.CAP_PROP_FRAME_WIDTH))
20         height = int(ravi_video_data.get(cv.CAP_PROP_FRAME_HEIGHT)
  )
21         f = 0
22         while ravi_video_data.isOpened() is True:
23             fetch_status, frame = ravi_video_data.read()
24
25             if fetch_status is False:
26                 print(' playing video is complete')
27                 break
28             re_frame = frame.view(dtype=np.int16).reshape(height,
  width)
29             actual_frame = re_frame[1:, :]
30             displace_frame = cv.normalize(actual_frame, None, 0,
  255, cv.NORM_MINMAX, cv.CV_8U)
31             disp_color = cv.applyColorMap(displace_frame, cv.
  COLORMAP_JET)
32             cv.imshow('Optris RAVI file output', disp_color)
33             if f==950:
34             #print(f)
35                 plt.imshow(displace_frame,cmap='RdYlBu_r',
  interpolation='None')
36                 plt.axis('off')
37                 plt.savefig(r"D:\STUDY_MATERIAL\document\
  optris_python"+str(f)+".png",dpi=600,bbox_inches='tight',
  transparent=True)
38             cv.waitKey(10)
39             print(f)
40             f += 1
41         ravi_video_data.release()
42         cv.destroyAllWindows()
43         pass
```

Listing 5.3: VarioTherm interpolation

We used OpenCV to extract the ravi file and convert into a rgb image. The rgb image is normalized to differentiate the substrate from the surrounding.

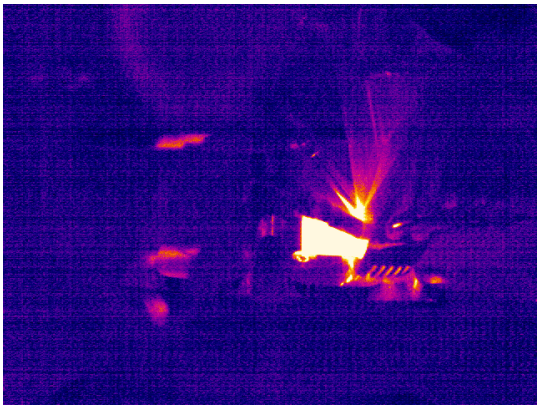


Figure 38: PIX connect

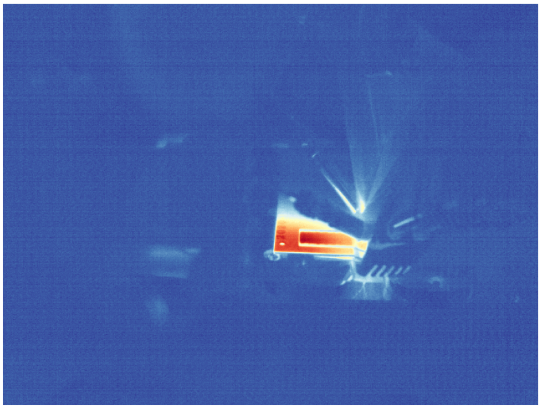


Figure 39: Python API

As shown in fig.39. For coating segmentation, the images are extraction without the temperature profile and based on the brightness contrasts, the coating is segmented.

5.3 Dataset Generation

5.3.1 Phase identification

The thermal image sequence consist of reflection phase and radiation phase. Identification of those phases are important to segregate the data for object segmentation and thickness estimation.

Fourier Transformation

Discrete Fourier Transform (DFT) is used to convert a sequence of signal information into a sine wave over the time domain. From the sine wave, the amplitude and phase change can be identified.[28]

$$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-i2\pi kn/N} = \sum_{n=0}^{N-1} x_n[\cos(2\pi kn/N) - i \cdot \sin(2\pi kn/N)] \qquad (5.3.1)$$

where N = number of samples ,n = current sample ,k = current frequency,x_n = the sine value at sample n,X_k = the DFT which include information of both amplitude

and phase.

The amplitude and phase of the signal can be calculated as

$$amplitude = \frac{|X_k|}{N} = \frac{\sqrt{\mathrm{Re}\,(X_k)^2 + \mathrm{Im}\,(X_k)^2}}{N} \tag{5.3.2}$$

$$\text{phase} \ = \operatorname{atan}2\,(\mathrm{Im}\,(X_k)\,, \mathrm{Re}\,(X_k)) \tag{5.3.3}$$

where $\mathrm{Im}(X_k)$ and $\mathrm{Re}(X_k)$ are the imagery and real part of the complex number.

The change in the phase can be identified using the change in the amplitude. The limitation of DFT is it's inability to accurately convert signal to sine wave for large number of data-points. To overcome this, we use Fast Fourier Transform (FFT).FFT is divide and conquer algorithm that breaks DFT into smaller parts and solves them recursively. FFT reduces the complexity of DFT from O(n2) to O(nlogn) which is a significant reduction in computational time for large n.

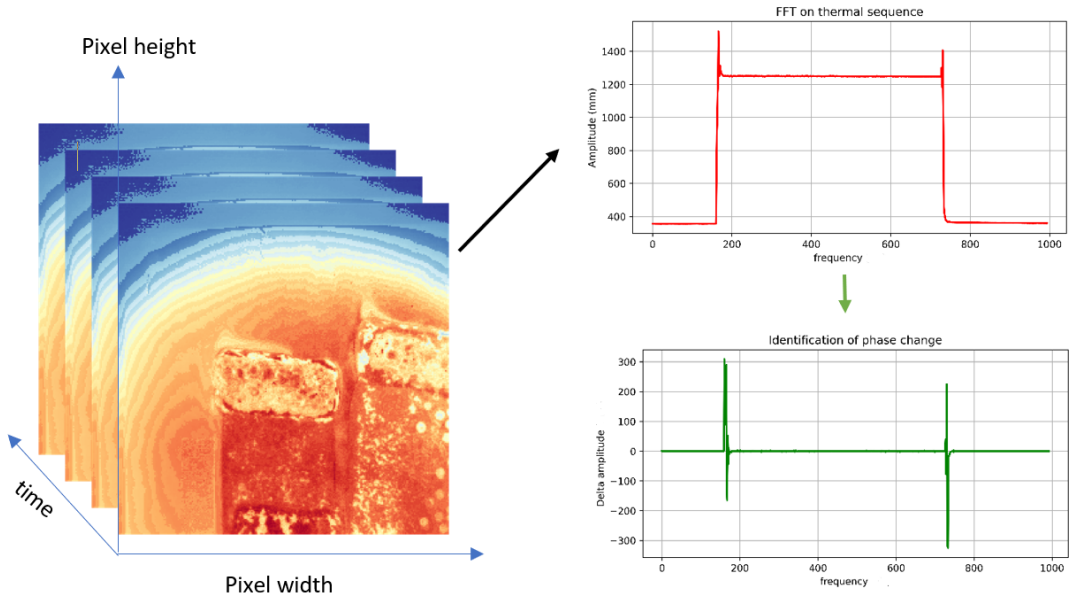


Figure 40: Thermal sequence signal processed using FFT.

As shown in fig.40. The maximum and minimum of the delta amplitude are identified through which the reflective and radiation phases are classified. The fig.[41], the reflection and radiation phases are plotted.

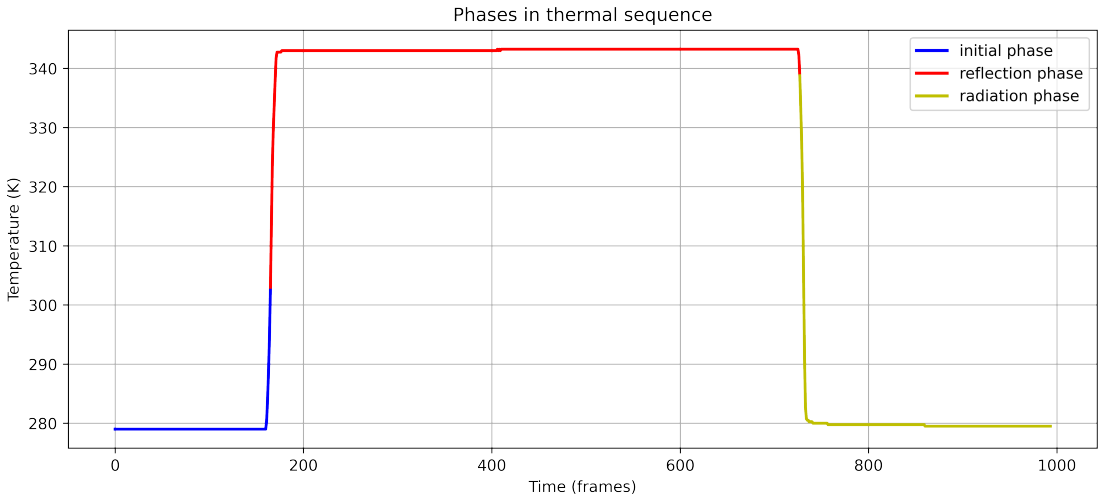


Figure 41: Phase identification in a thermal sequence.

5.3.2 Principle component analysis (PCA)

Principle component analysis is used to identify pattern within a spatial data. PCA is generally used to reduce the number of parameter which is known as dimensional reduction. This is achieved by decomposing the data into a singular values which map the higher dimensional data to newer dimensions.[29]

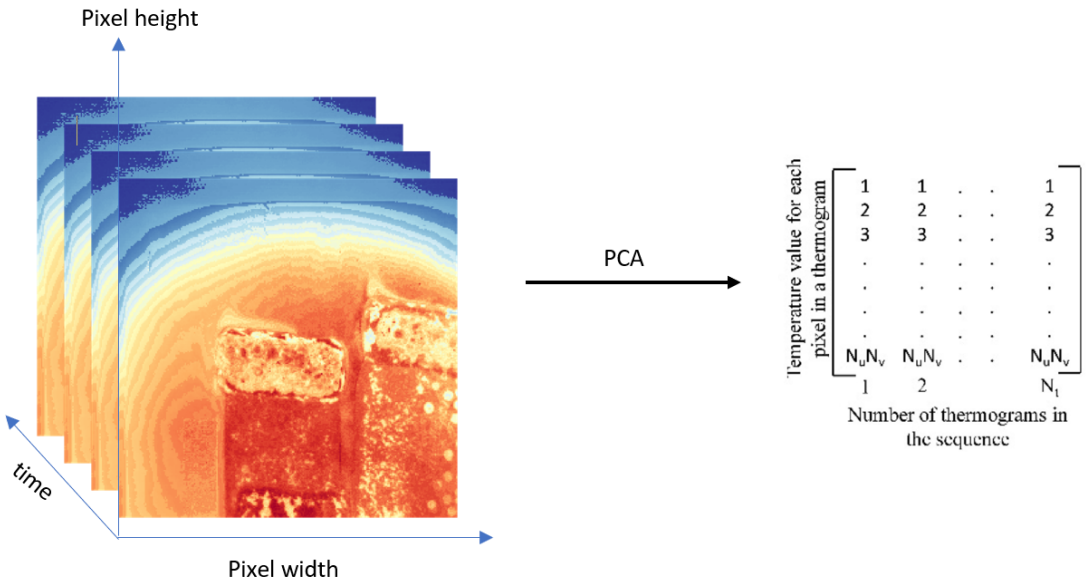


Figure 42: Thermal sequence processed using PCA.

For thermography, the thermal sequence is a 3D array consisting of N_t thermograms, each thermogram consists of u horizontal elements (pixels)N_u and v vertical elements (pixels)N_v.

PCA is applied to thermogram sequence by converting 3D array to 2D array. First, the covariance of the converted 2D array is calculated [29]

$$\mathrm{Cov} = \frac{1}{N_t} \mathrm{X}\mathrm{X}^{\mathrm{T}} \tag{5.3.4}$$

where X is the thermogram sequence rearranged as a 2D array.

Eigenvector of the covariance are calculated and stored in matrix P.

$$\mathrm{Cov}_D = \mathrm{P}^{-1}\,\mathrm{Cov}\,\mathrm{P}_1 \tag{5.3.5}$$

The Cov_D is a diagonal matrix which consist of eigen values as it diagonal element. The above operation is known is Single Value Decomposition (SVD). SVD is applied to compute decomposed matrices U,G and V. [29]

$$\mathbf{A} = \mathbf{U}\mathbf{G}V^{\mathrm{T}} \tag{5.3.6}$$

where A is the 2D initial matrix (which is rearrangement of temporal data (3D)), G is a diagonal matrix consisting of eigenvalues. Principal components, which represent time variations, have been rearranged into rows of matrix V_T.

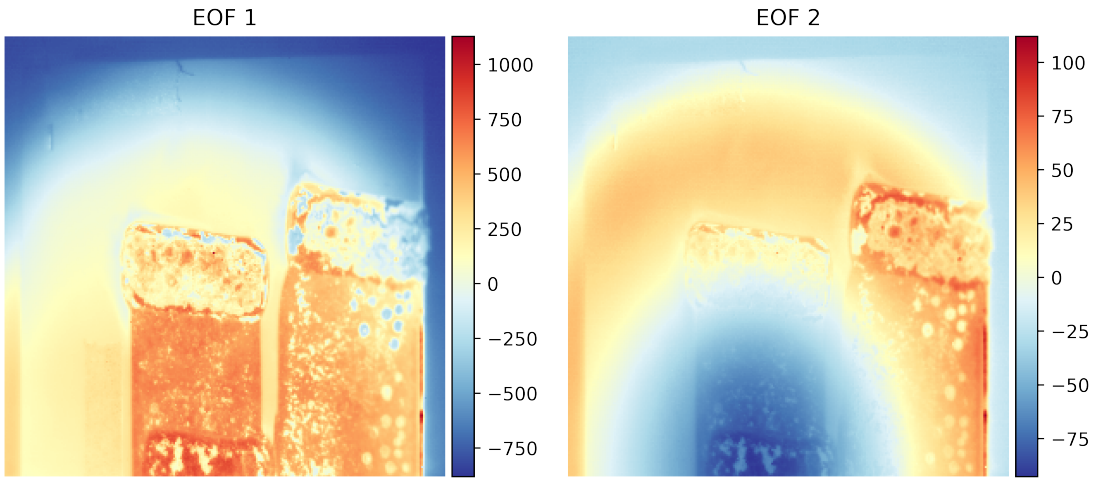


Figure 43: Empirical Orthogonal Functions (EOF) of the thermogram sequence using PCA.

The principal components and empirical orthogonal functions (EOF) acquired by using PCA have all non-zero matrix elements. If EOF images are depicted,

the first EOF (EOF1) corresponds to the first characteristic data variability, the second EOF (EOF2) corresponds to the second characteristic variability, etc [29]

5.3.3 Object Segmentation Masks (Output data)

LabelMe

For object segmentation, we require output data based on which the predicted mask is compared. To create the mask, we have used a annotation tool known as LabelMe. As shown in fig.44

In labelMe, we have used points to select the boundaries of a feature and convert them into a json file which contain the pixel location. The json files are imported and sci-kit learn module is used to map the points to get a polygon structure while assigning all the data points inside the polygon structure with the respective annotation value. The annotation of very small feature was done using OpenCV.

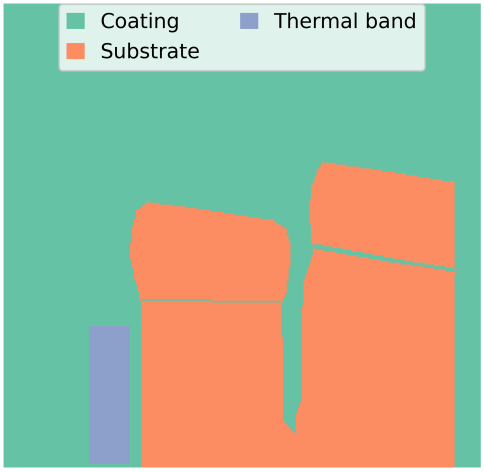


Figure 44: Object segmented manually using labelMe tool

OpenCV

OpenCV is a computer vision module
in python which deals with real time processing of images and videos. The data from PCA and labelMe are used to extract the other features in thermogram. we have morphological operation are used extract the required features.

The fig.[45] shown a manual mask built using LabelMe tool, the second image is the first EOF obtained after PCA. Bitwise AND is used to extract the coating. Bitwise OR is used to extract substrate without any damages. Bitwise NOR is used to extract the damaged substrate.

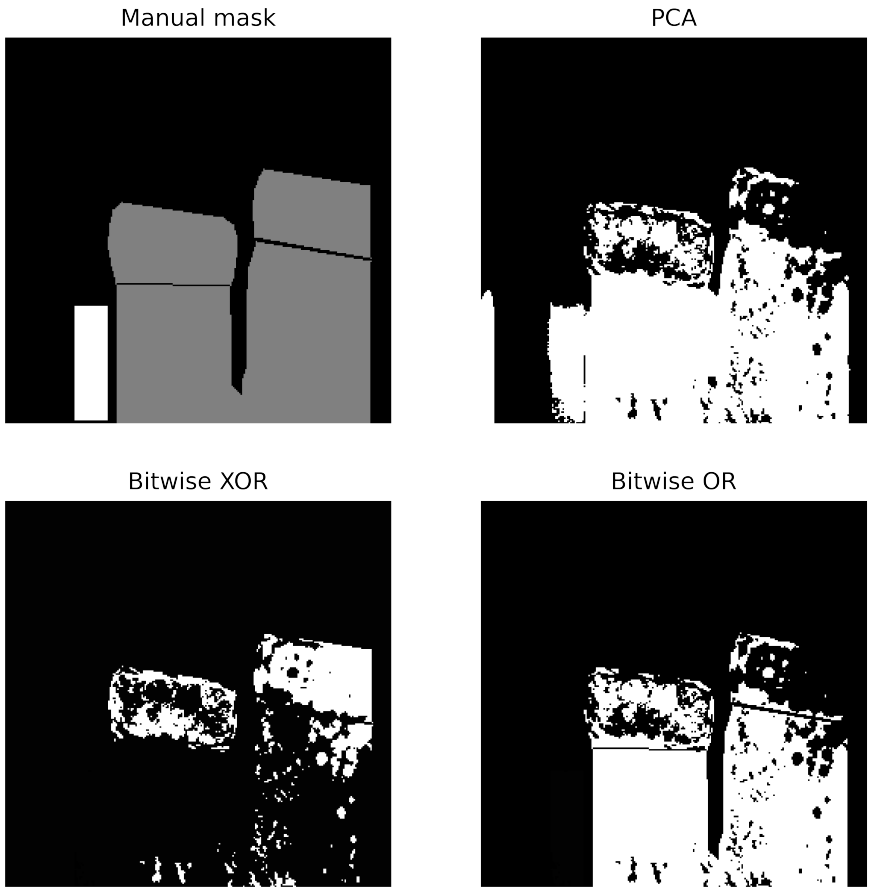


Figure 45: Feature Extraction for annotation using OpenCV.

5.3.4 Datasets

For object segmentation, the dataset consist of input thermogram and output segmentation mask. Only reflection phase data is used for object segmentation. Radiation phase data is used for thickness estimation. The dataset is divided into three part as given below.

Training Dataset

A training dataset is the primary data on which the model is trained and consist of large number of sample. The main goal is to train the model to generalizes well to new unknown datasets.

Due to the large number of parameters within the neural network. The network

can overfit on training data and becomes bias. To avoid this a validation and testing dataset are created from the initial training dataset.

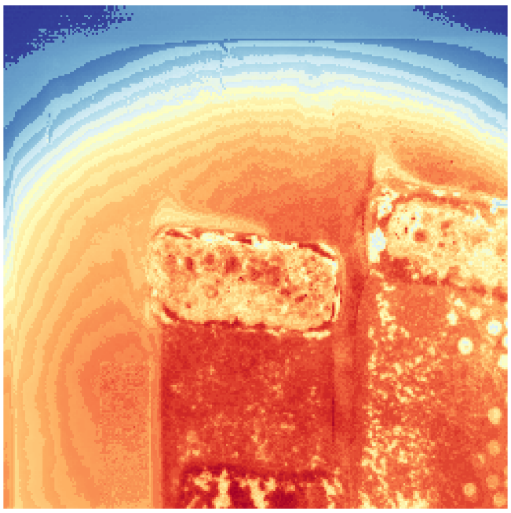


Figure 46: Input thermogram

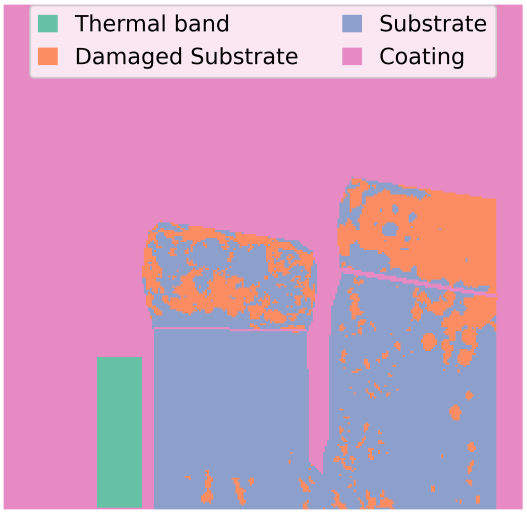


Figure 47: Output segmentation

For thickness estimation, the object is cropped and the average temperature is calculated.

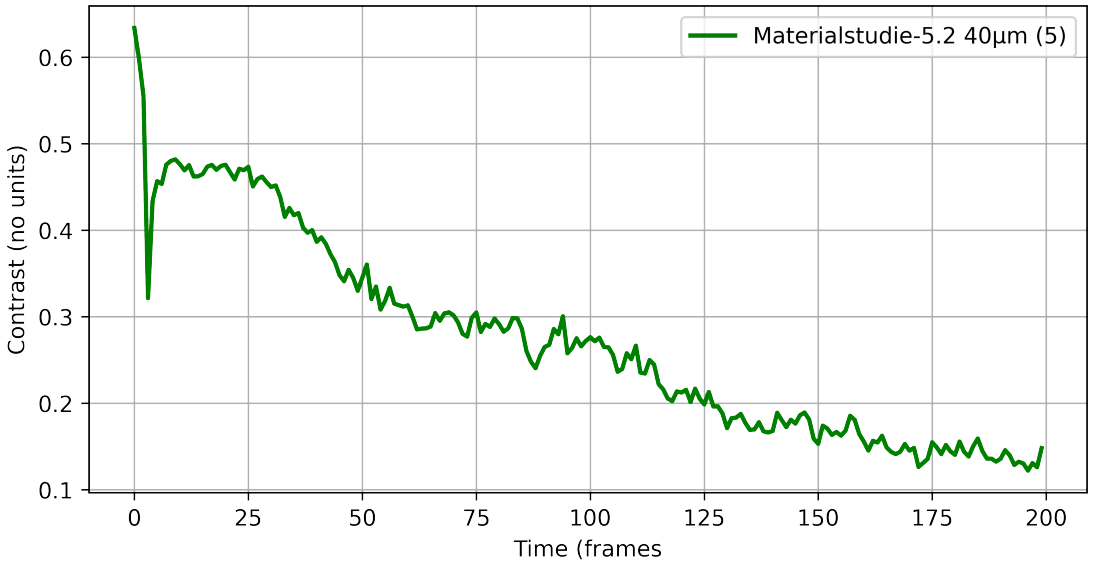


Figure 48: Radiation phase after scaling the thermogram using contrast function.

Validation Dataset

A neural network consist of large number of parameters and some hyperparameter like learning rate, number of hidden units, activation function etc. A validation dataset is used to validate the trained model and make necessary adjustment to achieve a better accuracy. In general, the part of training dataset is used as validation dataset. Validation dataset doesn't take part in the training, it is used only for the evaluation of the model. Training and validation learning curve help us evaluate the model performance if there is bias or variance in the neural network.

Test Dataset

The performance of the model is evaluated using the test dataset. Test dataset can be part of training dataset but test dataset consist of example which the training dataset doesn't contain. The evaluation of model on test dataset help us to check if the neural network is able to generalize on the new unknown data.

5.3.5 Scaling and Normalizing of data

Neural network need a huge computational effort. The gradient descent is used to update the parameters of the model iteratively until the difference between the expected and predicted output is minimum.

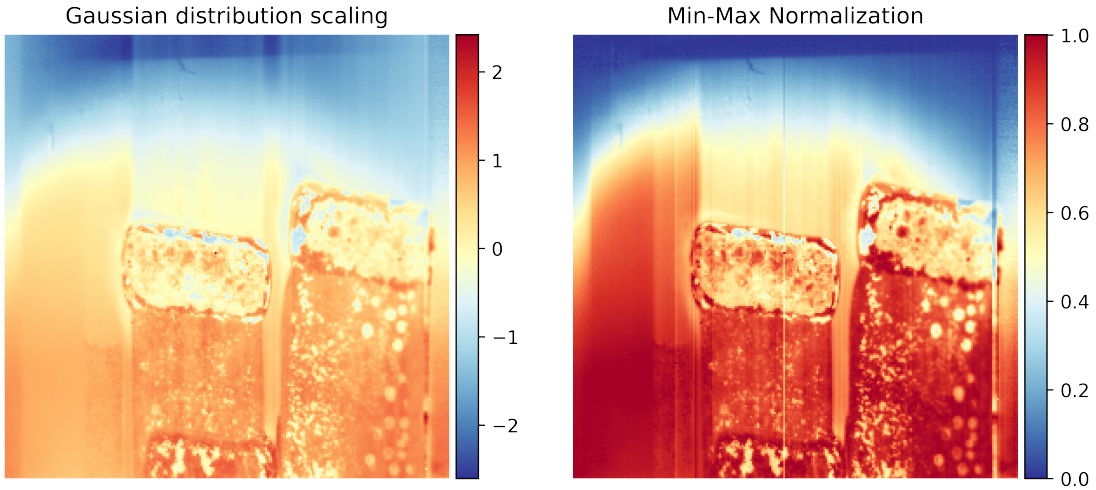


Figure 49: Scaling and normalization of thermogram.

To improve the overall performance of the neural network normalization of input

data to standard scale is performed. Neural network loss their ability to learn if the values are to high or low as they create gradient exploding and vanishing.

Based on the type of normalizing function, the typical range varies.

- Min-Max scaling : The typical range is -1 to 1 or 0 to 1.
- Standard scaling : Normalizes on Gaussian distribution.

5.3.6 Data Augmentation

Data augmentation is a set of techniques used to artificially increase the amount of data by generating new data points from existing data. This includes making small changes to data or using deep learning models to generate new data points.[30]

Data augmentation generates dataset which are rich and sufficient for the model to perform accurately. Data augmentation techniques enable machine learning models to train their parameter to the variations it may see in the real world.

Few data augmentation techniques are padding, random rotating, re-scaling, vertical and horizontal ,flipping, translation (image is moved along X, Y direction) ,cropping,zooming,color modification

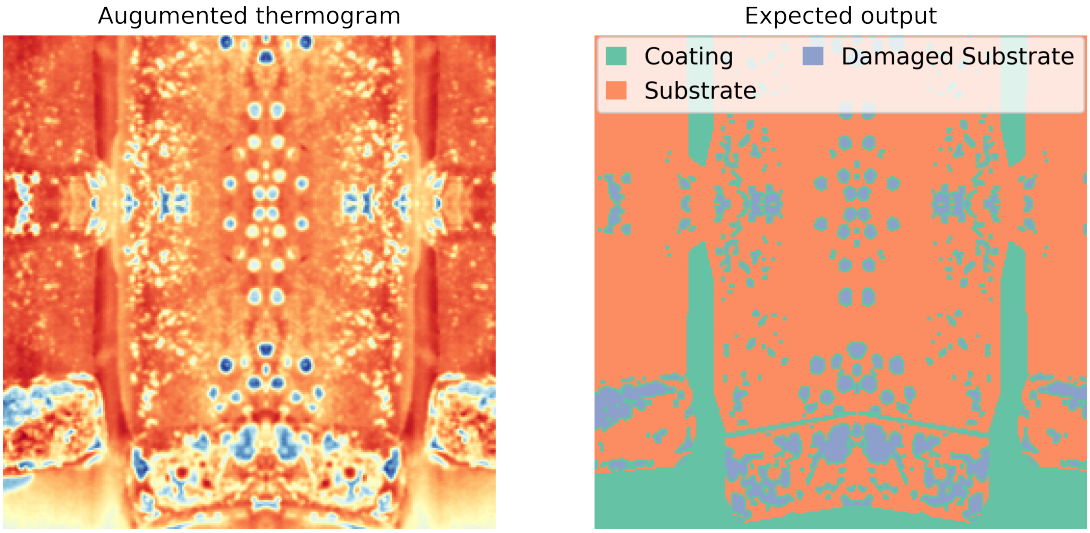


Figure 50: Augumentation of thermogram.

Chapter 6

NN: Strategies and Inferences

6.1 Neural Network parameters

Neural network consist of large number of parameter which have to be selected based on architecture, output data-type, size of the model and type of algorithms used in the model.

6.1.1 Loss Function

Most important parameter in neural network is the loss function which depend on the output type.

Object segmentation and thickness estimation: We use categorical cross-entropy loss function.

$$L = -\frac{1}{m}\sum_{i=1}^{m}\left(y_i \cdot \log\left(\hat{y}_i\right) + (1-y_i)\cdot \log\left(1-\hat{y}_i\right)\right) \tag{6.1.1}$$

Where y_i is the actual value and $\hat{y}_i$ is the predicted value. m is the number of sample (batch size).

Depth estimation: The values to be estimated are numeric values. A mean squared error (MSE) loss function is used for training this model.

$$\text{MSE} = \frac{1}{m}\sum_{i=1}^{m}\left(y_i - \hat{y}_i\right)^2 \tag{6.1.2}$$

Where y_i is the actual value and $\hat{y}_i$ is the predicted value. m is the number of sample (batch size).

6.1.2 Evaluation Metrics

The performance of neural network needs t be accessed to optimize the parameter and hyper-parameters. Evaluation metric does not take part in training but are used as performance indicator. Evaluation metrics are calculated during forward propagation.

$$\text{Accuracy } = \frac{TP + TN}{TP + FN + FP + TN} \tag{6.1.3}$$

where TP, TN, FP, and FN are true positive, true negative, false positive, and false negative. A threshold has to be set to obtain a confusion matrix and to be able to compute the evaluation metrics.

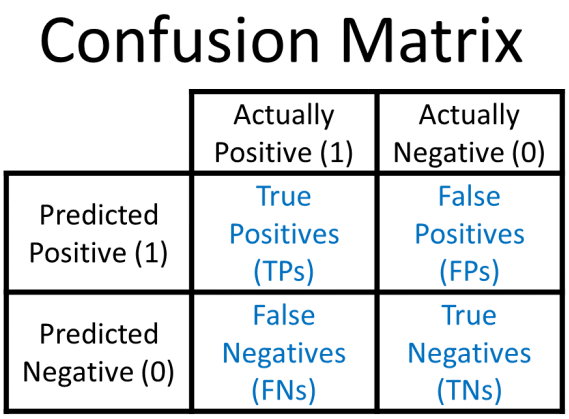


Figure 51: Confusion Matrix.

6.1.3 Optimizers

During training, gradient descent is required to update the parameters during back-propagation. Optimizers are algorithms which minimizes the loss function using gradient descent method.

Gradient descent algorithm finds the minima of a function by calculating the gradients and move iteratively in the direction of steepest descent(negative slope).

Few optimizers:

- **Gradient descent**-Batch gradient descent calculates the gradients for the entire training dataset which is computationally expensive.[31]

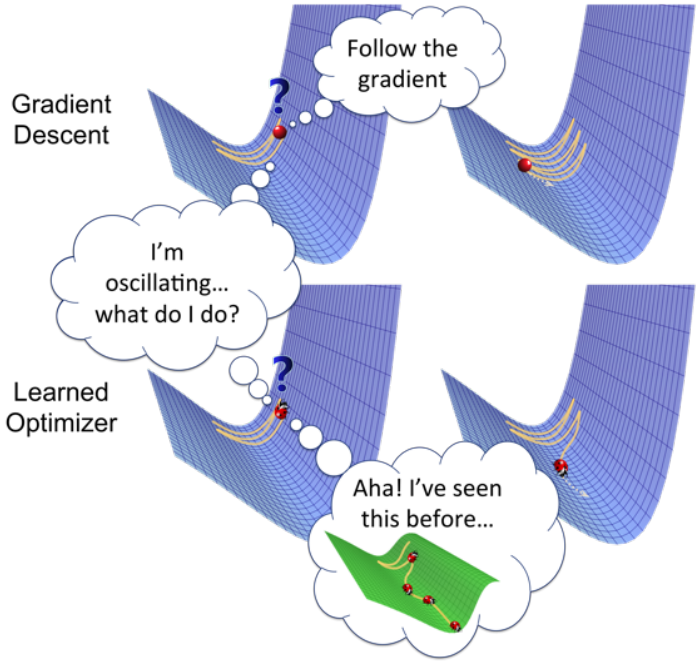


Figure 52: General theme of Gradient based Optimizers.[16]

- **SGD**-Stochastic gradient descent optimizer updates the parameters for each training example. It eliminates the method of computing gradients of the entire data in every epoch like batch gradient descent.[31]
- **RMSProp**-Root Mean Squared Propagation is a gradient-based optimizer, applies the exponential moving average of the squared gradients to adjust the learning rate.[31]
- **Adam**-Adaptive Moment Estimation is a gradient descent-based optimizer combined with the advantages of RMSProp and Adagrad. The method computes the adaptive learning rate for each parameter and applies bias-correction.[31]

6.2 Object Segmentation

The process of determining the pattern within image by identifying the boundaries and area of the pattern is known as object segmentation. Object segmentation can be classified as

- Semantic Segmentation :In this process the respective labels are assigned

to each pixel in the image. Semantic segmentation cannot identify multiple object, all object of same class are assigned the same label.

- Instant Segmentation : In this process, The multiple instances of the object of same classes are separated as different objects of same class.

For this project, we would perform semantic segmentation. The following network are built, trained and experimented.

1. Pyramid Scene Parsing Network (PSP Net)
2. Fully Convolution Network (FCN)
3. U-Net

6.2.1 Pyramid Scene Parsing Network (PSP Net)

Scene parsing is a method used for image segmentation. In this method, the images are segmented and parsed into different regions.

Network Architecture

Parsing refers to the transformation of data into a parse tree structure which is easier to read, understand or identify pattern.

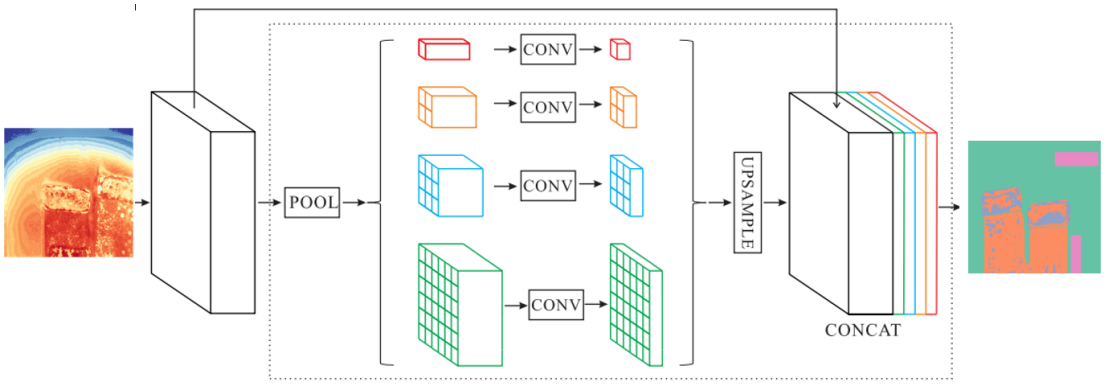


Figure 53: PSP architecture.[17]

In PSP-Net, the feature from images extracted are divided into four different pyramid scales. Based on the size of the pyramid scale, the features are identified

from coarsest to finest. The size of these pyramid modules can be modified. The size of the pyramid scale are modified by changing the kernel size, stride and pooling parameters in convolution block.

In general, the size of these pyramid modules depend on the size of the input image. The main advantage of this method is each pyramid module extract different features from the image which are concatenated with the up-sampled feature vector. This bi-linear interpolation of the pyramid feature with the original images extract low dimensional features. The number of trainable parameter in the network are less compared to other CNN networks.

Table 5: Number of parameters in PSP-Net

Total parameters	2,11,268
Trainable parameters	2,09,916
Non Trainable parameters	1352

Our pyramid pooling module is a four-level one with the kernel sizes of 1×1, 2×2, 3×3 and 6×6 respectively.

Training and Experimentation

The number of parameter to train depends on the number of hidden layers and size of each hidden layer. Training a neural network can be optimized by selecting the appropriate hyper-parameters. The process of optimizing multiple local optima along with global optima is known as non-convex optimization

Challenges in optimizing a Neural Network

1. Learning Rate- If it is too small, it takes too long to converge and if too large, it may never converge.
2. Batch size - If the batch size is small or very large, the parameter don't get updated during training which leads to under fitting.
3. Gradient descent optimizer - Parameters are updated during back-propagation by gradient descent method and selection of an effective gradient descent optimizer would speed up the training process.

Learning rate The learning rate is identified using a learning rate scheduler which reduces the learning rate iteratively based on the number of epoch and the slope were there is steep descent is taken as learning rate.As shown in fig.54

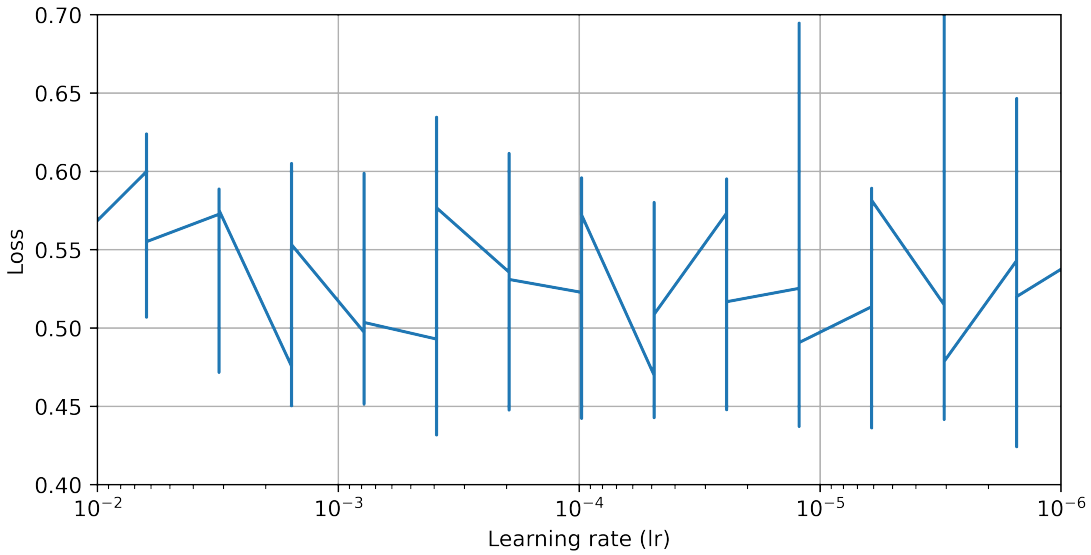


Figure 54: Learning curve for lr scheduler (PSP-Net).

The learning rate is selected based on the above fig.54. The loss should not explode when learning rate changes (which shows the instability)

Batch size The learning curve for different batch sizes are plotted and compared and the learning curve with least fluctuation and minimum loss is selected. As shown in fig.**??** and fig.56

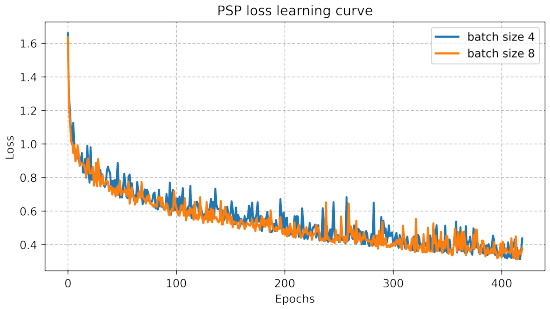


Figure 55: Loss curve batch size (PSP-Net)

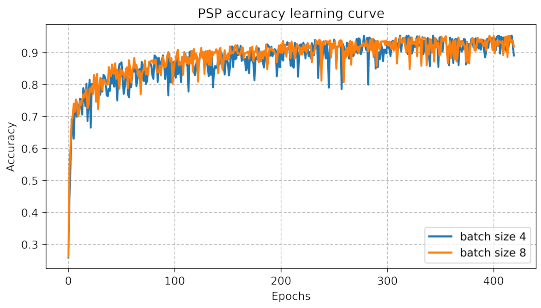


Figure 56: Accuracy curve batch size (PSP-Net)

Optimizers The learning curve for different optimizers are plotted and compared and the learning curve with minimum loss and maximum accuracy is selected. As shown in fig.57 and fig.58

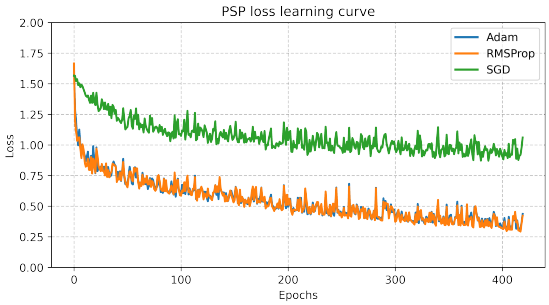


Figure 57: Loss curve optimizer (PSP-Net)

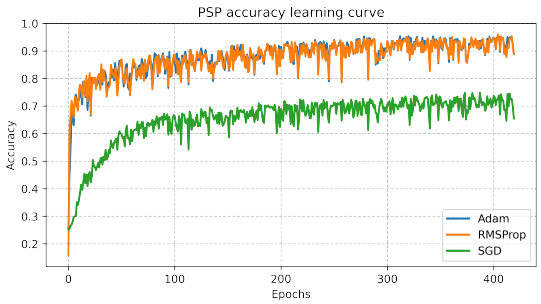


Figure 58: Accuracy curve optimizer (PSP-Net)

Final model parameters and Training

The final parameters are selected based on the above experimentation and the parameters which have high accuracy and low loss are selected.

1. Learning Rate- 0.0001
2. Batch size - 8
3. Gradient descent optimizer - RMSprop

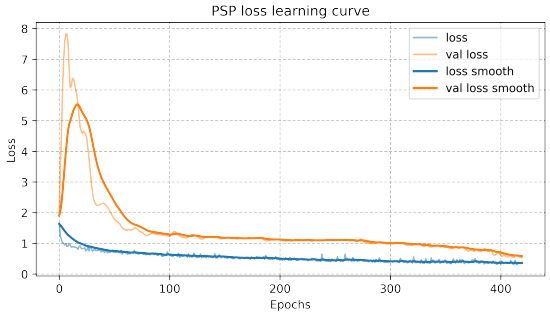


Figure 59: Loss curve optimizer for final hyperparameters (PSP-Net)

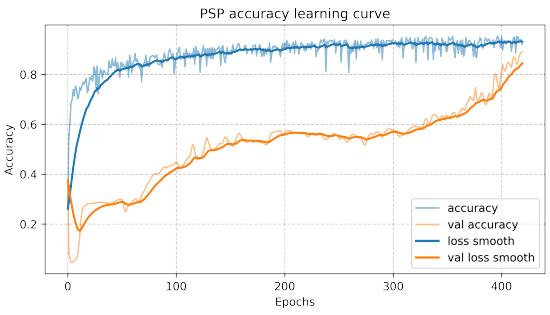


Figure 60: Accuracy curve for final hyperparameters (PSP-Net)

6.2.2 Fully Convolution Network (FCN)

FCN is a convolution based neural network which is used for segmentation. In FCN, there are no dense layers. It consist of up-sampling and down-sampling.

Network Architecture

There are 2 variants of the FCN architecture: FCN-32 and FCN-8.

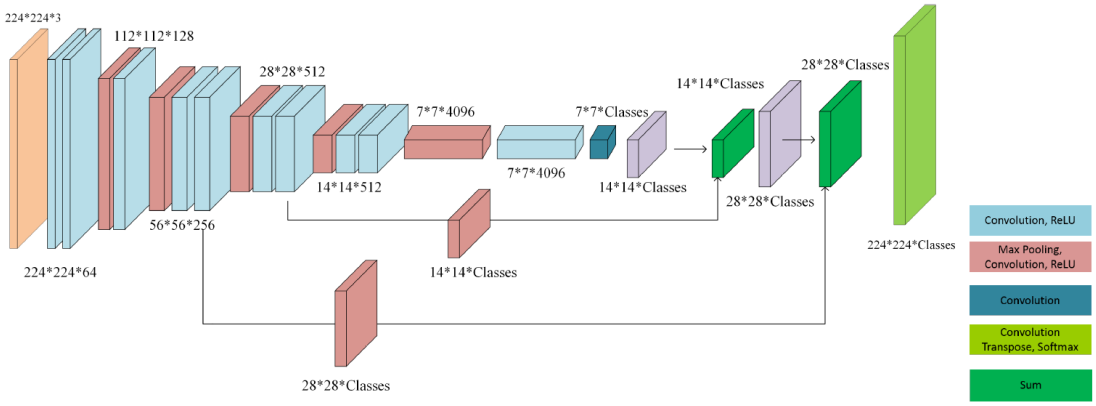


Figure 61: FCN architecture.[18]

In FCN, the convolution operation is coupled with pooling and batch normalization. The main feature of this network is the feature at intermediate down-sampling layer are extracted and added during the up sampling to predict the mask. IN FCN-32 is used, then features are mapped to the up-sampling only at a single point resulting in coarse mask. For our project, we built FCN-8, where 3 intermediate maps were extracted and added during up-sampling to generate finer mask.

Table 6: Number of parameters in FCN-8

Total parameters	14,751,956
Trainable parameters	14,751,956
Non Trainable parameters	-

Training and Experimentation

The number of parameter to train depends on the number of hidden layers and size of each hidden layer. Training a neural network can be optimized by selecting the appropriate hyper-parameters. The process of optimizing multiple local optima along with global optima is known as non-convex optimization.

Challenges in optimizing a Neural Network

Learning rate The learning rate is identified using a learning rate scheduler which reduces the learning rate iteratively based on the number of epoch and the slope were there is steep descent is taken as learning rate.As shown in fig.**??**

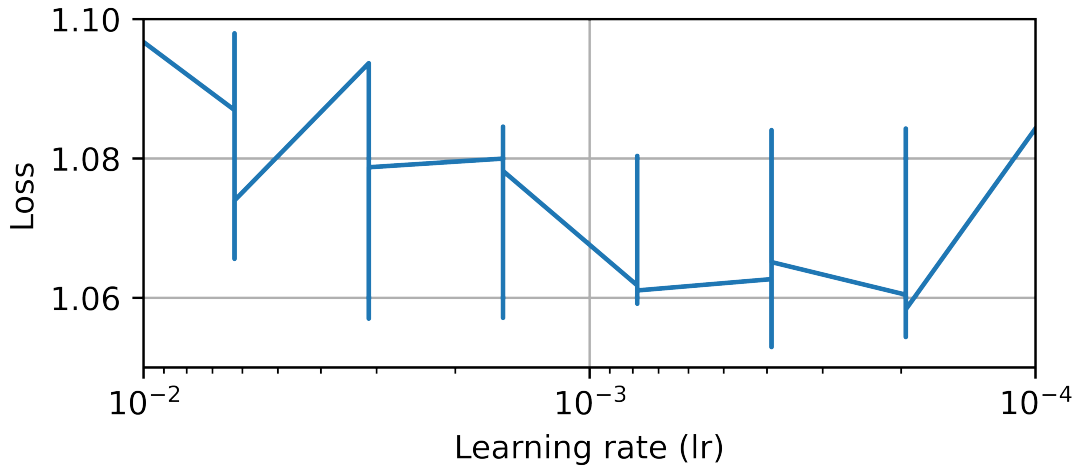


Figure 62: Learning curve for lr scheduler (FCN).

The learning rate is selected based on the above fig.62. The loss should not explode when learning rate changes (which shows the instability)

Batch size The learning curve for different batch sizes are plotted and compared and the learning curve with least fluctuation and minimum loss is selected. As shown in fig.63 and fig.64

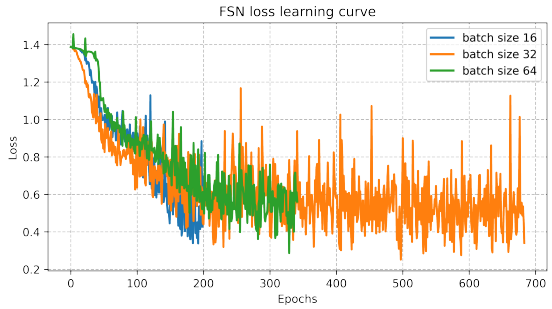


Figure 63: Loss curve batch size (FCN)

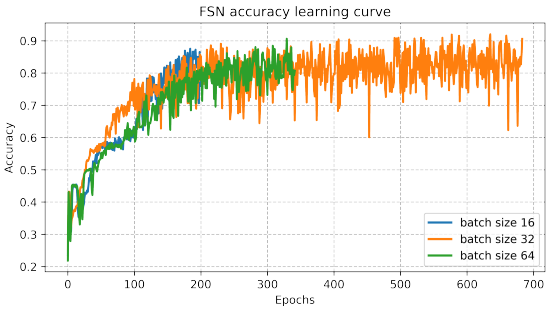


Figure 64: Accuracy curve batch size (FCN)

Optimizers The learning curve for different optimizers are plotted and compared and the learning curve with minimum loss and maximum accuracy is selected. As shown in fig.65 and fig.66

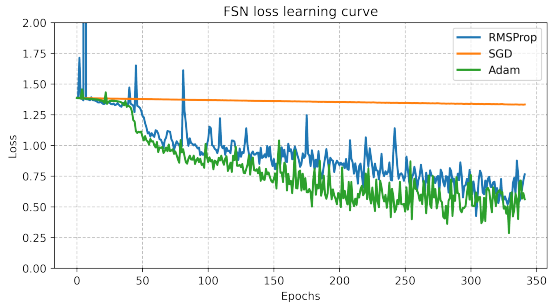


Figure 65: Loss curve optimizer (FCN)

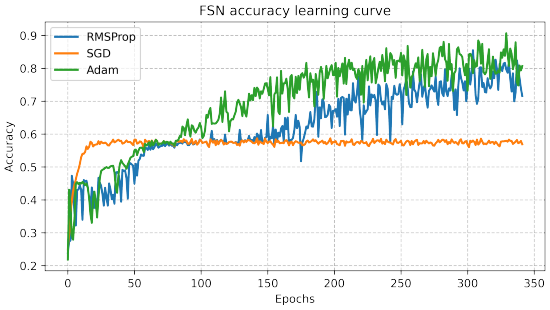


Figure 66: Accuracy curve optimizer (FCN)

Final model parameters and Training

The final parameters are selected based on the above experimentation and the parameters which have high accuracy and low loss are selected.

1. Learning Rate- 0.0001
2. Batch size - 64
3. Gradient descent optimizer - ADAM

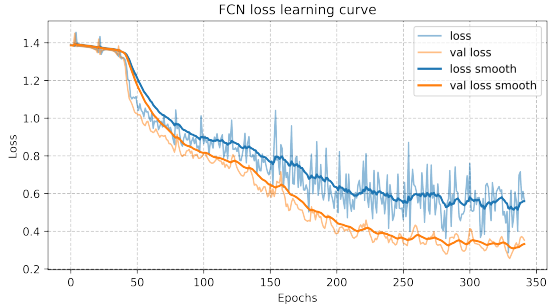


Figure 67: Loss curve for final hyperparameters (FCN)

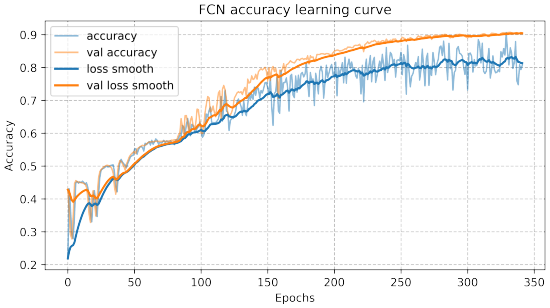


Figure 68: Accuracy curve for final hyperparameters (FCN)

6.2.3 U-Net

U-Net was developed in Computer Science department by University of Freiburg. The network were used for biomedical image segmentation yielding more precise segmentation for fewer training images.

Network Architecture

U-nets have the ability to extract spatial feature independent of it's position. U-Net architecture is consist of an encoder and decoder.

Encoder In encoder, the feature map is spatial dimensionality is reduced throughout the network path. A encoder network consist of encoder blocks which consist of convolution layer, dropout layer, activation function and pooling layer. The feature vector at each block are stored for concatenation during decoding. Due to the decrease in the feature vector dimensions, only the higher values are stored which identify important feature in from the input.

Decoder In decoder, the feature map is spatial dimensionality is increased throughout the network path. A decoder network consist of decoder blocks which consist of convolution layer, dropout layer, activation function and up-sampling layer. The feature vector at each block is concatenated with feature vector from encoder which enables a precise assemble of output.

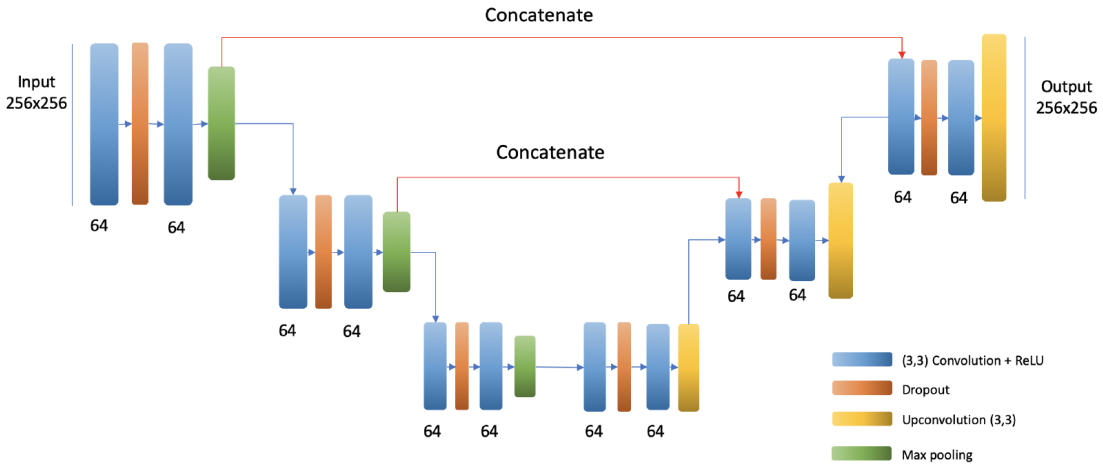


Figure 69: UNet architecture.[19]

The U-Net has a bottle-Neck in it's structure. As the size of the feature vector decreases in encoder. It losses most of the spatial information. Thus, the size of the filters are adjusted to achieve better performance.

Table 7: Number of parameters in U-Net

Total parameters	4,629,396
Trainable parameters	4,629,396
Non Trainable parameters	0

Training and Experimentation

The number of parameter to train depends on the number of hidden layers and size of each hidden layer. Training a neural network can be optimized by selecting the appropriate hyper-parameters. The process of optimizing multiple local optima along with global optima is known as non-convex optimization Challenges in optimizing a Neural Network

Learning rate The learning rate is identified using a learning rate scheduler which reduces the learning rate iteratively based on the number of epoch and the slope were there is steep descent is taken as learning rate.As shown in fig.**??**

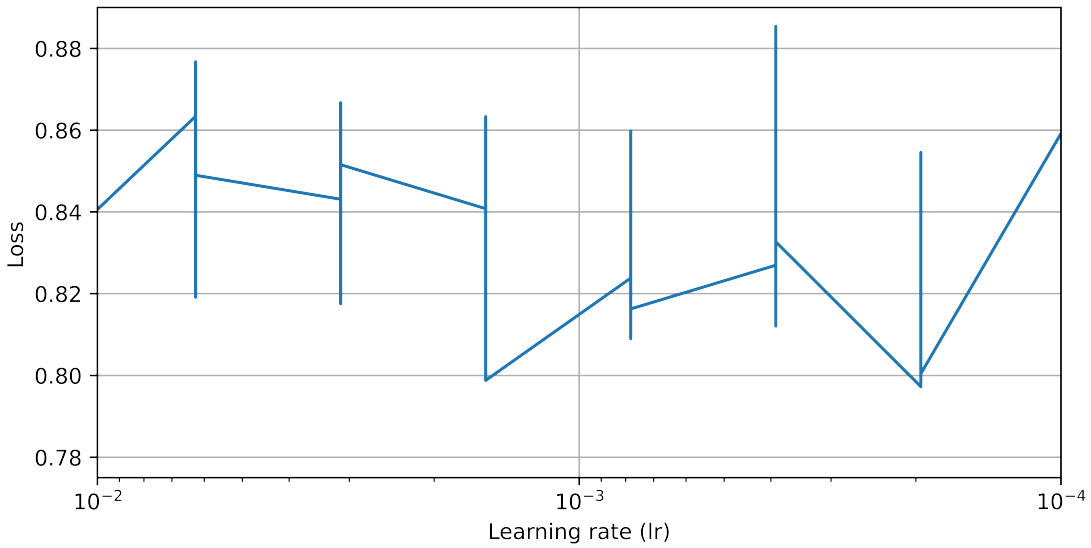


Figure 70: Learning curve for lr scheduler (U-Net).

Batch size The learning curve for different batch sizes are plotted and compared and the learning curve with least fluctuation and minimum loss is selected. As shown in fig.71 and fig.72

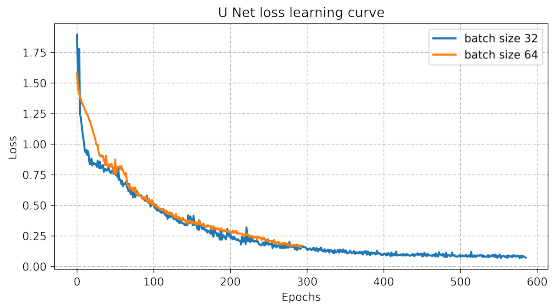


Figure 71: Loss curve batch size (UNet)

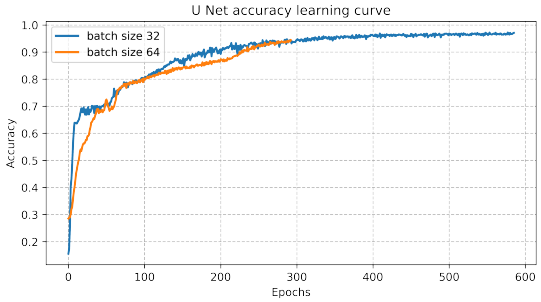


Figure 72: Accuracy curve batch size (UNet)

Optimizers The learning curve for different optimizers are plotted and compared and the learning curve with minimum loss and maximum accuracy is selected. As shown in fig.73 and fig.74

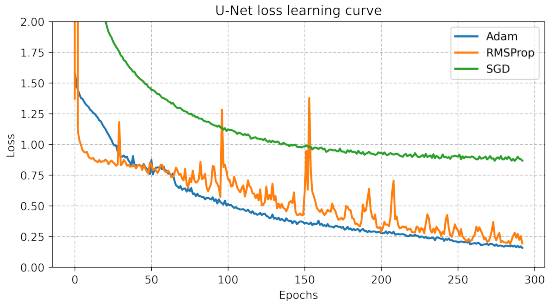


Figure 73: Loss curve filter (UNet)

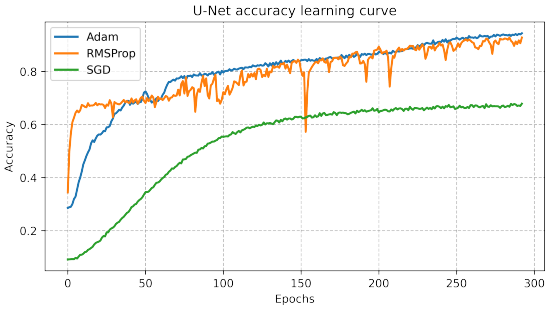


Figure 74: Accuracy curve filter (UNet)

Filter size The learning curve for different filter sizes are plotted and compared and the learning curve with least fluctuation and minimum loss is selected. As shown in fig.75 and fig.76

Final model parameters and Training

The final parameters are selected based on the above experimentation and the parameters which have high accuracy and low loss are selected.

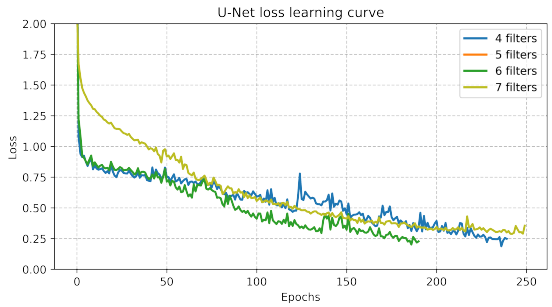


Figure 75: Loss curve filter size (UNet)

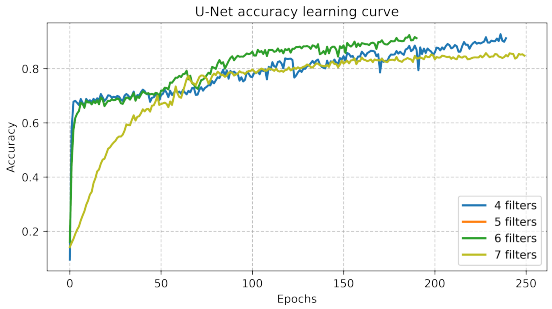


Figure 76: Accuracy curve filter size (UNet)

1. Learning Rate- 0.0001
2. Batch size - 64
3. Gradient descent optimizer - ADAM
4. Filter sizes - 6 [16,32,64,128,256,512]

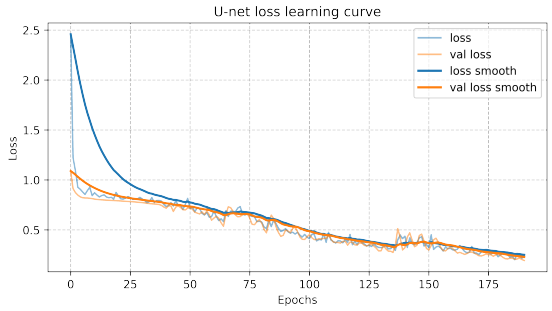


Figure 77: Loss curve for final hyperparameters(UNet)

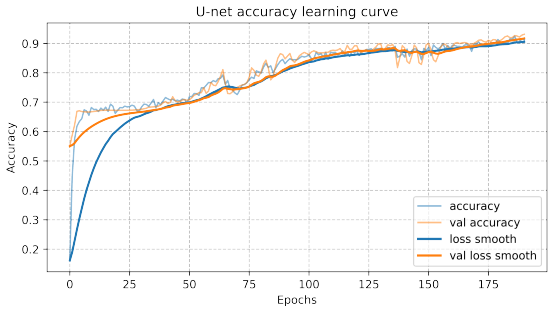


Figure 78: Accuracy curve for final hyperparameters(UNet)

6.3 Thickness Estimation

The process of estimating the thickness of the coating is a complex operation and depends on external factors like initial temperature and duration of heat source (reflection phase) and type of contrast function.

6.3.1 Recurrent Neural Networks Architecture

The ability of RNN to store data from previous time step allows us to make prediction of coating thickness. To estimate the thickness of an object, the radiation phase of the object is analysed. The variation in radiation of different coating is used to estimate their respective thickness. The architecture of the LSTM and GRU recurrent Neural Networks are discussed below.

LSTM : Long Short Term Memory

LSTM networks are designed to solve the vanishing gradient problem which occurs due to the loss of memory from previous states. This problem is rectified using a characteristic memory unit built within each neuron which can store the memory. The flow of information is regulated with the help of gates, Input gate- can restrict the flow of input data of current state, Forget gate - can restrict the memory to be stored from previous states. The magnitude of gradients are also learned from the parameters during training based on these gates. A unit LSTM is illustrated in fig.79

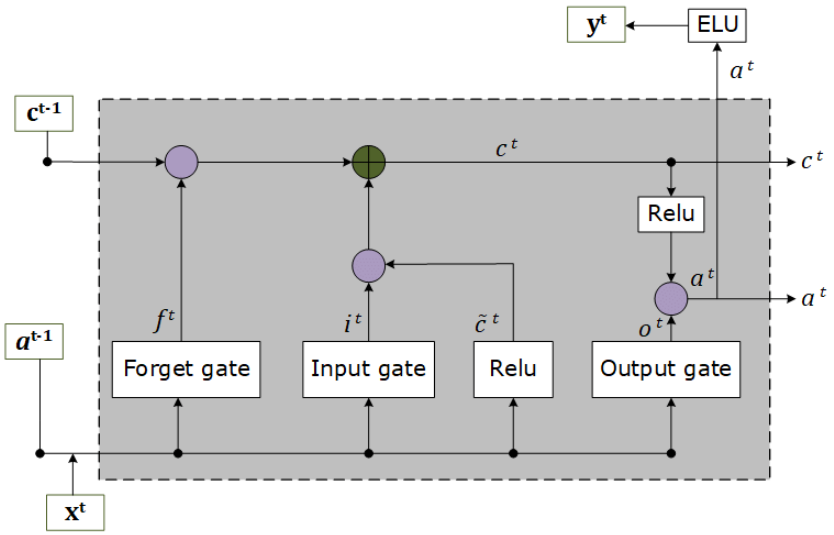


Figure 79: LSTM Architecture.[20]

GRU : Gated Recurrent Unit

GRU consist of update gate and reset gate. These two gate decide the information flow in the neural network. The main advantage of these gates is that they can

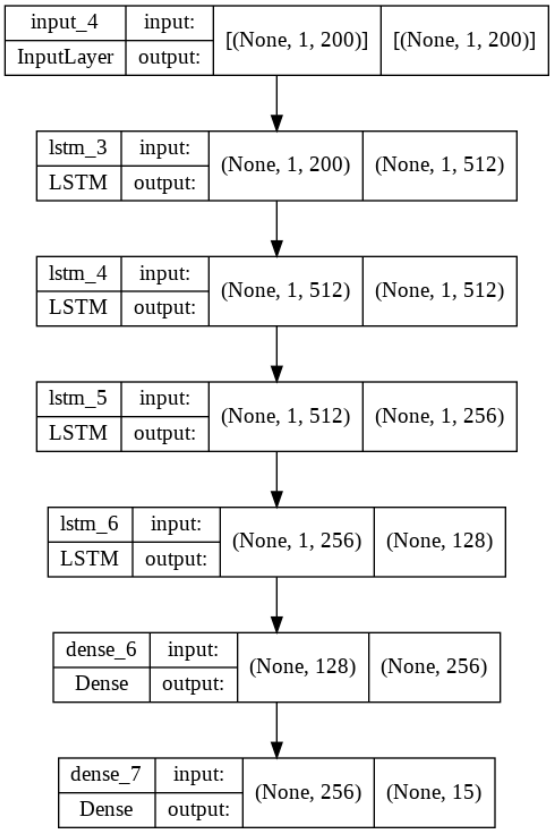


Figure 80: Long short term memory architecture (LSTM)

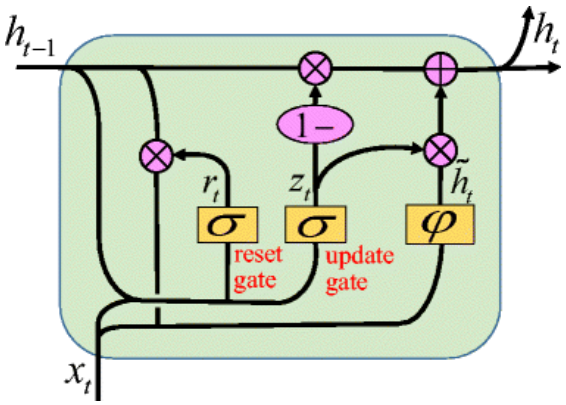


Figure 81: GRU Architecture.[20]

be trained to store information for longer layers which are required to predict accurately. Update gate- takes the inputs of the current time step and weights from previous time step. This gate restrict the flow of memory from previous

states. Forget gate - it decides how much of the previous memory can be forgotten. The combination of update and forget gate generate data or memory of current time step. A unit GRU is illustrated in fig.81

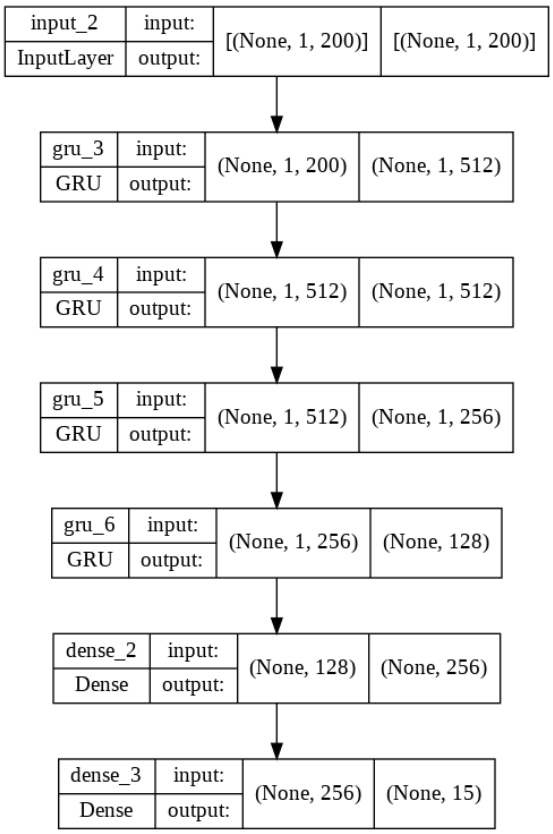


Figure 82: Gated recurrent unit architecture (GRU)

Bi-directional Layer

In RNN, the input to output flow would be from left to right. By using bi-directional layer , the input data can be given from either end. There is no change in the back-propagation as the flow of input only changes by using bi-directional layer.

The regularization of RNN are performed using drop-out layer. This prevent over-fitting on data.

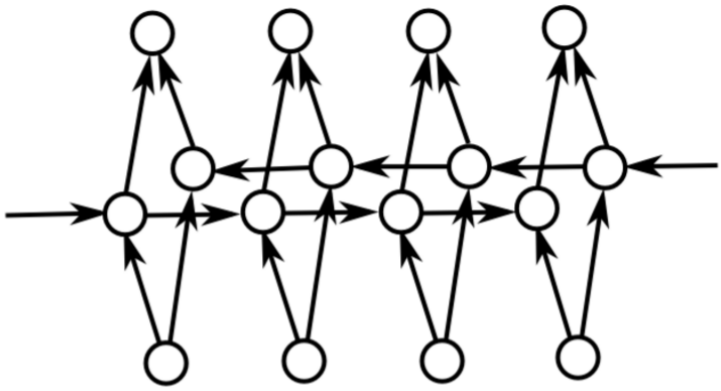


Figure 83: Work-flow of bi-directional layer.

Table 8: Number of parameters in different RNN architecture

RNN	**Bi-LSTM**	**LSTM**	**Bi-GRU**	**GRU**
Total params	44,66,191	45,80,879	33,63,087	34,49,103
Trainable params	44,66,191	45,80,879	33,63,087	34,49,103
Non-trainable params	0	0	0	0

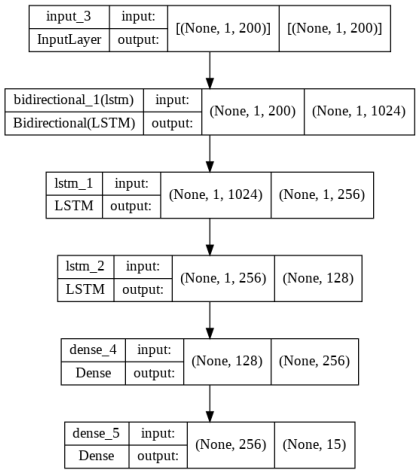


Figure 84: Bi-directional Long short term memory architecture (Bi-LSTM)

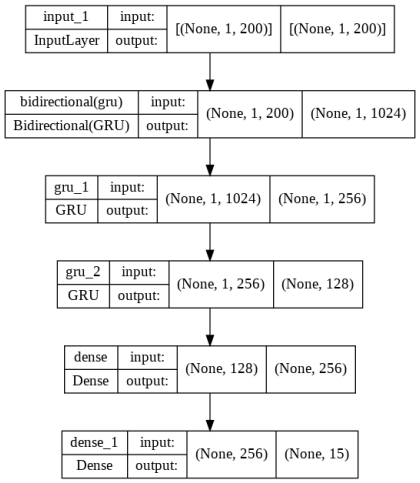


Figure 85: Bi-directional Gated recurrent unit architecture (Bi-GRU)

6.3.2 Training and Experimentation

As the network is a recursive algorithm, the parameters are update recursively and require huge computational effort. As the flow of data is sequential, parallelisation

of network is not possible.

Training Data

The input data is radiation phase thermograms and output are the class to which the thickness of the object belongs. From our experiment, the thickness of one class were high compared to other thickness classes. To avoid over-fitting on one class, we built a class weight dictionary using sci-kit learn **class weight** module. The class weight convert imbalanced dataset to a balanced dataset by adding class weight to loss during training.

Network selection

The performance of all the network is evaluated for same set of hyper-parameters.

- Optimizer: Adam
- Learning rate : 0.0001
- Batch size : 8
- Number of epoch : 500

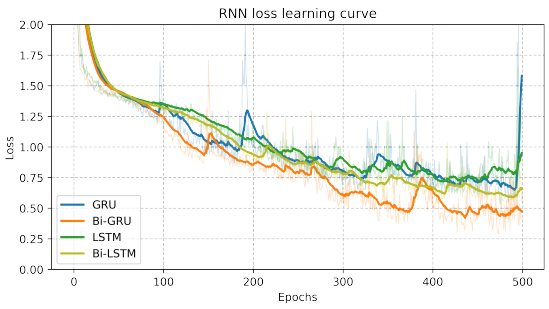


Figure 86: Loss curve for RNN types

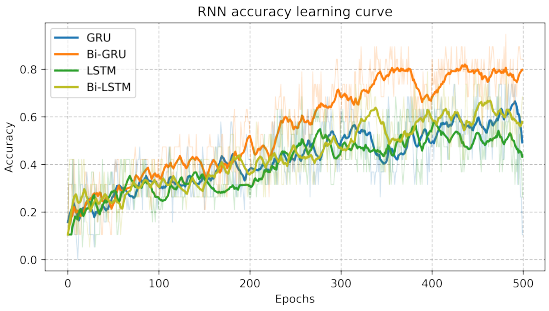


Figure 87: Accuracy curve for RNN types

The Bi-GRU was the best performing network among all the networks.

Contrast selection

Deep learning models tend to perform best if the inputs are between -1 to 1. To maximize the performance and improve the accuracy of the model different contrast functions are used.

- Normalizing function : Standardizing the thermography data in Gaussian distribution.
- Temperature contrast function :

$$x = \frac{T - T_i}{T_i} \tag{6.3.1}$$

where T is the present temperature, T_i is the initial temperature.

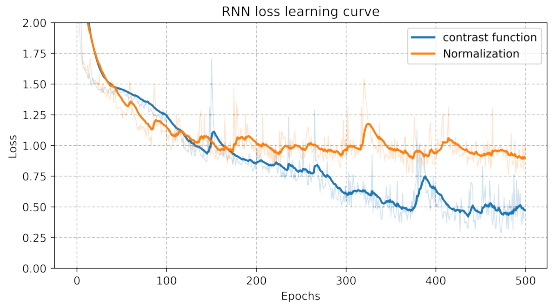


Figure 88: Loss curve for different contrast functions

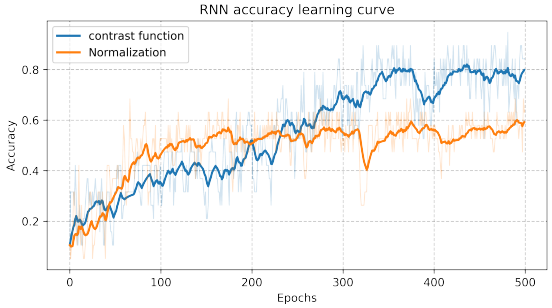


Figure 89: Accuracy curve for different contrast functions

Temperature contrast function is giving better accuracy as an unique profile of radiation thermography can be created.

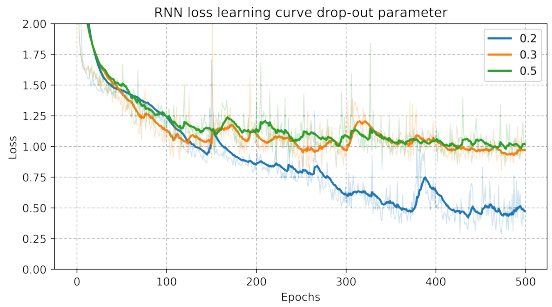


Figure 90: Loss curve for different drop-out parameters

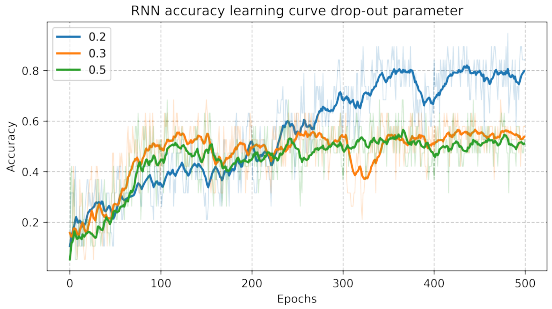


Figure 91: Accuracy curve for different drop-out parameters

Drop-out

RNN tend to over-fit on training data and to avoid this drop-out is used. Drop-out is a random parameter which kills the neurons in neural network by assigning 0 to it. This cause the neuron from passing any information. The process of selecting the neuron is done randomly.

Final model parameters

The final parameters are selected based on the above experimentation and the parameters which have high accuracy and low loss are selected.

1. Learning Rate- 0.0001
2. Batch size - 8
3. Gradient descent optimizer - ADAM
4. Contrast function =Initial temperature profile as in equ.[7.2.1].
5. Dropout value -0.2

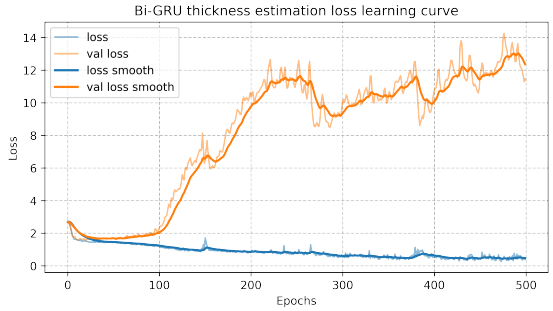


Figure 92: Loss curve of the model with final parameters

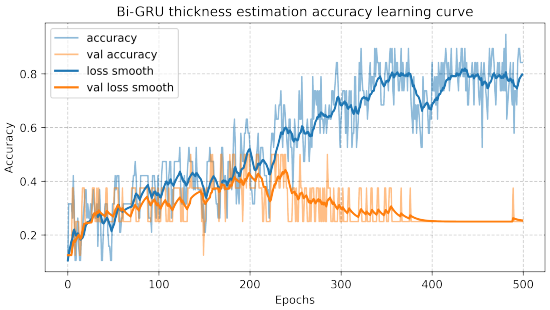


Figure 93: Accuracy curve of the model with final parameters

The validation dataset is the temperature profile of duration 5 seconds where as training dataset is of duration 10 seconds. The high loss is expected as it's a different dataset.

Chapter 7

NN:Performance Evaluation

7.1 Object Segmentation

The performance of different architecture on training, validation and testing dataset are evaluated using evaluation metric like accuracy.

7.1.1 PSP-Net

The performance of PSP-Net on training, validation and test datasets is evaluated.

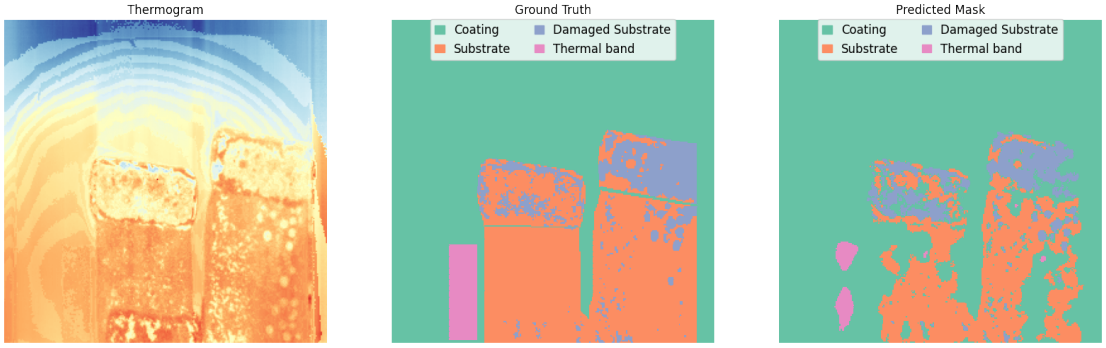


Figure 94: Input thermogram (left), expected output (center), predicted mask(right) for training dataset(PSP-Net)

The number of parameters in PSP-net are 2,09,916. The parameters of PSP-net are trained on training data. In general, the network predict accurately on data

similar to training data.

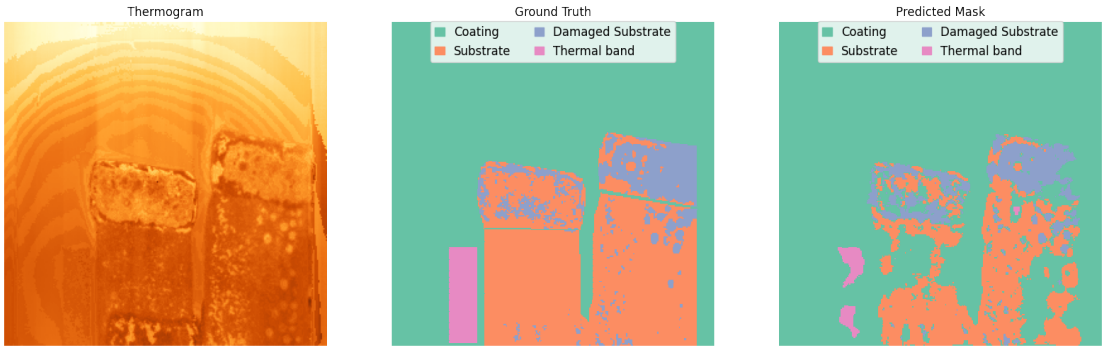


Figure 95: Input thermogram (left), expected output (center), predicted mask(right) for validation data(PSP-Net).

Validation dataset is a part of training dataset on which the network is not trained but the parameters are evaluated.

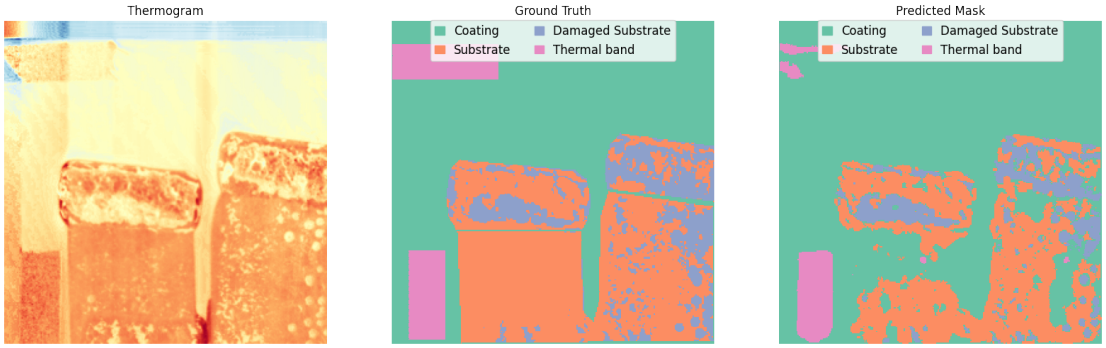


Figure 96: Input thermogram (left), expected output (center), predicted mask(right) for test data(PSP-Net).

The performance of PSP-Net for final parameters and hyper-parameters with varying batch size is given below

Table 9: Performance index (Accuracy) of PSP-Net for different batch sizes

Batch-size	**4**	**8**	**16**
Training dataset	72.51%	86.81%	85.79%
Validation dataset	72.02%	86.06%	85.66%
Test dataset	72.71%	86.30%	85.34%

The performance evaluation of PSP-net indicates that the network is unable to learn the complex features within the thermogram. As the architecture of the network is simple, the network is limited to the number of features it can learn.

7.1.2 FC-Net

FC-Net was first used for segmentation of objects in large images (740x1084). Training of networks with large input parameters requires huge computational size. The performance of FC-Net is evaluated on training data, validation data and test data.

The number of parameter in FC-Net are 14,751,956. FC-Net consist of huge number of trainable parameter and requires larger dataset. Initially, the network trained on basic dataset without any augmentation. This is generally done to train the network to identify important features. Later, it is trained on augmented dataset (rotation, translation) to work on varied data.

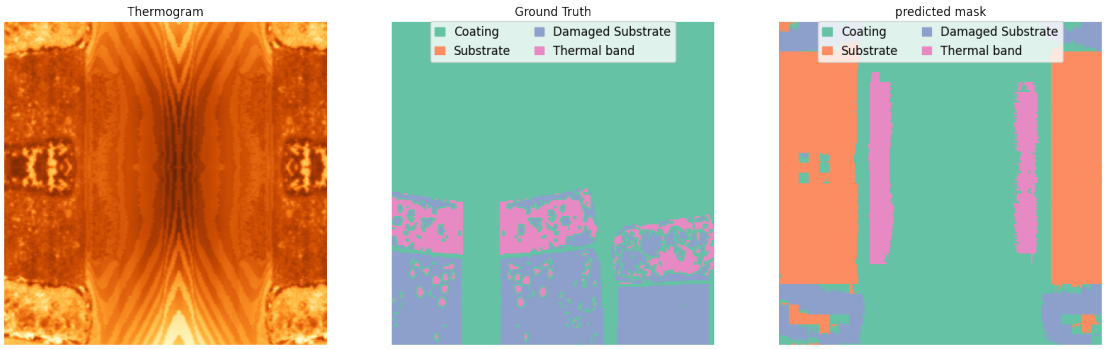


Figure 97: Input thermogram (left), expected output (center), predicted mask(right) for training data(FC-Net)

The validation dataset is non-augmented dataset which is used to evaluate the parameters and hyper-parameters.

The performance of FC-Net for final parameters and hyper-parameters with varying batch size is given below

The object segmentation operation by FC-net is quite effective. The main drawback of FC-net is it's inability to generate fine segmentation. FC-net generates coarse segmentation. The network performs moderately good compared to PSP-Net. Due to the coarse segmentation, the network can not identify minute feature like the damaged substrate.

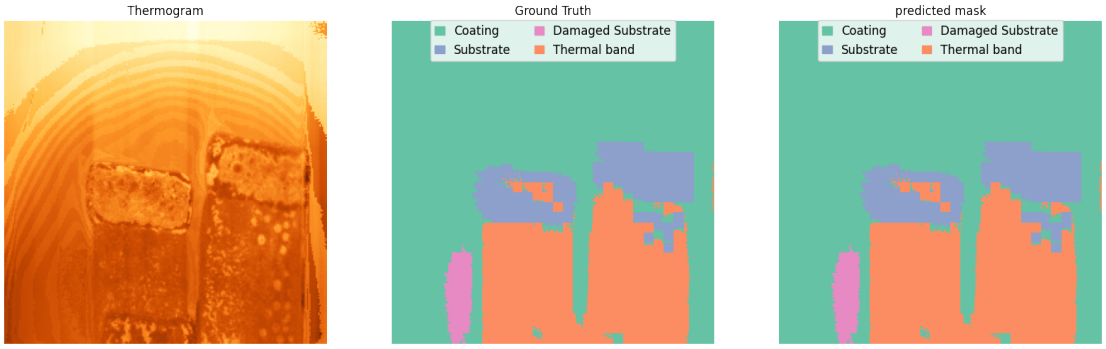


Figure 98: Input thermogram (left), expected output (center), predicted mask(right) for validation data(FC-Net).

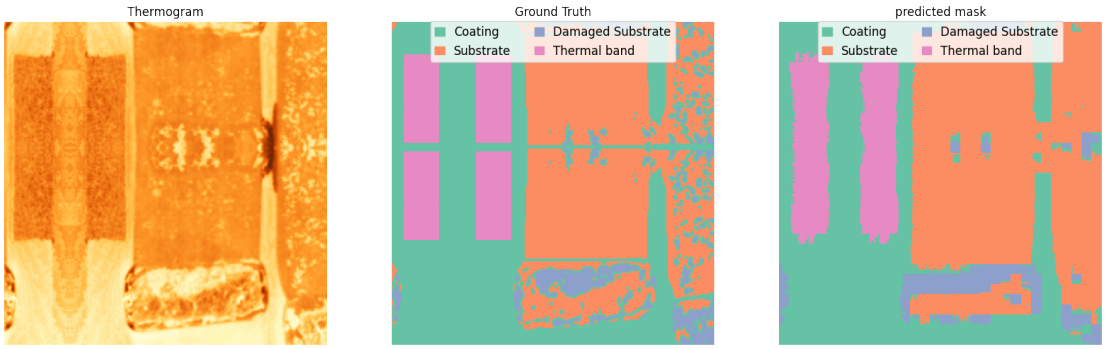


Figure 99: Input thermogram (left) expected output (center) predicted mask(right) on test data(FC-Net).

Table 10: Performance index (Accuracy) of FC-Net for different batch sizes

Batch-size	**16**	**32**	**64**
Training dataset	85.87%	86.55%	78.53%
Validation dataset	90.74%	90.35%	85.44%
Test dataset	86.83%	85.73%	82.94%

7.1.3 U-Net

U-net were first used to segment medical images like x-rays.

U-net can segment complex features. The performance of U-net is evaluated on training, validation and test dataset. U-net consist of 4,629,396 trainable parameter. In U-net the intermediate feature vectors from encoder are added to feature vectors in decoder. This method need to lot of memory to store all intermediate feature vector in encoder.

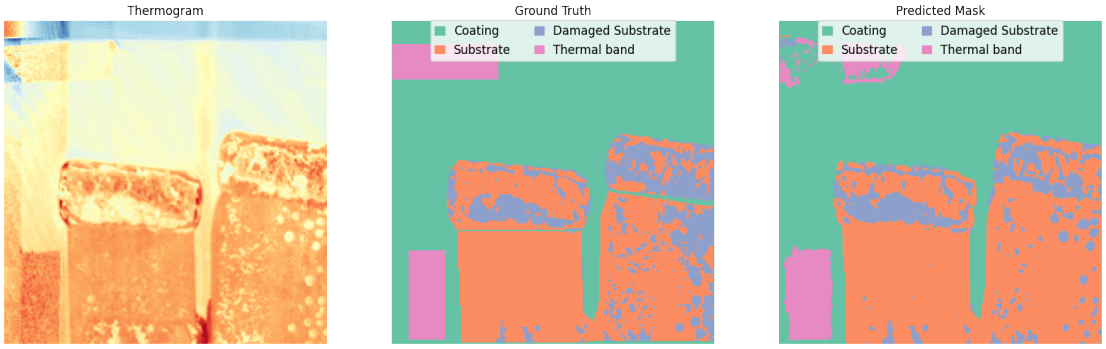


Figure 100: Input thermogram (left) expected output (center) predicted mask(right) on training data(U-Net)

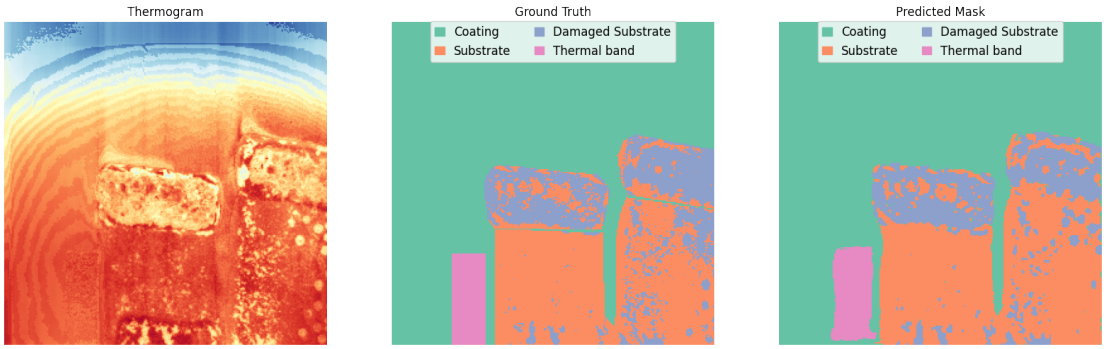


Figure 101: Input thermogram (left) expected output (center) predicted mask(right) on validation data(U-Net).

The validation dataset is used to evaluate the parameters and hyper-parameters.

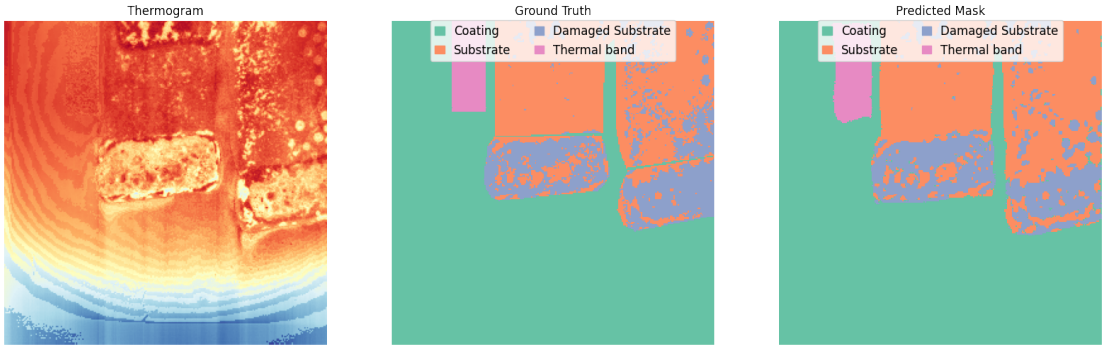


Figure 102: Prediction on test data(U-Net).

The performance of U-Net for final parameters and hyper-parameters with varying filter size is given below

Table 11: Performance index (Accuracy) of U-Net for different filter sizes

Filter size	**4**	**5**	**6**	**7**
Training dataset	91.10%	91.36%	92.47%	81.46%
Validation dataset	91.28%	91.57%	92.62%	81.50%
Test dataset	90.63%	91.03%	92.22%	81.40%

U-net generates very fine segmentation and perform the best among the all the neural network. U-net tends to over-fit on data. To avoid that the training is stopped when the loss reaches a tolerance limit of 0.2. As the number of filter increases a bottle-neck is created which restricts the amount information flow in the architecture. U-net are initially trained on non-augmented data and later experimented with augmented data and other dataset

Experimentation

The performance of U-Net on augmented data can be seen in fig.[103]

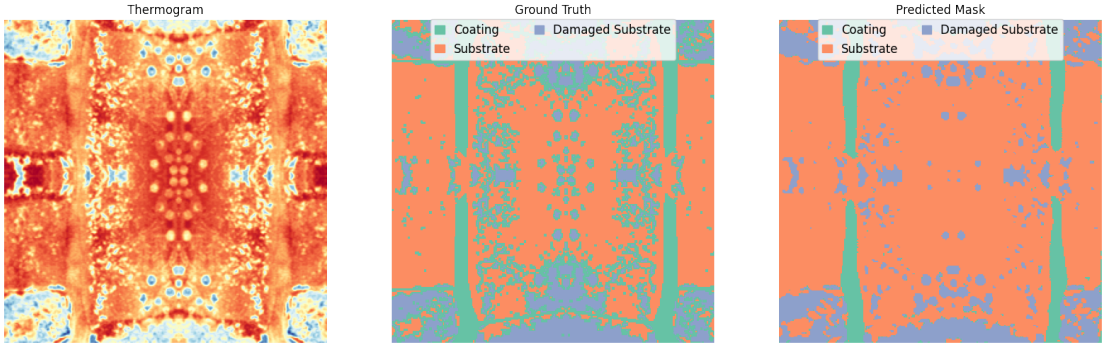


Figure 103: Input thermogram (left) expected output (center) predicted mask(right) on augmented data(U-Net)

The performance of U-Net on metal data can be seen in fig.[104]

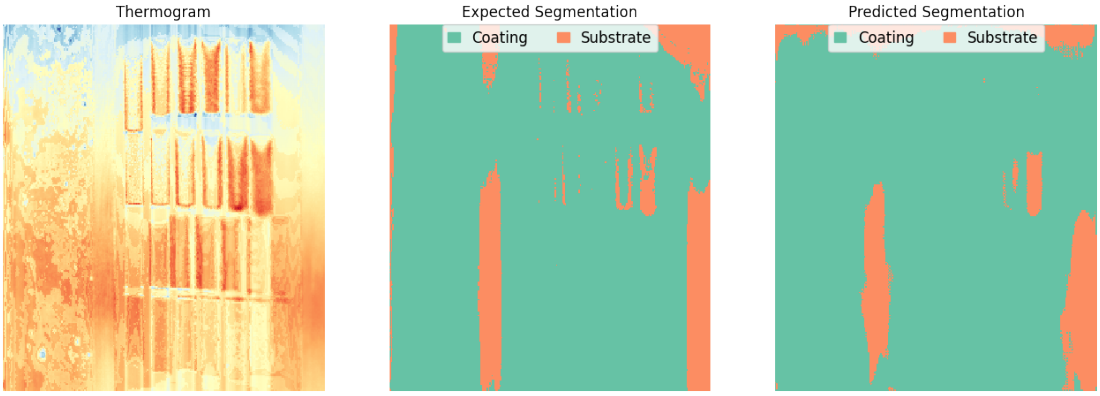


Figure 104: Input thermogram (left) expected output (center) predicted mask(right) on metal data(U-Net)

Table 12: Performance index (Accuracy) of all segmentation networks on thermograms

Architectures	**PSP-Net**	**FC-Net**	**U-Net**
Training dataset	86.81%	86.55%	92.47%
Validation dataset	86.06%	90.35%	92.62%
Test dataset	86.30%	85.73%	92.22%

7.2 Thickness Estimation

After the generation of segmentation mask, the mask can be used to plot the thermal sequence of the features in the thermogram.

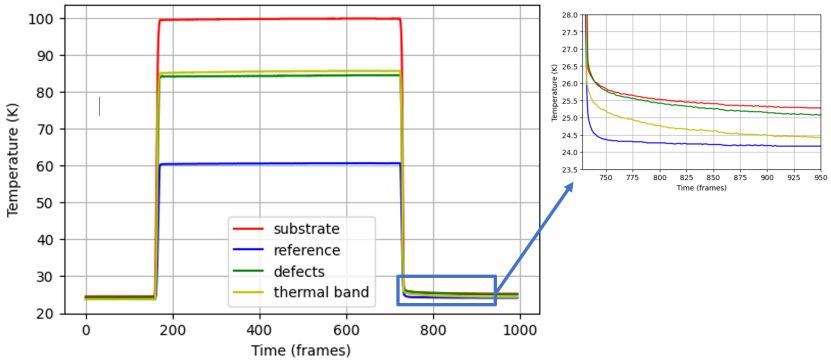


Figure 105: Thermal sequence of different feature in thermograms

The radiation phase data can be used to predict the depth or thickness of the

coating which are discussed in the next section. Based on the number of features detected during segmentation process depth analysis or thickness analysis can be performed.

7.2.1 Depth Analysis

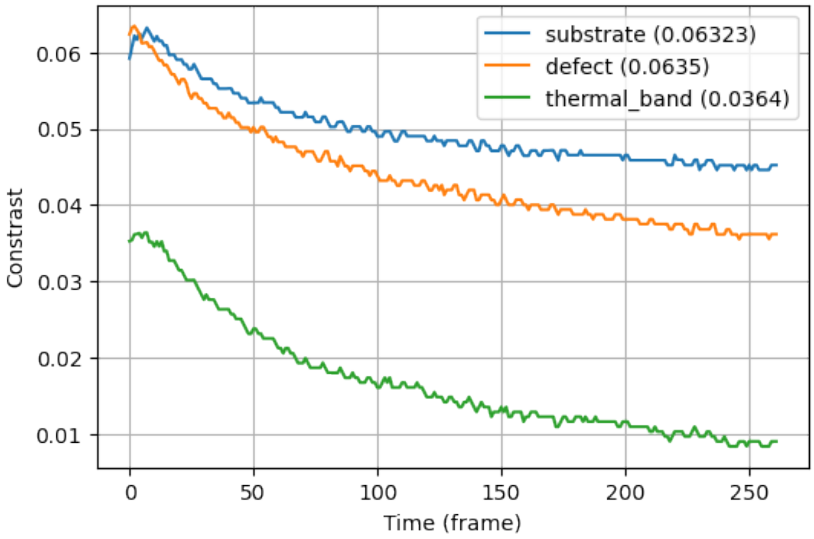


Figure 106: Radiation contrast function w.r.t time(frames)

The depth in the coating can be estimated using the time sequence data in which the temperature decrease over time due to loss of heat to the surrounding after the removal of heat source. We have used a contrast function which is given as

$$x_i = \frac{T_i - T_r}{T_r} \tag{7.2.1}$$

where x_i is the contrast value at time i ,T_i is the feature temperature at time i, T_r is the reference or surrounding temperature at time i.

The contrast value for fig.105 for time sequence of 250 frames are plotted in fig.106 To perform efficient depth analysis, we require thermography from various coating with different depth. The work-flow for training is prepared and trained on single training set given in fig.106 which is yielding 100% accuracy.

7.2.2 Thickness Analysis

The thickness are classified into different class which represent the range of thickness as shown in fig.107

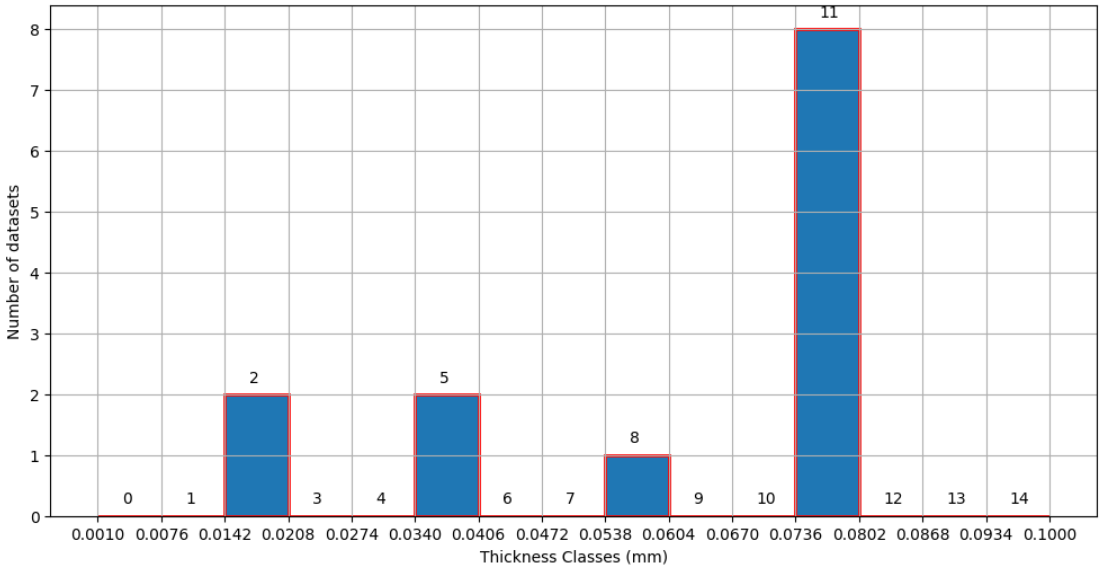


Figure 107: Coating thickness classification.

If the object segmentation doesn't show any features. Then the thermogram contains coating without any substrate. Thermography analysis on this type of objects can be used to estimate the thickness of the coating. Each coating has unique radiation phase which can be optimized to estimate thickness. For this time series analysis of radiation phase needs to be performed.

Table 13: Performance index (Accuracy) of all RNN architecture for thickness estimation using 1000 W heat source

Network	**MSE**	**Accuracy**
LSTM	1.2301	31.58%
GRU	2.3292	21.05%
Bi-LSTM	0.6114	73.68%
Bi-GRU	0.2288	89.47%

The higher the value, more energy is absorbed during heating phase. The performance of Bi-directional GRU for coating thickness on few sample is shown fig.108. The model is trained on heat source with 1000 W which yield better thermography analysis compared to 500 W.

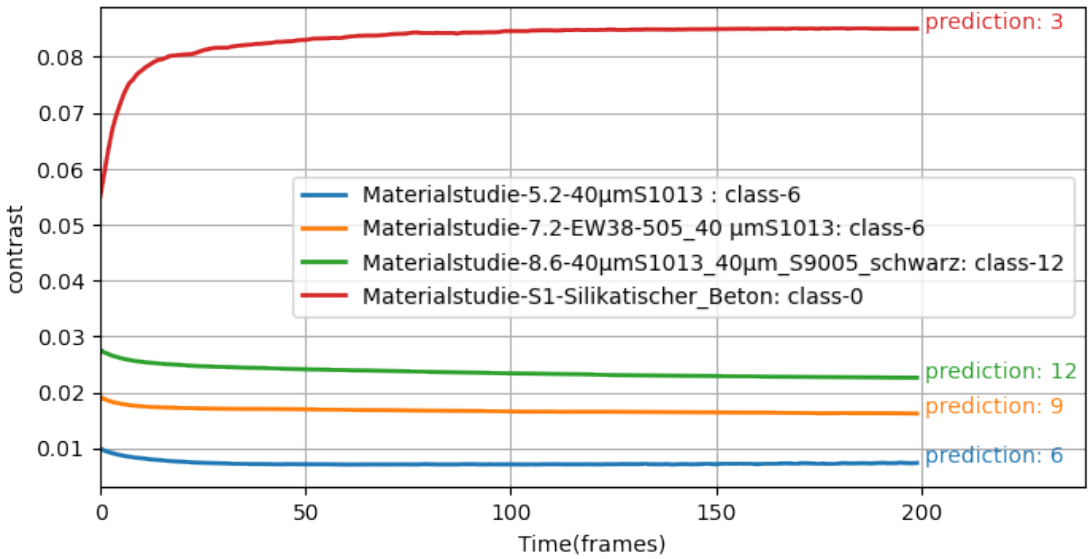


Figure 108: Thickness estimation for few samples with actual class in legend and predicted class

Table 14: Performance index (Mean Square error) of all RNN architecture for thickness estimation using 500 W heat source

Network	**MSE**	**Accuracy**
LSTM	2.6280	15.38%
GRU	5.3732	30.77%
Bi-LSTM	3.4682	30.77%
Bi-GRU	8.037	23.03%

As the number sample can not be generated and have to be collected. A lot of data need to be collected and trained before deploying it in a real-world scenario.

Chapter 8

Summary

In this student work, a MLOps software is developed in python. The data pipelines in this software are independent of any other third part software's. The workflow of ML-LaDECO is given in fig.[109]

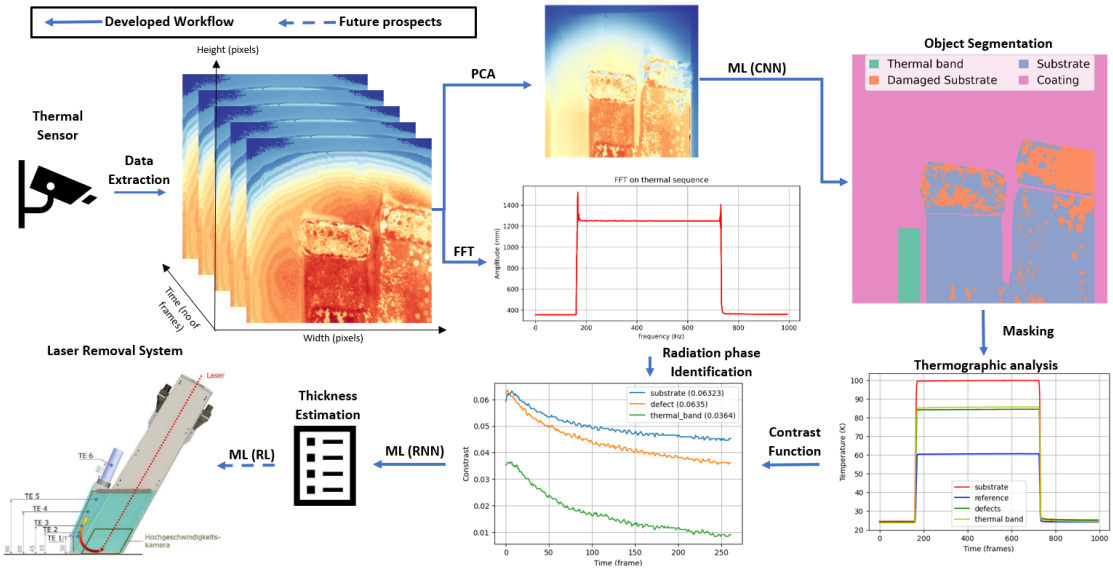


Figure 109: ML-LaDECO process workflow.

In this set-up, the thermograms (Input) from thermal sensor are extracted into an 3D array. After the extraction, reflection and radiation phases are identified using amplitude calculated from Fast Fourier transformation (FFT). The segmentation of features (coating, substrate and thermal band) in thermogram for various heat source and incident angles are investigated. Principle component analysis (PCA) is identified as a method which can extracts important features from ther-

mogram sequence with minimum effluence of other factors. The segmentation masks (Output) for training are generated using labelMe and OpenCV. Object segmentation models like FC-Net,U-net and PSP-Net are built and optimized.

The performance of the object segmentation neural networks for various parameter and hyper-parameters are evaluated. U-Nets are able to segment the features effectively achieving high accuracies. Data augmentation is performed to generate more complex data on which the performance of U-Nets are trained and optimized. This process increases the robustness and flexibility of the model. Thus predicting complex features in real world scenario. The segmentation object's thermographic analysis is performed to extract temperature profile of coating, substrate, damaged substrate and thermal band.

From temperature profile, thickness is estimation using time-series analysis of radiation phase in thermal sequence. For this process, the training data consists of different surface types with few colouration samples. The thickness are divided into different classes to perform classification process. This module depends on various external factors which are initial temperature, time period of heat source. A contrast temperature function is required to create a unique radiation phase profile for different coatings. The thickness or depth estimation module have performed well on the data available, but to check it robustness, a wide range of data is required.

The work-flow for quantification of laser-based coating removal system can be easily calibrated for real-time industrial application. This quantification process is urgently needed for analysing the radioactive PCB coatings which can not be disposed of into landfills due to health risks. From the quantification process, lasers can be calibrated in order to prevent vitrification of the concrete or soothing which can leave behind traces of PCB coating in concrete and metal.

The future prospects of the project can be expanding the present data flow till the laser system. The data from the laser system along with quantification data can be used to calibrate the laser system for optimal parameters for efficient coating removal. For this process, reinforcement learning have shown significant success.

Bibliography

[1] A. Anthofer, P. Kögler, C. Friedrich, W. Lippmann, O. Peise, S. Voss, A. Hurtado, and D. Trimis, "Laserdekontamination von pcb-haltigen lackschichten," *Energy Policy*, vol. 137, p. 111125, 2020.

[2] "Hershel experiement light scaattering." `https://skepticalscience.com/print.php?n=2275`. Accessed: 2022-02-22.

[3] R. Usamentiaga and D. F. García, "Infrared thermography sensor for temperature and speed measurement of moving material," *Sensors*, vol. 17, no. 5, p. 1157, 2017.

[4] K. I. Schultz, M. W. Kelly, J. J. Baker, M. H. Blackwell, M. G. Brown, C. B. Colonero, C. L. David, B. M. Tyrrell, and J. R. Wey, "Digital-pixel focal plane array technology," *Lincoln Laboratory Journal*, vol. 20, no. 2, pp. 36–51, 2014.

[5] "Thermography active thermography." `http://www.ir-ndt.de/en/active_thermography.php`. Accessed: 2022-02-24.

[6] T. Le, R. Garcia, P. Casari, and P.-O. Östberg, "Machine learning methods for reliable resource provisioning in edge-cloud computing: A survey," *ACM Computing Surveys*, vol. 52, pp. 1–39, 09 2019.

[7] "NN deep learning." `http://www.mplsvpn.info/2017/11/what-is-neuron-and-artificial-neuron-in.html`. Accessed: 2022-03-09.

[8] "ANN deep learning." `https://towardsdatascience.com/power-of-a-single-neuron-perceptron-c418ba445095`. Accessed: 2022-03-09.

[9] "AF deep learning." `https://medium.com/dataseries/basic-overview-of-convolutional-neural-network-cnn-4fcc7dbb4f17`. Accessed: 2022-03-09.

[10] "CL deep learning." https://medium.com/swlh/convolutional-neural-networks-for-multiclass-image-classification-a-beginners- Accessed: 2022-03-10.

[11] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," *Insights into imaging*, vol. 9, no. 4, pp. 611–629, 2018.

[12] "CNN netowork deep learning." https://www.jefkine.com/general/2016/09/05/backpropagation-in-convolutional-neural-networks/. Accessed: 2022-03-09.

[13] "RNN netowork deep learning." https://medium.com/deeplearningbrasilia/deep-learning-recurrent-neural-networks-f9482a24d010. Accessed: 2022-03-09.

[14] M. Boden, "A guide to recurrent neural networks and backpropagation," *the Dallas project*, vol. 2, no. 2, pp. 1–10, 2002.

[15] A. Pestel, S. Eckart, and H. Krause, "Entwicklung eines echtzeit-detektionssystems zur identifizierung von lack- und farbschichten auf betonoberflächen," 05 2021.

[16] "GD gradient descent." https://towardsdatascience.com/optimizers-for-training-neural-network-59450d71caf6. Accessed: 2022-03-09.

[17] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2881–2890, 2017.

[18] S. Piramanayagam, E. Saber, W. Schwartzkopf, and F. Koehler, "Supervised classification of multisensor remotely sensed images using a deep learning framework," *Remote Sensing*, vol. 10, p. 1429, 09 2018.

[19] Z. Hamdi, M. Brandmeier, and C. Straub, "Forest damage assessment using deep learning on high resolution remote sensing data," *Remote Sensing*, vol. 11, 08 2019.

[20] T. A. Shifat and J.-W. Hur, "Remaining useful life estimation of bldc motor considering voltage degradation and attention-based neural network," *IEEE Access*, vol. 8, 09 2020.

[21] T. Scherwath, B. Wealer, and R. Mendelevitch, "Nuclear decommissioning after the german nuclear phase-out an integrated view on new regulations and nuclear logistics," *Energy Policy*, vol. 137, p. 111125, 2020.

[22] B. B. Lahiri, S. Bagavathiappan, T. Jayakumar, and J. Philip, "Medical applications of infrared thermography: a review," *Infrared Physics & Technology*, vol. 55, no. 4, pp. 221–235, 2012.

[23] J. Speakman and S. Ward, "Infrared thermography: Principles and applications," *ZOOLOGY Zoology*, vol. 101, pp. 224–232, 01 1998.

[24] R. LUKAC and K. Plataniotis, "Single-sensor camera image processing," 10 2006.

[25] L. De Marchi and L. Mitchell, *Hands-On Neural Networks: Learn how to build and train your first neural network model using Python*. Packt Publishing Ltd, 2019.

[26] H. Szu and G. Rogers, "Generalized mccullouch-pitts neuron model with threshold dynamics," pp. 535 – 540 vol.3, 07 1992.

[27] E. Grossi and M. Buscema, "Introduction to artificial neural networks," *European journal of gastroenterology hepatology*, vol. 19, pp. 1046–54, 01 2008.

[28] H. Goh, "Continuous fourier transform: A practical approach for truncated signals and suggestions for improvements in thermography," 07 2019.

[29] B. Milovanović, M. Gaši, and S. Gumbarević, "Principal component thermography for defect detection in concrete," *Sensors*, vol. 20, no. 14, 2020.

[30] "Data augumentation deep learning." `https://research.aimultiple.com/data-augmentation/#:~:text=%20Benefits%20of%20data%20augmentation%20include%3A%20%201,closely%20to%20a%20limited%20set%20of...%20More%20`. Accessed: 2022-03-16.

[31] "GD gradient descent optimizers." `https://www.datatechnotes.com/2019/12/understanding-optimizers-in-neural.html#:~:text=Neural%20network%20optimization%20is%20a%20process%20to%20fit,implementations%20in%20neural%20network%20training%20with%20Keras%20API.` Accessed: 2022-03-09.