Automatic defect detection in industrial CT volumes of casting

Abdel Rahman Dakak

N° d'ordre NNT : 2022LYSEI013

THÈSE DE DOCTORAT **DE L'UNIVERSITÉ DE LYON**
opérée au sein de
L'Institut National des Sciences Appliquées (INSA) de Lyon

École Doctorale N° ED160
Électronique, Électrotechnique, Automatique

Spécialité/ discipline de doctorat :
Traitement du Signal et de l'Image

Soutenue publiquement le 09/03/2022, par :
Abdel Rahman DAKAK

Automatic Defect Detection in Industrial CT Volumes of Castings

Détection Automatique des Défauts dans des Volumes Tomographiques des Pièces de Fonderie

Devant le jury composé de :

M. OSMAN Ahmad	Professeur, HTW des Saarlandes, Président
Mme. ROMBAUT Michèle	Professeur, Université Grenoble Alpes, Rapporteur
Mme. RUAN Su	Professeur, Université de Rouen, Rapporteur
M. MERY Domingo	Professeur, UC de Chile, Examinateur
M. DUFFNER Stefan	Maître de Conférences, INSA de Lyon, Examinateur
Mme. KAFTANDJIAN Valérie	Professeur, INSA de Lyon, Directrice de thèse
M. DUVAUCHELLE Philippe	Maître de Conférences, INSA de Lyon, Co-encadrant

Département FEDORA – INSA Lyon - Ecoles Doctorales – Quinquennal 2016-2020

SIGLE	ECOLE DOCTORALE	NOM ET COORDONNEES DU RESPONSABLE
CHIMIE	**CHIMIE DE LYON** http://www.edchimie-lyon.fr Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage secretariat@edchimie-lyon.fr INSA : R. GOURDON	**M. Stéphane DANIELE** Institut de recherches sur la catalyse et l'environnement de Lyon IRCELYON-UMR 5256 Équipe CDFA 2 Avenue Albert EINSTEIN 69 626 Villeurbanne CEDEX directeur@edchimie-lyon.fr
E.E.A.	**ÉLECTRONIQUE, ÉLECTROTECHNIQUE, AUTOMATIQUE** http://edeea.ec-lyon.fr Sec. : M.C. HAVGOUDOUKIAN ecole-doctorale.eea@ec-lyon.fr	**M. Gérard SCORLETTI** École Centrale de Lyon 36 Avenue Guy DE COLLONGUE 69 134 Écully Tél : 04.72.18.60.97 Fax 04.78.43.37.17 gerard.scorletti@ec-lyon.fr
E2M2	**ÉVOLUTION, ÉCOSYSTÈME, MICROBIOLOGIE, MODÉLISATION** http://e2m2.universite-lyon.fr Sec. : Sylvie ROBERJOT Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 INSA : H. CHARLES secretariat.e2m2@univ-lyon1.fr	**M. Philippe NORMAND** UMR 5557 Lab. d'Ecologie Microbienne Université Claude Bernard Lyon 1 Bâtiment Mendel 43, boulevard du 11 Novembre 1918 69 622 Villeurbanne CEDEX philippe.normand@univ-lyon1.fr
EDISS	**INTERDISCIPLINAIRE SCIENCES-SANTÉ** http://www.ediss-lyon.fr Sec. : Sylvie ROBERJOT Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 INSA : M. LAGARDE secretariat.ediss@univ-lyon1.fr	**Mme Sylvie RICARD-BLUM** Institut de Chimie et Biochimie Moléculaires et Supramoléculaires (ICBMS) - UMR 5246 CNRS - Université Lyon 1 Bâtiment Curien - 3ème étage Nord 43 Boulevard du 11 novembre 1918 69622 Villeurbanne Cedex Tel : +33(0)4 72 44 82 32 sylvie.ricard-blum@univ-lyon1.fr
INFOMATHS	**INFORMATIQUE ET MATHÉMATIQUES** http://edinfomaths.universite-lyon.fr Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage Tél : 04.72.43.80.46 infomaths@univ-lyon1.fr	**M. Hamamache KHEDDOUCI** Bât. Nautibus 43, Boulevard du 11 novembre 1918 69 622 Villeurbanne Cedex France Tel : 04.72.44.83.69 hamamache.kheddouci@univ-lyon1.fr
Matériaux	**MATÉRIAUX DE LYON** http://ed34.universite-lyon.fr Sec. : Stéphanie CAUVIN Tél : 04.72.43.71.70 Bât. Direction ed.materiaux@insa-lyon.fr	**M. Jean-Yves BUFFIÈRE** INSA de Lyon MATEIS - Bât. Saint-Exupéry 7 Avenue Jean CAPELLE 69 621 Villeurbanne CEDEX Tél : 04.72.43.71.70 Fax : 04.72.43.85.28 jean-yves.buffiere@insa-lyon.fr
MEGA	**MÉCANIQUE, ÉNERGÉTIQUE, GÉNIE CIVIL, ACOUSTIQUE** http://edmega.universite-lyon.fr Sec. : Stéphanie CAUVIN Tél : 04.72.43.71.70 Bât. Direction mega@insa-lyon.fr	**M. Jocelyn BONJOUR** INSA de Lyon Laboratoire CETHIL Bâtiment Sadi-Carnot 9, rue de la Physique 69 621 Villeurbanne CEDEX jocelyn.bonjour@insa-lyon.fr
ScSo	**ScSo*** http://ed483.univ-lyon2.fr Sec. : Véronique GUICHARD INSA : J.Y. TOUSSAINT Tél : 04.78.69.72.76 veronique.cervantes@univ-lyon2.fr	**M. Christian MONTES** Université Lyon 2 86 Rue Pasteur 69 365 Lyon CEDEX 07 christian.montes@univ-lyon2.fr

*ScSo : Histoire, Géographie, Aménagement, Urbanisme, Archéologie, Science politique, Sociologie, Anthropologie

Acknowledgements

I would like to thank Mme. Michèle Rombault and Mme. Su Ruan for accepting to be the first reviewers. And I would like to thank M. Domingo Mery, M. Ahmad Osman and M. Stefan Duffner for agreeing to be on the viva-voce board.

Abstract

Industrial X-ray computed tomography (CT) has proven its value as a non-destructive method for inspecting light metal castings. The CT volume generated enables the internal and external geometry of the specimen to be measured, casting defects to be localized and their statistical properties to be investigated. On the other hand, CT volumes are very prone to artifacts that can be mistaken for defects by conventional segmentation algorithms. Based on CT data of aluminium alloy castings provided by industrial partners, we have developed an automatic approach to analyze discontinuities inside CT volumes based on a three-step pipeline: (1) 2D segmentation of CT slices with automatic deep segmentation to detect suspicious greyscale discontinuities; (2) classification of these discontinuities into true alarms (defects) or false alarms (artifacts and noise), using a trained convolutional neural network classifier; (3) localization of the validated defects in 3D to investigate their statistical properties such as sphericity, elongation and compactness. Based on this, the validated 3D defects are then classified into porosities or shrinkage cavities using an SVM classifier and a siamese neural network. The choice of each model and the training results are presented and discussed, as well as the efficiency of the approach as an automatic defect detection algorithm for industrial CT volumes.

Keywords: Casting, CT, X-Ray, Defect Detection, Machine Learning, Deep Learning, Few-Shot Learning.

Contents

List of Figures

List of Tables

List of Acronyms

2D	Two-Dimensional
3D	Three-Dimensional
adam	Adaptive Moment Estimation Optimizer
AUC	Area Under the Curve
CAD	Computer-Aided Design
CNN	Convolutional Neural Network
COM	Center-of-Mass
CT	Computed Tomography
DA	Data Augmentation
DL	Deep Learning
FN	False Negatives
FP	False Positives
FCN	Fully Convolutional Network
GPU	Graphics Processing Unit
IoU	Intersection over Union
MSE	Mean Squared Error
ML	Machine Learning
NDT	Non-Destructive Testing
NN	Neural Network
POD	Probability Of Detection
PR Curve	Precision Recall Curve
ReLU	Rectified Linear Activation Unit
ROI	Region of Interest
SGD	Stochastic Gradient Descent
SNN	Siamese Neural Network
SVM	Support Vector Machine
TN	True Negatives
TP	True Positives
TPR	True Positives Rate

Introduction

This work is part of a collaborative project between the Technical Centre CTIF, LVA laboratory of INSA Lyon, and an industrial group including: RENAULT, Groupe SAB, MONTUPET, EUROCAST and CONSTELLIUM.

Aluminium alloy castings are widely used in the automotive and aerospace industries, e.g. wheel rims, cylinder housings, engine blocks etc., as they have an excellent combination of mechanical and technological properties in addition to their stability. Although these components are subjected to high mechanical stress and fatigue, they allow fuel economy due to their light weight and high durability and safety. On the other hand, the production of castings is a difficult task and the manufactured specimens are never free of flaws. Various forms of defects, such as gas voids and shrinkage cavities can occur during the casting of the molten metal due to reactions with the sand of the mould or the ambient air, and due to material contraction during the subsequent cooling and solidification [1]. Although defects reduce the stability of the specimens, they are not all equally damaging. To maintain the longevity and reliability of the production, we need to ensure that a specimen has only the least harmful forms of defects and that they occur only in non-critical regions. We can achieve this either by destructive methods such as cutting the part open [2] or by non-destructive testing (NDT).

To ensure the integrity of aluminium alloy components and to increase their lifetimes, most foundries perform non-destructive testing after casting their specimens, to locate defects and study their impact. One of the most common methods is 2D radiography, which provides access to discontinuities inside the casting that are not visible to the naked eye. However, because this method projects the volume of

the casting onto a plane, it can sometimes be difficult to locate the defects along the thickness variations or to clearly project the sides of the casting. For these reasons, many foundries are moving to 3D computed tomography (CT) to inspect their castings, which allows to directly access the 3D volume of the casting and locate the discontinuities in 3D. CT initially had great success in evaluating and measuring the castings where there is no time constraint; and recent advances in computer technology and the increasing speed of image reconstruction algorithms have made it possible to use CT for inspecting castings on the production line [3]. In addition, tomography can be used to determine whether the discontinuities disappear or clog on the surface during machining or whether they are located in an area that is crucial for the mechanical resistance of the component.

Eliminating defective specimens in the early stages of production after inspection with CT saves time and money and increases the lifespan of validated specimens [4]. However, this requires a manual subjective interpretation of the CT volume, which is time consuming and depends mainly on the quality of the CT data. Indeed, the interpretation of industrial CT images is challenging for several reasons:

- To speed up production, inspectors prefer to limit the scanning time, which in turn leads to compromises in the spatial resolution of the images.
- Casting processes are very prone to small porosities and shrinkage cavities whose contrast in the output images is very low, and can easily be missed during a visual inspection.
- CT is prone to various types of artifacts that can arise during the 3D reconstruction of the volume of the specimen. These artifacts can have the same contrast as real defects, and can easily be mistaken for defects by non-trained users.
- Image interpretation is usually done upon binary images. This requires binarizing greyscale CT images, which can lead to deviant results if a validation procedure is not correctly applied.

All these obstacles raise the need for an automatic defect detection algorithm that can shorten the interpretation time and automatically process the industrial CT volumes, which is the ultimate goal of this study. To fully automate the processing, several deep learning algorithms were used to: (1) locate suspicious discontinuities that could be defects, but also artefacts; (2) classify these discontinuities as true defects or false alarms; (3) and finally identify the type of true defects, porosities or shrinkage cavities. To train these deep learning models, several industrial CT volumes of aluminium alloy castings were provided by the industrial partners, scanned with different tomography machines under unique scanning conditions, providing unique contrast and spatial resolutions.

In chapter I, we give a brief introduction about industrial computed tomography, explaining its foundations and the various forms of artifacts. This is followed by a literature review concerning defects detection algorithms with respect to CT data.

In chapter II, we present two segmentation algorithms, 2D and 3D, which serve as tools to generate ground-truth binary volumes from the greyscale CT volumes, which in turn serves as training datasets for the deep learning models and to validate their predictive abilities.

In chapter III, the state-of-the-art model U-Net is trained from ground up with our data to over-segment CT slices and include real defects as well as artifacts and noise in its output.

In chapter IV, a deep learning classifier named CT-Casting-Net is developed and trained to recognise real defects in CT images and distinguish artifacts as false alarms.

U-Net and CT-Casting-Net are coupled in chapter V to make up a new approach for automatic defect detection, which takes a 3D CT volume and generates a binary volume in which casting defects are highlighted. This approach was validated on manually segmented CT volumes, using validation metrics relevant to the non-destructive testing field.

And finally, Using a siamese neural network presented in chapter VI, defects validated by the above approach can be classified into porosities or shrinkage cavities, after studying their geometrical properties such as sphericity, elongation and compactness.

Chapter I

Tomography for Casting Inspection

In this chapter we look at the theoretical aspects of computed tomography (CT) and the use of this non-destructive testing (NDT) method for inspecting aluminium alloy castings, as well as its advantages compared to other NDT methods. We present the challenges in developing automatic defect detection algorithms for the analysis of CT casting images and the available CT data for the study. A literature review on casting defect detection methods such as guided image processing techniques, machine learning and deep learning algorithms , is provided.

I.1 Industrial Computed Tomography

The manufacturing industry is constantly searching for a compromise between the life cycles and the geometrical features of its products. Advances in manufacturing technology allow the production of components with high geometric complexity in terms of freedom surfaces and internal structures, which in turn complicate the conformity and integrity of the products. Consequently, this rises the need for new methods to check the production in terms of dimensions, tolerances and manufacturing defects in order to reduce economic costs and waste on the production line. One of the most advanced solutions for this task is X-ray computed tomography (CT), a non-destructive method that allows external and internal inspection without accessibility restrictions.

CT was developed in the late 1970s for medical imaging as a non-invasive imaging technique. It revolutionized the field of medical diagnosis by providing more information and details about the patient and was later widely applied to industrial applications such as material analysis, crack and void detection and metrology [5].

CT can be used during conception to validate the conformity of the design by performing a double quality check on the same CT data, such as dimensional quality and material quality [6]. This makes it the only technology that measures internal and external geometry without having to cut and destroy the component. Similarly, CT can also be applied after the specimen has been assembled into a complex structure. This allows gaps, surface deviations and forced tolerances to be detected, ensuring the proper functioning of the assembled system. Figure I.1 shows the quality control possibilities that can be applied to the same CT volume of a casting, such as geometrical deviations, thickness verification and material integrity (density and chemical composition).

I.2 Cone-Beam CT System

Before giving a brief description of the setup of a CT system, we must explain a few terms that will be used very frequently throughout the following chapters:

- **Volume**: a 3D array which represents the body of the specimen. It is reconstructed based on 2D X-ray projections at different angles.
- **Slice**: a CT volume can be sliced along a specific direction, producing a stack of 2D images called slices.
- **Voxel**: corresponding to the pixel in a 2D image, a CT volume is a 3D array of volumetric pixels called voxels, arranged in a regular grid and representing a small region of the specimen as illustrated in Figure I.2. The dimensions of the voxel along each direction represent the spatial resolution of the volume, and a CT volume is considered as isotropic if the 3 dimensions of the voxel are equal. In the following chapters, we will use the term voxel when referring to a 3D volume unit, as well as 2D slice unit since each slice has a physical

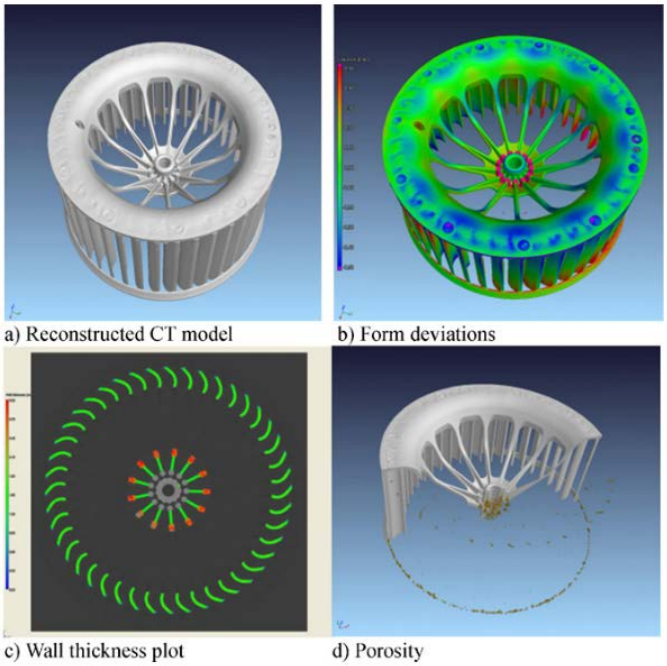


Figure I.1: Quality control with CT: (a) reconstructed CT volume, (b) geometry control, (c) thickness conformity, (d) porosity analysis. Source: [7]

thickness with respect to the inspected specimen.

- **Greyscale Value**: Each voxel has a greyscale value, which represents the X-ray attenuation level, i.e. the proportion of X-ray energy absorbed by the specimen at a defined region. The attenuation level, and consequently the greyscale value, depends on the X-ray energy spectrum and the composition and density of the material.

A variety of CT systems are available in the market, depending on the application and the size of the specimen [8]. A basic CT system consists of an X-ray source, a detector, a moving system that holds the specimen to be examined, and computation devices. The detector captures the X-rays attenuated by the specimen at different angles forming 2D radiographic images; and the computation devices reconstruct a

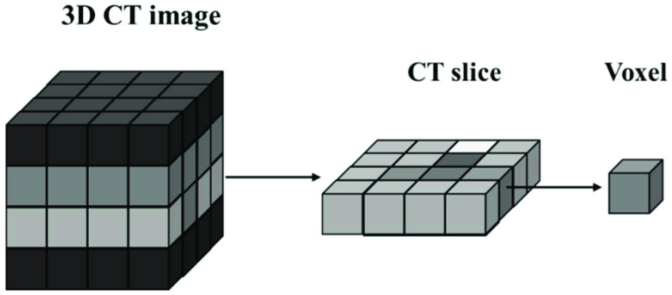


Figure I.2: Each 2D slice of the CT volume has a physical thickness determined by the voxel size.

3D volume based on these radiographic projections. In contrast to medical systems, the specimen is not fixed but rotated on a movable table, and is usually exposed to higher X-ray energy as there is no health hazard during the examination.

The industrial partners who provided CT data for the study use cone-beam CT systems, which are capable of scanning and producing 3D volume of a specimen in a single rotation. Figure I.3 shows the geometry of an industrial tomograph that captures the X-ray attenuation of a cone-shaped beam. Although this system is more prone to artifacts compared to the fan-beam, it is preferred for many applications because of the speed advantages and the ability to scan large specimens, either in stepwise or helical manners [9]. In the following, the influence of each component on the CT data quality is discussed.

I.2.1 Source

In the X-ray tube, the cathode shoots accelerated free electrons at a target metal (anode) to pull electrons out of the its atoms and produce X-ray energy. The amount of this energy is controlled by two factors set by the manufacturer: current and voltage. The first defines the number of electrons to be accelerated and the second the acceleration of the free electrons, which in turn defines the minimum wavelength of the polychromatic spectrum of the extracted electrons. Both factors

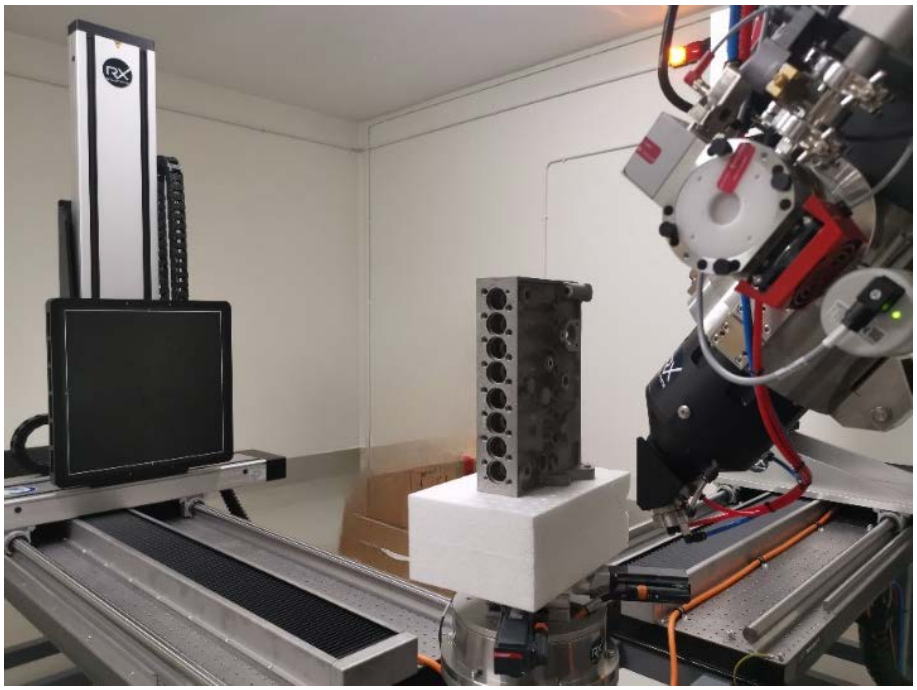


Figure I.3: Cone-beam industrial computed tomograph machine at INSA Lyon, with max X-ray voltage of 300 kV and current of 1000 μA. The object is placed on a rotary table, and the respective distances between the object, source and detector control the magnification factor, i.e. the spatial resolution of the output volume.

control the amount of X-rays emitted by the source and consequently the **contrast resolution** of the CT volume after reconstruction.

Another important design factor is the size of the focal spot, i.e. the point from which the X-rays are emitted. Increasing the size of the focal spot increases the maximum intensity of the X-ray beam, but at the expense of geometrical sharpness of the specimen volume, as this reduces the maximum permissible magnification. Consequently, enlarging the focal spot reduces the **spatial resolution**, i.e. the dimensions of each voxel in the CT volume [10].

I.2.2 Detector

Opposite the source, the detector measures the unabsorbed X-ray radiations after penetrating the specimen. Flat panel detectors are the most common X-ray detectors in industrial CT and consist of three successive layers: the scintillating layer, the photodiodes and the thin film transistors. When the scintillating layer is irradiated with X-ray energy, it produces visible light, which is then converted into electrical charge by the photodiodes. The thin-film transistors pass this electrical charge on to the readout electronics, which amplify this analogue signal and convert it into a digital representation.

The detector is structured as an array of X-ray sensors, each called a detector element, whose size, together with the thickness of the scintillator layer, has a major influence on the **spatial resolution** of the output image. On the other hand, the process of digitizing the analogue electrical charge controls the limits of the **contrast resolution** [11].

I.2.3 Object

The object or the specimen to be examined is positioned on a rotating table between the source and the detector. If the table is closer to the source, we have a higher magnification and the spatial resolution is more influenced by the size of the focal spot of the source. If the table is close to the detector, we have a lower magnification and the spatial resolution is determined by the size of the detector elements. In addition, the total distance between the detector and the source plays a major role in the contrast resolution, as doubling the distance reduces the X-ray radiation received by the detector by a quarter according to the inverse square law [12].

If the X-ray beam is not attenuated by the object during exposure, it is represented as black voxels in the reconstructed greyscale CT volume. If it is completely attenuated by a dense object, it is represented as white voxels. The level of attenuation controls the contrast resolution and is determined by several physical effects that depend on the material properties of the object, such as Rayleigh scattering, the photoelectric effect and the Compton effect. The sum of the contributions of the individual effects is the total attenuation coefficient, which is shown in Figure I.4

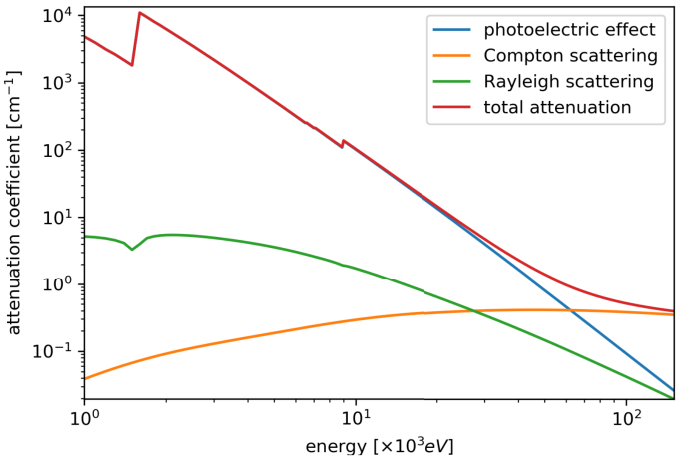


Figure I.4: The attenuation coefficient of the aluminium alloy EN AW 2014 [13] as a function of the energy of the X-ray radiation.

for an aluminium alloy and can be defined as follows:

$$\mu = \mu_R + \mu_{pe} + \mu_{Compt} \tag{I.1}$$

When an X-ray beam with intensity I_0 travels through *matter* with attenuation coefficient $\mu > 0$, it suffers an exponential loss of intensity. This X-ray attenuation can be expressed with the Lambert-Beer law, integrated over the energy spectrum of the beam and along the crossed path length:

$$I(L) = \int_0^{E_{max}} I_0(E) e^{-\int_0^L \mu(E,x)dx} dE \tag{I.2}$$

I, the intensity after interaction with the matter; E, the energy of X-rays; L, the thickness of the material to be penetrated by the X-ray beam.

Since attenuation depends on the properties of the material, the source voltage must be adjusted according to the thickness of the specimen, as shown in Table I.1, to achieve better contrast resolution. To avoid underexposure, the longest dimension of the specimen is usually mounted perpendicular to the rotating table.

I.2.4 Exposure

CT volumes are prone to artifacts caused by various types of noise during exposure. This can arise from the process of X-ray emission, its attenuation by the object and quantization by the X-ray detector; all of which are stochastic processes. Due to the low probability of photons being emitted from the focal spot of the emitter, and consequently the amount of X-rays received by the scintillating layer of the detector, the actual X-rays received can be estimated using the Poisson distribution [15].

Accordingly, to measure the quality of the X-ray projections before reconstructing the 3D volume, we can use the signal-to-noise ratio (SNR), which is usually equal to the square root of the theoretical intensity of the X-ray radiation. This means that the contrast resolution depends strongly on this noise, so much so that a doubling of the SNR requires a quadrupling of the X-ray dose.

In addition, the projections suffer from other sources of noise, such as the detector's readout electronics after receiving the attenuated radiation at the scintillating layer. All these noises manifest as artifacts in the reconstructed 3D volume, as explained in the following section.

I.3 3D Reconstruction & Artifacts

After taking X-ray radiographic projections at different angles around the specimen to obtain information about attenuation, a 3D map of the internal volume of the specimen can be reconstructed without overlapping features, as can be seen in Figure I.5. The contrast of the final volume is a mixed combination of density and composition information of all individual projections and can be visualized either as a 3D array, or as 2D slices after cutting the volume along a certain direction.

In mathematical terms, the reconstruction process solves an inverse problem by drawing a conclusion about the cause from an observation. This can be done using 3D reconstruction algorithms [17]. A widely used algorithm for cone-beam CT data is the Feldkamp algorithm, which projects the captured intensities from the detector back into object space [18]. Due to the difficulty of implementing reconstruction algorithms, optimized versions are usually used in industrial CT machines for fast computations. However, this shortcut implementation, together with the varying sources noise in the CT system, has disadvantages for the quality of the reconstructed volumes, which can manifest as reconstruction artifacts, as shown in Figure I.6.

These artifacts and noise manifest themselves in the form of greyscale variations or discontinuities that untrained users might mistake for defective zones. There is a long list of possible artifacts found in CT volumes [19], and the following are the most common in industrial casting data:

- Noise is a spontaneous greyscale variation in the image. It is usually the result of stochastic attenuation and detection during exposure. Note that some of

Table I.1: Typical maximum penetrable material thicknesses for common industrial materials. Source: [14]

X-Ray Voltage	130 kV	150 kV	190 kV	225 kV	450 kV
Steel/Ceramic	5 mm	<8 mm	<25 mm	<40 mm	<70 mm
Aluminium	<30 mm	<50 mm	<90 mm	<150 mm	<250 mm
Plastic	<90 mm	<130 mm	<200 mm	<250 mm	<450 mm

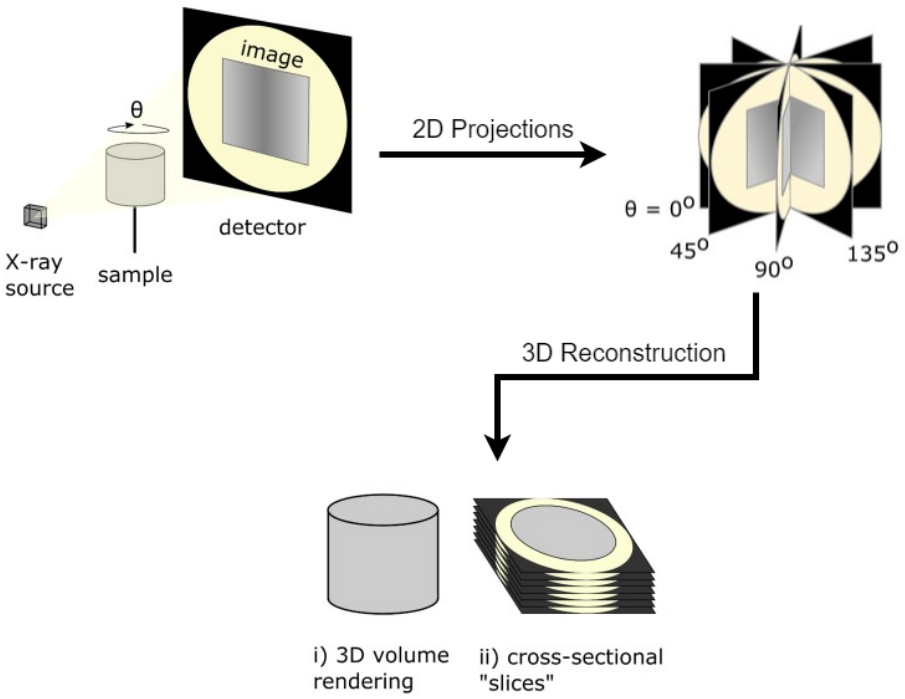


Figure I.5: 3D reconstruction of CT volume workflow. Projection images at different angles θ are digitally reconstructed using reconstruction algorithms in order to generate an internal representation of the specimen. Results can be viewed as a 3D rendered volume (i) or as 2D cross-sections called slices. Source: [16]

the noise in the 2D projections is removed after reconstruction by averaging the projections with the reconstruction algorithm.

- Figure I.6a: Cupping artifact occurs when the material appears darker as we move towards the centre of the image. This effect, also known as "beam hardening artifact", is due to the fact that the X-ray spectrum moves towards higher energy radiation as it passes through the object, while lower energies are preferentially absorbed [20].
- Figure I.6b: Streaks appear at the periphery of the specimen, which receives less energetic radiation. These streaks are due to large differences in thickness through which the X-ray beam passes (e.g. the corners of the specimen).
- Figure I.6c: Ring artifacts are caused by a miscalibration of the detector or by the presence of dead detector elements. If a sensor element has a systematic greyscale difference compared to its neighbours, a line appears in the projections, which is then reconstructed as a ring in the 2D slices.
- Figure I.6d: Feldkamp artifacts are reconstruction artifacts in the 3D volume as a result of using modified approximations of the Feldkamp algorithm.

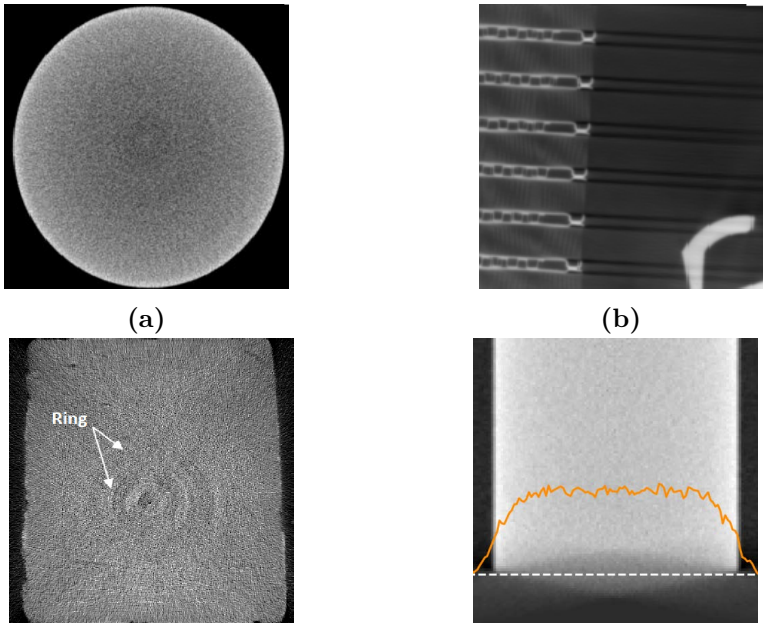


Figure I.6: Different types of industrial CT artifacts: (a) beam hardening, (b) streaking, (c) ring artifact, and (d) Feldkamp artifact.

I.4 Casting Inspection: Tomography vs. Radiography

Once the casting is finished, manufacturers must verify internal shapes, features size and clearances, which is not an easy task when dealing with complex structures. As mentioned earlier, CT provides a non-destructive method to scan the internal volume and look for cracks, voids or inclusions of foreign materials. Compared to radiography and radioscopy, CT provides more contrast information about the specimen, and from different angles. As illustrated in Figure I.7, the output image of radiography inspection is perpendicular to the X-ray beam plane and represents the total integral of μ along the X-ray path through the specimen, i.e. a single image represents the total attenuation along the entire thickness penetrated by the X-ray radiation. On the other hand, CT generates a 3D map representing the local attenuation μ in each small volume element (voxel) of the specimen, making the contrast resolution largely independent of the geometry and the output data more informative about the casting specimen.

Radiography detects defects of various sizes and shapes present along the penetration path, but do not provide information on the depth of the defect, as shown in Figure I.8. CT, on the other hand, provides volumetric data about each defect and gives its true dimensions relative to the specimen thickness. However, the detectability of defects with CT can be affected by many factors, including spatial resolution, contrast resolution and artifacts as stated earlier.

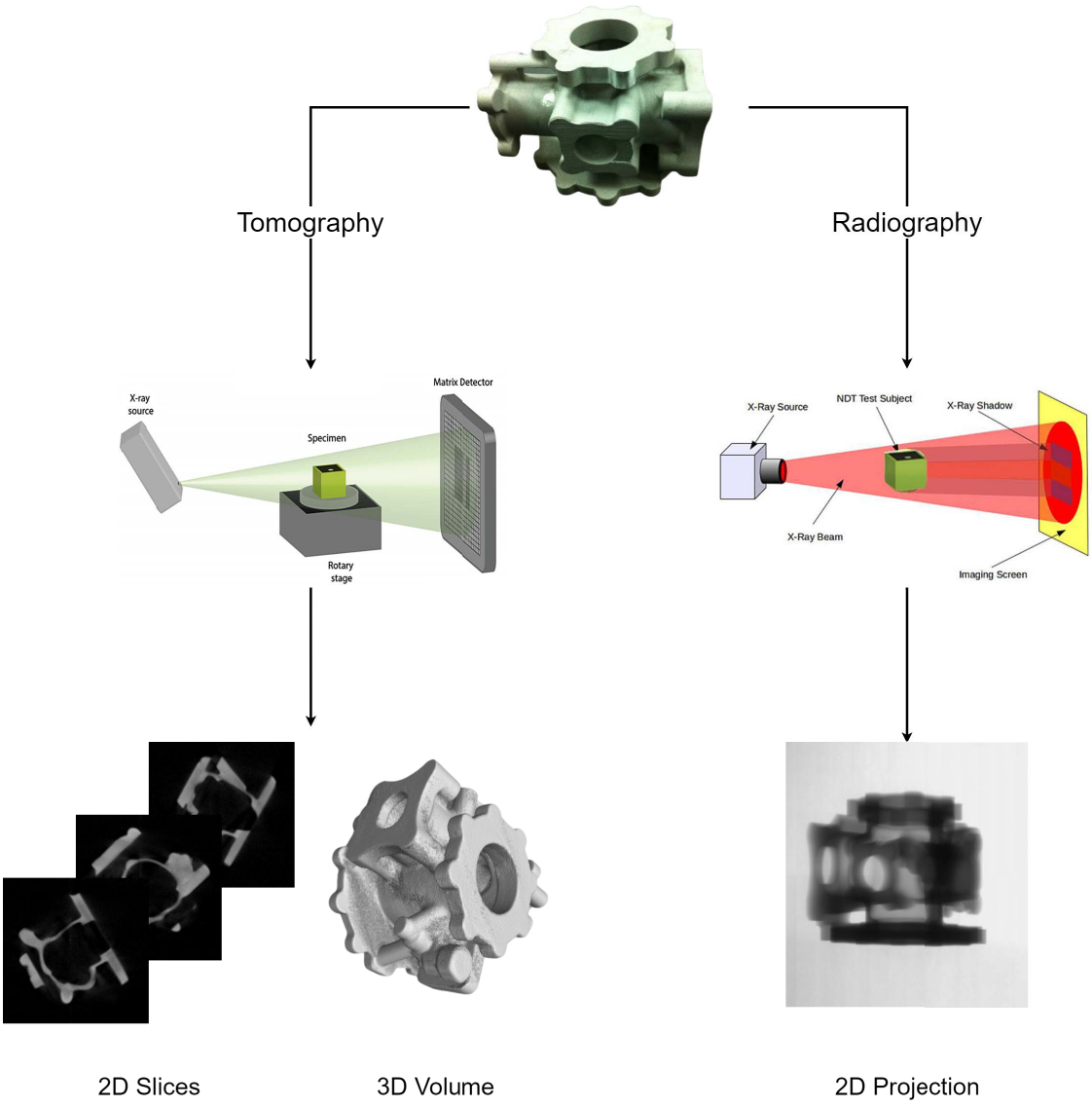


Figure I.7: Inspection of casting with radiography and tomography. Radiography is not able to distinguish between thickness and density changes, but CT provides a clear and quantitative characterization by giving access to the internal volume. Source: [21, 22]

Today, the acceptance of castings tested by radiography is based on ASTM X-ray reference images. There are many standards depending on the alloy, the casting process, the spatial resolution and type of the defect. For aluminium alloys, the standards of reference images are ASTM E155/E505 for radiography with film, and ASTM E2422/E2973 for radiography with digital detectors [23, 24, 25, 26]. A sample of reference radiographs is shown in Figure I.9. By comparing the radiographic projection of a defective zone with these images, a qualified operator can manually assess the criticality level and decide whether or not the casting meets the requirements.

However, this method of comparison is not always reliable for the following reasons:

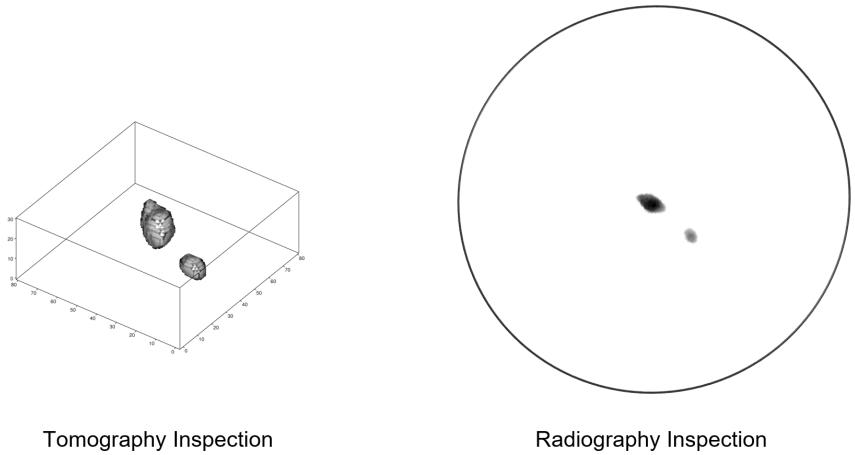


Figure I.8: Defects representation by radiography and tomography. Unlike CT, radiography does not give depth information about the defects.

- The progression of ASTM images is not linear in terms of defect size and density. Some levels are very close to each other, while between two successive levels there is sometimes a very large difference, as is visible between levels 3 and 4 in Figure I.9.
- Visual comparison is subjective and there are often differences between the operators' decisions, or between one client and another.
- Comparison with ASTM images leads to high rejection rates and economic costs [4] because they are not based on quantitative characteristics.
- Castings may have sufficient strength and stiffness for the intended application despite some defects in the body, and subsequent machining and processes sometimes determine the reliability of the specimens despite comparison with ASTM images [27].

Such disadvantages can be avoided with tomography, as it can provide much relevant information for assessing the impact of a defect on the functionality of the casting.

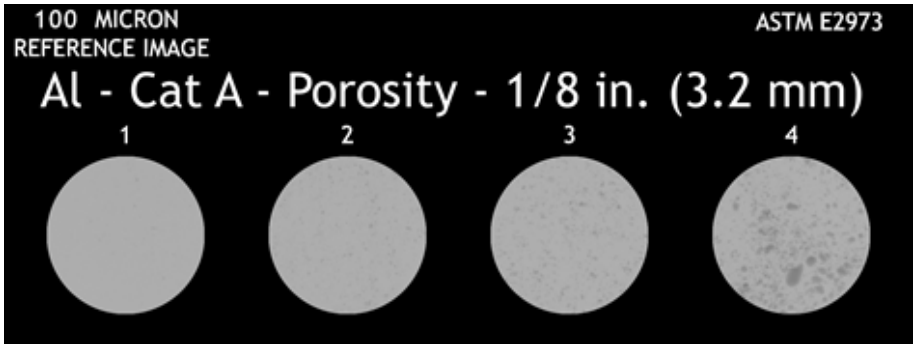


Figure I.9: ASTM E2973 radiographs for interpreting radiographic projects of a high-pressure aluminium casting specimen. If a defective zone shows a high porosity density after inspection with radiography, the level of criticality of the zone is decided after visual comparison with the ASTM images.

In 3D, the defect size and its position are the decisive parameters and not the complete coverage with limited visual criteria. For this reason, foundries using CT to inspect their casting productions are constantly looking for automatic defect detection algorithms that help eliminate false alarms such as noise and artifacts, locate and isolate 3D defects in the volume and investigate their geometrical properties.

I.5 Automatic Defect Detection Challenges

The quality of inspection of a casting production using CT requires a trade-off between the cost of the inspection device, the spatial resolution of the volume, the contrast resolution, and the time to scan and analyse the data, as shown in Figure I.10. Although the detection of small discontinuities requires very high spatial resolution, it is unreasonable to expect such feature after scanning large specimens, as the time and cost of inspection becomes very prohibitive. Manufacturers usually prefer to reduce the cycle time when implementing CT, even if this is at the expense of data quality. As a result, contrast resolution is reduced due to lower exposure and more noise and artifacts are introduced into the reconstructed images.

In addition, CT is still used today as an "in-lab" inspection method because it takes a lot of time to analyse the data manually, which in turn could affect production if CT is used directly on the production line. In fact, it is very time-consuming to assign a human inspector to slice through the volume looking for discontinuities that belong

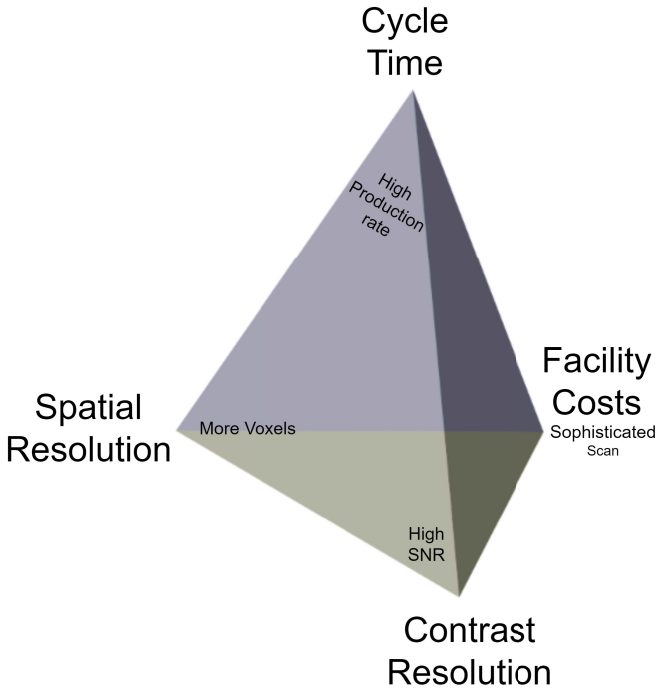


Figure I.10: Industrial CT scan is a compromise between spatial resolution, contrast resolution (image sharpness), components cost and desired cycle time. Source: [28]

to real defects. Therefore, automatic methods for locating and detecting discontinuities are needed to reduce analysis time and enable the future implementation of CT on the production line.

Our work focuses on the development of a new approach to detect discontinuities inside any CT casting volume, to correctly classify these discontinuities into true alarms (defects) or false alarms (artifacts, noise or geometrical features) without user intervention in order to reduce the cycle time. As mentioned earlier, the scope is focused on aluminium alloy castings as the available data for this project was provided by industrial partners working with light alloys. As shown in Figure I.11, our work focuses on post-processing volumetric data of aluminium alloy castings, as the available data has already been reconstructed after scanning different specimens.

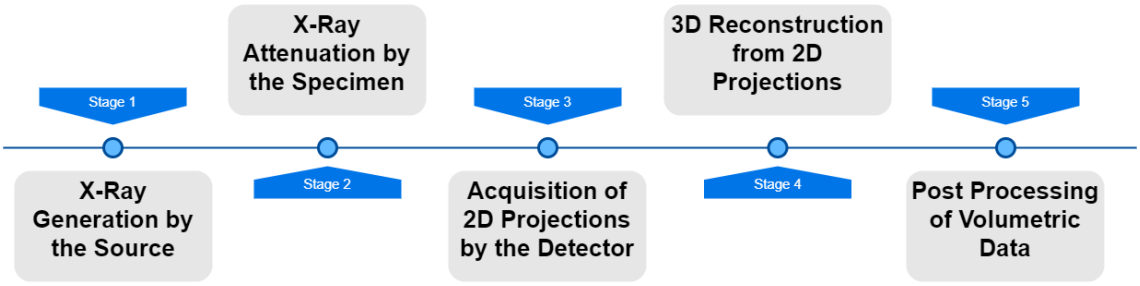


Figure I.11: CT scanning workflow. Our work focuses on the last phase, where the volumes are already reconstructed and ready to be analysed.

I.5.1 Available Data

In this work, we have developed several algorithms for processing CT images of aluminium alloy castings, trained and tested on CT images provided by our industrial partners, namely CTIF, Renault, Groupe SAB, Montupet, Constellium and Eurocast. To avoid conflicts of interest, the contents of these volumes are not presented entirely in this manuscript.

A total of 20 volumes were made available. Each represents a unique specimen scanned under unique conditions and with unique spatial and contrast resolutions. The volumes can be divided into two categories depending on the casting process: high-pressure die casting and gravity die casting.

I.6 Literature Review: Automatic Defect Detection in Industrial CT Images

There is a gap in the literature when it comes to automatic defect detection algorithms inside CT volumes of castings. This can be attributed to the following reasons:

- Compared to 2D radiography, tomography inspection is still new and sometimes unknown to some foundries. And for some, this method is only used in

the design phase of the components, but not on the production line.
- As mentioned earlier, tomography inspection leaves many artifacts in the CT volumes after reconstruction. This problem requires user intervention to ensure that these artifacts are not counted as defects. Therefore, many foundries still consider it a big risk to leave the entire interpretation to automatic defect detection systems.
- There is no publicly available CT data of castings in open-source to conduct research and experiments with defects detection algorithms.

In the field of radiography, there is the GDXray dataset [29], which contains radiographs of welding and aluminium casting tests. Much work has been done on this dataset to automatically detect flaws in 2D X-ray images using image processing techniques [30], traditional machine learning algorithms [31, 32] and deep learning models [33, 34, 35, 36]. On the other hand, there is no open-source CT data, as CT reveals everything about the specimens, which in turn may raise concerns about the manufacturer's intellectual property. Furthermore, the industrial CT data exceeds the terabyte limit, which makes public release difficult.

Our literature review study does not include the work developed based on 2D radiographs of casting, nor medical CT images for the following reasons:

- Because they show the total attenuation along the thickness of the specimen, 2D radiographs do not give a complete representation of the defects, and in some cases defects could be suppressed by the geometrical variations, as shown in section I.4. Conversely, CT offers different semantics, as the depth of each defect is equally represented in the image, as well as artifacts that do not exist in 2D radiographs (cf. section I.3).
- To avoid health hazards during medical CT examination, patients are usually exposed to a lower X-ray dose compared to mechanical specimens, resulting in lower contrast resolution [37]. In addition, anomalies that should be detected by medical CT, such as tumours or cysts, have different semantics than casting defects [38].

In the following section, we discuss work related to automatic defect detection methods used on CT images of castings. These methods or algorithms can be divided into three categories:

1. Algorithms based on image-processing techniques, such as region growing and image subtraction.
2. Traditional Machine learning, which classifies suspicious discontinuities after calculating handcrafted features.
3. Deep learning models, where the raw greyscale images are used directly as inputs before applying deep segmentation or region classification.

I.6.1 Image Processing Techniques

Hadwiger et al. [39] proposed an interactive approach to exploring industrial CT volumes. They performed region growing segmentation by manually selecting a voxel seed and repeating the process until the features merge, as shown in Figure I.12. The algorithm checks all neighbouring voxels in 3D that have a greyscale value within a certain range and make up the same entity (defect or artifact). A 3D transfer function classifies each entity based on volumetric size and density in order to decide whether it is a true alarm (defects) or a false alarm (artifact or noise). Although their method is helpful in segmenting and identifying the type of discontinuities, it requires manual tuning of various parameters, such as target ranges of density and size alongside the selection of seed points for each volume, which increases the processing time.

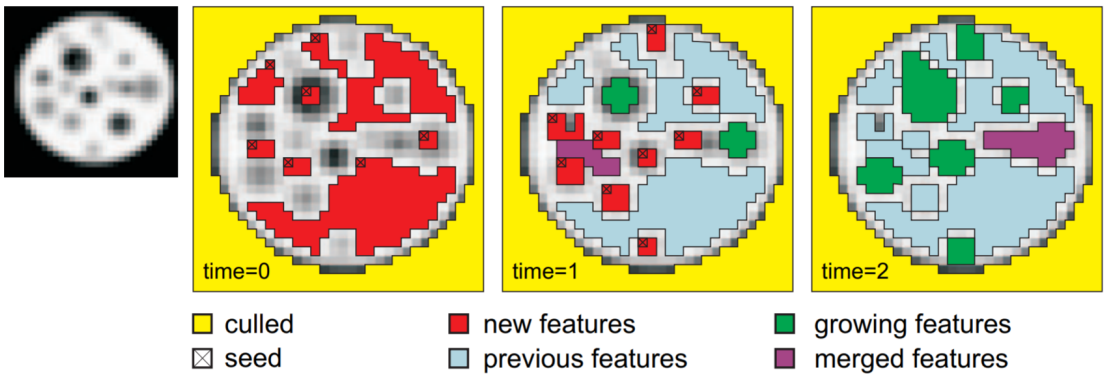


Figure I.12: In each pass, new features can be created and already existing features may grow and merge. Background voxels can be suppressed in order to improve pre-processing performance. Source: [39]

Another common approach to detecting anomalies in industrial CT volumes is the defect-free reference approach., where the CT slices of the specimen are compared with CT slices of a defect-free volume. The latter can be obtained by scanning a defect-free specimen of the same shape, with the same tomograph and under the same scanning conditions [40]. Consequently, the volume of this "golden" part has the same artifacts as the examined specimen. By subtracting the slices of both volumes, geometrical structures and artifacts are removed from the difference images, and defects are highlighted. The pipeline of this approach comprises five steps:

1. CT slices enhancement by reducing noise with Wiener filter [41], for example, followed by histogram equalization to account for the dynamic range of the greyscale values distribution.
2. Compensation for deviations in orientation and translation between the defective and defect-free specimens. An optimization algorithm estimates the parameters of an affine transformation based on the divergence of greyscale values between the two volumes.

3. Absolute subtraction operation takes place in order to obtain the difference image.
4. The difference image is segmented by thresholding, where the threshold is chosen to maximize the sum of the entropy values of the foreground and background entropy values.
5. Morphological opening operation is applied in order to filter small indications and refine the segmentation results.

This method gives good results in detecting defects inside the CT volumes, including cold fills. Theoretically, it can be applied to the inspection of a mass production line, i.e. one golden defect-free specimen is required for all specimens to be inspected. However, it is an impossible task to place all specimens in the same position and at the same angle as the golden specimen, and the compensation for deviations is never perfect. Gondrom-Linke [42] suggested scanning the same specimen a hundred times and averaging all volumes to obtain a defect-free version, and Rieter et al. [43] suggested the use of robots to ensure accurate placement, but both solutions would slow down the production line. In addition, Reiter et al. suggested using simulation algorithms to create a defect-free volume. However, this simulated replica should include all physical effects, including detector response, to account for all artifacts, and it requires a registration step between the simulated replica and the real specimen.

Defect-free reference algorithms are very prone to false alarms in the case of small deviations, as absolute subtraction could create additional discontinuities in the difference image. Furthermore, they are 100% dependent on the casting geometry. Every time the geometry of the casting is changed, or in the case of a new casting, the process has to be repeated all over, which can be tedious and expensive.

Proposition: in chapter I, we present two segmentation algorithms based on image processing techniques to localize discontinuities inside the casting, which are independent of the shape of the specimen. We show their segmentation results and discuss their advantages and limitations.

I.6.2 Traditional Machine Learning

The second category includes traditional machine learning algorithms such as support vector machine (SVM) or random forest that take handcrafted features as inputs. The latter are calculated after isolating suspicious discontinuities in 3D.

Zhao et al. [44] have proposed a very interesting approach to detect casting defects in 3D CT images of aluminium castings. Their pipeline uses random forest (RF) to classify discontinuities into defects or artefacts based on their greyscale values and shape curvature. The pipeline consists of 4 main steps: locating and selecting candidate discontinuities, isolating these discontinuities, extracting features and classification. Each step can be summarised as follows:

1. Candidate selection, with the help of morphological closing followed by a 3D convolution with Gabor filters to remove streaks artifacts [45]. This step can be assimilated to a thresholding where low response voxels are removed, which reduces the number of discontinuities to be processed.
2. In order to isolate the selected discontinuities, their zones are locally subtracted from a defect-free volume, as introduced in the previous section. However, the method of obtaining the defect-free volume was not elaborated in the paper.
3. 25 texture features and 4 geometrical features are calculated from each discontinuity based on the greyscale level and the shape curvature. Texture features are calculated using the "grey-level co-occurrence matrix" (GLCM) [46], which describes the frequency of occurrence of greyscale values in relation to other values (voxel-pairs occurrence) at a given distance and direction, as well as the uniformity of the greyscale distribution, contrast, entropy and standard deviation. Curvature-based features are calculated by computing the voxel-wise derivatives to find the principal curvatures [47]. Based on the latter, a "shape descriptor" is calculated for each discontinuity to assign a shape category and calculate mean, standard deviation and entropy.
4. Random forest model is used to classify each discontinuity into defect or normal zone based on the extracted features.

Without giving information on the training process, the proposed method achieved a high recognition rate of 94% on 31 CT volumes with 49 porosities and shrinkage cavities. However, this method has some limitations in practice: (1) it requires a defect-free volume to isolate the candidates for discontinuities in future inferences, which is time-consuming as explained in the previous section; (2) most greyscale features do not contribute equally to the RF decision, and their computation has a disadvantage for processing time; and (3) the shape descriptor is only sensitive to roundish gas pores.

Osman, Kaftandjian et al. proposed a data fusion-based classifier (DFC) to inspect CT images of aluminium castings [48]. This classifier was originally developed for 2D radioscopy data [49] and then extended to 3D CT images of castings. Based on 30 features calculated from each discontinuity, DFC decides whether it is a real defect or a false alarm (artifact or noise). The model was trained on a highly imbalanced dataset with 26 defects and 200 false alarms and validated on 18 defects and 198 false alarms. The DFC was then compared with the support vector machine (SVM) in terms of performance.

Performance was quantified using the classification rate per class and the area under the receiver operating characteristic (ROC) curve [50] on a validation dataset. The area under ROC allowed the authors to determine which features had the greatest impact on DFC performance, reducing the number of inputs to the model. Combining only two features resulted in the same recognition performance with DFC compared to SVM that required eleven features. However, SVM performed better in classifying false indications, i.e. noise and reconstruction artifacts.

Proposition: to decide whether a zone is defective or not, we avoided traditional machine learning algorithms because selecting the right features is a very time-consuming task and strongly depends on the contrast resolution and the shape of the defects in the training dataset. However, in chapter VI we tried to categorize 3D defects into porosities or shrinkage cavities using SVM based on geometrical features such as sphericity, elongation and compactness.

I.6.3 Deep Learning

Deep learning-based algorithms, which have been at the forefront of computer vision in recent years [51], make up the third category of automatic defect detection techniques. Image classification is the best known deep learning application that has reached the state-of-the-art [52, 53]. Deep learning classifiers have reached a stage where they can often outperform humans in image labelling tests, according to [54]. The core of this technique is to use raw images as inputs, and instead of manually extracting handcrafted features, a convolutional neural network learns how to automatically extract features in a hierarchical fashion.

Deep learning has been explored to reduce artifacts in CT data, either before or after reconstructing the CT volume. Deep scatter estimation [55] approximates the result of Monte-Carlo function, which in turn removes the scatter artifacts from 2D projections before 3D reconstruction. On the other hand, in medical imaging where low X-ray dose is used, removing noise and artifacts after reconstructing CT volumes with low contrast resolution is an active research area [56, 57, 58]. However, when applied to industrial CT volumes to improve data quality, there is a risk of removing small porosities in homogeneous regions.

In our research area, two leading software companies in the field of industrial CT data analysis and visualization, $Zeiss^{TM}$ and $VolumeGraphics^{TM}$, are constantly exploring the possibilities of deep learning to detect defects in CT casting data. In 2020, Schlotterbeck et al. [59] in collaboration with $Zeiss^{TM}$ have *presumably* developed an approach to detect defects inside CT volumes of aluminium castings using the following pipeline:

1. Anomaly detection using SVM model, which was trained on difference volumes after subtracting many specimens from their defect-free versions. The latter are each calculated by averaging many 3D volumes of the same specimen.
2. Deep learning by transfer-learning in order to classify each anomaly detected by SVM into porosities or voids, or even geometrical features such as core breaks or wall shifts.
3. To decide if each anomaly is a defect or a geometrical feature, the authors used two models: 3D U-Net for deep segmentation [60], or random forest after calculating handcrafted features. The majority votes of the decision trees are used to determine the type of the anomaly.

The authors did not give any information about the training data, the learning

process, nor the training results. The article focused only on the processing time. Moreover, SVM will always require a defect-free volume before detecting anomalies in a new volume, which is a time-consuming task as explained earlier. In addition, applying 3D segmentation with the convolutional neural network U-Net is not efficient because the model does not perform well on anisotropic volumes where voxels do not have the same size along the three axes [61].

In 2021, Fuchs et Al. [62] in collaboration with $VolumeGraphics^{TM}$ worked on an approach to detect defects inside CT scans of aluminium casting using deep segmentation. The authors developed a CT mesh simulation pipeline that generates a huge amount of synthetic training data with labelled discontinuities. Casting defects (gas holes, cavities, cracks) and artifacts can both be generated by this pipeline to train a modified version of U-Net (state-of-the-art deep segmentation model) to binarize CT slices. The model achieved an accuracy of 65% when tested on real low-quality CT data, outperforming a reference-based method and a random forest algorithm.

Synthetic data is useful to train deep learning models when real data is not available. However, using a large amount of fake data, not to mention unrealistic defects, to train DL models inevitably leads to overfitting, at least towards the real data on which the simulation pipeline was designed.

Proposition: In chapter III, we used the state-of-the-art U-Net model to segment CT volumes by binarizing each slice individually and then stacking all slices to form a binary CT volume. In chapter IV, we trained a new convolutional network from ground up, and compared its performance with state-of-the-art models in classifying discontinuities in 2D CT slices into true alarms (defects) or false alarms (artifacts or noise). All these models were trained with different CT slices coming from different volumes scanned by different tomography machines (various spatial resolutions and X-ray imaging conditions). This diversity is a very important factor as it helps the models to generalize better and increases their future reliability in detecting defects inside new CT inferences. Finally, in chapter VI, we trained a siamese convolutional neural network to predict the type of 3D casting defects.

I.7 CNN Under the Loop

A deep learning model arranges a large number of linear classifiers, called neurons, into a large number of successive layers. The weights (and bias terms) of these neurons are the trainable parameters of the network, and its output depends on the training task, e.g. a label map in the case of segmentation or membership probabilities in the case of classification. Since we are working with CT images, a multilayer perceptron (MLP) or dense feed-forward neural networks must be avoided. These neural networks cannot learn how to correctly detect defects in the image since they always look at fixed positions. For this reason, convolutional neural networks (CNNs) must be used because they apply convolutional operations to the input looking for suspicious discontinuities, regardless of their positions on the image. The most common CNN layers that we have used to make up the models of the following chapters are described below.

I.7.1 Convolutional Layers

In convolutional neural networks, the first layer is always a convolutional layer that acts as the first feature extractor. In this layer, as well as in subsequent convolutional layers, several convolutional operations are performed on the input by sliding some filters (kernels) of size $m \times m$ to search for texture information. The output of the convolutional layers is a set of feature maps, which are in fact the cross-correlation product (despite the term used) between each filter and the input. These feature maps provide information about the semantics of the image, learned

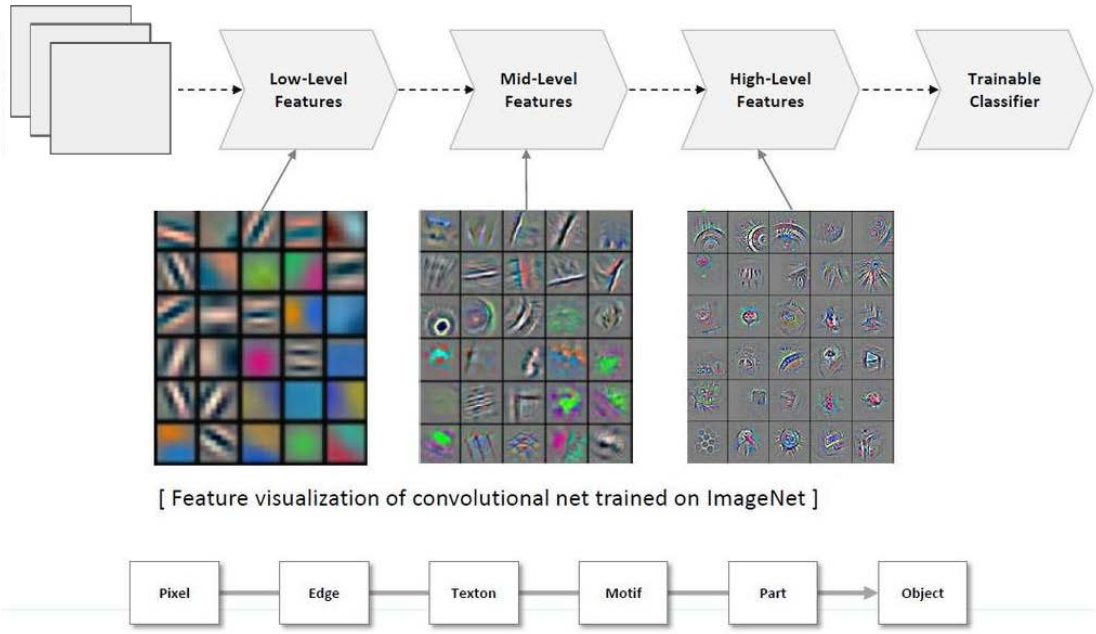


Figure I.13: Hierarchical semantics learned by the consecutive convolutional layers of a CNN from the pixel (or voxel) to the very object. Source: [63]

in a hierarchical fashion throughout the architecture, as shown in Figure I.13. For a 2D image I and a 2D filter or kernel K, the feature map G is obtained as follows:

$$G(i,j) = (I \circledast K)(i,j) = \sum_u \sum_v I(i+u, j+v) \cdot K(u,v) \tag{I.3}$$

I.7.2 Activation Functions

One of the most important components of a neural network is the activation function, which is a mathematical equation applied to the output of a previous layer (e.g. a convolutional layer) [64]. Choosing the right activation function is crucial because it can cause a model to converge faster and achieve high performance, and sometimes prevent it from converging at all because of saturation. Figure I.14 illustrates the most commonly used activation functions in deep neural networks. Each convolutional layer in the network is coupled with an activation function that is applied to each feature map outputted by the layer. The use of an activation function improves the learning process by performing a kind of non-linearity on the weights of the network.

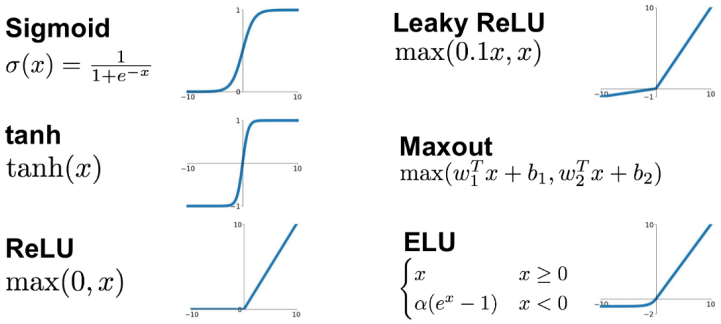


Figure I.14: Common activation functions used in neural networks. Each one is a mathematical equation that can be applied to the output coming from the previous layer.

I.7.3 Max-Pooling Layers

After applying an activation function, a convolutional layer is usually followed by a pooling layer. The main advantage of the latter is to reduce the computational cost by reducing the size of the feature maps. Figure I.15 shows the reduction of the dimensionality of the feature maps in the case of max-pooling, which uses the maximum value in a kernel of size $n \times n$ to generate a new feature map. Unlike convolutional layers, pooling layers are not followed by an activation function.

I.7.4 Fully-Connected Layers

Fully-connected layers are the most commonly used layers in NN and they are essential components in classification models. Each fully-connected (dense) layer is

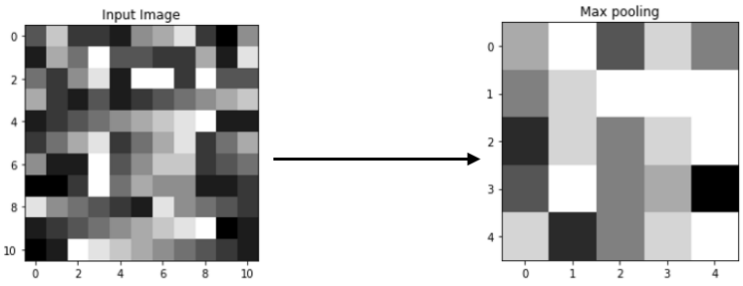


Figure I.15: Resolution reduction of a greyscale image by applying max-pooling, which holds the maximum value in each kernel to generate a new image.

interconnected with the previous layer and the next layer as shown in Figure I.16. This means that each neuron in this layer receives information from all neurons of the previous layer, and sends information to all neurons of the following layer. It performs matrix-vector multiplication, where the vector is the input and the matrix values are the weights of the layer that needs to be trained in order to map the underlying relationship between the input and the output. In the case of classification, the final output is a vector with k dimension that represents the membership distribution of the input over the k classification classes.

I.7.5 Transpose Convolutional Layers

As mentioned above, CNNs use convolutional layers followed by a pooling layer to learn the contextual information (semantics) in the input image by generating feature maps. In a segmentation task, where the final output is a segmented version of the input, the model needs to upsample these feature maps to undo the down-

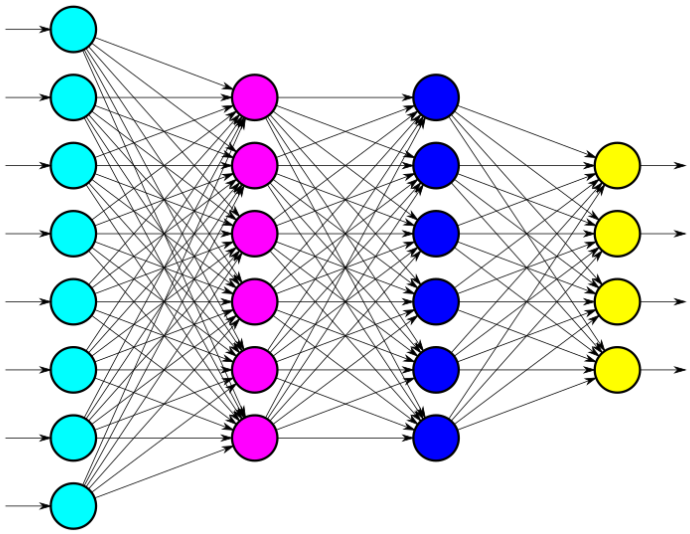


Figure I.16: A series of deeply interconnected dense layers. The final output of the dense block is a membership distribution over the classes.

sampling by the pooling layer and restore the feature maps to the original size of the input. One way to do this is to interpolate the individual feature vectors to increase their resolution. However, it is almost impossible to find an interpolation that works well with every input image. An alternative solution is the transposed convolutional layer (incorrectly called deconvolutional layer), which works like an upsampling layer [65, 66].

When training a segmentation model, the kernels (filters) of this layer learn how to correctly upsample the feature vectors while preserving the contextual information. The simplest way to think of the process is to calculate the output shape of a direct convolution and then invert the input and output shape. The layer then learns a set of weights that can be used to reconstruct the inputs. Figure I.17 shows the most common configuration of upsampling with a transposed convolutional layer, with a 2×2 kernel, a stride of 2 and a padding of 1.

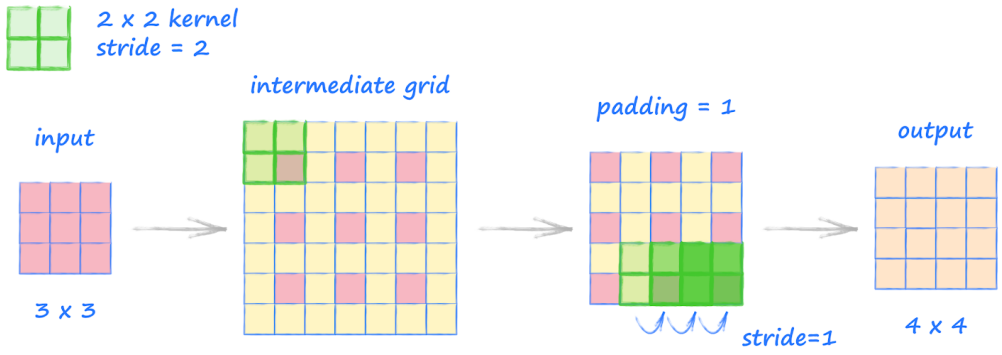


Figure I.17: A schematic diagram of convolutional transpose operation of a small input of 3×3. An intermediate grid is created by stretching the input, followed by padding removal before applying a convolution with one of the layer's kernels in order to find the right output size. Source: [67]

I.8 Deep Learning Models Challenge: Overfitting & Underfitting

The main goal in training a neural network model is to ensure that it can perform well on new unseen data points (in our case, segmentation or classification). This ability to process new inputs that the model has not been exposed to during training is called generalization. To measure this, the model tests itself at the end of each training iteration (**epoch**) on a vector of data called the validation set. The error measure on the training set is called **training loss**, and that for the other set is called **validation or generalization loss**. The latter is defined as the value of the error on new inputs. To ensure generalization, we need to make the training loss as small as possible, as well as the gap between the training loss and the validation loss. The optimal state is shown in Figure I.18.

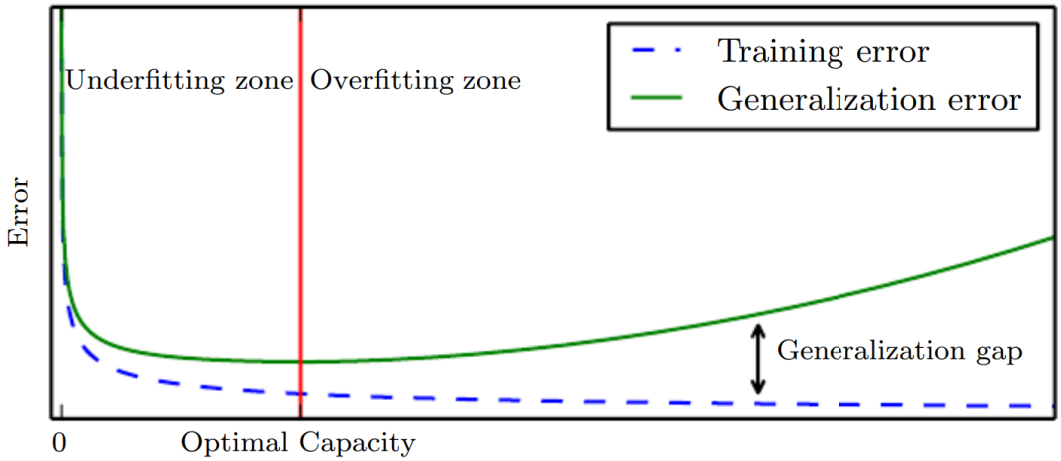


Figure I.18: After training, overfitting or underfitting can be spotted by looking at the training and validation losses. The model is well trained when it reaches the optimal capacity: both losses are close to zero, and the gap between them is relatively small. Source: [68]

These requirements are not met if one of two phenomena occurs: (1) underfitting, when the model is too shallow to capture the underlying relationship between the input and output variables, resulting in a high training loss; or (2) overfitting, when the model has been overtrained on the particularities of the training dataset so that it cannot perform well on new unseen data that does not exhibit the same noise trends, resulting in a large gap between the training and validation losses. Optimal capacity can be achieved by increasing the size of the dataset, applying regularization techniques or changing the training conditions, as explained in the following chapters.

I.9 Implementation Setup

The algorithms in the following chapters were developed using Python 3.6, Matlab and C++17. Training deep learning models was performed using Keras [69] (Tensorflow-gpu 2.1.0 backend) as Python deep learning API. To ensure fast training, CuDNN 7.6 —a GPU-accelerated library for deep neural networks— was used with this framework. The workstation is equipped with an Nvidia P3200 GPU (6GB memory) and an Intel Core i7-8850H @2.60GHz CPU.

Conclusion

CT is a powerful non-destructive testing method for industrial production inspection. It provides a large amount of data with internal and external details about the specimens. However, as it leaves artifacts inside the volumes, the foundry industries are always looking for algorithms that can automatically isolate defects in the CT volumes and ignore artifacts without user intervention. Furthermore, this

detection algorithm should be able to process any CT volume regardless of its contrast and spatial resolution. In the literature, algorithms intended for this endeavour fall into 3 categories: image processing techniques, traditional machine learning and deep learning algorithms. The available algorithms today are either semi-automatic approaches that can guide a human examiner, or fully automatic approaches that rely on a defect-free reference specimen.

In the following, we present two segmentation algorithms that we have used to binarize CT data, either in a slice-by-slice fashion or by using the 3D volume directly as input. These algorithms were used to create ground truth data for training and validation of deep learning models, presented in the subsequent chapters.

Chapter II

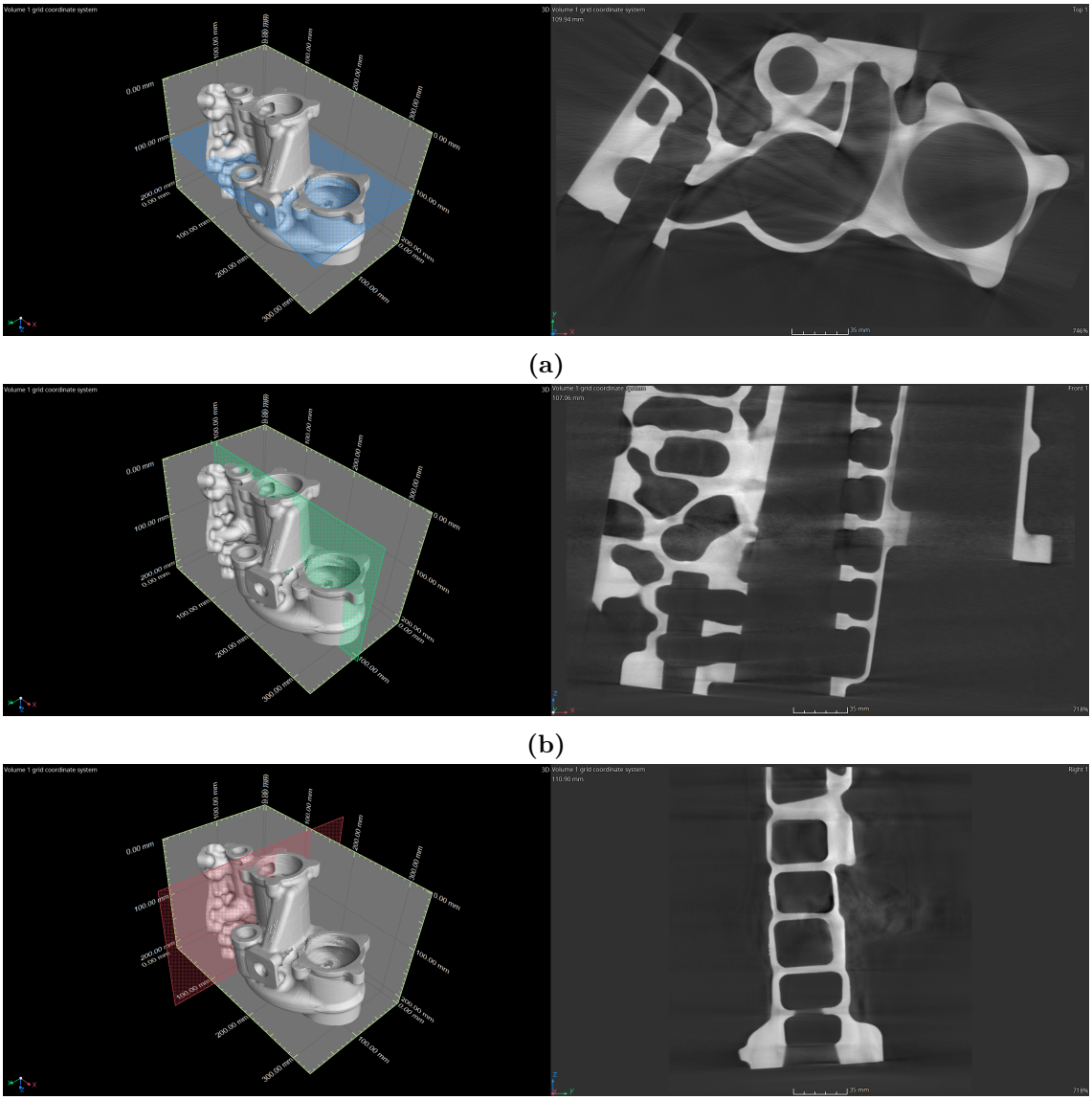
2D & 3D Image Processing of Industrial CT Data

In this chapter, we present two algorithms for locating defects inside CT images. Two edge-based segmentation algorithms are presented, one takes 2D CT slices as input and the other takes directly a 3D CT volume. The output of each algorithm is a binary version of the input in which defective zones are highlighted. These algorithms need to be tuned manually, which hinders their use as automatic inspection tools. They are used to generate ground truth images in order to train or verify the efficiency of future methods.

II.1 CT Volumes Analysis

After inspecting a specimen with tomography, a mathematical process generates the 3D volume of the specimen from X-ray projections taken at different angles around it [70] (cf. Figure I.5). This volume is a 3D grid of voxels that can be described as a cartography of the local attenuation of X-rays by the specimen. The denser the material, the higher the attenuation and the brighter the voxels. Conversely, if the region is defective (lack of material), the voxels are darker. Such volume can be sliced along any direction, generating a set of CT slices, as illustrated

in Figure II.1. This volume was provided by one of our partners: it is a greyscale volume of a gravity die casting specimen, with a size of $1118 \times 776 \times 1349$, bitmap of 16 bits per voxel, and a spatial resolution of $320\mu m$.



(a)

(b)

(c) Right view

Figure II.1: A 3D CT volume can be sliced along any direction to generate 2D slices. The highlighted slices within the 3D volume (left column) are shown in the right column: (a) a slice along the top view; (b) a slice along the front view; and (c) a slice along the right view.

Since this volume has 16 bits per voxel (8 bits for others), each can have a greyscale value ranging from 0 to $2^{16} - 1 = 65536$. In Figure II.2, we can see the intensity histograms of 2 slices, as well as the histogram of the whole volume, which shows the distribution of voxels with respect to their greyscale values. Each histogram is model, i.e. has two obvious relative modes, or data peaks. This means that there

are 2 majorly represented classes of voxels: (1) the hump on the left, representing the voxels with lower greyscale values, i.e. the empty zone outside the specimen; and (2) the hump on the right, representing the body of the specimen, which has bright contrast due to the high x-ray absorption or attenuation. On the other hand, any voxel that belongs to a *lack of material* or an artifact lies between these two humps. This shows that the classes of voxels, defective vs. normal, are very imbalanced as the defects make up a very small proportion (that lies between the two humps) compared to the total number of voxels.

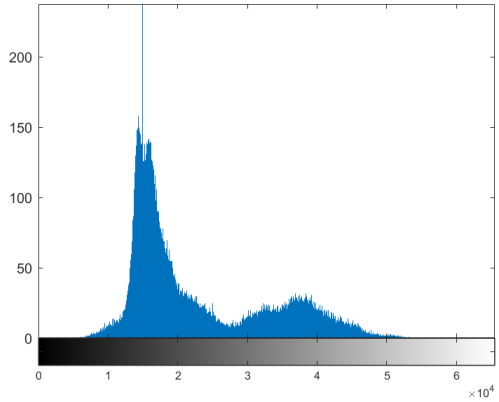
II.1.1 True Alarms and False Alarms

Since the castings are made in a single operation, the presence of defects inside the specimen is inevitable. In fact, there is no such thing as a flawless casting and the real endeavour is to know the impact of the defects on the lifetime of the castings and how to control them [71]. When a casting is inspected by tomography, we can find inside its CT volume some discontinuities that can belong to real defects (True Alarms), as in Figure II.3, but also other discontinuities that can represent artifacts or noise (False Alarms), as in Figure II.4. Unlike defects that arise during the casting process and physically exist, artifacts are inconsistencies between the CT images after reconstruction and the real material density and geometry of the specimen, i.e. they have no physical existence and yet unavoidable [72].

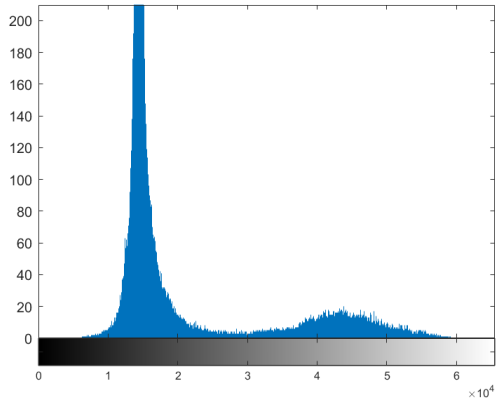
Most factories using this NDT method employ a human inspector to distinguish the potential defects from other discontinuities since CT is filled with artifacts. However, this human inspection leads to dubious quality control because visual inspection suffers from relative bias and eye fatigue. And from a methodological point of view, the number of inspectors, the zone to be inspected and the inspection interval have to be planned, which is very time-consuming [73]. Therefore, foundries are always looking for an automatic means to assist the inspection process. As explained in the last chapter, such methods have been extensively studied in various applications over the last decades, and computer vision and image processing algorithms are widely used for such tasks [74]. To achieve the goal of detecting defects in CT volumes, we present two algorithms based on image processing techniques that require tweaking some parameters. The aim of these algorithms is to enhance the display and binarize the volume, i.e. isolate the defects from the background and generate ground-truth binary volumes from the originals. Each algorithm processes the CT in a different way: one processes the entire volume directly in 3D, and the other proceeds more slowly and processes each slice separately in 2D.

II.2 Image Segmentation Techniques

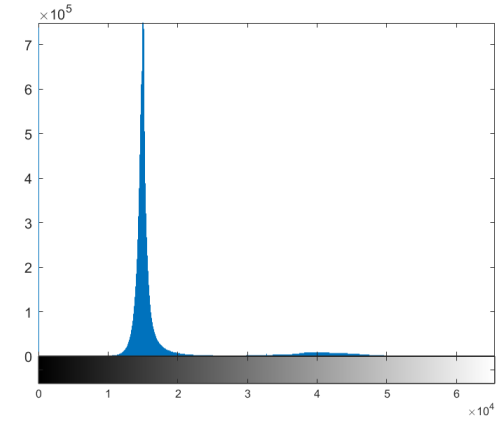
Image segmentation is a process that partitions an image I into n non-intersecting subregions I_{i}, such that $I = \bigcup_{i=1}^{n} I_i$ and each region I_{i} is homogeneous. Image segmentation algorithms can be categorized into two classes based on the greyscale values of the voxels as illustrated in Figure II.5:



(a) Slice №150/1349



(b) Slice №750/1349



(c) Entire 3D volume

Figure II.2: 65536 bins histograms showing the greyscale value distribution in CT images: (a-b) histograms of some 2D slices along the frontal direction, plotting the number of voxels for each greyscale value; (c) histogram of the 3D volume. The large hump represents the voxels belonging to the empty zone outside the specimen and the smaller one represents the normal regions filled with material. The voxels of the defects and artefacts lie in between.

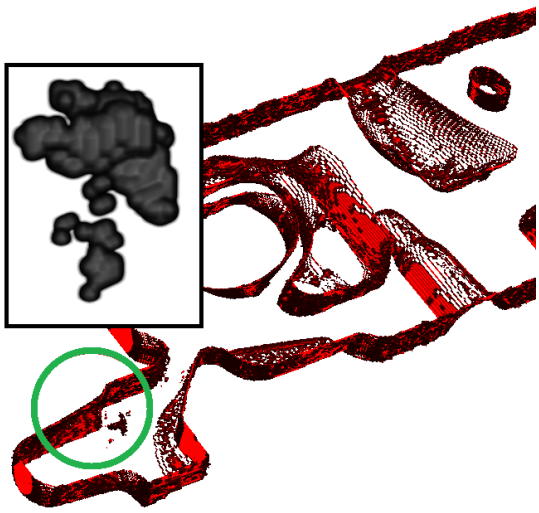


Figure II.3: Shrinkage cavities inside a CT volume after applying a segmentation algorithm that separates discontinuities from the background.

- Discontinuity-based: an image is partitioned based on the boundaries between two regions, which represent abrupt changes in greyscale value.
- Similarity-based: an image is partitioned into regions by grouping neighbour voxels that have similar values according to a set of predefined criteria.

In recent decades, many segmentation algorithms have been developed that fall into these categories [75]. In our two approaches, we have combined two algorithms that come from different categories:

- Edge-based segmentation, which is the most common method used in the discontinuity-based category, where an edge operator is applied to the image: depending on the output of the edge operator, a voxel is classified as an edge if it represents an abrupt greyscale variation; if not, it is assigned to the background class.

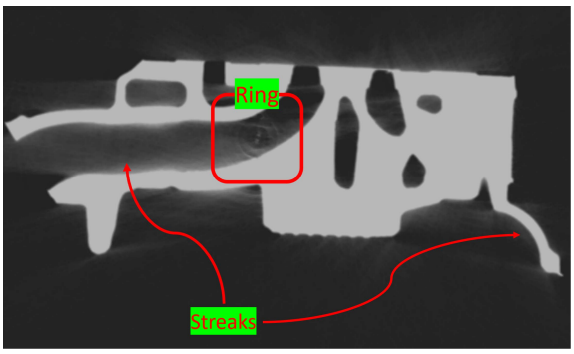


Figure II.4: CT slice from our database that contains ring and streaks artifacts.

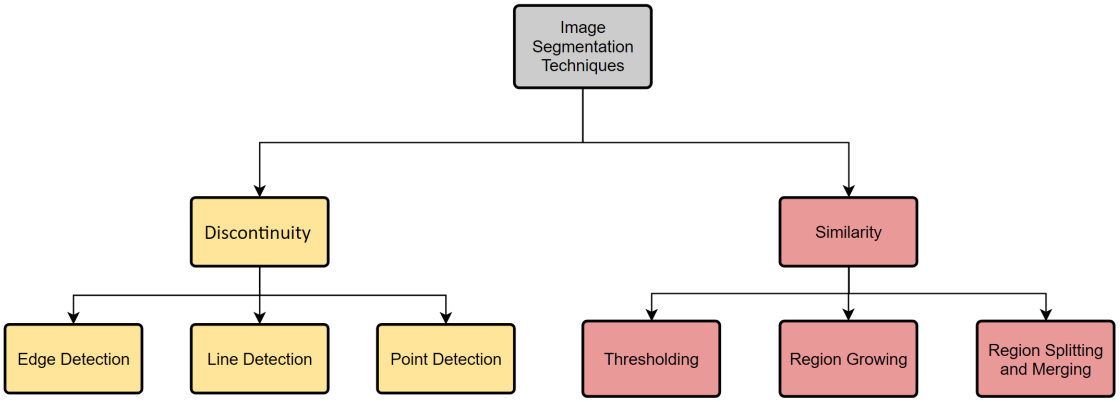


Figure II.5: Image segmentation techniques categories based on the discontinuity or similarity of regions in the image.

- And thresholding, which is a similarity-based algorithm that assigns voxels to classes based on intensity ranges. In global thresholding, only one threshold is selected for the entire image, while in adaptive thresholding, local thresholds are selected for each region of voxels.

II.2.1 Edge-Based Segmentation

When we move from one region to another, for example from the body of the specimen to the empty zone outside, or from a defective zone to a normal zone, the greyscale value changes. This discontinuity belongs to the boundary of the specimen or the boundary of the defect, also called edges. Edge-based segmentation techniques locate these edges by applying an edge detection operator [76]. Edge detection methods require high image quality, which is not always the case with CT images (cf. section I.2). Photonic and electronic noise, as well as the blur created by the limitations of the focusing mechanism of the source and the spatial resolution of the detector, reduce the SNR of the images. This gives the edges a blurred noisy ramp profiles as illustrated in Figure II.6, and make the edge detection more difficult. On the other hand, edge operators have a tendency of amplifying the noise in the image even further [77]. For this reason, a prior denoising of the input must be considered before applying an edge operator [78]. The resulting image of the latter, called edge map, is not the final segmented image. Additional steps are required in order to reduce the number of unnecessary segments in the edge map (global thresholding in our case), followed by morphological operations to fill the edge-based segmented indications. In summary, 3 steps are required in order to apply edge-based segmentation:

1. Image filtering, or denoising, in order to reduce the noise in the image.
2. Image differentiation with an edge operator in order to detect potential edge-points in the denoised image.
3. Edge localization, where only the points that belong to a real edge, such as specimen boundary or defect zone boundary, are kept in the output image.

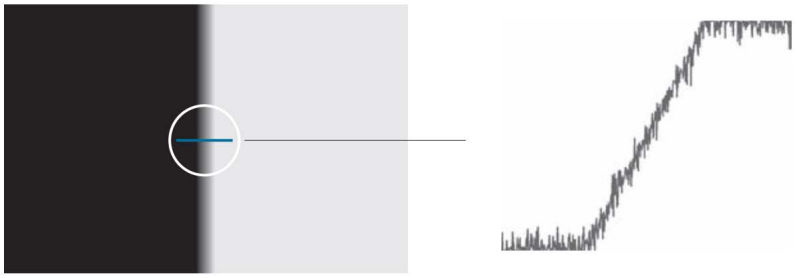


Figure II.6: Two regions separated by a blurred edge in a digital image. Source: [76]

II.3 2D Segmentation Algorithm

The 2D segmentation algorithm takes a tomographic slice as input and returns a binary image where defects are isolated from the background. The pipeline illustrated in Figure II.7 consists of three main steps as mentioned in the last section, preceded by preprocessing.

II.3.1 Preprocessing

As seen in Figure II.2, CT slices have bimodal histograms where the background and the specimen have two separate humps. An automatic thresholding with Otsu can binarize the image and separate the specimen from the background. The preprocessing block consists of removing noise and artifacts (greyscale values) in the empty zone outside the specimen by multiplying the image by its Otsu binary version:

a) Before applying Otsu's thresholding, the image is sharpened in order to enhance the borders of the specimen. The output image, also called the gradient ∇I, is obtained by convolving the image I with a 3×3 sharpening kernel as follows:

$$\nabla I(x, y) = I(x, y) \circledast \begin{pmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{pmatrix} \tag{II.1}$$

For each 3-by-3 block of voxels in the slice, we multiply each voxel by the corresponding entry of the kernel and then take the sum. This sum becomes a new voxel in the output gradient, which eventually represents a sharpened version of the original image.

b) The second step consists of segmenting this gradient using thresholding [79]. The output is a binary image, where the voxels that belong to the specimen are set to white, while the rest of the image, i.e. the background, to black. We used Otsu's thresholding algorithm [80], which selects the threshold by minimizing the within-class variance of two majorly represented groups of voxels.

c) The third step is an emphasizing operation. By multiplying the binary slice with its original version (before sharpening), we obtain a new image where the specimen is highlighted while the background and its artifacts are concealed.

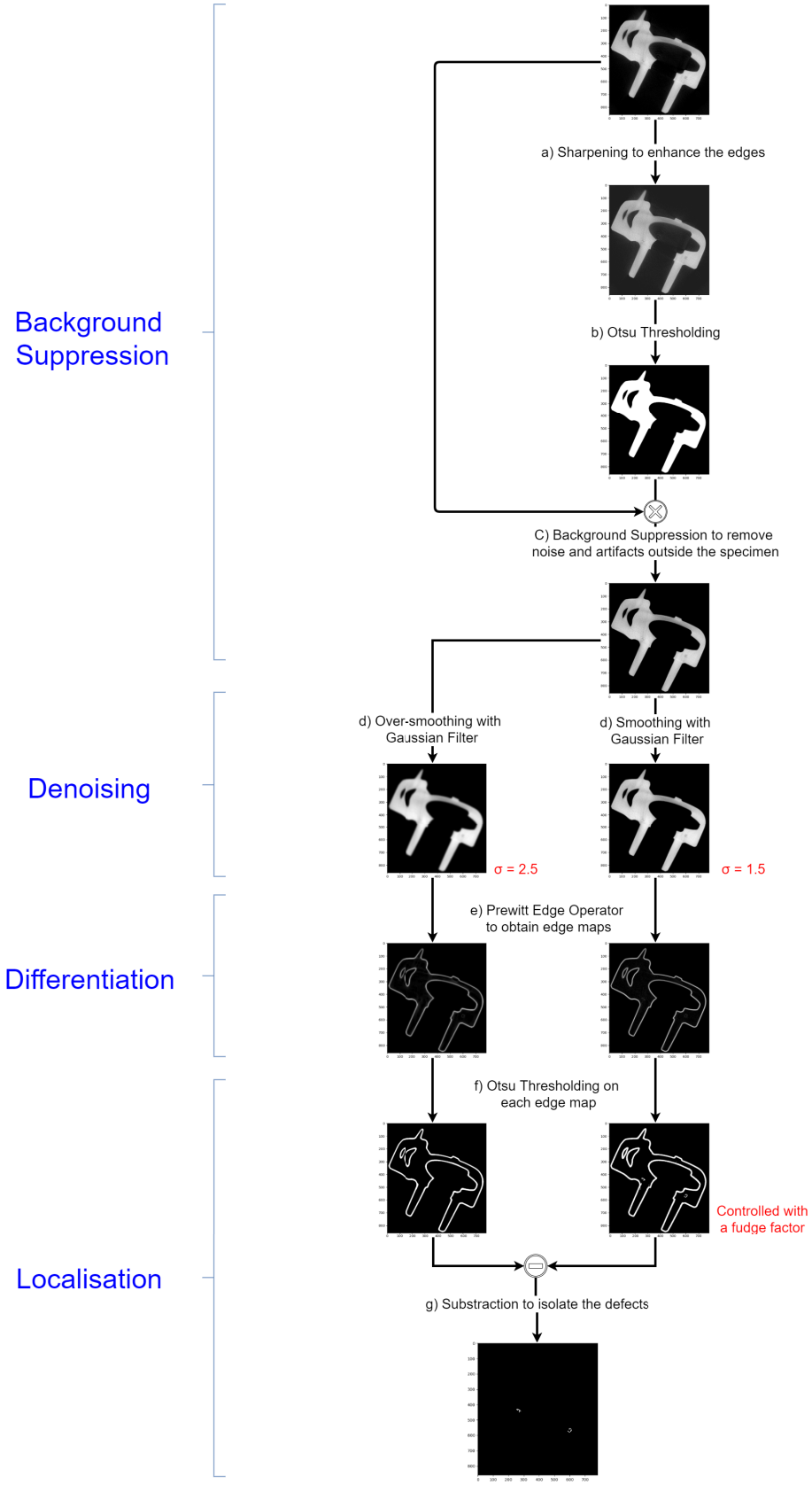


Figure II.7: 2D segmentation pipeline for detecting or localizing casting defects. The input is a greyscale CT slice (Scale $14cm \times 14cm$ in this example), and the output is a binary version where casting defects are highlighted.

II.3.2 Denoising

d) The denoising step consists of smoothing with a gaussian kernel. By applying a 2D convolution between the image with suppressed background and the latter, we can smooth or blur the image and remove noise [81]. A gaussian kernel has the shape of Gaussian distribution with a size of 3×3 or 5×5, and its weights are calculated using the following 2D gaussian function:

$$g(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{II.2}$$

where (x,y) is the coordinate of each element in the kernel and σ is the standard deviation of the gaussian distribution.

Although technically the kernel can be of any size, σ must be scaled in proportion to the kernel size. In the case of a large kernel with a small sigma, the kernel elements at the extremes will have no real effect on the computation. Moreover, the value of the standard deviation σ controls the extent of smoothing. The higher the σ, the stronger the smoothing effect.
At the end of the pipeline there is a subtraction operation where the borders of the specimen are removed and the edges of the defects are maintained. For this reason, we applied two separate convolutions with two gaussian kernels as illustrated in Figure II.7:

- Image on the right: the image with suppressed background is denoised with a 3×3 kernel and $\sigma = 1.5$.
- Image on the left: the image with suppressed background is over-smoothed with a 5×5 kernel and $\sigma = 2.5$ where the defects are removed.

II.3.3 Differentiation

e) The differentiation block has one step, which is the essential operation of edge-base segmentation. Both images outputted by the filtering block are convoluted with an edge operator. The output of this convolution is a greyscale gradient where the abrupt greyscale discontinuities are highlighted. At each voxel, the operator calculates the gradient of intensity. In the case of an edge, this gradient shows how abruptly the image has changed, as well as the orientation of this edge from dark to light. Various edge operators can be used to find finite-difference approximations of the gradient [82], and after many trial-error attempts we have used Prewitt operator as it has been found to be more sensitive to small variations compared to other operators due to the absence of an additional smoothing module [83]. This ensures that poorly contrasted defects are not overlooked during segmentation.
Prewitt operator uses two 3×3 kernels which are convoluted with the smoothed image in order to calculate its horizontal and vertical derivatives. If we define $S(x, y)$ as a smoothed image by the gaussian filter, S'_x and S'_y, the horizontal and vertical derivatives approximations, are calculated as follows:

$$S_x^{'} = S(x,y) \circledast \begin{pmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{pmatrix}; \quad S_y^{'} = S(x,y) \circledast \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{pmatrix} \tag{II.3}$$

At each point of $S(x,y)$, the resulting gradient approximations can be combined to give the gradient magnitude, which is a greyscale image with potential edge segments:

$$\nabla S(x,y) = \sqrt{(S_x')^2 + (S_y')^2} \tag{II.4}$$

This gradient is calculated for the smoothed and over-smoothed images from the last step of the pipeline. Each contains segments of edges that belong to the specimen borders, defects, artifacts and even noise.

II.3.4 Localization

The last block of the pipeline is the localization block where edges that belong to noise or artifacts are eliminated as much as possible.

f) In order to get rid of the edges that belong to noise or artifacts without affecting the edges that belong to defects, we decided to apply Otsu's thresholding method to the greyscale gradients. As explained earlier, this segmentation method finds the threshold that separates the two majorly represented classes of voxels - background and foreground. By multiplying this threshold by a **fudge factor**, we can change the amount of edges that should be preserved in the output image. For example, if we set the fudge factor to 0.1, this means that the segmentation threshold is set to 10% of the value calculated by Otsu and the output image will have more edges after segmentation. Conversely, the higher the fudge factor, the less edge segments in the output.
By looking at the step **f)** in Figure II.7, Otsu was applied to both gradients, or edge maps, coming from the previous step:
   - On the left, the output binary image does not have the defects because of the over-smoothing at step **d)** with the gaussian filter.
   - On the right, the output binary image contains the borders of the specimen, as well as the defects in the CT slice.

g) For clarity, the 2 images from the last step can be subtracted to remove the borders of the specimen and keep the defects isolated. Then, if needed, a morphological operation can be applied to the image to fill the hollow edges segmented by the pipeline.

II.4 3D Segmentation Algorithm

The 3D algorithm is straightforward compared to the 2D algorithm because processing 3D images requires time-consuming calculations. The algorithm takes a

3D image as input, and outputs a binary 3D volume, as in Figure II.8. The segmentation is edge-based and consists of three main steps: preprocessing, differentiation and localization.

II.4.1 Preprocessing

Before applying an edge operator, we need to suppress the empty zone outside the specimen and its artifacts. As with the 2D algorithm, this can be done by multiplying the volume by its Otsu binary version. Otsu thresholding works well in our case because the histogram of all 3D CT volumes (Figure II.2) is bimodal, i.e. the voxels can be classified into one of 2 major classes: Background and Foreground. The Otsu method gives a binary volume with the specimen voxels set to 1 and the background voxels set to zero. When we multiply the volume by this binary version, the specimen remains unchanged, but the grey zone outside the specimen and its noise are deleted or suppressed. Unlike the 2D algorithm, the preprocessing is not followed by a denoising step with a gaussian filter, as this could remove small discontinuities. However, we have chosen a 3D operator that has a smoothing effect to reduce the noise along the 3 directions.

II.4.2 Differentiation and Localization

The second step is to find the edges of the volume. This is done by applying an edge operator to the volume to obtain a greyscale edge map, followed by an adjustable Otsu threshold to remove the edges that belong to false alarms. Edge detection is performed by convolving the volume with a set of three 3D operators, also called kernels, which find a discrete approximation of the partial derivatives of the volume with respect to each direction. Depending on the layout of the 3D CT volume, the first kernel detects the discontinuities in the right direction, the second in the top direction and the third in the front direction. In our application, we used Sobel-Feldman operator, which has three $3 \times 3 \times 3$ kernels applicable in three consecutive frames along the 3 directions. The convolution operations between the denoised volume $V(x, y, z)$ and the 3D Sobel-Feldman edge detector in the directions X, Y, Z are as follows:

$$V_x^{'} = V \circledast \left\{ \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}_{x-1}, \begin{pmatrix} -2 & 0 & 2 \\ -4 & 0 & 4 \\ -2 & 0 & 2 \end{pmatrix}_{x}, \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}_{x+1} \right\} \tag{II.5}$$

$$V_y^{'} = V \circledast \left\{ \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}_{y-1}, \begin{pmatrix} 2 & 4 & 2 \\ 0 & 0 & 0 \\ -2 & 4 & -2 \end{pmatrix}_{y}, \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}_{y+1} \right\} \tag{II.6}$$

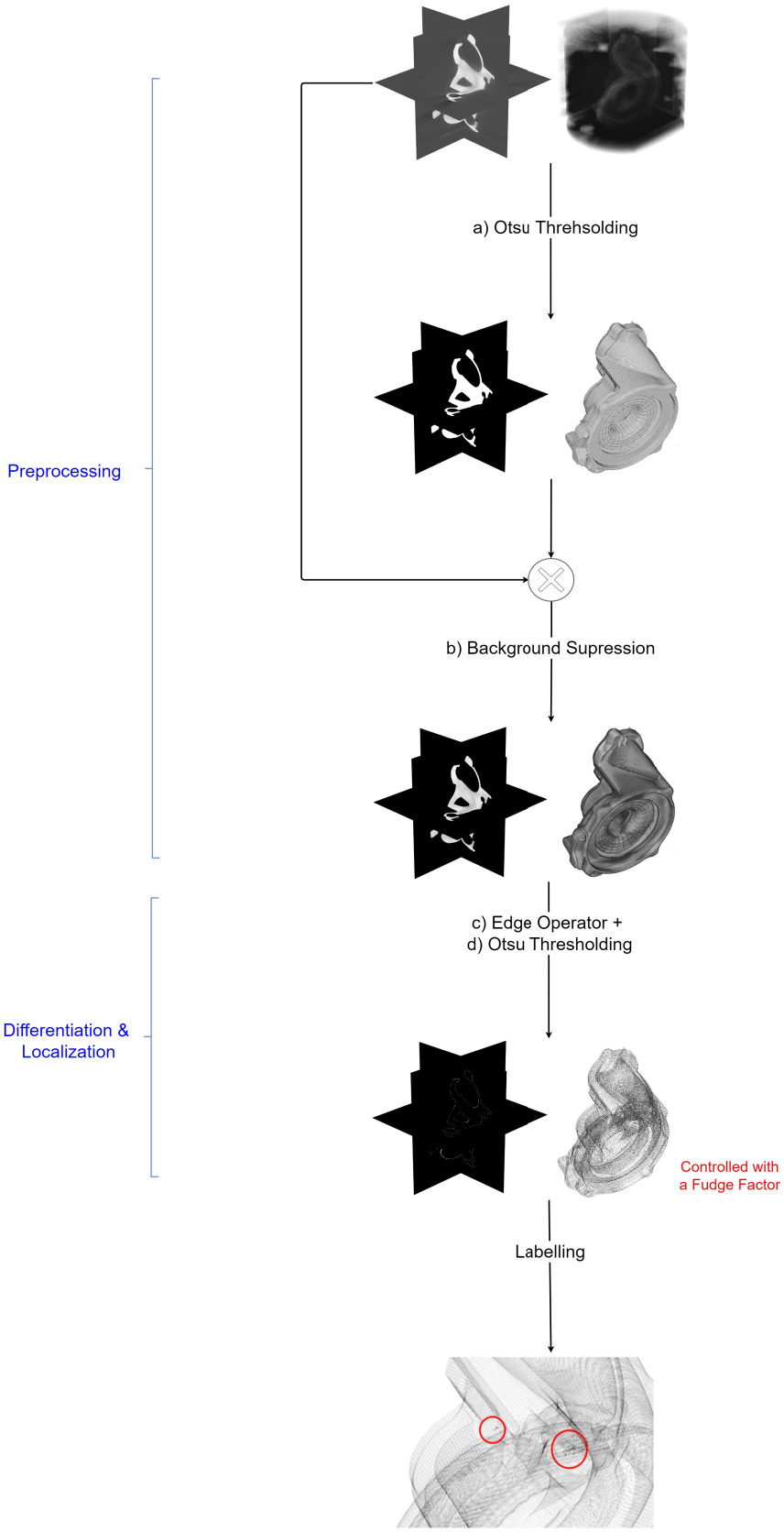


Figure II.8: 3D segmentation pipeline for localizing casting defects. The input is a greyscale 3D X-ray volume and the output is a labelled 3D binary volume. For clarity, the opacity of each volume is adjusted for better visualization, and each volume is displayed with a 3D ortho-slice view.

$$V_z^{'} = V \circledast \left\{ \begin{pmatrix} -1 & -2 & -1 \\ -2 & -4 & -2 \\ -1 & -2 & -1 \end{pmatrix}_{z-1}, \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}_{z}, \begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}_{z+1} \right\} \tag{II.7}$$

where $V_x^{'}$ is the gradient approximation or partial derivative along the X-axis, and so on. The final edge map, or the total gradient, is tracked by combining the 3 dissimilarities maps into a single magnitude greyscale map as follows:

$$\nabla V(x,y,z) = \sqrt{(V_x^{'})^2 + (V_y^{'})^2 + (V_z^{'})^2} \tag{II.8}$$

We used the Sobel-Feldman operator because it gives good approximations of the gradients along each direction, has a denoising effect on the volume, and is not very sensitive to isolated point fluctuations with high intensity [84]. The edge map contains many edge-points of false alarms that need to be eliminated with thresholding. Using Otsu's algorithm, we can find the optimal threshold that eliminates these unwanted edge segments. By manually adjusting this threshold with a **fudge factor**, a binary volume $W(x,y,z)$ can be obtained in which defects and borders of the specimen are highlighted:

$$W(x,y,z) = \begin{cases} 1 & if \quad \nabla V(x,y,z) > \textbf{fudge factor} \times Thresh \\ 0 & otherwise \end{cases} \tag{II.9}$$

A morphological operation was then applied to the volume to fill the hollow edges segmented by the pipeline. An additional labelling step can be applied at the end of the pipeline to isolate each defect separately. In the binary volume outputted by the previous step, each set of adjacent white voxels is assigned a unique integer representing a single defect, as in Figure II.9.

II.5 Results & Discussion

II.5.1 2D Algorithm

The 2D algorithm detects discontinuities in CT slices belonging to defective zones, as in the pipeline in Figure II.7. This algorithm is very fast and requires no computational cost. However, some of the discontinuities detected by this algorithm may belong to noise or artifacts if the fudge factor is not set properly. The fudge factor controls the value of Otsu threshold applied to the edge map outputted by Prewitt operator and consequently has a large impact on the result, as can be seen in the following images:

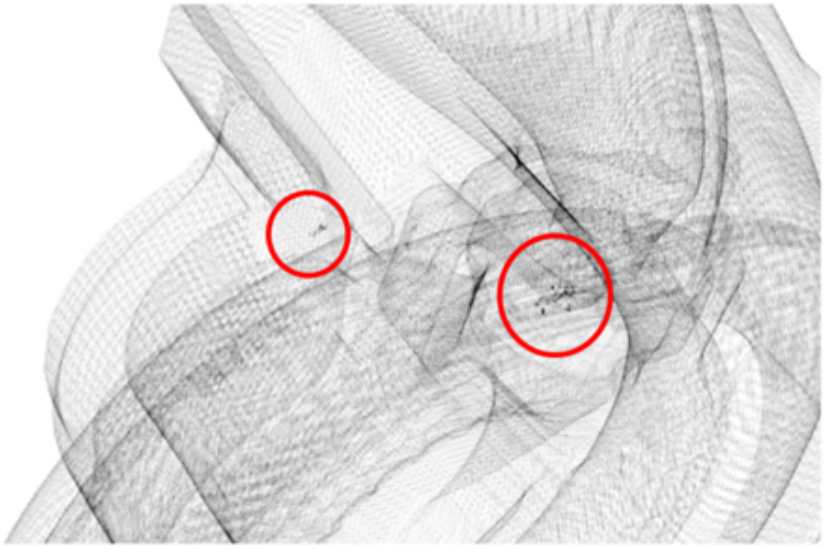


Figure II.9: Some detected defects inside the 3D CT volume, segmented by the 3D edge-based segmentation algorithm ($fudgefactor = 26\%$).

- Figure II.10 shows that for the same image, a decrease in the threshold value can increase the number of detected noise and artifacts. Conversely, increasing the value decreases the detected discontinuities to the point where true defects could be missed in the output binary image.
- Figure II.11 shows that a defined fudge factor is not suitable for all slices of the same volume because the contrast changes along the volume and therefore constant adjustment is required to reduce the number of unwanted segments.

Because of this need for user intervention, this algorithm cannot be used to automatically inspect CT slices of casting specimens. We will use it to generate ground truth images to train neural networks and validate their efficiency, as shown in chapter IV.

II.5.2 3D Algorithm

The 3D segmentation algorithm in Figure II.8 can be applied directly to a CT volume, yielding a binary volume in which discontinuities are highlighted. However, as with the 2D algorithm, it is almost impossible to find the right fudge factor that works well for the entire volume, as the contrast changes along the thickness of the specimen. In Figure II.12, we applied the algorithm to small regions of interest (ROIs) along the volume to detect embedded defects. In this example, the fudge factor is adjusted as we move along the height of the specimen in order to avoid segmenting noise or artifacts when the fudge factor is very low, or missing the true defects when the fudge factor is very high.

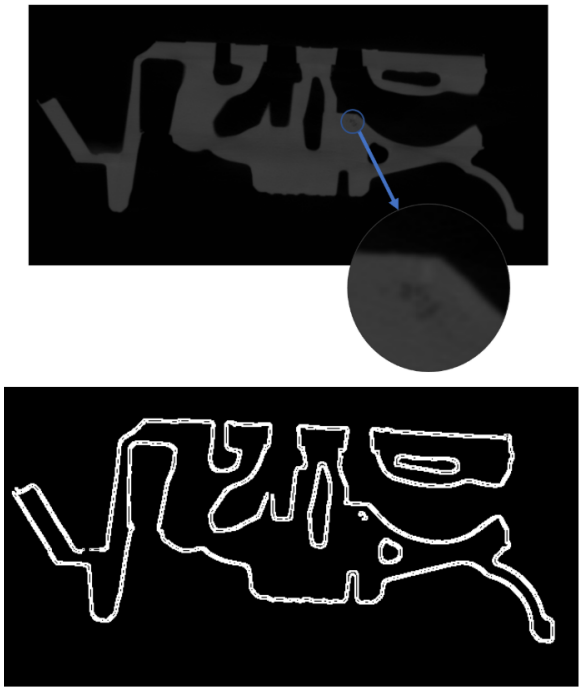


Figure II.10: The sensitivity of the 2D segmentation algorithm is controlled with a fudge factor. To find the right value (32 % for this image), the user must intervene to avoid segmenting artifacts (low fudge factor) or missing the defect itself (high fudge factor).

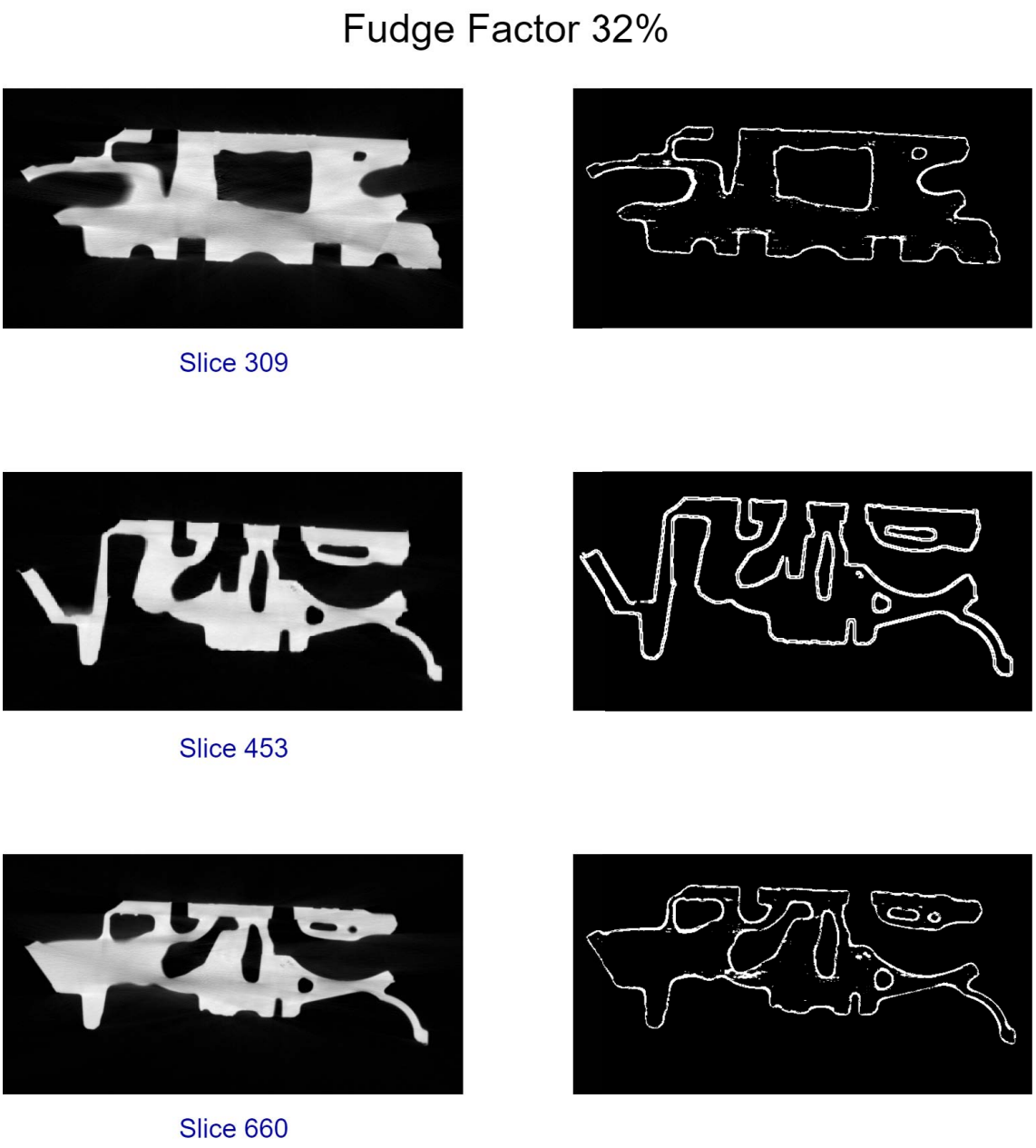


Figure II.11: The same fudge factor does not give good results for all slices of the same volume. As the thickness changes along the volume, the X-ray absorption changes and so does the contrast of the images. Therefore, the fudge factor must be adjusted independently for each slice.

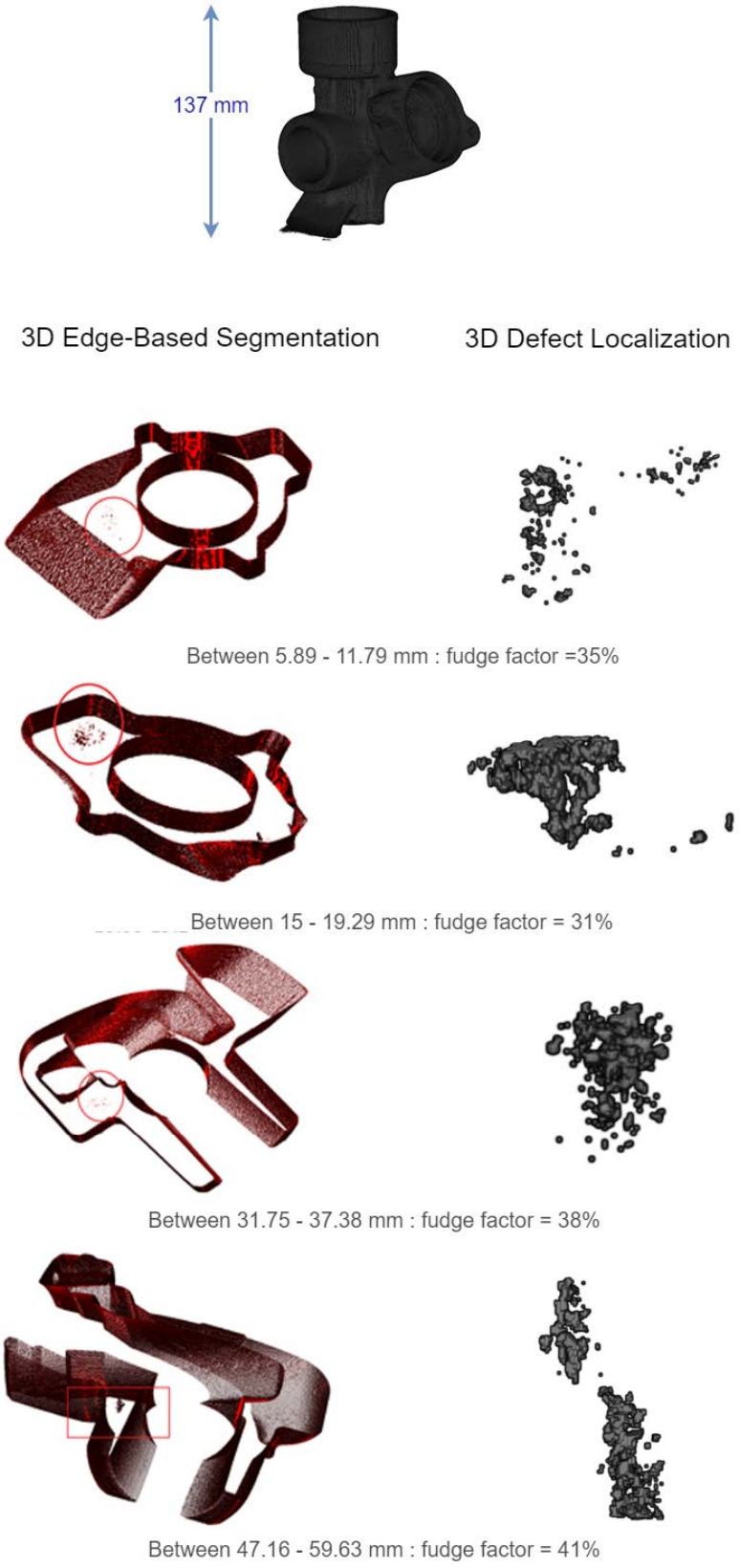


Figure II.12: 3D segmentation algorithm applied to small ROIs along the height of the specimen. As we move across the volume, the fudge factor needs to be adjusted so that defects are not missed.

Conclusion

In this chapter, two edge-based segmentation algorithms are described in detail. The first is a 2D algorithm that takes a CT slice as input, and outputs the defects present in the image; the second takes a 3D ROI, and outputs a binary 3D volume with the defects embedded. Both algorithms give good results if their parameters, called fudge factors, are tweaked correctly. However, such human intervention slows down the inspection process when used on the production line.

In the next chapter, a potential neural network, called U-Net, is trained to perform image segmentation automatically without tweaking any parameters. The training data (ground-truth images) were manually segmented using the 2D segmentation algorithm presented in this chapter. And in chapter VI, a geometrical characterization study is conducted to investigate the properties of 3D volumes containing casting defects, segmented using the 3D segmentation algorithm.

Chapter III

U-Net for Industrial CT Images Segmentation

In chapter II we presented a 2D segmentation algorithm based on image processing techniques that extracts defects in CT images. It requires adjusting a fudge factor for each slice along the volume to avoid under- or over-segmentation. An ideal algorithm binarizes all CT slices of the same volume without tweaking any severity factor. However, such segmentation potential is very difficult to achieve with standard image processing algorithms because CT volumes have different contrasts and artifacts depending on the tomography system and the slice index along the slicing axis.

In the biomedical field, a fully convolutional-based network called U-Net has shown

robust efficiency in segmenting CT slices [85, 86]. In this chapter, we explore the potential of U-Net in segmenting CT slices of the same casting volume as well as slices derived from other volumes without user intervention. This is achieved by training this model from ground up with a **manually segmented dataset** that we built by cropping 2D slices from different volumes of different specimens scanned under different acquisition conditions.

III.1 Segmentation Task

Some tasks are too difficult to solve with traditional algorithms in terms of processing, execution time and implementation settings. In recent decades, advances in the deep learning have made some tasks easier to solve, such as segmentation, classification, regression, transcription, machine translation, anomaly detection, synthesis and sampling, denoising and density estimation [87]. Deep Learning tasks can be defined as the process by which a deep learning model learns from a set of examples, or data points, to perform a particular task. Each example from this set, represented as a vector $x_i \in R_n$, is a collection of features x_{ij}. In the case of image segmentation, the model learns to output a segmented version of x_i, called a label map or mask p_i, by assigning each feature (pixel or voxel) x_{ij} to a certain class.

Image segmentation tasks with deep learning models, more specifically convolutional neural networks (CNNs), fall into two broad categories depending on the output. Looking at Figure III.1, these two categories can be distinguished as follows:

- In semantic segmentation, an image is divided into several contiguous regions. This pixel (or voxel)-wise partitioning is based on the correlations between the pixels in the image. If the input image contains several elements of the same type, such as a chair, all these elements would be assigned the same colour in the output. In the case of binary classification, the output contains only black and white areas, as is the case with chapter II segmentation algorithms.
- Instance segmentation is a more difficult task. It is done at the pixel (or voxel)-level of the individual elements in an image. If an image contains two chairs, instance segmentation distinguishes which pixels belong to which chair. This category of algorithms distinguishes not only the semantics, but also each individual object of interest. In classical image processing, this task is equivalent to semantic segmentation with an additional labelling step.

III.2 Learning Experience

Deep learning models can be divided into two categories, depending on the learning experience and the amount of information contained in the dataset: (1) unsupervised learning algorithms, which are trained on data points without labels so that the algorithm learns the underlying structure or distribution in the data itself; and (2) supervised learning algorithms, which are trained on a dataset where each data point is given a label, also known as a target or ground truth. The term

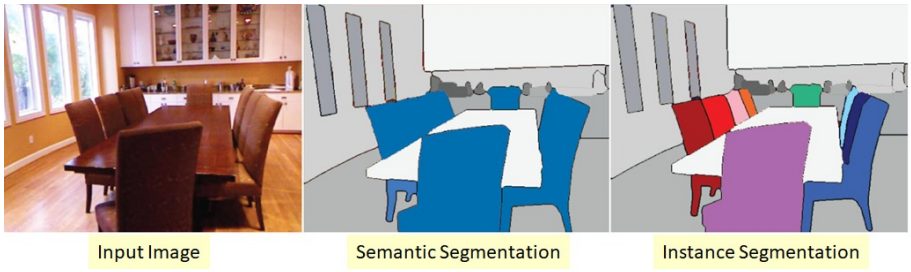


Figure III.1: Semantic segmentation (middle) and instance segmentation (right). Semantic segmentation assigns a single colour to all elements of the same category, while instance segmentation processes each element individually.

"supervised" stems from the fact that for each data point x_i the target y_i is given by the "teacher", whereas in the unsupervised case there is neither a teacher nor a target and the algorithm is supposed to draw inferences from the dataset without human intervention. To teach U-Net to segment CT images, we trained it with the following manually segmented dataset making the learning process supervised. We chose to train the segmentation model with 2D slices instead of directly using the entire volumes, as we only have a limited amount of 3D CT data.

III.2.1 Training Dataset

We built a dataset by cropping 512×512 images from 2D CT slices of high-pressure and gravity aluminium alloy castings provided by our industrial partners. A snapshot of one of the CT volumes and the cropping process is shown in Figure III.2. Due to GPU memory constraints, 512×512 was chosen as the cropping size and the input to the U-Net model was limited accordingly. A sample of this dataset can be seen in Figure III.3 where we can see a variety of contrasts and spatial resolution in the CT slices.

After normalization, each greyscale 512×512 image was segmented *individually* using the 2D segmentation algorithm explained in Figure II.7 after manual adjustment of the fudge factor, resulting in a dataset with a total of 3000 data points. Each segmented mask contains the following classes of voxels:

- Black voxels, which belong to non-defective zones filled with material, or the empty zone outside the specimen, or artifacts or noise.
- White voxels, which belong to real defects.

During each training epoch (iteration), the segmentation model is trained on 80% of the dataset, and at the end of the epoch it validates itself on the rest to give an unbiased estimate of the skill of its architecture with the current weight update, as explained in section I.8.

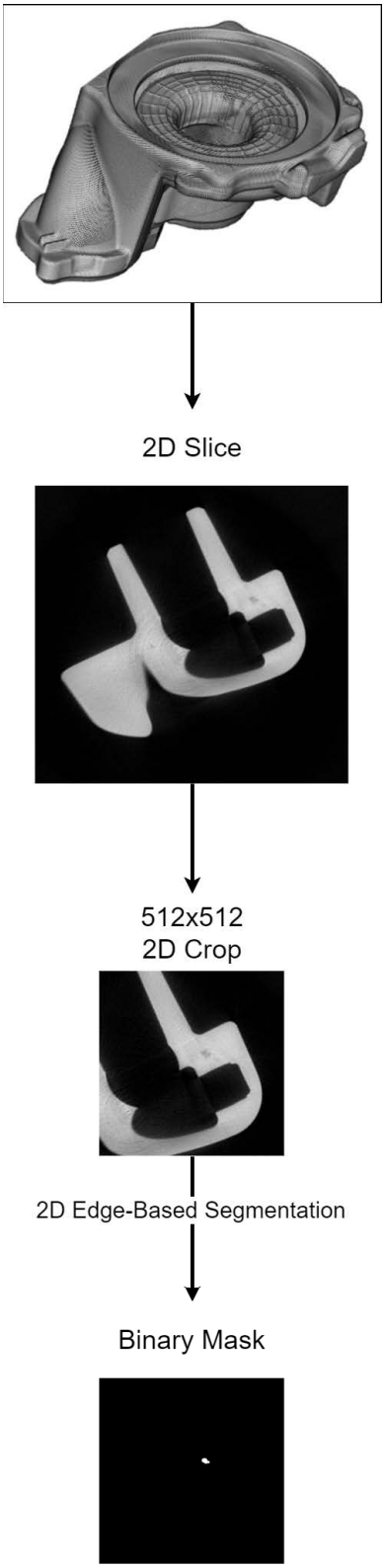


Figure III.2: The segmentation database contains 512×512 images with their target binary masks. Each was cropped from a 2D image that was sliced from a 3D volume.

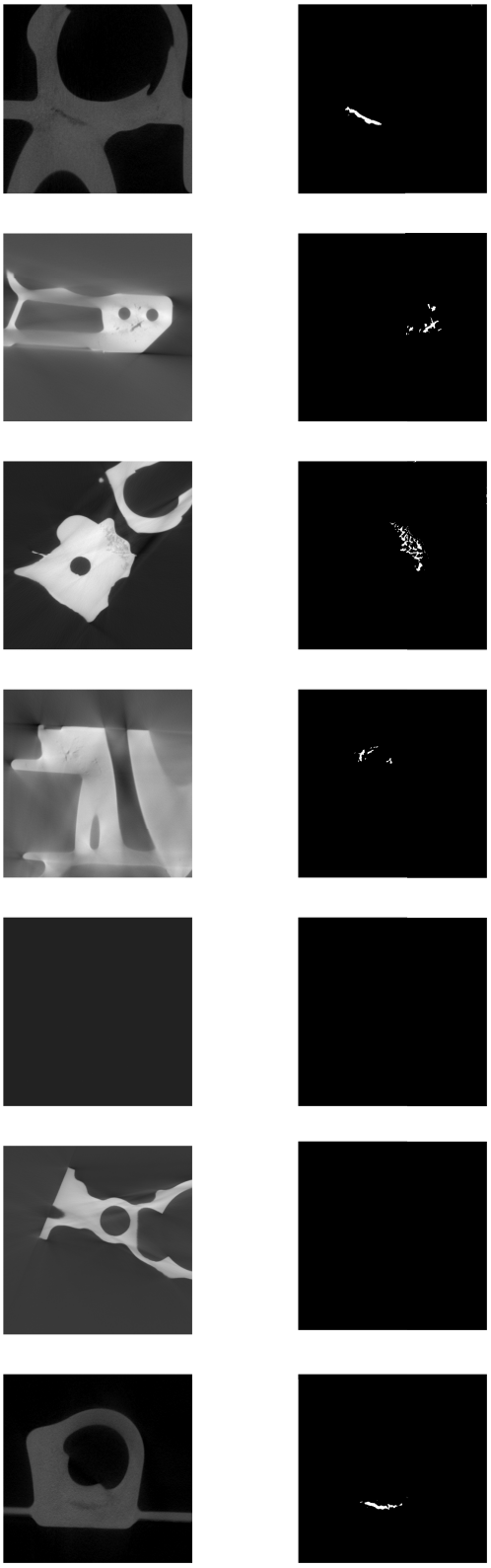


Figure III.3: Examples of images from the segmentation database of 3000 images: (Left) 512×512 images cropped from the CT volumes, (Right) their binary target masks where only defects are segmented. This segmentation is to ensure that the model only responds to discontinuities that belong to real defects.

III.3 Performance Measurement

The proper use of state-of-the-art models to perform semantic segmentation can achieve high accuracy and reduce the time required to optimize a new architecture. However, these advanced models, such as U-Net, need appropriate metrics to evaluate their learning process and predictive ability on new inferences. It should be noted that U-Net does not output a binary black and white mask like the usual image segmentation algorithms. During training, for each CT image in the dataset, U-Net learns to output a greyscale mask that resembles the target binary mask in the dataset as closely as possible. An image correctly segmented by U-Net, also called a label map, must fulfil the following conditions:

- If the binary target in the dataset contains voxels set to 1 (white voxels belonging to defects), U-Net must set the same voxels as close to 1 as possible in its greyscale output.
- In the case of voxels set to 0 (black voxels belonging to the background), U-Net must set the corresponding voxels in its output as close to 0 as possible.

During training, the model's learning is evaluated using a loss function based on how far the voxels are from the target values 0 or 1. However, when the model is tested on new unseen images after training, we need to convert the outputs of U-Net into binary images to compare them with the binary versions of the new inferences. This conversion from greyscale images to binary images could be done with a simple thresholding, as explained in chapter V.

Common loss functions, such as cross-entropy, are not relevant in our semantic segmentation task due to class imbalance. This occurs when classes of regions, in our case "defects" or "background", are not equally represented in the image. Indeed, CT images are dominated by voxels belonging to a normal zone filled with material or to the empty zone outside the specimen, as in Figure II.2. Therefore, we need to find a new metric that does not reward the model for correctly classifying the background voxels without equally considering the voxels of defects. In other words, we need other metrics that help the model to converge optimally during training, as shown in Figure I.18, while preventing it from developing a bias towards the largely represented class. For this reason, we measured the training and validation phases during each training epoch with a loss function based on an extended version of the Sørensen-Dice coefficient [88].

III.3.1 Dice Loss

When it comes to the segmentation task, a variety of loss functions can be used as metrics to evaluate the learning process [89]. However, common loss functions evaluate the performance of the segmentation model in predicting the class of each voxel and then average over all voxels. As explained earlier, this evaluation approach is misleading for images with **imbalanced classes**. The dice coefficient is a well-known evaluation metric for image segmentation tasks and can also serve as a loss

function [90]. Such a loss resolves the class imbalance by rewarding the model not only for correctly classifying the background voxels as true negatives (TN), but also for classifying the defect voxels as true positives (TP), with the same penalty.

What is the dice coefficient?

The dice coefficient measures the overlapping between 2 masks $Mask_1$ and $Mask_2$ for each class of regions $k = 1, \ldots, M$. The total dice coefficient is the average of overlapping over all classes and it can be calculated as follows:

$$Dice = \frac{1}{M}\sum_{k=1}^{M} Dice_{class=k} = \left.\frac{1}{M}\sum_{k=1}^{M}\frac{2\ \times \left|Mask_1^k \cap Mask_2^k\right|}{\left|Mask_1^k\right| + \left|Mask_2^k\right|}\right|_{class=k} \tag{III.1}$$

- M, the total number of classes in both images.
- $|Mask_1^k|$, the total number of voxels in $Mask_1$ that belong to class k.
- $|Mask_2^k|$, the total number of voxels in $Mask_2$ that belong to class k.
- $|Mask_1^k \cap Mask_2^k|$, the number of voxels in common that belong to class k.

In the case of 2 binary masks, there are only two classes of voxels in the images, $k = 1, 2$, i.e. white or black voxels. In this case, the total dice coefficient can be calculated as follows:

$$Dice = \frac{1}{2}\left\{\frac{2\ \times \left|Mask_1^{white} \cap Mask_2^{white}\right|}{\left|Mask_1^{white}\right| + \left|Mask_2^{white}\right|} + \frac{2\ \times \left|Mask_1^{black} \cap Mask_2^{black}\right|}{\left|Mask_1^{black}\right| + \left|Mask_2^{black}\right|}\right\} \tag{III.2}$$

By dividing the overlap of each class by the total number of voxels, the background class (black voxels) is prevented from dominating over the smaller class, so that each class is penalized with a balanced magnitude.

Dice Loss for U-Net Evaluation

During training, for each image x_i with a target mask y_i, the segmentation model predicts a label mask p_i, where the overlap between the regions k of both masks must be as high as possible, i.e. the dice coefficient must be maximized. Alternatively, the dice loss must be minimized, and can be deduced as follows:

$$L_{dice}(y_i, p_i) = 1 - \frac{1}{M}\sum_{k=1}^{M} Dice(y_i^k, p_i^k) \tag{III.3}$$

As explained earlier, each training epoch consists of training the model on a subset of images X, followed by validation on a subset X'. To evaluate the learning of a segmentation model, we need to compute the error on each subset, also called training and validation losses. Plotting both losses can help spot signs of underfitting

or overfitting, as explained in section I.8. In the case of binary segmentation ($k = 1, 2$), the dice loss on a subset of length N is calculated for all images and then averaged to obtain a final score:

$$L_{dice} = \frac{1}{N}\sum_{i=1}^{N} L_{dice}(y_i, p_i) = \frac{1}{N}\sum_{i=1}^{N}\left\{1 - \frac{1}{2}\sum_{k=1}^{2} Dice(y_i^k, p_i^k)\right\} \tag{III.4}$$

$$L_{dice} = \frac{1}{N}\sum_{i=1}^{N}\left\{1 - \frac{1}{2}\left\{\frac{2\ \times \left|y_i^{white} \cap p_i^{white}\right|}{\left|y_i^{white}\right| + \left|p_i^{white}\right|} + \frac{2\ \times \left|y_i^{black} \cap p_i^{black}\right|}{\left|y_i^{black}\right| + \left|p_i^{black}\right|}\right\}\right\} \tag{III.5}$$

This equation works well in the case of a segmentation model that outputs a binary mask in which voxels have discrete values 0 or 1. However, as mentioned earlier, the output of U-Net is an array of probabilities, i.e. a greyscale image where each voxel has a value between 0 and 1. For this reason, we need a modified version of this dice loss to make it differentiable. Sudre et al. introduced the soft dice loss in [90], which can be used as a metric to evaluate U-Net training without computational cost. As depicted in Equation III.6, the overlap is calculated by element-wise multiplication between the target mask y_i and the predicted mask p_i as follows:

$$L_{dice}(y_i, p_i) = \frac{1}{N}\sum_{i=1}^{N}\left\{1 - \frac{2\ .\ \sum y_i \cdot p_i}{\sum y_i^2 + \ \sum {p_i}^2}\right\} \tag{III.6}$$

III.4 Adam Optimizer

Now that we have defined the performance metrics that measure the learning process, an important question arises: *How do we find the right weights of the network that can achieve the lowest possible loss value?*. For this reason, we need an optimization algorithm that updates the weights as the training progresses. The optimizer updates the weights of the network at the end of each training epoch until the loss function reaches the lowest possible local minimum, as shown in Figure III.4. This updating process is controlled by a parameter called **learning rate**, which must be tweaked during training [91].

After many trial-error attempts, Adaptive Moment Estimation (adam) [93] was adopted, which uses momentum and adaptive learning rates to make the loss function converge faster. This means that a separate learning rate is maintained for each weight, which is adjusted during the training process. As these learning rates decrease, the gradients of the weights are updated very slowly, which prevents the loss function from overstepping a local minimum. Furthermore, during each epoch, the training data is divided into small **batches**, and at the end of each batch, adam optimizer estimates which weights need to be changed to achieve the minimum loss. This ensures faster convergence towards a local minimum as the weights are updated more frequently.

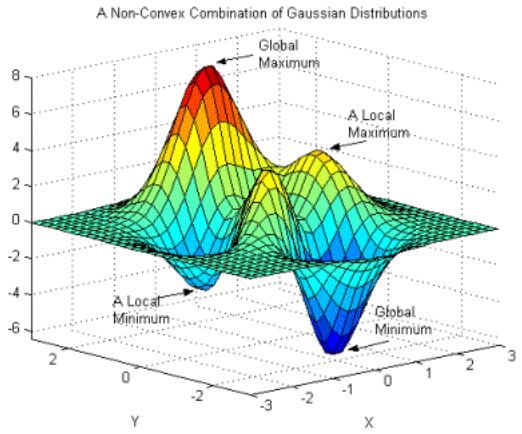


Figure III.4: The loss landscape has many peaks and valleys based on the weights of the network. Since it is impossible to reach the global minimum, the ideal set of weights ensures that the model reaches the lowest local minimum. Source: [92]

III.5 Semantic Segmentation with CNN

Semantic segmentation is the task of assigning each pixel or voxel in an image to a class based on its value (RGB or greyscale). Many algorithms have been developed for this pixel-wise classification task, including neural networks. In 2015, Shelhamer et al. [94], trained a supervised Fully Convolutional Network (FCN) for pixel-wise prediction on the PASCAL VOC, a dataset with 20 object categories where each image contains pixel-level segmentation annotations. The FCN consists of two phases, as shown in Figure III.5:

1. Downsampling phase, in which feature maps are generated using convolutional layers (cf. subsection I.7.1).
2. Upsampling phase, in which transposed convolutions are applied to these feature maps to restore the original image size and regenerate the semantics in the final output (cf. subsection I.7.5).

At the time, this network was revolutionary and led researchers to realize the potential of deep learning models in the field of image segmentation. Later, SegNet was proposed in [95], which is similar in structure to FCN. This network has good real-time performance as it uses max-pooling (cf. subsection I.7.3) followed by feature map stretching as an upsampling method to reconstruct the segmented image instead of transposed convolution, which in turn reduces the training parameters and time.

Also in 2015, Ronnerberger et al. [97] introduced U-Net for medical image segmentation. The U-Net architecture outperformed the FCN network due to the increase in number of feature maps by the same amount during the upsampling and downsampling phases. This allows the deep and shallow features of the image to be com-

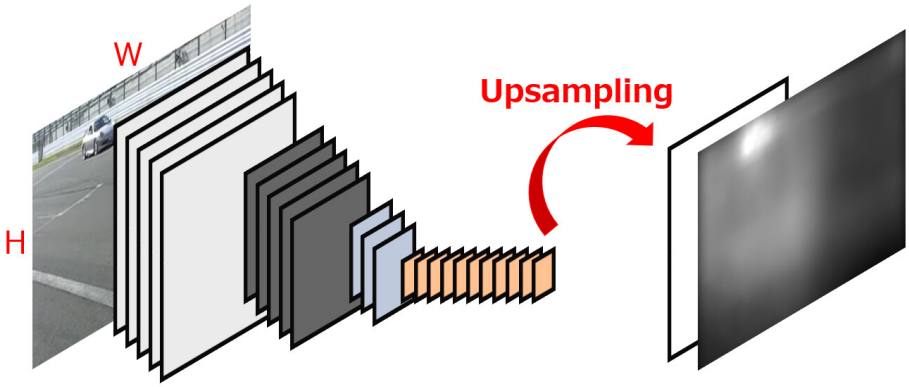


Figure III.5: A diagram of the structure of FCN. The original input is downsampled by pooling and upsampled again to the original image size by transpose convolution. Source: [96]

bined and solves the problem of the vanishing gradient. The latter occurs when the loss improves slowly and the weights are only updated by a small amount, resulting in a vanishing gradient and very slow convergence. To preserve the fine-grained and local information of the image, U-Net uses connections between the upsampling and downsampling, similar to FCN, and concatenates the feature maps of the different phases, as shown in Figure III.6.

When it comes to segmenting CT images with neural networks, there are two categories of convolutional neural networks (CNNs) that can be used for this task: (1) 3D CT volumes are sliced along a specific direction into a series of 2D slices, so that 2D CNNs can segment CT volumetric images [98]; or (2) 3D fully-convolutional architectures such as 3D U-Net [60] and V-Net [98] can be trained directly on the CT volumes without the need for slicing. However, 3D CNNs have several shortcomings compared to 2D CNNs; For example, less stable training could be achieved since there are not many pre-trained 3D CNNs that could be used as reference [99]. Furthermore, according to Isensee et al. [100], a 2D U-Net may outperform 3D U-Net if the data is anisotropic, i.e. if the voxels have different dimensions along the three axes, which is the case for one of our available volumes. Moreover, training 3D models is very time-consuming compared to 2D models and requires a tera-scale amount of data.

We decided to use U-Net for segmenting CT slices of aluminium alloy as it offers the following advantages:

- After reducing the number of filters in the convolutional and transposed convolutional layers compared to the original U-Net model, the modified network consists of only 1,940,817 parameters compared to the original U-Net (11,593,424), SegNet (29,457,797) and FCN (134,325,766) – with the same input size–, and thus requires less computational cost and GPU memory allocation.

- When trained with the correct evaluation parameters, it can separate two elements of the same class in the output image, even if these elements are extremely close to each other in the original image.
- Since U-Net has skip connections between the downsampling and upsampling phases, and its architecture has enough feature maps per convolutional block, it preserves the contextual information of the image very well, especially the neighbourhood information between voxels. Contextual information, including the relationship between two or more voxels and the fine-grained detail features such as the edges of the image, is needed during upsampling (reconstruction of the segmented image) and it is provided by the skip connections from downsampling, as shown in Figure III.6.

III.6 U-Net Under the Loop

When U-Net was proposed as a semantic segmentation algorithm for biomedical images, it performed exceptionally well on the required task, outperforming the best segmentation methods to date, despite being trained with very few images [101]. U-Net belongs to the autoencoders family, which means that it consists of two paths, as shown in Figure III.6:

1. On the left, a contractive path called the **encoder**. This part of the network takes the input image, and outputs a set of feature maps called a *feature vector* containing information and features representing the input.
2. On the right side is an expansive path called the **decoder**. It has the same network structure as the encoder, but in reverse orientation. This part of the network takes the *feature vector* from the encoder, which is in the middle (bottleneck) of the network, and outputs the closest representation of the input image. The feature vector in the bottleneck is also called the code or *sparse representation* of the input.
3. In the middle, skip connections between both paths that add the feature maps of the encoder to the decoder in order to reduce the coarseness of the output.

The architecture of U-Net is fully-convolutional, i.e. it has no fully-connected layer (cf. subsection I.7.4) and consists of several convolutional blocks divided into encoder and decoder paths. In Figure III.7, the version of U-Net architecture we used to segment 512×512 CT casting images is shown. We decided to keep the original U-Net architecture [97], although we reduced the number of kernels in each convolutional layer to reduce the trainable weights and computational costs.

Encoder

The encoder consists of 5 convolutional blocks, each containing two consecutive convolutional layers followed by a downsampling layer called max-pooling, except for the bottleneck block. All conv layers have 3×3 kernels and the number of feature maps per block is doubled between two consecutive blocks. On the way from 1$^{\text{st}}$

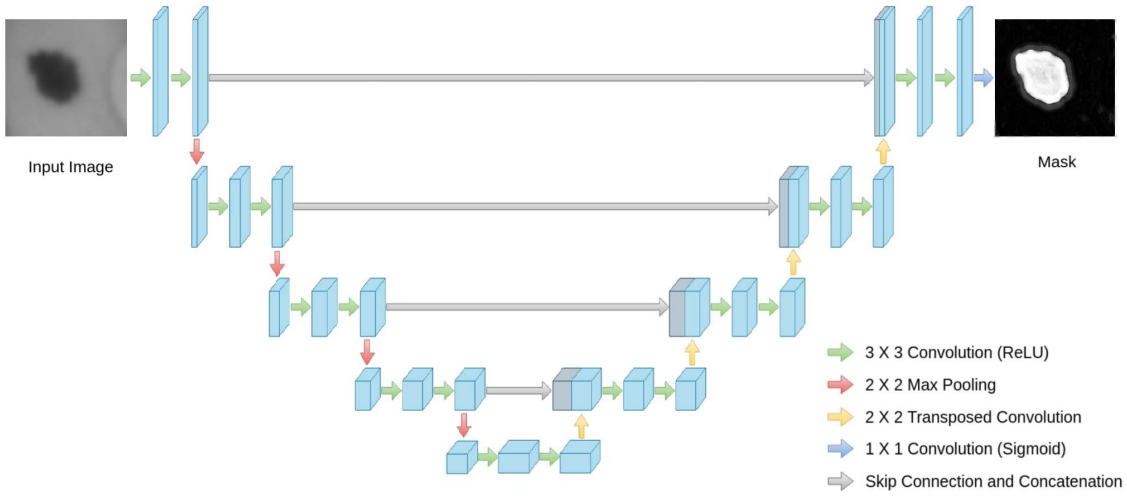


Figure III.6: Illustration of U-Net Architecture. The model consists of an encoder and a decoder pathways connected with skip connections, which concatenate the feature vectors of the first to those of the second. The output of a well-trained U-Net architecture is a greyscale mask, which is the closet representation of a binary version of the input.

block to 5^{th} block, the number of filters is 16, 32, 64, 128 and 256 respectively. These conv layers are used to extract or model features from the original image in hierarchical order.

The max-pooling layer applied at the end of each block has a kernel size of 2×2 and a stride of 2, reducing the size of the feature vector by half each time. If the size of the input image is not even before applying max-pooling, a checkerboard effect may occur during downsampling, resulting in some loss of information [102]. This was avoided in the case of our dataset with 512×512 images.

Decoder

To reconstruct a quasi-segmented mask at the output of U-Net with the same size as the original image, we need to upsample the feature vectors along the decoder to increase their size. The decoder starts after the 5^{th} block of the encoder and consists of 4 convolutional blocks, each of which contains: (1) a transposed convolutional layer with a kernel size of 2×2 and a stride of 2 like the max-pooling layers (cf. subsection I.7.5), (2) two successive convolutional layers with the same number of filters (feature maps) as their counterparts in the encoder, (3) and a concatenation layer explained below. Each convolutional layer and each transposed convolutional layer is coupled with a ReLU activation function (cf. subsection I.7.2). The latter $f(x) = max(0, x)$ sets negative voxels of each feature map to zero without further computational cost. In fact, the dice loss at the beginning of the training is always close to 1, which can lead to instability by exploding the weights gradients. Using the ReLU function after each conv layer can resolve this gradient explosion [103].

At the output of the architecture, an additional convolutional layer is used to output a greyscale mask that is closest to a binary mask, as shown in Figure III.6. This convolutional layer has a 1×1 kernel and is used to map 16 512×512 feature maps of the 9^{th} block to a single feature map of size 512×512, where each voxel has a value between 0 and 1. This layer, unlike the other convolutional layers, is coupled with a sigmoid activation function.

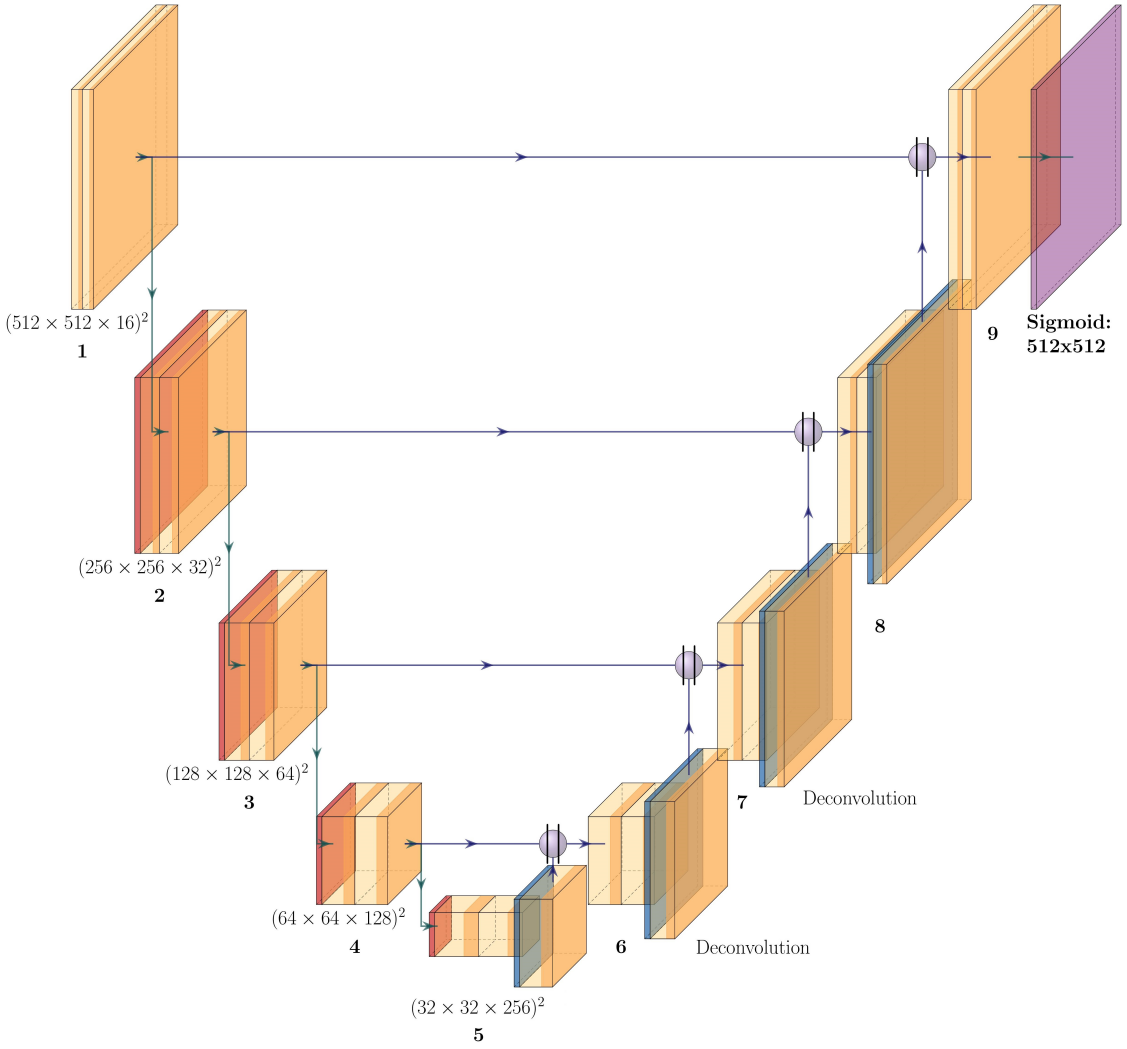


Figure III.7: U-Net architecture with 512×512 input size. Throughout the encoder, the size of the input is reduced by applying a max-pooling operation at the end of each convolutional block. And in the decoder, it is upsampled until it reaches its original size.

Skip Connections Between Encoder and Decoder

The concatenation layers, also called skip connections, come directly from the encoder, as shown in Figure III.7. For example, when an input arrives in the 6^{th} block of the decoder, its feature vector from the 4^{th} block of the encoder is sent to the 6^{th} block to be concatenated before passing through the convolutional layers. And the

feature vector from the encoder's 2nd block is concatenated to the input of block 8, and so on. It should be noted that the feature vector coming from the encoder is sent to the decoder before max-pooling is applied.

This concatenation process distinguishes U-Net from other encoders and makes it a powerful segmentation tool. It preserves contextual information that helps the decoder to more accurately reconstruct the original image as a quasi-segmented image.

III.6.0.1 Sigmoid Activation Function

The last layer of the architecture is a convolutional layer with only one filter. This layer takes the feature vector from the previous convolutional layer and outputs a new feature map with a depth of 1. Then an activation function is coupled at the output of this layer to map the voxels of this feature map to new values between 0 and 1 to reconstruct the final mask. Sigmoid activation function, shown in Figure III.8, is the right choice for this task because it ranges between 0 and 1. This function takes the feature map G_i of the last convolutional layer and outputs a greyscale image P_i by mapping the voxels as follows:

$$P_i = \Phi\left(G_i\right) = \begin{cases} 0 < p_{ij} < 0.5 & if\ g_{ij} < 0 \\ p_{ij} = 0.5 & if\ g_{ij} = 0 \\ 0.5 < p_{ij} < 1 & if\ g_{ij} > 0 \end{cases} \tag{III.7}$$

Consider a scenario where the final activation function of U-Net is a binary step function that outputs a binary image with voxels 0 and 1. In this case, the soft-dice loss would be a function with discrete values, as explained in subsection III.3.1. This would prevent the network from converging, as a loss function must be differentiable in order for the optimizer to compute the gradients and update the weights after each epoch. Outputting greyscale images whose voxels lie between 0 and 1 using the sigmoid function ensures the differentiability of the soft-dice loss function and consequently convergence to a local minimum. For example, if the voxel in the target

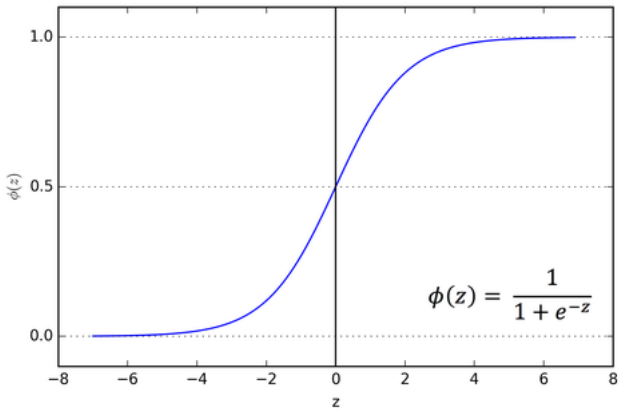


Figure III.8: Plot of the sigmoid function which exists between 0 and 1.

mask is 1 (white), the soft-dice loss function penalises the weights of the network depending on how far the voxel in the predicted mask is from 1.

III.7 Regularization

III.7.1 Dropout

To help the model generalize better and reduce the risk of overfitting (cf. section I.8), e.g. learning during training that the casting defects to be segmented have only the shapes represented in our dataset, we added a dropout layer between successive convolutional layers as a regularization method. Consequently, a percentage of the convolutional *filters* are temporarily dropped out during training by resetting them to their initial kernel values. This removes dependencies and prevents all filters from synchronously optimizing their weights [104]. The decision of which filters to drop is random and dropout layers have no parameters to be learned [105]. In the results section, we study the impact of the number of dropout layers in the architecture on the training and validation dice losses.

III.7.2 Data Augmentation

When the model is trained on a limited number of observations, it tends to over-learn from the available examples without being able to make correct predictions on new images. This challenging problem requires some kind of regularization to improve the performance of the model on new unseen inferences [106]. One such regularization technique is data augmentation (DA), which consists of artificially increasing the size of the dataset using image manipulation methods. DA is implemented through operations that change the appearance of each training instance without changing its semantics. For example, by rotating the image or flipping it horizontally and vertically. In our study, we used online augmentation, which means that the modified versions of each original instance were generated in real time during training. This ensures that the network is trained on different images during each epoch, as these new images, which should be as realistic as possible, are not stored in memory and are never generated more than once.

The transformations used include random zooming into the image by a factor between $0-0.2$, random rotation by a value between $0°-30°$, horizontal and vertical shifts by a factor between $0-0.1$, shearing by a factor between $0-0.15$ and vertical or horizontal flipping. These techniques help to make the model more tolerant to variations in position, orientation and size of discontinuities in CT slices. It is important to note that we did not use a transformation that changes the greyscale values in order to preserve the texture information. During training, transformation operations are randomly combined to augment the dataset by generating new 512×512 images *with their corresponding masks*, as shown in Figure III.9.

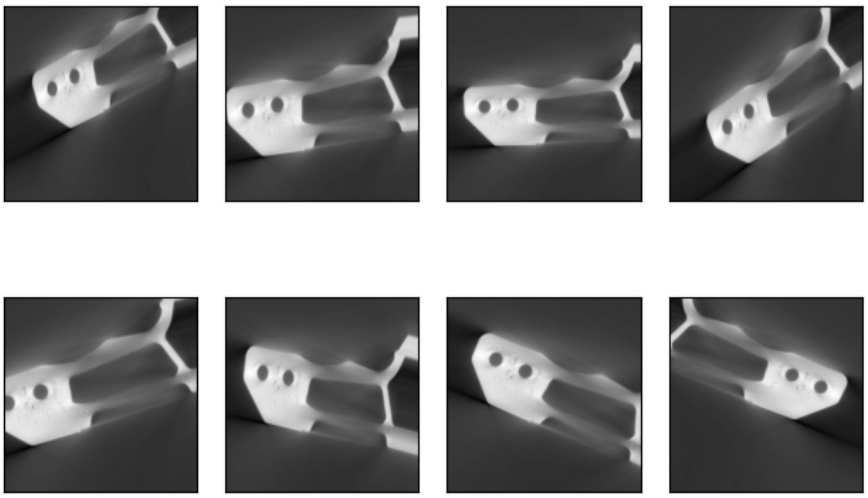


Figure III.9: Generation of new instances from the same image via random data augmentation.

III.7.3 Hyperparameters

Training U-Net to correctly segment CT slices requires a series of trials in which the hyperparameters are tweaked interchangeably. Hyperparameters are not model parameters, such as the weights of the layers, and they cannot be trained directly on the data. The model parameters are learned during training when the weights are updated based on the loss function using an optimizer, while the hyperparameters are manually tweaked before training begins. Since we decided to keep the same architecture of U-Net as in the original paper, the remaining hyperparameters that can be tweaked are: initial learning rates of the optimizer, number of epochs, batch size, number of filters per convolutional layer, kernels weights initialization algorithm, and dropout layer rate. Finding the optimal set of hyperparameters, included stopping training, changing the hyperparameters and restarting training again until the training and validation dice losses were as close to zero as possible. The options for setting these hyperparameters are default values from the software package, manual configuration by the user, or configuration for optimal predictive performance through a tuning procedure.

III.8 Performance Results

As mentioned in subsection III.2.1, during each training epoch, U-Net was trained on 2400 augmented images, and at the end of the epoch it validates itself on 600 images. The goal is to teach U-Net to output a greyscale image that resembles the target masks as closely as possible. Therefore, the average training and validation losses at the end of each epoch were recorded to monitor the training and stop it if there were signs of overfitting or underfitting. As can be seen from the training curves in Figure III.10, the latter phenomenon occurred because the validation dice

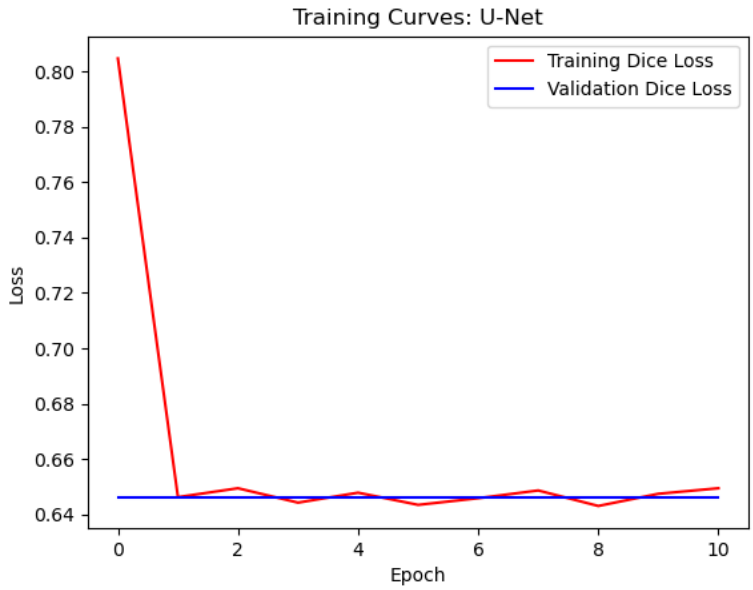


Figure III.10: Evolution of dice loss evolution on training and validation data with respect to training epochs. The stagnation of the validation loss at 0.6459 is a sign of underfitting, which means that the model could not map the underlying relationship between the training images and the corresponding binary target masks.

loss did not show any improvement during the training, no matter how many times we changed the hyperparameters before starting the training.

This underfitting could be due to the fact that the CT images in the dataset contain various artifacts, as explained in section I.3. In most cases, these artifacts may have the same contrast range as the defects, creating an ambiguity that could prevent the model from distinguishing the right voxels to be segmented. In Figure III.11 we see that a defect and a region filled with streaks have the same greyscale ranges, and only the defects are supposed to be segmented in the corresponding binary target mask.

After a number of failed attempts, we decided to change the approach to CT slices segmentation by following the steps below:

1. Train U-Net with over-segmented images as explained in the following section. This will teach U-Net to recognise suspicious greyscale discontinuity that could belong to a real defect, artifacts, noise or even geometrical borders.
2. Send these segmented discontinuities to a CNN classifier, whose training is presented in the following chapter, to decide whether the discontinuities belong to true alarms (defects) or false alarms.

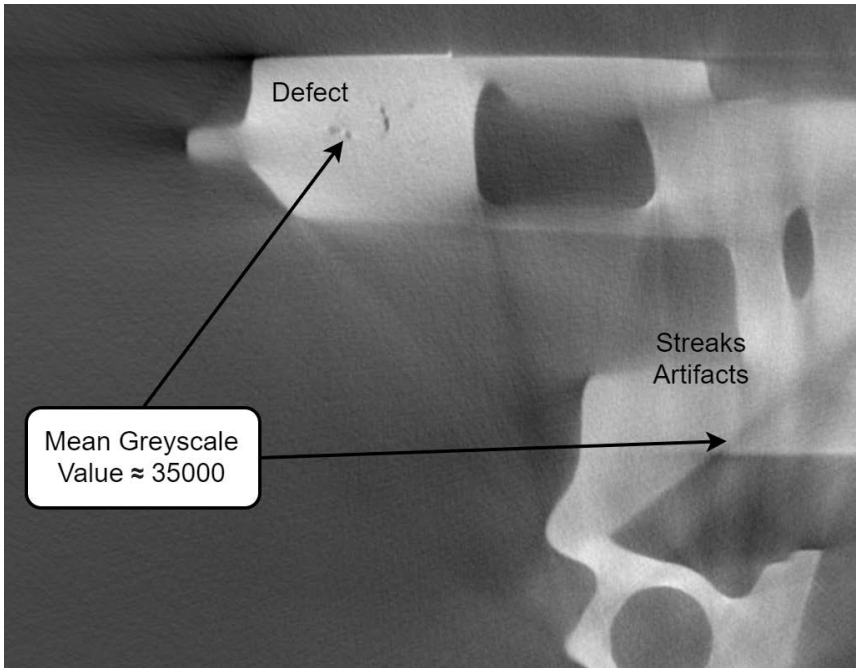


Figure III.11: 16-bit greyscale CT image from the training dataset. Both artifacts and defects voxels have close contrast or greyscale values.

III.9 Training U-Net with Over-Segmented Images

III.9.1 Dataset

For this dataset, we segmented the same greyscale 512×512 images of the dataset individually using the 2D segmentation algorithm of section II.3, but with a lower fudge factor. A sample of this dataset is shown in Figure III.12. – compare with Figure III.3. Each mask contains the following classes of voxels:

- Black voxels, which belong to non-defective zones filled with material or the empty zone outside the specimen.
- White voxels, which belong to suspicious discontinuities such as defects, artifacts, or even the abrupt change of greyscale at the borders of the specimen.

The fudge factor for each image was carefully chosen to include all voxels belonging to such discontinuities. A side effect of such over-segmentation is that U-Net learns to produce similar results when used to segment new images. On the other hand, this has the advantage of reducing the false negative rate, i.e. the number of undetected or missed discontinuities that could belong to real defects. **It should be noted that over-segmentation does not mean that the shape of the defects is over-extended, but that there is an excess of discontinuities.** Furthermore, these additional false alarms are expected to be eliminated using trained classifiers, as described in the following chapters.

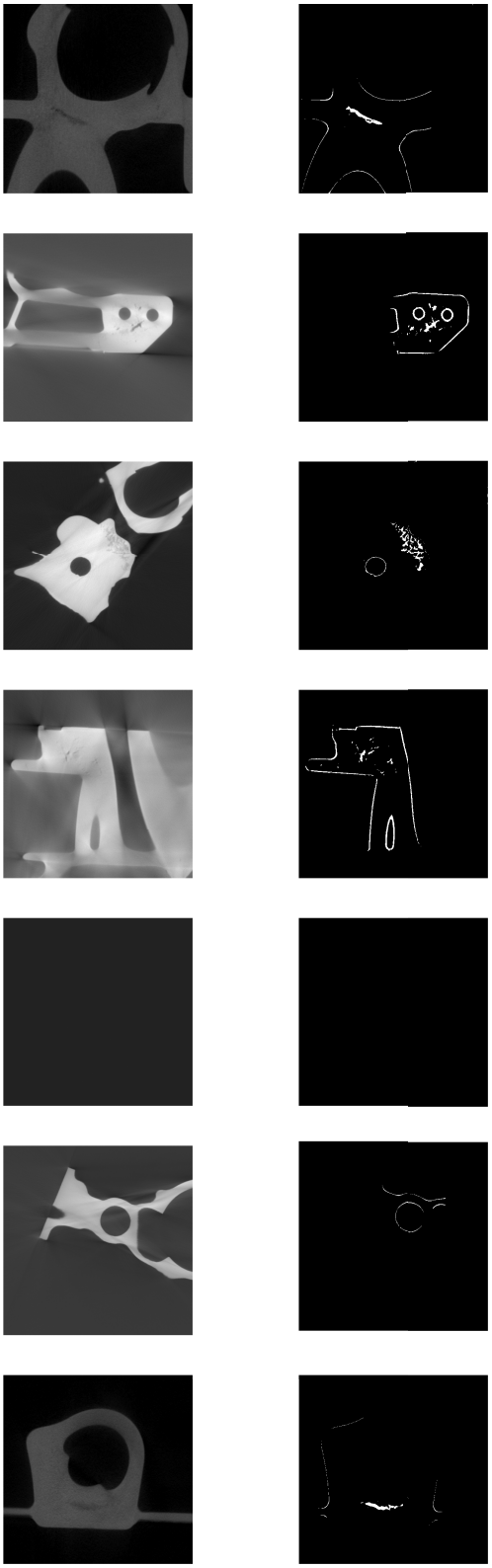


Figure III.12: sample of images from the segmentation database of 3000 images: (Left) 512x512 images cropped from the CT volumes, (Right) their target masks where defects, borders and false alarms are segmented. This over-segmentation of the dataset is intended to make the model sensitive to slight discontinuities.

III.9.2 Training Results

Training a deep learning model requires monitoring the loss values and restarting the training process if the results are not satisfactory after tweaking the hyperparameters interchangeably,as explained in subsection III.7.3. Table III.1 gives the final values or choices of hyperparameters that led to the best training results.

Table III.1: U-Net training hyperparameters that gave best training results on CT segmentation dataset. The final values were obtained after a series of interchangeable tweaking.

Hyperparameter	**Final Value or Choice**
Loss Function	Soft-Dice Loss
Optimizer Choice	adam
Optimizer Learning Rate	10^{-3}
Batch Size	12
Dropout Rate	0.2
Encoder: filters in Conv layers	16, 32, 64, 128, 256 consecutively
Decoder: filters in Conv layers	128, 64, 32, 16, 1 consecutively
Filters Initialization Algorithm	"He algorithm" suitable for layers with ReLU [52]
Epoch	Training is stopped when validation loss stops improving

Although we decided to use the same layers in U-Net as in the original paper, we changed the number of dropout layers to study their impact on training and validation losses. Table III.2 shows the dice loss values depending on the number of dropouts in the encoder and decoder. In each case, the number of dropout layers in the contractive path (encoder) is equal to the number of dropouts in the expansive path (decoder) to maintain the symmetrical property of the architecture. According to the results, completely removing the dropout layers from the architecture has a disadvantage for the dice loss. As explained in subsection III.7.1, the dropout layers prevent the convolutional filters from synchronously optimizing their weights and thus provide better generalization by letting the network rely only on robust features in the feature vectors. On the other hand, adding 4 dropout layers did not lead to the best results, as it causes the network to lose the contextual information about the images that is essential for reconstructing the quasi-segmented version of the input.

The optimal architecture contains 2 dropouts in the encoder and 2 more in the decoder: one dropout layer is introduced between two convolutional layers in blocks 2^{nd} and 4^{th} of the encoder, and blocks 6^{th} and 8^{th} of the decoder (cf. Figure III.7). With the set of hyperparameters in Table III.1, the architecture has achieved a training loss of 0.1231 and a validation loss of 0.1456 after 72 training epochs. Figure III.13 illustrates the evolution of the training and validation losses over the course of training.

The spikes on the curves are caused by mini-batch weights updates by adam optim-

Table III.2: The impact of dropout layers in the contractive path (encoder) and the expansive path (decoder) on training results. Adding two dropouts to each path achieved the lowest training and validation dice losses.

Number of dropout layers per path	Training dice loss	Validation dice loss	Number of epochs (before stagnation)
0	0.1335	0.1612	32
1	0.1272	0.1670	35
2	**0.1231**	**0.1456**	72
3	0.1263	0.1471	42
4	0.1519	0.1756	30

izer. During each epoch, data is sent to the network in batches of 12 and at the end of each batch adam optimizer updates the weights of the network based on the dice loss value. In some batches, there are training instances that have more discontinuities belonging to geometrical borders compared to other images (cf. Figure III.3). This difficulty manifests itself on the training curve by a sudden increase in the dice loss, or spikes. From epoch 16, the model started to adapt to this particularity in the dataset which was intended to reduce the number of missed defects.

Conclusion

This chapter explored the potential of U-Net as a segmentation tool for CT images. Due to the presence of artifacts in CT, the model was not able to learn from

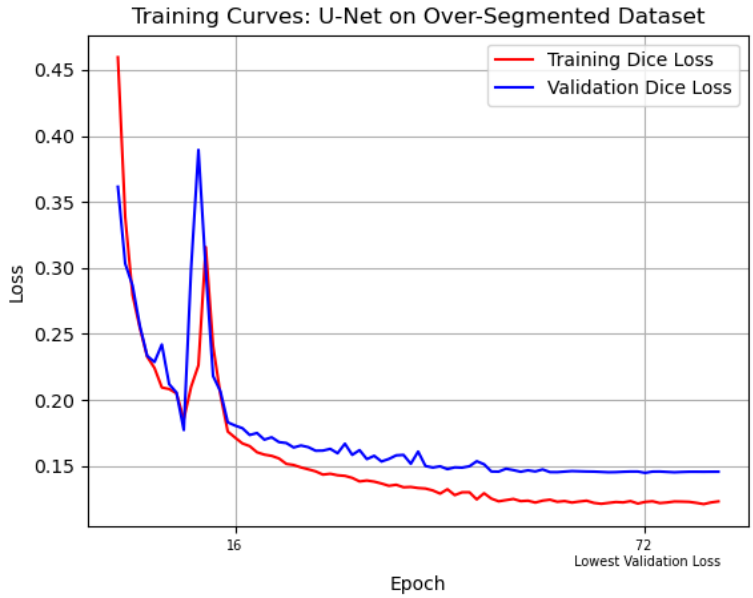


Figure III.13: Evolution of training and validation losses over training epochs. The lowest validation dice loss was achieved after 72 training epochs.

the manually segmented dataset to minutely segment CT slices. In contrast, when the model was trained on an oversegmented dataset, it achieved higher training and validation losses and shifted the segmentation task from detecting defects to detecting discontinuities that could belong to a defect, artifact, noise or even the borders of the specimen.

In the next chapter, we present a CNN classifier that can be coupled with U-Net to distinguish true defects from false alarms after it has been trained on a binary dataset. In chapter V, we explain the implementation details and validate this coupling approach on CT volumes of castings.

Chapter IV

CNN for Defects Recognition in Industrial CT Images

In chapter II, we have proposed a 2D segmentation algorithm that can be used on 2D images after slicing the CT volumes along a certain direction. This algorithm outputs the discontinuities that could represent real defects if the fudge factor is set correctly. This means that the algorithm always requires human intervention to avoid artifacts or noise segmentation. In chapter III, we trained U-Net model to over-segment CT images, highlighting casting defects and reducing false negatives. Unlike the 2D segmentation algorithm, U-Net does not require any parameter to be adjusted, even when applied to images from different volumes. The only drawback is the amount of false positives in the output that do not belong to real defects.

In this chapter we propose an automatic classifier that can be combined with U-Net to reduce false alarms. Once the discontinuities are located by U-Net, a 64×64 greyscale ROI (and not binary) is cropped around each discontinuity to classify it using a trained deep learning algorithm, specifically a convolutional neural network

(CNN). This may be a new architecture trained from ground up, or a pre-trained state-of-the-art model adapted for this classification task.

In this chapter we will discuss the following: (i) building a diverse dataset from 2D CT images of high-pressure and gravity castings, (ii) training and optimizing a new convolutional network from ground up, (iii) comparing this new model with deep state-of-the-art models and demonstrating that high performance can be achieved with a relatively less deep architecture.

IV.1 Classification Task

Classification is a process of dividing a given set of data into classes, often called targets, or labels. In the last decade, deep learning models such as convolutional neural networks have emerged as the most powerful solutions for modern object classification [107, 108]. The goal of this chapter is to use CNN to classify 64×64 CT images into true alarms or false alarms after learning from a binary dataset.

Our classification task consists of learning to classify x_i into k classes. x_i is an image from a dataset of length N, i.e. $i = 1, ..., N$. Each image x_i is an array of length M of voxels, and each voxel value, also called feature, can be represented as x_{ij} with $j = 1, ..., M$. The CNN must output a vector p_i associated with x_i, where p_i is the probability distribution over the possible classes. In our classification task, $k \in 1, 2$ represents the number of classes, where 1 is the false alarm class (normal zone or artifact) and 2 is the true alarm class (defect). In practice, these classes are converted into array-like categorical variables such that 1 is encoded as $[1, 0]$ and 2 as $[0, 1]$.

Given these variables, we can now define the mapping function $f : R_n \rightarrow 1, ..., k$, which maps each image to a vector that determines the probability of distribution after applying an activation function. The mapping function f can be represented as follows:

$$p_i = f(x_i; W; b) = W \cdot x_i + b = \sum w_m \cdot x_{im} + b \tag{IV.1}$$

The weight matrix, denoted $\mathbf{W}$, and the bias vector b are referred to as the parameters of the network. These parameters should be optimized throughout the training process until the network is able to output the best probability distribution. The latter is a vector whose values represent the membership probability to each class, $y_i = [y_{i1}, y_{i2}]$. For example, if the second value of this vector is the maximum probability value, it means that the image belongs to class 2 (true alarm).

IV.2 Learning Experience

Since we want to train a CNN to classify small regions in CT slices into true or false alarms, we built a binary dataset in which each data point x_i is associated with a target label y_i that indicates whether x_i represents a defect or not. As the labels are provided, this makes the learning experience supervised as the model observes

multiple examples x_i with their assigned labels y_i and learns to predict y_i from x_i by estimating $p_i(y_i \mid x_i)$.

IV.2.1 Training, Validation and Test Data

In metal casting, aluminium specimens can be made either by injecting molten metal into a mould under high pressure or by cooling the metal in a mould under the force of gravity. As explained in subsection I.5.1, the data available for this study include CT volumes of different specimens fabricated by one of these processes. These specimens are very diverse in terms of thickness and shape and mostly represent mechanical parts from the automotive and aerospace industries that have been inspected using different CT systems. Due to this diversity, each volume was scanned under different imaging conditions in terms of source voltage and current and magnification factor (cf. section I.2).

Consequently, a total of 20 CT volumes have been cropped to build a binary dataset to train the neural networks in this chapter. Although this diversity poses a challenge for learning, it should help the networks to generalize better and avoid overfitting by training them on CT images that have different contrasts, spatial resolutions and artifacts. Building the dataset involved two steps:

1. Each 3D volume was sliced along the three axes to generate series of 2D slices as explained in chapter II.
2. Several defective and normal zones of 64×64 were cropped manually from each slice. The choice of 64×64 as cropping size served the purpose of capturing the extended shape of any potential casting defect in CT slices, given a spatial resolution between 150 and 450 μm.

Figure IV.1 shows a typical 2D CT slice of a casting specimen where different "to-be-cropped" zones are highlighted:

- Cropped casting defects are mostly porosities and shrinkage cavities, and they were labelled as "True Alarm".
- Normal zones may be empty zones outside the casting (black zones), and/or homogeneous grey zones of material representing the body of the casting. Typical tomography artifacts such as streaks and ring artifacts are also considered normal zones as they are generated after the 3D reconstruction of the CT images and have no physical existence (cf. section I.3). Since these crops do not represent a defect, these zones were labelled as "False Alarm".

It is important to note that the black background may be included in the "true alarm" images, as some defects may occur near the borders of the specimen. On the other hand, artifacts in CT volumes are very peculiar and depend on the scanning system. Ring artefacts occur, for example, when the sensitive elements of the X-ray detector respond differently to photon energy. For thick castings, the calibration of the CT system is not very efficient due to the difference between the calibration energy

and the X-ray spectrum reaching the detector after passing through the specimen, resulting in coaxial rings along the axis of rotation, as shown in Figure IV.1. Another example of artifacts are the streaks resulting from scattered X-rays or caused by large thickness differences within the specimen or by the presence of materials with high attenuation coefficients compared to aluminium (cf. Figure I.6). With typical image processing techniques, these artifacts could be incorrectly segmented as defects due to their high contrast compared to their surrounding regions. Since their shapes are relatively regular compared to the casting defects, different types of artifacts were included in the dataset and the neural network models were trained to recognize them as normal zones, "false alarms".

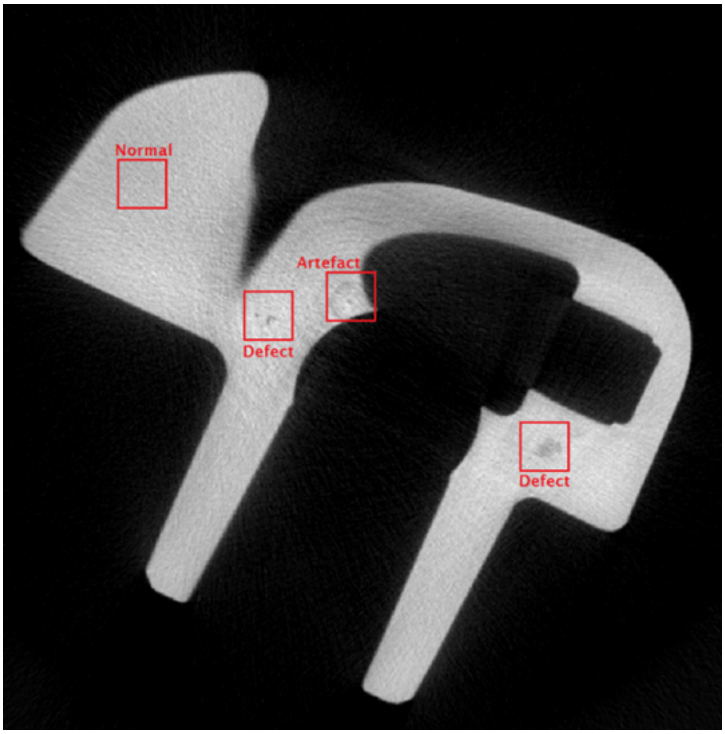


Figure IV.1: Two defects and a ring artifact highlighted in a CT slice, as well as a normal non-defective zone. The training dataset is built by cropping 64×64 images around such zones.

A sample from this dataset is illustrated in Figure IV.2. At the end of the cropping process, we built a binary dataset with the following number of 64×64 images:

- 5500 true alarms, i.e. defective regions.
- 8500 false alarms, i.e. normal regions, artifacts or noise.

As explained in section I.8, training a neural network on a dataset requires splitting the data for each phase of the learning process. 80% of the images in the dataset are used as the training set, and 20% as the validation set, with a batch size of 32. The latter is a hyperparameter that specifies the number of images the model must observe before updating the weights of the network using an optimizer, as

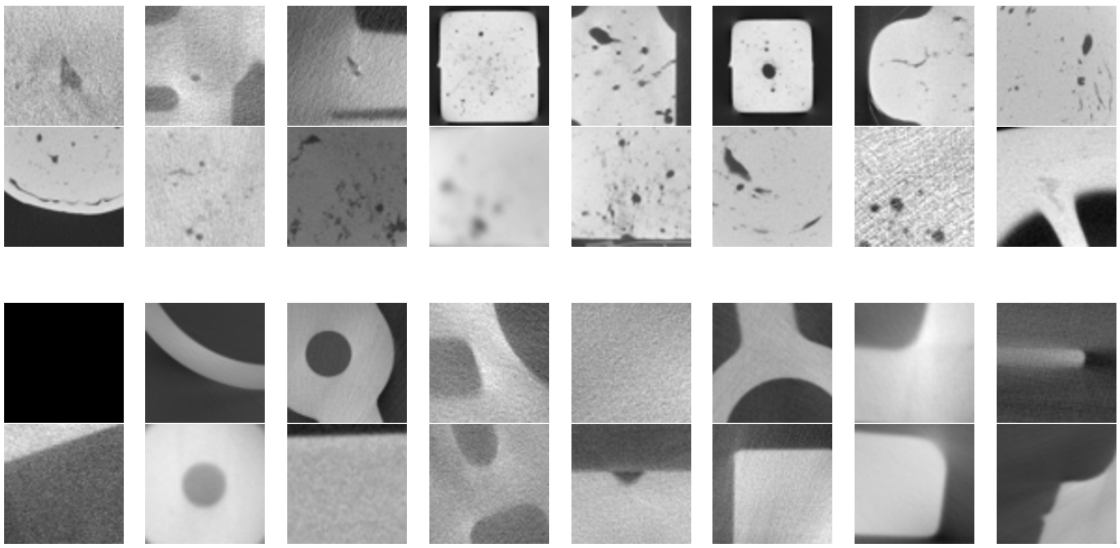


Figure IV.2: A sample from the binary casting dataset: (top) defects, (bottom) false alarms. This dataset is very diverse in terms of contrast and spatial resolution.

explained in section III.4. After training is done and the models are saved, each can be tested on a new vector of images called the test vector. This vector contains 350 images representing each class (700 in total) cropped from new volumes. This test evaluates the models' ability to recognize defects as true alarms and artifacts and normal zones as false alarms by plotting the confusion matrix.

It was decided to work with 2D CT slices instead of using 3D CT volumes directly as input. Although the defects in CT have 3D geometries, building a dataset of 3D zones requires a large number of defect samples. In practice, to increase the number of 2D 64×64 images in the above dataset, we sometimes cropped multiple zones from the same defect along different slicing axes. Conversely, working in 3D means that the entire defect is considered as one data point, which limits the size of a possible 3D dataset and hinders efficient training.

IV.3 Performance Measurement

To select the best CNN architecture and evaluate its learning ability, we need to measure its performance quantitatively. In the case of a classification task, accuracy is a reliable parameter that measures the proportion of examples for which the model predicted the correct labels. Another important parameter is the loss function, which indicates the magnitude of the prediction error that the model made during training and validation, as explained in section I.8.

On the other hand, to evaluate the predictive ability after training, the confusion matrix can be used to calculate the F1-score on a new set of images. This parameter helps to compare the predicted labels p_i with the ground truth labels y_i of a small

dataset that the model did not observe during training.

IV.3.1 Evaluation Metrics

Accuracy is the intuitive metric that can be used to measure the learning ability of a model during training. In binary classification, accuracy can be calculated in terms of positive and negative results as follows:

$$Accuracy = \frac{TP \ + \ TN}{TP \ + \ TN \ + \ FP \ + \ FN}, \ where \tag{IV.2}$$

TP = True Positives, correctly classified defects,
TN = True Negatives, correctly classified normal zones,
FP = False Positives, normal zones misclassified as defects,
FN = False Negatives, defects misclassified as normal zones.

After training is done and the model is saved, the performance of the model on new unseen data points can be measured using **F1-score**. This metric is established by measuring the precision and recall of the model given a vector of new data points. These two parameters can be explained as follows:

- When the possibility of false positives is very high, as is the case in our classification task where artifacts and noise could be mistaken for defects, we can use precision to measure how often the model is correct in predicting positive outcomes.

$$Precision = \frac{True \ Positives}{True \ Positives \ + \ False \ Positives} \tag{IV.3}$$

- With the same understanding, recall measures the model's ability to identify true positive outcomes (defects). This metric is useful when the cost of missing defects is high, as its denominator takes into account the detected defects (TP) and the missed defects (FN).

$$Recall = \frac{True \ Positives}{True \ Positives \ + \ False \ Negatives} \tag{IV.4}$$

F1-score combines precision and recall. When testing the model on a new dataset, a high F1-score means that the model predicted few false positives and few false negatives, i.e. this metric measures the performance of the model, considering all classification categories equally [109]. This metric is used to evaluate the models on new inferences (unseen images).

$$F1 - Score = 2 \ \frac{Precision \ \times \ Recall}{Precision \ + \ Recall} \tag{IV.5}$$

IV.3.2 Loss Function: Binary Cross Entropy

As explained in subsection III.3.1, a loss function indicates how efficient a model is at classifying the data points in the training dataset. The smaller the loss, the better the model at mapping the relationship between the input images and the output labels (false or true alarm). Conversely, a large loss value means that we need to optimize the training parameters to improve the training process [110, 111]. In binary classification tasks, the most common loss function is **binary cross entropy**, which can be described as the average log-likelihood that the model assigns to the training data points. In information theory, entropy is considered a measure of uncertainty. For this reason, it is logical to logarithmically transform what we are uncertain about; and in our classification task, uncertainty is represented as the predicted probabilities.

Binary cross entropy log loss compares the predicted probabilities $p_i = [p_{i1}, p_{i2}]$ to the real label $y_i = [y_{i1}, y_{i2}]$ of the input. As mentioned earlier, the latter can be either $[1, 0]$ = "false alarm" or $[0, 1]$ = "true alarm". The equation of this log loss over a dataset of length N is as follows:

$$BCE = \frac{1}{N}\sum_{i=1}^{N} -[y_i \; log(p_i) + (1 - y_i) \times log(1 - p_i)] \tag{IV.6}$$

If the model predicts that an observation should have a higher membership probability to the first class (e.g. $[0.98, 0.02]$), a large penalty will occur if the true label is $[0, 1]$. On the other hand, a smaller penalty will occur if the true label of the input is $[1, 0]$. In other words, the binary cross entropy loss calculates the score that penalizes the probabilities based on how close or far the predicted probabilities $[p_{i1}, p_{i2}]$ are from the target values $[y_{i1}, y_{i2}]$. To summarize the performance metrics, the accuracy and loss function are used while training the network. And once the model is trained and saved, the F1-score is used to evaluate the model on a vector of new images.

IV.4 SGD Optimizer

Gradient descent [112] is a famous optimization algorithm that works over the loss landscape, as shown in Figure III.4. After many trial-error attempts, stochastic gradient descent (SGD) optimizer [113] with momentum was used with each model in this study to find the weights of the network that need to be updated, at the end of each training epoch. The momentum parameter accelerates the updating of the gradients towards a local minimum, leading to even faster convergence. For the t^{th} iteration, the weights update is denoted by Equation IV.7, where θ_t denotes the current weight update and θ_{t-1} the previous update. W represents the weights, E is the average loss over the dataset, and $\nabla E(W)$ is the negative gradient. α is the learning rate and $\gamma \in [0, 1]$ represents the momentum used for speeding up the

convergence of the gradient and preventing oscillations [114]:

$$\begin{aligned} \theta_t &= \gamma\theta_{t-1} - \alpha\nabla E(W_{t-1}) \\ W_t &= W_{t-1} + \theta_t \end{aligned} \tag{IV.7}$$

The initial value of the learning rate α was set to 0.01 and the momentum γ to 0.9. Using the callback "reduce learning rate on plateau" in the Keras library, the learning rate is automatically reduced by a factor of 0.1 each time the validation loss stagnates over five consecutive epochs.

IV.5 State-of-the-Art Models

To achieve high performance, deep convolutional neural networks require a large dataset of labelled images, which can be a costly process in many domains. One possible solution to this challenge is to use one of the pre-trained "state-of-the-art" CNNs, such as VGG19, GoogLeNet, AlexNet, etc. Each of these networks has been trained on a very large dataset consisting of natural images, such as ImageNet [115]. Consequently, their convolutional layers have very sophisticated filters that aptly respond to different details in the images. The learning ability of these convolutional filters could be applied to a new classification task through a procedure called transfer learning [116].

Transferring the learning of these models to a new classification task requires fine-tuning the weights of these models, **without initialization**, with a new dataset, usually similar to ImageNet. And before conducting fine-tuning, the last block of layers must be replaced to prevent the model from classifying a new image according to its previous configuration. The last block, called FC head, consists of fully connected layers (cf. subsection I.7.4). In Figure IV.3 we see the FC head of a very famous deep state-of-the-art model VGG19.

After replacing the FC head, we can start fine-tuning by adjusting the hyperparameters until we find the best configuration. This process consists of two steps:

1. Warming-up, in which the model is trained on the new dataset with frozen weights except those of the FC head layers. This prevents the gradients of the newly initialized layers from propagating backwards through the entire network and weakening the powerful filters of the convolutional layers.
2. Fine-tuning, where the model is completely unfrozen and its weights are updated with a very small learning rate so that the original convolutional filters do not deviate dramatically.

When the dataset was still under construction, we fine-tuned VGG19 architecture with our casting dataset **without initialising its weights**. Since ImageNet is an RGB dataset, this required a preprocessing step where each image in the dataset was converted from greyscale to RGB by repeating each image three times along the channel dimension. As a result, the fine-tuned VGG19 achieved near-perfect

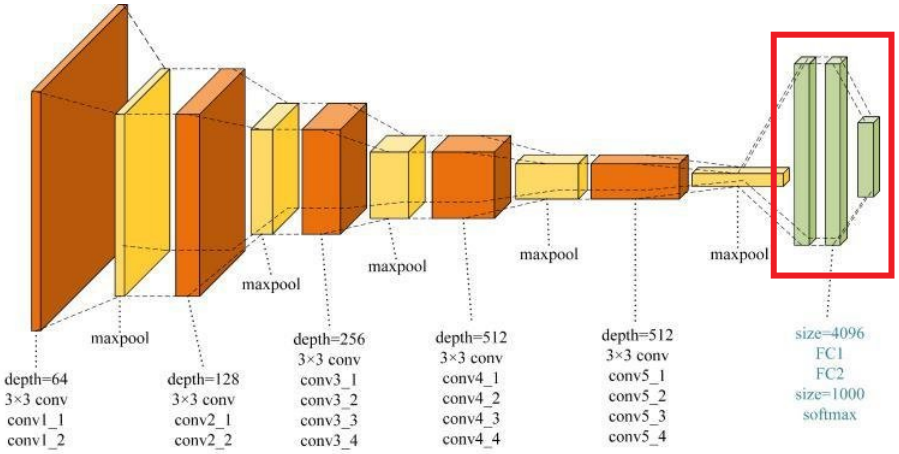


Figure IV.3: Fine-tuning consists of removing the original FC layers, highlighted in red in VGG19 architecture, and replacing them with new layers that are better adapted to the new classification task. Source: [117]

performance, suggesting overfitting as the model sees each voxel three times during training. An alternative solution was to develop a new architecture and train it from ground up with our casting dataset.

To support the claim that CNNs can be used as classifiers to distinguish greyscale discontinuities in CT slices of casting volumes and that, when trained on a diverse dataset, they can correctly distinguish artifacts from true defects, we compared this new model with five state-of-the-art models after changing their inputs to greyscale images, modifying their FC heads and **initializing their weights**. This means that we trained these models from ground up without transfer learning, just like the new model. To show that a dataset of greyscale CT images does not require much depth, the state-of-the-art models were selected based on their depth to compare performance.

IV.6 Building a New CNN Architecture

The process of finding an efficient convolutional architecture begins with selecting an arbitrary design and modifying it progressively according to learning losses and accuracies. In the early stages of training, the loss function was very high when the dataset was still under reconstruction. This is due to the fact that our dataset contains two types of defective images, as shown in Figure IV.4: (a) images with uniform background, a normal zone around the defect filled with material; and (b) images with multiple backgrounds, a normal zone around the defect and a dark zone outside the casting. The contrasts of the defect and the dark background (outside the specimen) are of the same range. Consequently, this ambiguity slowed down the improvement of accuracy throughout the optimization process until we increased the size of the dataset to solve this problem.

Deep learning models have many hyperparameters that control the learning process,

Figure IV.4: Two types of defective images in the dataset: (a) defect surrounded by a uniform background, (b) defect surrounded by a uniform background and a dark zone outside the casting body.

as explained in subsection III.7.3. These parameters have a great impact on the training process as well as on the training time. The list of hyperparameters includes: learning rate of the optimizer, number of epochs, batch size, number of layers, number of filters per convolutional layer, size and stride of convolutional filters, number of neurons per FC layer, weights initialization algorithm, dropout rate and choice of activation function. Finding the optimal architecture shown in Table IV.1 involved a long series of hyperparameters adjustment and tweaking while observing the training loss and accuracy curves after each training attempt.

The first factor to consider when designing an architecture is the number of layers. Increasing the depth of the network could lead to an increase in the number of learning parameters, which in some cases could lead to overfitting, not to mention higher space and time complexity. For this reason, we started with a very shallow architecture, and then increased the number of layers progressively while observing the improvement of loss and accuracy. In the following sections, we explain the selection of each component that makes up our new architecture of 10 layers, named **CT -Casting-Net**.

Table IV.1: The optimal architecture of the CT-Casting-Net after many trial-error attempts for an input size of $64 \times 64 \times 1$, with the shape of the output of each layer, as well as the number of filters or units.

#Layer	Layer Type	Filter Size	Number of Filters or Units	Output Shape
1	Conv2D + ELU	5x5	64	64, 64, 64
2	MaxPooling2D	2x2	64	32, 32, 64
3	Conv2D + ELU	5x5	128	32, 32, 128
4	MaxPooling2D	2x2	128	16, 16, 128
5	Conv2D + ELU	5x5	256	16, 16, 256
6	MaxPooling2D	2x2	256	8, 8, 256
7	Conv2D + ReLU + Dropout	5x5	512	8, 8, 512
8	MaxPooling2D	2x2	512	4, 4, 512
9	Flatten + FC + ReLU	-	256	256
10	FC + Softmax	-	2	2

IV.6.1 Convolutional Layers

Convolutional filters are applied to the training images to look for zones that might trigger class activation, i.e. a discontinuity representing a casting defect. Each image is convolved with multiple filters whose weights are initialized with the "He algorithm" suitable for layers with ReLU activation function [52]. The He function draws samples from a truncated normal distribution centred at 0 with a standard deviation $\sigma = \sqrt{\frac{2}{t_{in}}}$, where t_{in} is the number of filters in the layer.

One of the most influential hyperparameters besides the number of layers is the number of filters per convolutional layer. Decreasing this parameter makes the neural network too shallow to detect the class activation zones, and increasing it provokes a time complexity. Each filter convolves the feature map coming from the previous layer and looks for a shape that could be voted as a defect. Increasing the number of filters implies that the number of votes that decide whether the zone is defective or not also increases. To reduce the processing time and to avoid detecting noisy trends in the images, the comparison was stopped at $[64, 128, 256, 512]$.

In CT-Casting-Net architecture, after many trial-error attempts, four convolutional layers with filter sizes of (5×5) were used. Small filters were used in the convolutional layers because the training images are relatively small (64×64 voxels) and because defects in casting CT images, such as porosity, are usually small. A stride of 1×1 was used, which corresponds to the scanning step of the kernel over the image, meaning that a convolutional layer outputs feature maps of the same size as the input image.

IV.6.2 Activation Layers: ReLU

As explained in section IV.4, SGD optimizer updates the weights of the network, including the values of the convolutional kernels or the weights of the FC layers, by computing the gradient of these weights with respect to the loss value. However, when the loss improves slowly, the weights are updated by only a small amount leading to a vanishing gradient and very slow convergence. One solution to this vanishing gradient problem is to use ReLU (rectified linear unit) as the activation function [107].

When the accuracy started to stagnate after many training trials, the ReLU function was replaced by ELU in the first two convolutional layer and the training process was restarted with the same hyperparameters. As shown in Figure I.14, ELU does not have the ReLU dying problem, i.e. setting the negative pixels (or voxels) of the feature maps to zero. This advantage ensures that the loss converges faster and gives more accurate results.

IV.6.3 Max-Pooling Layers

After each convolutional layer, a max-pooling operation with a kernel size of 2×2 is applied. The architecture includes 4 max-pooling layers with the same kernel. Each of these layers creates a new reduced matrix by dividing each feature map into non-overlapping grids and then taking the maximum value in each grid [107]. Combining responses at different locations increases robustness to small spatial variations, and reducing the size of the feature maps reduces the learning parameters of the network and speeds up the learning process.

IV.6.4 Fully-Connected Layers

After extracting features using a series of convolutional blocks, we need to classify the input into one of the classes. This can be done using a fully connected (FC) layers of neurons or units, explained in subsection I.7.4. All feature maps of the same input image are flattened into a one-dimensional vector and passed to a fully-connected layer, also called dense layer, as shown in Figure IV.5. Like the convolutional layers, FC layers are also coupled with an activation function, and these layers form together the classifier block. Using the SGD optimizer, the weights of this block are updated during the training process until the classifier block correctly assigns the feature maps of each image to the correct class (true alarm or false alarm).

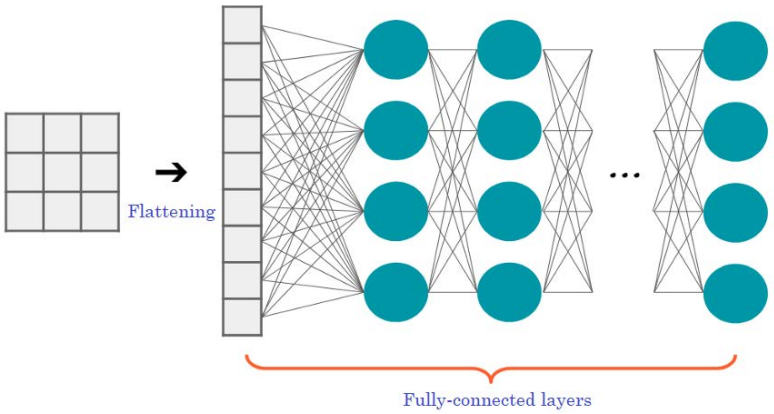


Figure IV.5: The output of the convolutional layers is flattened to create a single long feature vector. This is then sent to a block of fully-connected layers called classifier block, which decides the class of the input image.

After four convolutional blocks, the feature maps are flattened and sent to an FC layer with 256 units, coupled with a ReLU. The last layer of the architecture is also a FC layer with a number of units equal to the number of classes (two in our case). This layer is coupled with a softmax activation function that maps the output to a vector $[p_{i1}, p_{i2}]$ so that the total sum is 1. In other words, the output of softmax is a probability distribution over the possible classes, as shown in Figure IV.6.

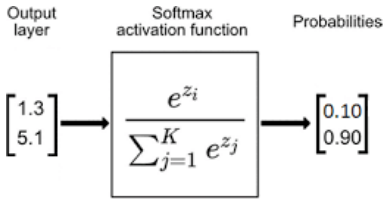


Figure IV.6: Softmax converts the output of the last FC layer into a vector of membership probabilities that sum to 1 over classes.

IV.6.5 Regularization

IV.6.5.1 Dropout

In the last convolutional block, a dropout layer was added between the convolutional layer and the max-pooling layer with a rate of 35% as shown in Table IV.1. This means that before applying convolution, 35% of the filters (randomly selected) are set to their initial kernel values [105]. This type of regularization improves the learning process by reducing the noisy trends that could be learned by the convolution layer filters that come after the dropout. Indeed, when looking at the feature-maps before adding the dropout in Figure IV.7, it is clear that the convolutional filters were triggered by the defects and the borders of the images. This is an example of a noisy trend learned by the network that can lead to overfitting if no regularization is applied.

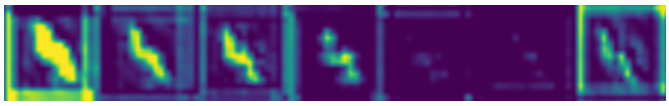


Figure IV.7: Some feature-maps before adding dropout. The convolutional filters detected the irregular shape of the defects, as well as the noisy trends at the borders.

IV.6.5.2 Data Augmentation

As explained in subsection III.7.2, one of the most common ways to improve the generalization of a CNN model is to train it with more data. In practice, obtaining and labelling more data is an expensive and difficult process. To get around this problem, we can create fake data by twisting and modifying the existing data, which is very beneficial in a classification task. The CNN is supposed to map a high-dimensional input x_i into a probability distribution p_i over the possible classes. When x_i is modified by applying aspect transformations, the model learns not to rely on the particularity of the original example, improving its ability to generalize and prevents overfitting. We applied the same transformations of subsection III.7.2 to each greyscale training instance.

In practice, during each training epoch, the CNN is trained on a dataset generated by the data augmenter that has the same length as the original dataset. Since these

augmented images are not stored in memory, this means that the model does not see the same version of the data point x_i more than once during the learning process. At the end of training, the model would be trained with $P \times Q$ images, where P is the number of images in the dataset and Q is the number of training epochs. And when comparing two CNN models, it is important that both models are trained using the same hand-designed dataset augmentation schemes.

IV.7 Performance Results & Discussion

As explained earlier, the search for the architecture of CT -Casting-Net involved a long series of trial-error attempts: After determining the number of layers and the number of filters in the convolutional layers, we began to change the other hyperparameters interchangeably until it achieved high accuracy on the CT casting datasets. Finally, to evaluate the learning process, we compared the performance with state-of-the-art models trained from ground up. During each training epoch, the training loss, which is the output of the binary cross-entropy loss function, and the training accuracy, which evaluates each model on examples it was constructed on, were plotted. Similarly, at the end of each training epoch, the model was validated on the 20% of the data and the validation loss and accuracy were plotted as shown in Figure IV.8. The resulting curves help to understand the learning process and spot any sign of overfitting. Ideally, the loss and accuracy should be closer to zero and one respectively.

The training was started with a learning rate of 0.01 and the weights of the networks were updated using the SGD optimizer. In the first trials, training and validation losses diverged after a few epochs, indicating overfitting as explained in section I.8. This was attributed to the small size of the dataset. Subsequently, this divergence was systematically reduced by increasing and diversifying the dataset, adding dropout regularization, and applying data augmentation techniques. For each model, the training process is stopped when the validation accuracy no longer improves over five consecutive epochs.

Table IV.2: Comparison of the performance of CT -Casting-Net and five state-of-the-art models based on training and validation results. CT -Casting-Net (10 layers) achieved the best performance in terms of loss and accuracy.

Model	Depth	Training Loss	Training Accuracy	Validation Loss	Validation accuracy
CT-Casting-Net	**10**	**0.0364**	**98.80%**	**0.0488**	**98.30%**
VGG19	22	0.4764	77.74%	0.4775	78.44%
ResNet50	50	0.1481	94.58%	0.1278	95.47%
ResNet101	101	0.2367	91.68%	0.2620	91.55%
Xception	126	0.3596	88.46%	0.3025	90.01%
InceptionV3	159	0.3588	81.67%	0.3694	82.88%

The first step of the comparison study is to eliminate the state-of-the-art models that do not achieve high accuracy on the dataset. For this study, 5 state-of-the-art models were selected based on their depth: VGG19 (22 layers), ResNet50 (50), ResNet101 (101), Xception (126) and InceptionV3 (159). Table IV.2 shows the performance of CT-Casting-Net trained from the ground up and the initialized state-of-the-art models as explained in IV.5. The performance comparison is based on losses and accuracies of training and validation.

Of all the state-of-the-art models, ResNet50 performed best with a training accuracy of 94.58% and a validation accuracy of 95.47%. CT -Casting-Net, which consists of only ten layers, outperformed the state-of-the-art models, achieving a training accuracy of 98.80% and a validation accuracy of 98.30% with less training time. This shows that increasing the depth does not necessarily increase the classification accuracy and does not guarantee better performance. The final training curves of CT-Casting-Net are shown in Figure IV.8.

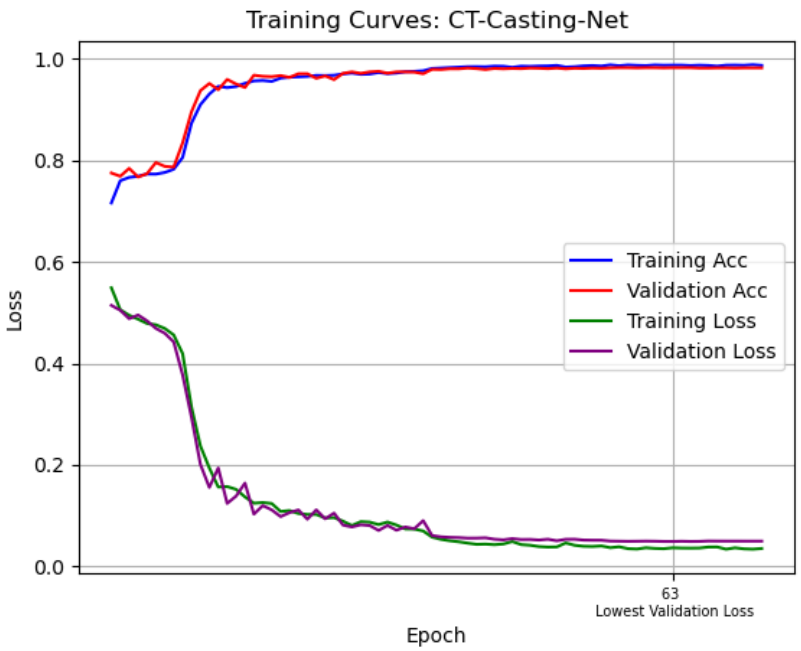


Figure IV.8: Casting Dataset: CT-Casting-Net training curves with respect to training epochs. The model achieved training and validation accuracies close to 1, and training and validation losses close to 0. The best performance was obtained after 63 epochs of training.

IV.7.1 Inference Results

After training is done, the best performing model CT-Casting-Net was tested on a set of 700 new images cropped from a new volume, in order to check its performance. The output of this test is a confusion matrix that summarises the classification results by checking how often the predicted labels are correct compared to the actual labels. As stated in subsection IV.3.2, for each inference image, the

Table IV.3: Inference results of CT-Casting-Net on a balanced vector of 350 defects and 350 false alarms. The model achieved high F1-score of 96.85%.

	CT-Casting-Net		Metrics
False Alarm	TN= 340	FP= 10	Recall= 96.57%
True Alarm	FN= 12	TP= 338	Precision= 97.12%
	False Alarm	True Alarm	

model outputs a vector $p_i = [p_{i1}, p_{i2}]$, where p_{i1} is the probability of membership to the false alarm class and p_{i2} to the true alarm class. The highest probability defines the class predicted by the model for each image.

According to the confusion matrix of CT-Casting-Net shown in Table IV.3, CT-Casting-Net achieved high recall of 96.57% and precision of 97.12%, indicating an F1-score of 96.85% (Equation IV.5) even though the test images come from a new volume, with different contrast and spatial resolution compared to the training dataset. In fact, in aerospace and automotive industries it is very important to detect all defects in order to evaluate their consequences on the lifetime of the casting specimen. On other hand, misjudging normal zones or artifacts as defects could lead the high rejection rate and economic problems. For that reason, having high F1-scores ensures that the models are able to classify both defects and normal zones correctly.

IV.7.2 Visual Test

To understand how CNNs make their classification decisions, the gradients map of the final convolutional layer of a model can be calculated for a specific image in order to highlight the zone that triggered the class decision. This gradient map is calculated with the help of Gradient-weighted Class Activation Mapping, or more simply, Grad-CAM. As introduced by [118]: "Grad-CAM uses the gradients of any target concept, flowing into the final convolutional layer to produce a coarse localization map highlighting the important regions in the image for predicting the concept".

In Figure IV.9, two Grad-CAM of the best performing model are shown after sending test images of shrinkage cavities. The model labelled both images as true alarm (defect). Looking at the grad-cams, it is clear that the filters of the last convolutional layer actually detect the correct patterns in the image and are activated around these patterns. These gradient maps ensure that the high accuracies achieved did not occur by accident or happenstance and that the model was not activated on irrelevant patterns.

Figure IV.9: Grad-cam output of the final convolutional layer showing the class trigger: (a-c) 64×64 zones with shrinkage cavities; (b-d) Grad-CAM by CT-Casting-Net.

Conclusion

Classical defect detection algorithms cannot perfectly distinguish between artifacts and defects, as both indications represent a zone of high contrast compared to the surrounding backgrounds. This ambiguity is also reported by operators when using automatic image recognition algorithms as an interpretation aid. The proposed CNN classifier CT-Casting-Net, which directly processes greyscale images, proved to be very efficient in correctly classifying casting defects as true alarms and artifacts or noise as false alarms. In addition, CT-Casting-Net achieved near-perfect performance compared to widely recognised deep CNN models, despite having only 10 layers, in terms of training and inference metrics.

Since the segmentation model U-Net from the last chapter has been trained to oversegment CT images, CT-Casting-Net is coupled with U-Net in the next chapter to recognize real defects in 3D volumes and eliminate false alarms. This represents a new approach for automatic defect detection in industrial CT volumes of castings.

Chapter V

Automatic Casting Defect Detection: Approach Validation

The ultimate goal of this chapter is to design a generic approach to automatically detect defects inside CT volumes of aluminium alloy castings without user intervention. In chapter III we trained U-Net to over-segment CT slices in order to detect all suspicious discontinuities. And in chapter IV we trained a CNN model, CT-Casting-Net, to classify discontinuities into true alarms (defects) or false alarms. By coupling U-Net and CT-Casting-Net, we can detect and validate true defects inside CT volumes and eliminate artifacts and noise.

As shown in Figure II.1, CT volumes are mostly filled with empty void outside the specimen and only 20% of the data show the actual aluminium casting. Furthermore, this 20% is mostly filled with material and defects make up only a very small proportion. Therefore, in order to validate the deep learning approach, we need validation metrics that primarily consider the voxels with defects and indicate whether they were detected or not. This chapter presents the details of this coupling approach and its validation on 6 CT volumes at the object level using the probability of detection (POD) and at the voxel level using Intersection-over-Union (IoU).

V.1 Automatic Defect Detection Approach

As illustrated in Figure I.11, the proposed approach is intended to be used as a post-processing algorithm after reconstructing the CT volume of an aluminium

specimen from X-ray projections. This approach consists of four steps as illustrated in Figure V.1:

a) The volume is sliced along the longest dimension of the specimen, and the 2D greyscale slices are individually over-segmented by U-Net. The binary slices are then stacked together to make up a binary volume. The latter is labelled by assigning a unique integer to each object. This can be a defect, noise, artifact or even borders segmented by U-Net.
b) The coordinates of the centre-of-mass (COM) of each object are retrieved and located in the original greyscale volume.
c) By locating the COMs of each object, three greyscale zones of 64×64 are cropped around each COM along the three planes XY, YZ and XZ.
d) The three 64×64 crops are sent to the CT-Casting-Net to validate the type of each image, defective zone or false alarm, and consequently the object from which they were cropped. If the three crops are classified as defects, the object is retained, otherwise it is removed from the binary volume of step a).

V.1.1 Slice-by-Slice Segmentation with U-Net

After slicing the volume along the longest dimension of the specimen, the user chooses a random ROI of $512 \times 512 \times (Z)$ to send to U-Net model. Unlike the 3D algorithm in section II.4, the input is not limited along the Z direction, but along the X and Y axes due to the size of the images on which U-Net was trained. If an inference slice is smaller than 512×512, the edges of the slices along the smaller dimension are padded with zeros. And if the slice is larger, an automatic crop is applied along the larger dimension. In the last case, parallel segmentation can be performed on different crops of the image, and the final results are concatenated to obtain a segmented slice with the same size as the original.

As explained in chapter III, the output of U-Net is a greyscale image in which each voxel has a value between 0 and 1. Each represents the confidence with which U-Net decides the type of voxel; e.g. voxels of suspicious discontinuities are set as close to 1 as possible. When U-Net is put into deployment after training, an additional threshold must be applied to its greyscale output in order to convert it to a binary mask. For a greyscale mask p_i estimated by U-Net, the binary slice is obtained by setting the voxels to 0 or 1 depending on how far they are from the threshold τ. The thresholding function is depicted as follows:

$$f(p_i) = \begin{cases} 0\,(Black) & if\ p_{ij} < \tau \\ 1\,(White) & if\ p_{ij} \geq \tau \end{cases} \tag{V.1}$$

Where p_i is the output of U-Net, and p_{ij} are the voxels of this output.

Stacking the binary slices together after thresholding results in a binary volume in which defects, artifacts, noise and some borders are highlighted. In the Results

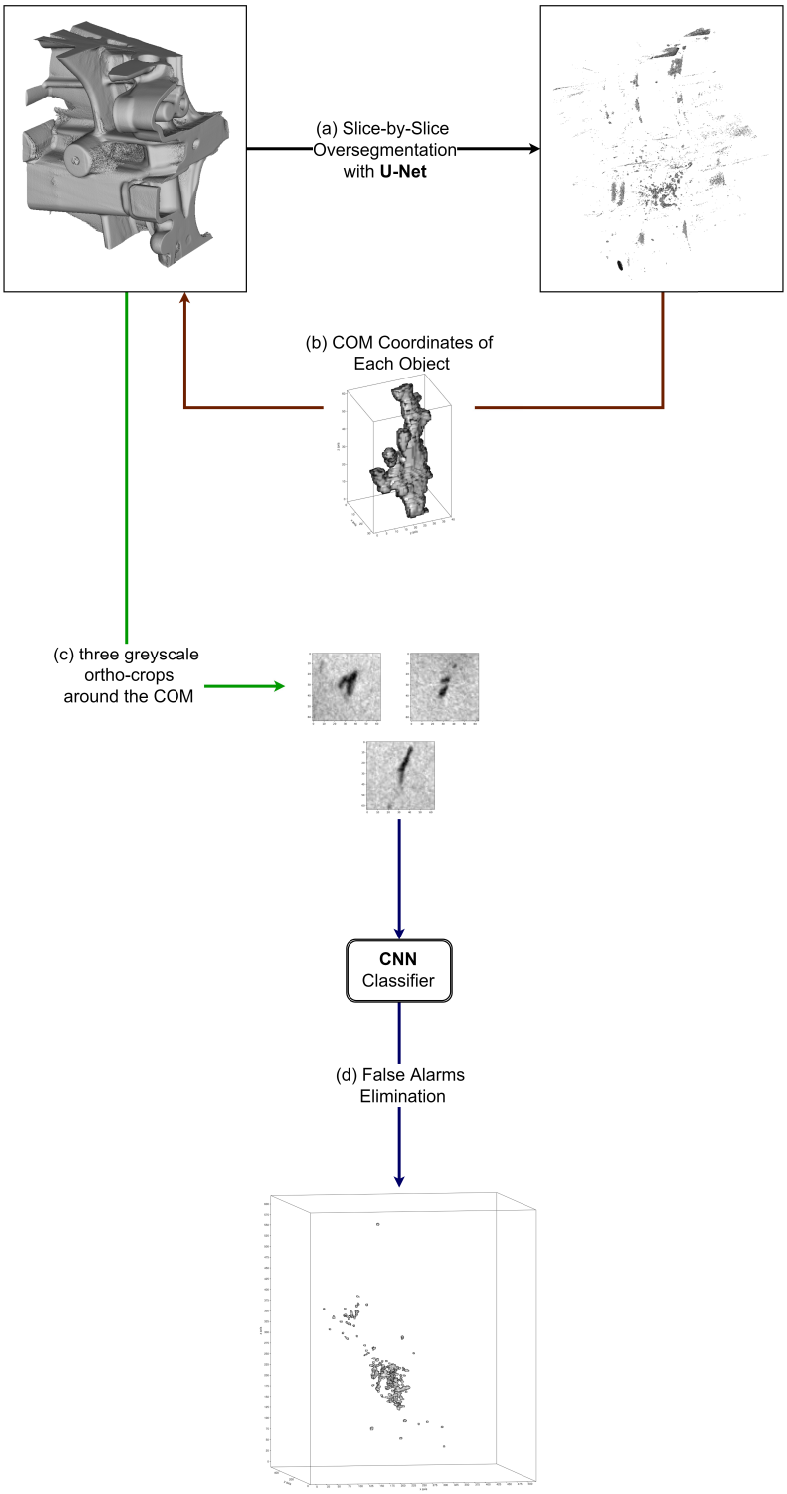


Figure V.1: Automatic defect detection approach that takes a $512 \times 512 \times (Z)$ ROI as input and outputs a volume containing only the defects. The U-Net model is used to isolate discontinuities, followed by CNN classifiers to classify these discontinuities as defects or false alarms. In this example, only 153/550 indications were validated as defects.

section, we will explore the impact of the threshold τ on the final output of the approach, in terms of true positives (detected defects), false negatives (missed defects) and false positives (misclassified false alarms).

V.1.2 3D Labelling and CNN Classification

Since U-Net was trained on over-segmented dataset, the binary volume will contain many irrelevant discontinuities that belong to false alarms, regardless of the threshold. For this reason, we need to call each object individually and validate its type using the CNN classifier. To handle each object individually, we first need to label the binary volume by assigning a unique integer to all voxels belonging to the same object. Voxels labelled 1 form object number one, voxels labelled 2 form object number two and so on. The labelling process is controlled by a parameter called "voxel connectivity", as shown in Figure V.2.

To avoid including the borders of the specimen in the case of a defect near the surface, and to process each defect individually in the case of a region with high density of porosities, we chose a neighbourhood of 6 as the optimal connectivity. This means that if a voxel has been segmented as white by the U-Net model, only 6 voxels in the neighbourhood that share a common surface with it will be tested whether they have also been segmented as white or not. In this way, all white voxels in the 6-neighbourhood are assigned the same integer as the voxel in the centre, so that together they form the same object. The process is repeated systematically with each white voxel of this object, checking its own neighbourhood to determine the final shape.

It should be noted that any object with a size of less than 10 voxels was removed

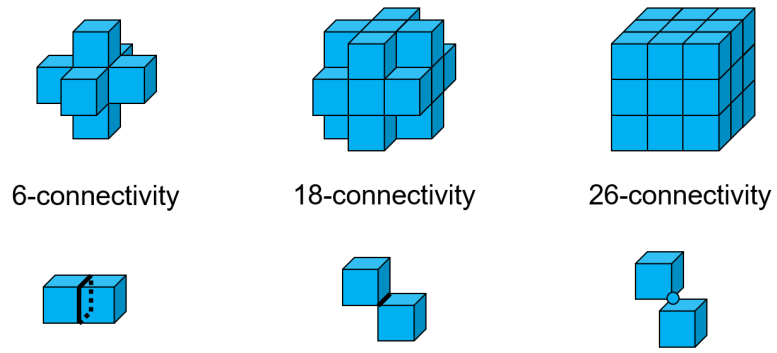


Figure V.2: Voxels considered as neighbours of a given central voxel under different connectivities: In the case of 6-connectivity, two voxels are considered neighbours and belong to the same object only if they share a common surface. In the case of 18-connectivity, two voxels are considered neighbours if they share at least one edge. And in the case of 26-connectivity, it is sufficient if they share a vertex point to be considered neighbours.

during processing. This is due to the fact that the largest spatial resolution available in our database is $450\mu m$, which means that a defect of 10 voxels corresponding to $0.911mm^3$ can be safely neglected. By iterating the list of labels after removing small objects, the coordinates of each object COM are rounded and located inside the original greyscale volume. Within this volume, three 64×64 images are cropped around each COM along the three ortho-planes and then sent to the trained CNN model *CT-Casting-Net* of chapter IV to predict whether all the crops represent a real defect. If, on the other hand, at least one of the crops is classified as a false alarm, the corresponding object is removed from the binary volume.

V.1.2.1 Why 3 Ortho Crops?

In the early phase of validation, the number of crops around the COMs was set to 1: a single crop along the thickness of the specimen is sent to CT -Casting-Net to validate the nature of the object. However, there were some cases of ambiguity where false alarms had been incorrectly validated as defects. Figure V.3 shows a ROI inside the output of the approach under this condition, where coaxial ring artifacts had been validated as true alarms. In fact, these artifacts are close to real defects contained in the 64×64 crop around the COM of each artifact, which in turn created the ambiguity. Processing 3 crops along the XZ, XY and YZ planes reduced this ambiguity by examining each object from 3 angles before determining its final class.

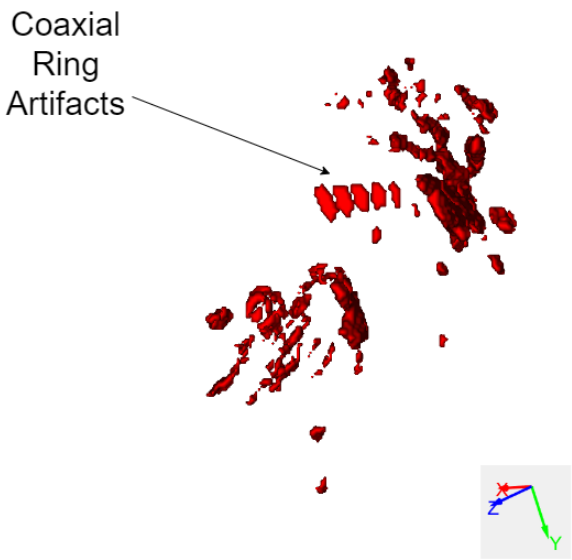


Figure V.3: ROI inside a binary volume at the output of the approach that processes only one crop around each object COM. Coaxial ring artifacts were wrongly classified as true defects because the classification model rather considered the near defects that exist at a distance of less than 64 voxels.

V.2 Validation Metrics

The output of the detection algorithm is a binary volume in which voxels of validated defects are set to 1 and all others to 0. To validate the approach, we performed a comparison between this output and manually-segmented ground-truth volumes according to the following classes of voxels:

TP = True Positives, correctly classified voxels of defects,
TN = True Negatives, correctly classified normal voxels,
FP = False Positives, normal voxels misclassified as defects,
FN = False Negatives, voxels of defects misclassified as normal.

The validation metrics were carefully chosen to ignore TN and focus mainly on voxels of defects that were either correctly detected (TP) or missed (FN). While varying the threshold τ that controls the output of U-Net (Equation V.1) and consequently the output of the whole approach, the detection performance was evaluated using the following metrics that ignore normal or background voxels.

V.2.1 Precision Recall Curve

A metric very commonly used in NDT is the Receiver Operating Characteristic (ROC) curve, which plots the rate of true positives (TPR) against the rate of false positives (FPR) for each system threshold (Equation V.2). However, the FPR takes into account the TN voxels (correctly classified background voxels), which leads to a bias towards the voxels that are largely represented in the images, i.e. the background class. This means that the ROC curve is not suitable for data with imbalanced classes such as CT images [119, 120].

$$\begin{aligned} TPR &= \frac{TP}{TP+FN} \\ FPR &= \frac{FP}{FP+TN} \end{aligned} \tag{V.2}$$

The equivalent of the ROC curve for imbalanced data is the Precision Recall (PR) curve, which answers two important questions while varying the binarization threshold [121, 122]:

- How many voxels in the output volume actually belong to real defects?, i.e. the precision:
$$Precision = \frac{TP}{TP+FP} \tag{V.3}$$
- And how many voxels of defects in ground truth can actually be found in the output?, i.e. the recall:
$$Recall = TPR = \frac{TP}{TP+FN} \tag{V.4}$$

In other words, the PR curve represents the percentage of detection, not in terms of the number of defects, but the number of voxels belonging to any defect. By plotting the PR curve for each threshold, we can evaluate the performance of the model at the voxel level, taking FP into account.

V.2.2 Intersection-over-Union

The Jaccard index or Intersection-over-Union (IoU) [123, 124] is a standard metric for evaluating object detection algorithms for imbalanced data. For each threshold, this parameter represents the overlap between the predicted binary volume p_i and the ground truth binary volume y_i divided by their union, as shown in Equation V.5, without considering TN. IoU ranges from 0 to 1, where 0 represents no overlap and 1 represents perfectly overlapping masks. In machine learning, an IoU value greater than 0.5 is usually a sign of "good" performance [125].

$$IoU(y_i, p_i) = \frac{Area\ of\ Overlap}{Area\ of\ Intersection} = \frac{|y_i \cap p_i|}{|y_i \cup p_i|} = \frac{TP}{TP + FP + FN} \tag{V.5}$$

V.2.3 Probability of Detection

A metric commonly used in non-destructive testing is the probability of detection (POD), which measures the performance of a segmentation system as a function of the number of defects detected (hits) or not segmented (misses) [126, 127], which has the same meaning as recall. Unlike the previous metrics, hit-or-miss POD compares the ground truth volume to the predicted volume at the object level rather than at the voxel level, which is more reliable for digital images with different spatial resolutions. In other words, POD processes each object (or eventually a defect) as a whole, regardless of its size.

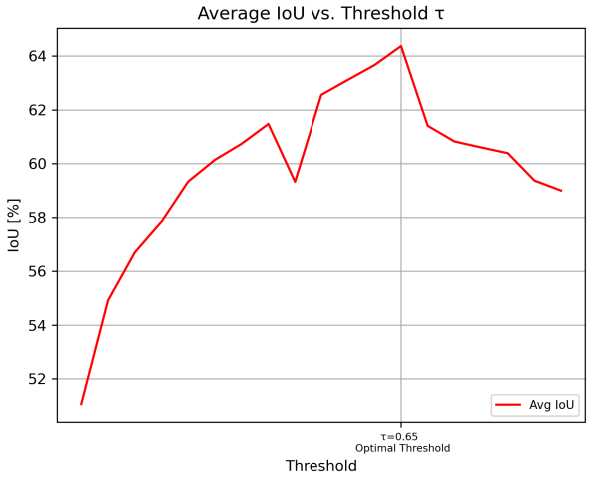
Plotting POD versus the binarization threshold gives a better insight into the reliability of the segmentation approach. For each threshold τ, we crafted this metric to consider each ground truth defect as a single entity and check whether it was detected by the approach or not according to the following equation:

$$POD(\tau) = Recall = \frac{n_{hit}}{n_{hit} + n_{miss}} \tag{V.6}$$

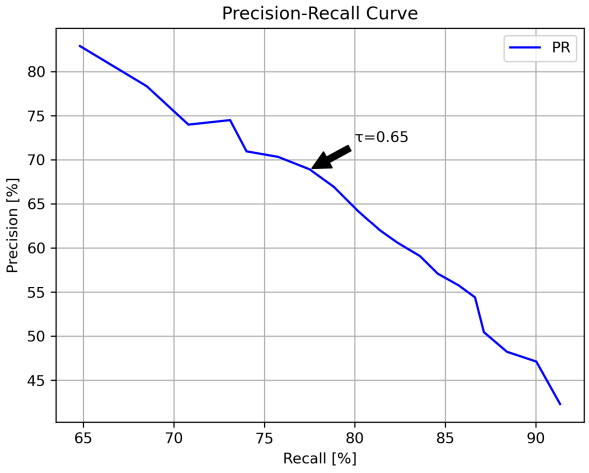
V.3 Validation Results & Discussion

The approach was evaluated on 6 CT volumes of high-pressure and gravity cooled specimens representing automotive and aircraft components. Based on the above metrics, the approach can be evaluated at the voxel level with PR-curve and IoU, and at the object level with POD. **It should be noted that the threshold is applied to U-Net output, but the performance is measured after eliminating false alarms with CT-Casting-Net.**

By calculating the IoU for each volume against the binarization threshold with a step size of 0.05, the average IoU can be plotted, giving insight into the similarity between the ground-truth binary volumes and the results of the approach. As can be seen in Figure V.4a, the IoU increases with the increase of the binarization threshold. In fact, an increase in τ means that the thresholds of U-Net label maps are stricter, resulting in fewer irrelevant voxels being sent to CT-Casting-Net. In addition, the objects that are validated as defects by CT-Casting-Net have smaller shapes and are less over-segmented, which increases the overlap between the ground-truth and the final output (IoU). On the other hand, the side effect of the strict thresholding is that small defects are usually suppressed, which explains the low IoU at small values of τ.



(a)



(b)

Figure V.4: Voxel level metrics to evaluate the performance of the approach: (a) Average IoU and (b) Precision-recall curve, against the binarization threshold.

The average IoU plot not only shows how accurate and how sharp the label map is

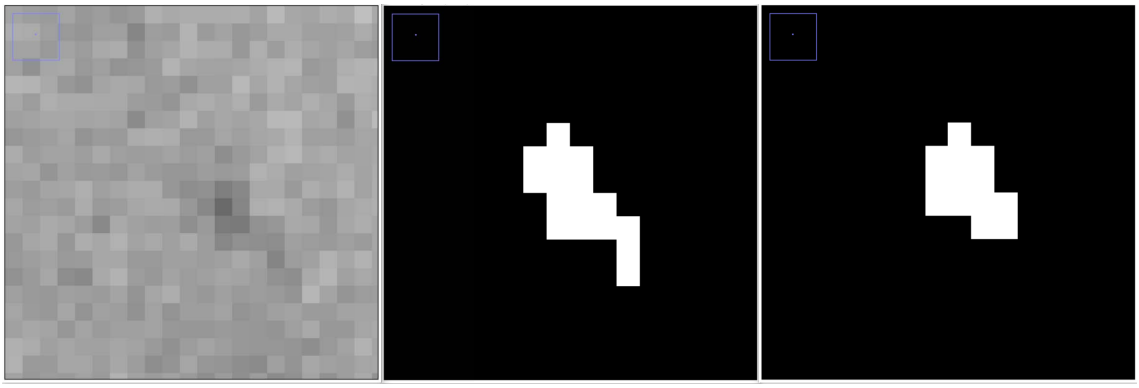


Figure V.5: (left) A crop along Z-axis of a casting porosity in a CT volume with low spatial resolution; (middle) the contour of this defect segmented and validated by the approach; (right) manually segmented contour of the same defect.

after applying a threshold and eliminating false alarms with CT-Casting-Net, but also provides the best threshold for binarization in the case of good data quality. The average IoU between the results of the approach and the ground-truth binary volumes is higher than 51% depending on the threshold, and the highest average value of 64.371% was obtained with a threshold $\tau = 0.65$.

Always at the voxel level, the PR-curve in Figure V.4b shows the impact of the threshold on the number of FP and FN voxels. We see that the average precision and the average recall of the results on 6 volumes are inversely proportional to each other: increasing recall (finding more voxels of defects) leads to decreasing precision (producing more voxels of false alarms). An ideal PR-curve has two perpendicular segments with a corner representing high average precision and high average recall at the same time.

As shown in Figure II.6 and Figure V.5, digital images in general and CT images in particular suffer from various sources of noise that make the contours of defects noisy and thus difficult to define, even when the slices are manually segmented to find the ground-truth. This makes the IoU and PR values relatively low and insufficient to determine the optimal value of the system threshold. These metrics require high spatial resolution, which is not the case in industry where inspection time is reduced at the expense of quality, as explained in Figure I.10. For this reason, it is common in non-destructive testing to evaluate the performance of a system based on the number of defects detected or missed, rather than examining each voxel individually.

A commonly used metric POD can be crafted to measure the performance of the approach at the object level. Figure V.6a shows the average of POD (or recall) on 6 CT volumes with respect to the binarization threshold τ, which is always higher than 93%. At higher τ, POD begins to decline as the strictness of the thresholding leads to under-segmentation and missing defects in the binary output.

Conversely, a low threshold leads to a high POD, as over-segmentation ensures that more greyscale discontinuities are detected, while FP increases the evidence of additional false alarms.

Since POD only considers hit defects (TP) and missed defects (FN), the amount of false alarms (FP) was plotted separately against τ as shown in Figure V.6b. At higher thresholds, the amount of FA decreases due to under-segmentation, but at the expense of POD. On the other hand, low thresholds lead to over-segmentation and increase FA. This is *possibly* due to false alarms generated near true defects that cannot be eliminated by CT-Casting-Net.

An optimal threshold corresponds to the highest IoU and POD and the lowest FA as shown in Figure V.6c. For example, a $\tau = 0.6$ gives an average POD of 99.03%, an average FA of 4.33, an average IoU of 63.66% and an average recall and precision of 78.84% and 66.94% respectively, as shown in Table V.1. It can be noticed that the number of objects or indications before classification with CT-Casting-Net is very high, as U-Net has been trained to over-segment CT images. However, the relatively high number of FA in V1 and V3 is not due to the over-segmentation, but to the poorly contrasted small defects that we missed when manually creating the binary ground-truth volumes, but which were nevertheless detected by the approach, contributing to its reliability.

On the other hand, processing a ROI volume of $512 \times 512 \times 100$ takes 15-18 seconds, which is relatively fast, although this depends on the number of over-segmented objects in the volume that need to be classified. In order to find the optimal range of the threshold, this study needs to be performed in the future with a larger amount of test data with ground-truth in order to increase the confidence rate.

Table V.1: Validation results with a threshold $\tau = 0.6$. The number of detected objects is reduced after each object is classified as a true or false alarm. The high average probability of defect detection POD and the low amount of FA show the efficiency of the approach in interpreting CT data of castings.

Test Volume	Size	Nb Obj Before Classification	Nb Obj After Classification	FA	POD %	IoU %	Processing Time (s)
V1	$512 \times 512 \times 100$	504	24	12	100	57.25	17.9
V2	$512 \times 512 \times 100$	676	30	0	100	72.18	18.3
V3	$512 \times 512 \times 96$	587	45	14	100	59.68	16.0
V4	$512 \times 512 \times 100$	220	19	0	100	68.79	12.2
V5	$512 \times 512 \times 86$	167	40	0	100	68.77	11.8
V6	$512 \times 512 \times 100$	354	24	0	94.23	55.28	13.8
			Average:	**4.33**	**99.03**	**63.66**	**15**

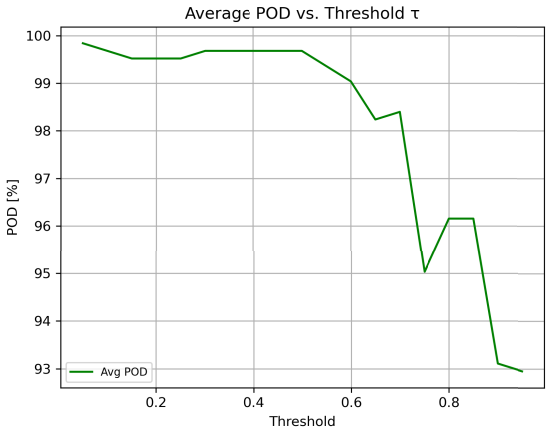
The great advantage of this approach is the low rate of FN (missed defects), i.e. high POD. Defective zones are very common in castings, especially those made by injecting molten metal into a mould under high pressure. Overlooking these defects

can lead to quality drawbacks at the level of the specimen itself and economic damage at the level of the production line. Such problems can only be avoided if quality control is carried out during the design stage of production using the right inspection tools. Our methodology ensures this capability through two important steps:

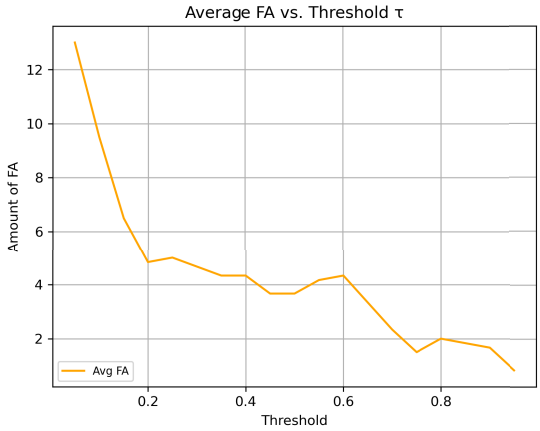
- Increase false and true positives by over-segmenting the CT slices with U-Net model to capture all potential defects.
- Reduce false positives by eliminating false alarms with the CNN classification model, resulting in a correct true positive rate in the final output.

Conclusion

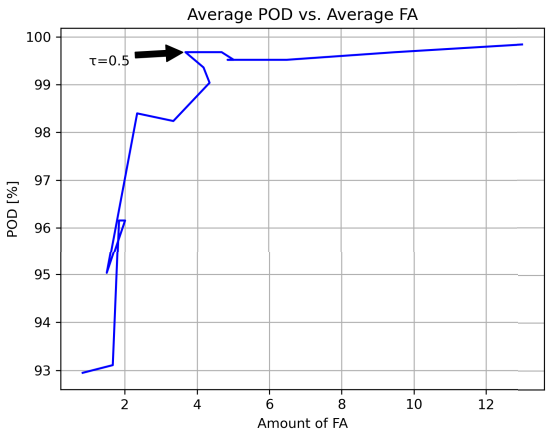
In this chapter we have interpreted the reliability of a defect detection approach in the processing of CT data of castings. This approach uses a deep segmentation model to over-segment CT slices, followed by a CNN classifier to decide whether the segmented objects are true defects that should be retained or false alarms that need to be eliminated. To go beyond deep-learning metrics, performance was measured using voxel-level and object-level metrics related to the non-destructive testing literature, which, in addition to the fast processing time, ensured the reliability of the approach in accomplishing the required task.



(a)



(b)



(c)

Figure V.6: Object level metrics to evaluate the performance of the approach: (a) Average POD and (b) average FA, against the binarization threshold. The optimal threshold at the object level (τ=0.5, average POD = 99.68%, average number of FA = 3.6) can be found by plotting (c) average POD against average FA.

Chapter VI

Few-Shot Learning for Casting Defects Categorization

Deciding whether or not a specimen is crucially defective in the early stages of production can bring economic benefits. This decision depends on the regions where defects occur during the cooling of the metal and the nature of the embedded defects. In this chapter, we explore the 3D geometrical properties of casting defects and the potential of traditional machine learning and deep learning algorithms in classifying these defects into porosities and shrinkage cavities.

VI.1 Casting Processes

Before dividing the defects into porosities or shrinkage cavities, we must explain their origin. As explained in subsection I.5.1, the cooling of a specimen into a mould (or casting die) can be done either under pressure (high-pressure die casting) or by the force of gravity (gravity die casting) [128].

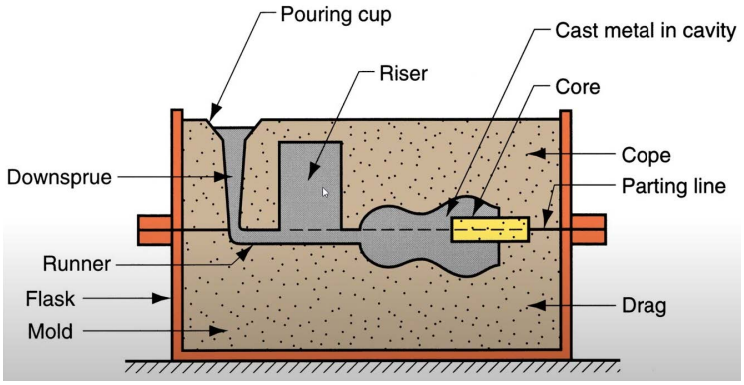


Figure VI.1: Diagram of gravity die casting in a sand mould. Source: learnengineering.net

In gravity die casting, the molten metal is poured into the mould slowly and very gradually to avoid any stirring and turbulence of the metal on the surface, as shown in Figure VI.1. The mould may be made of sand (sand casting) or steel (shell casting) and has a riser which acts as a reservoir to compensate for the cumulative shrinkage of the metal during the solidification process, and sometimes a core is introduced into this mould to create hollow structures in the casting. In this type of process, the casting may have a variety of embedded defects that emerge during solidification and can be divided into two categories: (1) gaseous porosities, either due to reaction between the mould and the metal, or to the gassing of the metal; (2) shrinkage cavities due to the lack of sufficient molten metal when the casting cools.

In high-pressure die casting, the molten alloy is injected at high velocity ($40\ to\ 60 m/s$) into a steel mould of type X38 CrMoV5 or equivalent under a very high pressure ($80\ to\ 100\ MPa$) during solidification, as shown in Figure VI.2. The possible defects after solidification can be divided into three categories: (1) gaseous porosities, which are due to the incorporation of air into the injected metal. These porosit-

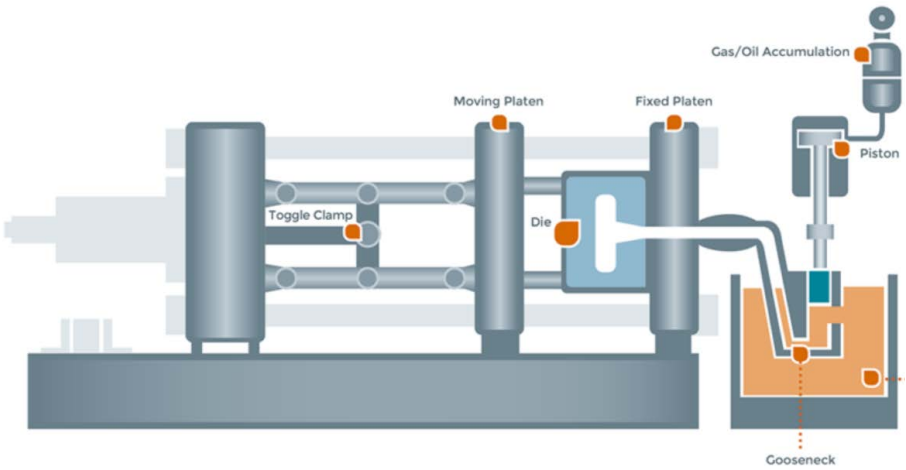


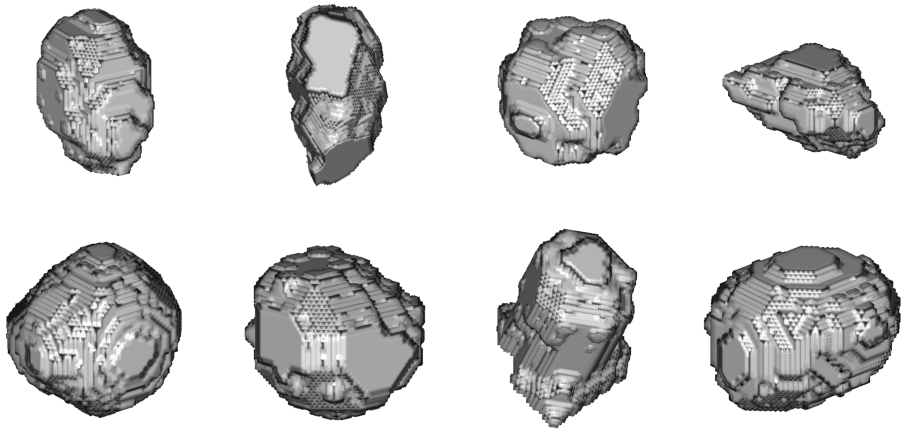
Figure VI.2: Hot chamber die casting machine. Source: bscdiecasting.co.uk

ies can be very dense in thick regions of the specimen, and are characterised by a spherical shape and smooth sides; (2) shrinkage cavities, which correspond to an internal contraction in the metal and usually have an elongated shape; and (3) cold fills, which are due to the turbulent injection associated with this process.

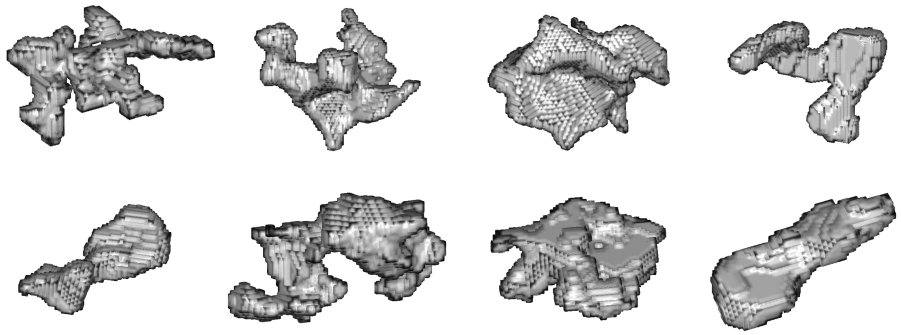
VI.2 Casting Defects Database

To investigate the geometrical properties of casting defects belonging to each category, the industry partners provided a set of isolated 3D greyscale defects retrieved from **CT volumes** of high-pressure and gravity-cooled specimens. Using the 3D segmentation algorithm in Figure II.8, these 3D volumes were segmented as shown in Figure VI.3 to investigate the geometrical properties of each embedded discontinuity. This dataset consists of 462 defects:

- High-pressure casting: 263 shrinkage cavities and 101 porosities.
- Gravity casting: 57 shrinkage cavities, 41 porosities.



(a) Porosities



(b) Shrinkage Cavities

Figure VI.3: 3D casting defects inspected by CT, after binarization: (a) porosities, (b) shrinkage cavities. (No scaling applied)

VI.2.1 Geometrical Properties

The aim of this chapter is to examine the validated defects inside the output of the approach presented in chapter V. As the output is a binary volume, the 3D defects in the dataset were binarized for consistency. Consequently, texture features (such as greyscale features) were avoided. Only the following geometrical features were calculated for each defect, as presented in Figure VI.4 that shows these properties for two typical defects:

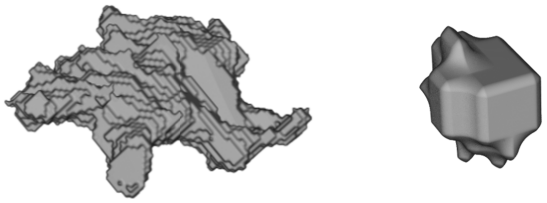
- Volume V: the number of 'on' voxels in the region within the volumetric binary image.
- Surface area S_A: by iterating across the volume after blurring its image, machine cubes algorithm extracts a 2D surface mesh from the volumetric binary image and measures its surface [129, 130].
- Sphericity Ψ: it is a measure of how spherical an object is. Proposed by [131], the sphericity of a particle is defined as the ratio of the surface area of an equal-volume sphere to the actual surface area of the particle:

$$\Psi = \frac{\pi^{1/3}(6V)^{2/3}}{S_A} \tag{VI.1}$$

- Elongation: it is a measure of how elongated an object is. It is the ratio between the lengths of the major and minor axes of the ellipse that has the same normalized second central moments as the object [132]:

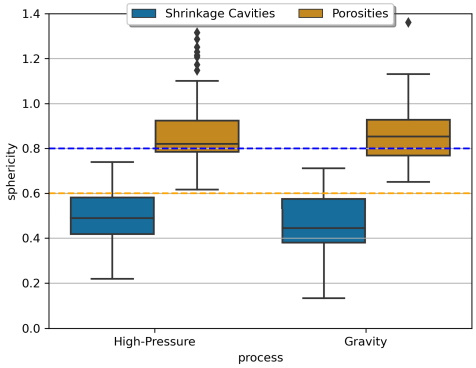
$$Elongation \ = \frac{major\ axis\ length}{minor\ axis\ length} \tag{VI.2}$$

- Compactness: Ratio of voxels of the object to voxels in the total bounding box, the smallest box that can contain the object.

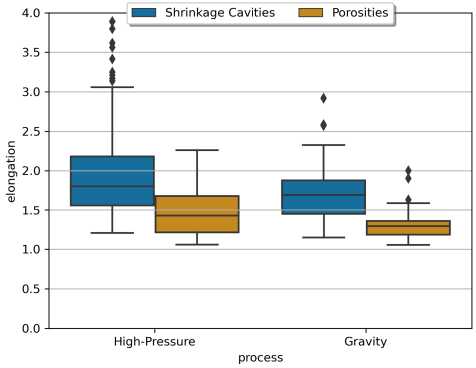


Process	High-Pressure	Gravity
Defect	Shrinkage Cavity	Porosity
Volume (voxel)	15763	61
Surface Area (voxel)	7664.75	89.0
Sphericity	0.3966	0.8419
Elongation	1.635	1.149
Compactness	0.1387	0.5083

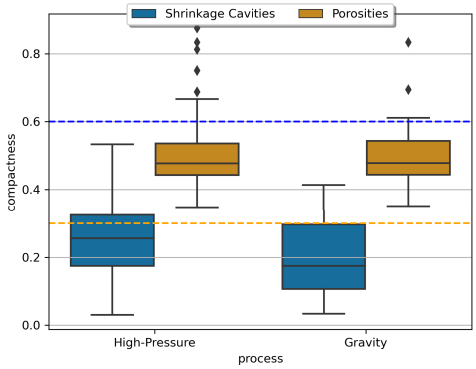
Figure VI.4: 3D geometrical properties of binary CT ROIs containing typical 3D casting defects. The shrinkage cavity is less spherical and compact than a porosity, but more elongated.



(a)



(b)



(c)

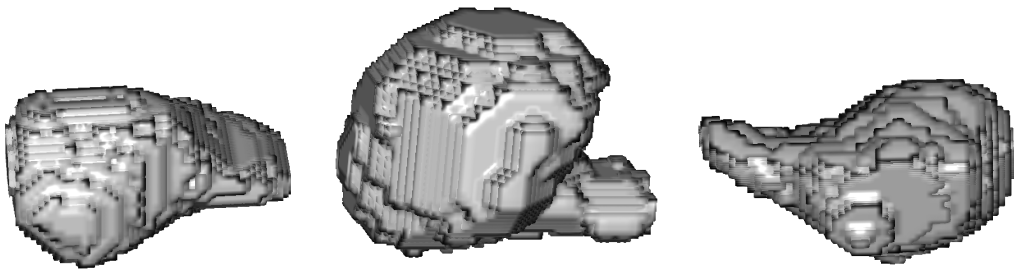
Figure VI.5: Geometrical properties distribution with respect to the casting process, and the category of the defect: (a) sphericity distribution; (b) elongation distribution; (c) compactness distribution. The interquartile range in the box plot contains 50% of the values, while the whiskers and their ends represent the other 50%. The dashed lines in plots (a-c) represent the range in which each category of defects exists with respect to the geometrical property.

VI.2.2 Properties Analysis

Figure VI.5 shows the distribution of the above properties with respect to the defect types and the casting process. Shrinkage cavities in high-pressure and gravity casting have median sphericity of 0.49 and 0.45, respectively, as shown in Figure VI.5a. And porosities have a median value of 0.82 for high-pressure casting, and 0.85 for gravity casting. The dashed lines represent the range in which each category of defects exists with respect to sphericity: porosities always have a sphericity greater than 0.6 and shrinkage cavities always have a sphericity less than 0.8. It should be noted that some porosities can have a sphericity value of more than 1, as can be seen in the boxplot. Such defects usually have a cylindrical shape with convex fillet joints.

Figure VI.5b shows that shrinkage cavities have an median elongation of 1.8 and 1.7 for high-pressure and gravity casting, respectively, and 1.4 and 1.3 for porosity. On the other hand, as shown in Figure VI.5c, shrinkage cavities tend to have a compactness of less than 0.6 and greater than 0.3 for porosities.

Regardless of the casting process, porosities have a higher sphericity and compactness because they are in fact gaseous bubbles. Conversely, shrinkage cavities have a lower sphericity and a higher elongation, as they represent a lack of material due metal contraction that can have any shape. In the intermediate range, i.e. with sphericity between 0.6 and 0.8 and compactness between 0.3 and 0.6, the defect can have a shape that is initially spherical and then becomes elongated, as shown in Figure VI.6. To determine the type of defect in this intermediate range, a trained user or a classification algorithm is required.



(a) **(b)** **(c)**

Figure VI.6: Defects with a sphericity value between 0.6 and 0.8: (a) shrinkage cavity. (b) porosity attached to a shrinkage cavity. (c) shrinkage cavity.

VI.3 SVM for Defects Categorization

Deciding the type of each defect inside the binary volume at the output by the approach of the last chapter requires a trained classifier that can identify the defect category by assigning one of two labels: 0= shrinkage cavity or 1= porosity. The above features or properties can be used to train a traditional machine learn-

ing algorithm such as the Support Vector Machine (SVM). The latter search for an optimal hyperplane that can separate each category in the training data using support vectors, where the margin between two different defects is maximal [133]. This concept can be seen in Figure VI.7, where the dashed lines represent the support vectors and the solid line the optimal hyperplane.

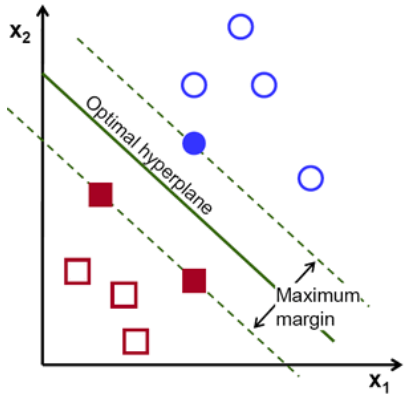


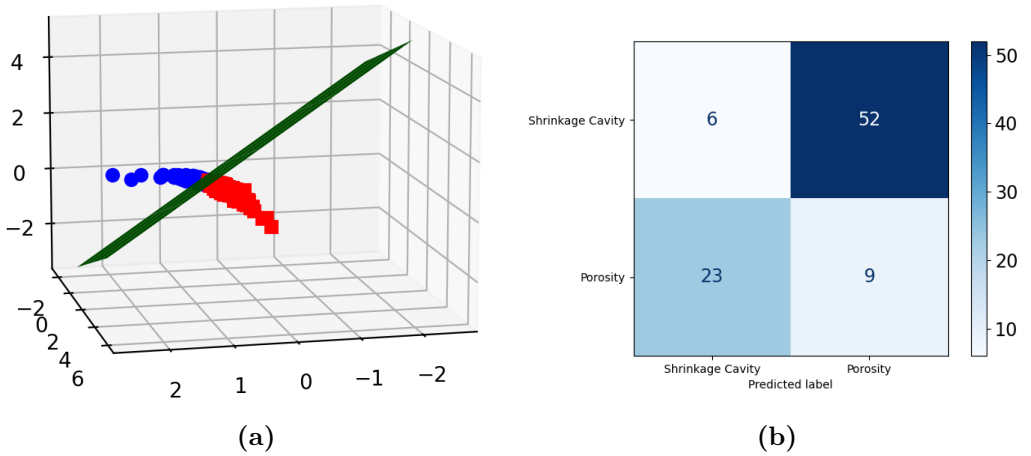
Figure VI.7: 2D hyperplane that separates 2D data points into two categories. Source: medium.com

We trained the SVM with the properties of all available 3D defects. This means that we mixed high-pressure cavities and porosities with their gravity counterparts because: (1) the defects in both casting methods have the same range of sphericity, elongation and compactness and (2) the gravity database (98 defects) is very small to train a classifier on its own. The final dataset was split into a training dataset (80%) to train the SVM and a test dataset (20%) to evaluate the predictive ability using a confusion matrix.

The first training attempt consisted of training SVM with the 3 properties of each defect to classify it into porosity or shrinkage cavity. Figure VI.8a shows the most approximate 3D hyperplane found by SVM that can separate the two categories, where each point represents a defect whose coordinates are its geometrical properties. After training, the model is tested on 20% of the data and the confusion matrix of this test is illustrated in Figure VI.8b. The recall of porosities class is 10.34%, and 28.12% for the shrinkage cavities class, which make the model unready for deployment.

In the second training attempt, we added the volume of each defect to the list of training properties. The confusion matrix showed a slight improvement, but the overall performance is still insufficient, as shown in Figure VI.9. Indeed, SVM requires more features to achieve high inference results [48]. But finding relevant features is a very tedious task, and since we have already binarized the defects and have no texture features, we are limited to geometrical features. For this reason, we need to find a model that can be trained with the available data, with the following hypothesis and limitation:

- Defects with sphericity lower than 0.6 (shrinkage cavities) and higher than 0.8



(a) **(b)**

Figure VI.8: Training results of SVM with 3 features: (a) SVM found the most approximate hyperplane in $\mathbb{R}^3$ that separates as much as possible the red points (porosities) and blue points (shrinkage cavities); (b) Confusion matrix that shows the classification results of SVM on a test vector of 58 shrinkage cavities and 32 porosities.

(porosities) do not need to be classified, and can be used as references to asset the type of other inferences.

- Our database contains only 462 defects, or examples, or shots.

VI.4 Few-Shot Learning for Defects Classification

Few-shot learning is the practice of training a model with few training examples [134]. Siamese neural networks (SNN) are deep learning models that fall into this category [135]. Suh models take 2 (or more) images as input and give the probability of similarity as output, as shown in Figure VI.10. SNNs are a special class of neural networks because they consist of two (or more) identical subnetworks that have the same architecture, the same parameters and the same weights. During training,

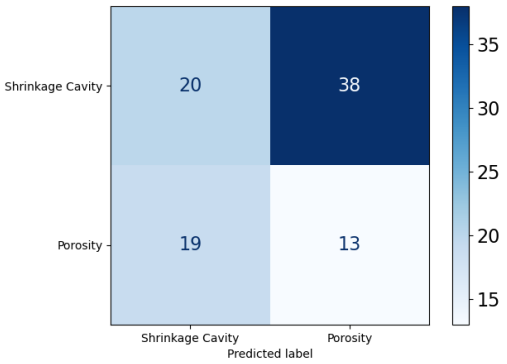


Figure VI.9: Confusion matrix that shows the classification results of SVM trained with 4 features, on a test vector of 58 shrinkage cavities and 32 porosities.

the update of the weights in both subnetworks is mirrored, which means that the update of the weights in one subnetwork leads to the update of the weights in the other subnetwork.

During training, the model is exposed to a set of pairs labelled "0= negative pair" (different objects) or "1= positive pair" (similar objects). This means that SNN does not learn to classify the objects in the traditional sense by selecting 1 of N possible classes, but rather decides whether the objects in the pair represent the same category or not. The first layer after the convolutional subnetworks is usually (but not always) an embedding layer that calculates the Euclidean distance between the latent representation of the two images before giving the probability of similarity. In our application case, each training pair contains two 3D images representing defects belonging to the same category, or different categories, without calculating any geometrical features.

VI.4.1 Preprocessing

When building a neural network, we must define its input dimension. The 3D images must be the same size, which is not the case with our database, so a preprocessing step is required before sending them to the network. The size of the input data can be arbitrary, but there are some important factors to consider when choosing a reasonable size: (1) neural networks perform better when trained with square or cubic images; (2) the size must be divisible by the size of the max-pooling filters, e.g. if we have 3 layers of max-pooling in the architecture with a filter size of $(2 \times 2 \times 2)$, the cubic input must be divisible by 2 at least three times to avoid the checkerboard effect during convolutional operations [136]; and (3) the size of the

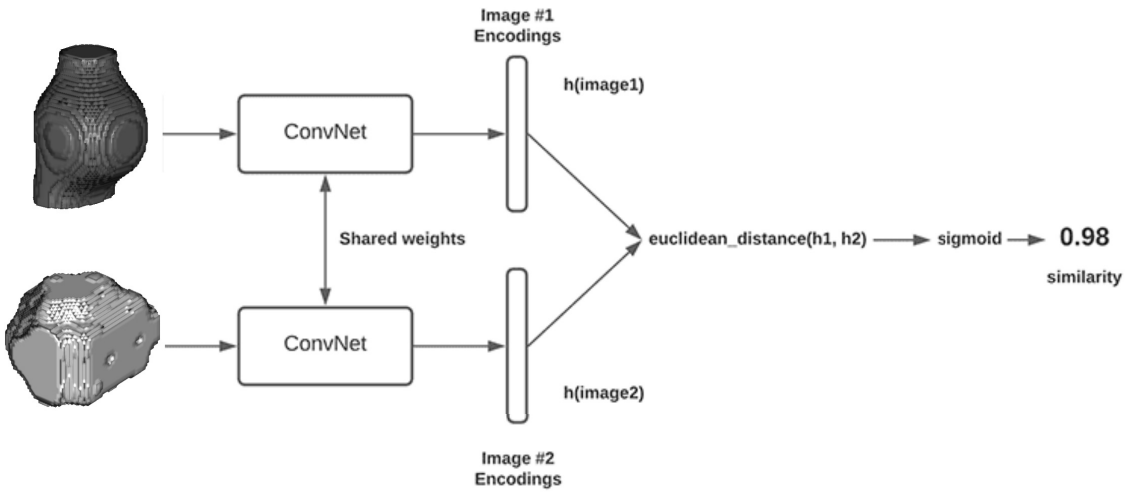


Figure VI.10: Siamese neural network consists of two convolutional subnetworks that take a pair of images as input, encode each image into a possible latent representation, calculate the Euclidean distance between these two representations and finally decide if they are similar or not.

cube must not be very large to avoid GPU saturation. The input size was set to $(40 \times 40 \times 40)$, and the preprocessing consists of two stages:

1. The bounding box that contains each defect is padded with zeros along the smallest dimensions to create a cubic volume. For example, if the bounding box has a shape of (4, 3, 5), the edges of the smallest axes are padded symmetrically to give a cubic final shape of (5, 5, 5).
2. Each cubic bounding box of the previous step is zoomed in or out to $(40 \times 40 \times 40)$ using spline interpolation [137].

VI.4.2 Data Augmentation

As explained in subsection III.7.2 and subsubsection IV.6.5.2, DA helps reduce the risk of overfitting by training the model with synthetic data. The latter are generated by applying transformations to the original dataset, while preserving the original shape. This means that shear and twist transformations were avoided to preserve a realistic shape for each defect. Finally, SNN model was trained during each epoch on an augmented version of the dataset that undergone the following transformations in a random manner:

- Flipping along one of the three axes.
- Rotation at an angle in a range of $]-180, 0[\cup]0, +180[$, along one of the three axes.

VI.4.3 Creating Pairs

As explained earlier, SNN does not decide whether a defect belongs to the porosities class or cavities class, but reports whether 2 defects are the same (belong to the same class) or different (belong to different classes). For this reason, the training dataset must contain positive and negative pairs:

- Positive pairs: 2 defects that belong to the same class.
- Negative pairs: 2 defects that belong to different classes.

By pairing each defect at least once with every other defect in the dataset of size N, the result is a dataset with a total number of pairs equal to N', as depicted in the equation below. As a result, 106491 pairs were generated and split 85% for the learning process and 15% for testing the predictive ability of the model after training is done the weights are saved. During each training epoch, 70% of the training data are augmented to train the model, and 30% for validating the weights of the network at the end of the epoch.

$$N' = \frac{(N-1)\left[(N-1)+1\right]}{2} \tag{VI.3}$$

VI.5 SNN Architecture and Hyperparameters

As shown in Figure VI.10, each 3D image of the pair passes through a branch of the siamese neural network. Each branch starts with a convolutional subnetwork that systematically converts the 3D input into a series of feature vectors that are selected after several trials. Table VI.1 shows the architecture of each branch (which are identical), consisting of eight layers: Each 3D defect is convoluted with a 3D filter and its size is gradually reduced until it reaches the encoding layer, which holds a possible latent representation of size $(5 \times 5 \times 5 \times 256)$.

Table VI.1: The optimal architecture of the convolutional subnetworks of the SNN model, for an input size of $(40 \times 40 \times 40)$.

# Layer	Layer Type	Filter Size	Number of Filter or Units	Output Shape
1	Conv3D + ELU	$5 \times 5 \times 5$	16	$40 \times 40 \times 40 \times 16$
2	MaxPooling3D	$2 \times 2 \times 2$	16	$20 \times 20 \times 20 \times 16$
3	Conv3D + ReLU	$5 \times 5 \times 5$	32	$20 \times 20 \times 20 \times 32$
4	MaxPooling3D	$2 \times 2 \times 2$	32	$10 \times 10 \times 10 \times 32$
5	Conv3D + ReLU	$5 \times 5 \times 5$	64	$10 \times 10 \times 10 \times 64$
6	MaxPooling3D	$2 \times 2 \times 2$	64	$5 \times 5 \times 5 \times 64$
7	Conv3D + ReLU + Dropout	$5 \times 5 \times 5$	128	$5 \times 5 \times 5 \times 128$
8	Enc Layer = Conv3D + ReLU	$5 \times 5 \times 5$	256	$5 \times 5 \times 5 \times 256$

After finding the latent representation of each 3D defect, the Euclidean distance between the two is calculated, i.e. the square root of the sum of the squared differences between the two vectors, with a size of $(5 \times 5 \times 5 \times 256)$. As can be seen in the diagram of Figure VI.11, the output is flattened using a global average pooling layer, followed by a dense classification block consisting of three dense layers, and a sigmoid activation function at the output. The latter gives the probability of similarity between the defects of the pair, which is between 0 and 1. During inference, we can binarize the output by applying a threshold of 0.5 to this probability to get 0 (belong to the same category) or 1 (different categories).

During each epoch, training and validation were evaluated using binary cross entropy (Equation IV.6) and accuracy (Equation IV.2), as our comparison task is binary (similar or different). Since a large amount of data was generated after pairing each defect with every other at least once, the batch size was set to 100. This means that the SGD optimizer, selected after trial-error attempts, updates the weights of the network after exposing the SNN to only 100 pairs, instead of waiting until the end of the epoch, which ensures faster convergence of the loss function towards a local minimum, as explained in subsection IV.3.2.

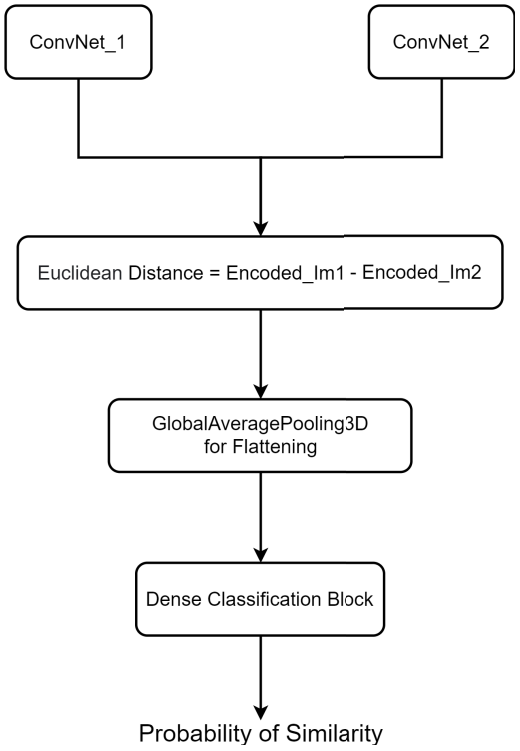


Figure VI.11: Our siamese neural network consists of two convolutional branches to find a latent representation for each defect, followed by flattening with global average pooling, and classification with a series of dense layers that yield the final probability of similarity between the two input defects.

VI.6 Results & Discussion

By training SNN with a large number of pairs and applying data augmentation during each epoch, the model learned the underlying relationship between the inputs (defect pairs) and the corresponding ground-truth outputs (similar or different) very quickly. Figure VI.12 shows that SNN needed only a few epochs to achieve low loss and high accuracy. Achieving low losses and high accuracies with a small gap between training and validation losses means that the model was neither under- nor over-fitted.

Training was stopped after 20 epochs with training and validation losses of 0.0221 and 0.005 and training and validation accuracies of 99.11% and 99.97%. To test the predictive ability of the model, it was tested on the remaining data to gain insight into its sensitivity to similar and different pairs of defects. Table VI.2 shows the inference results on a test vector of 7302 pairs of similar defects and 5478 pairs of different defects. As the confusion matrix shows, the model achieved perfect results with an F1-score of 99.63%, classifying only 40/5478 different defects as similar. This means that the model is able to recognise whether a pair of defects represents two inferences of the same category or whether both belong to a different category (porosities or shrinkage cavities).

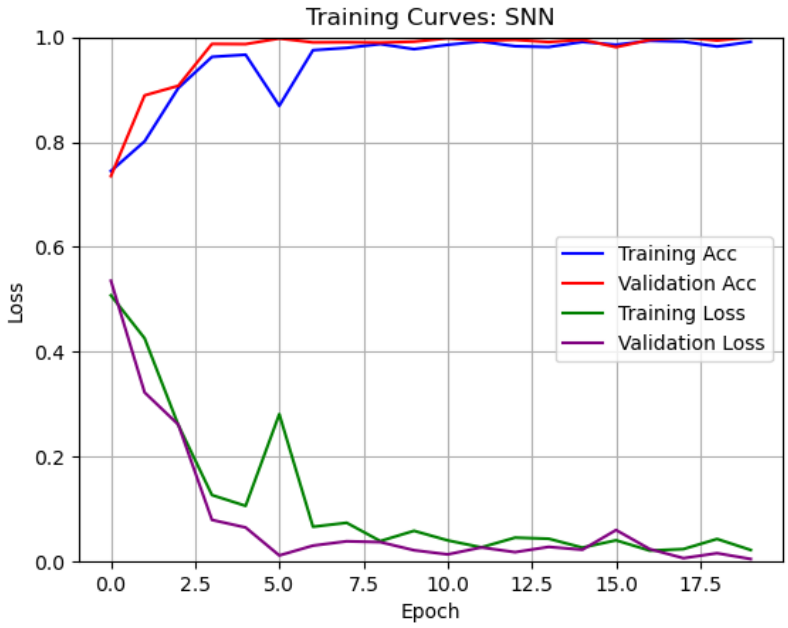


Figure VI.12: Training curves of the SNN model on a dataset of defect pairs. The model achieved the lowest training and validation losses after 20 epochs.

VI.7 Deployment Case

At the output of the approach in Figure V.1, the binary volume contains many embedded defects that can be categorized into porosities or shrinkage cavities. With the help of the SNN model, each defect can be classified using the following pipeline:

1. Defects with high sphericity and compactness, i.e. porosities; and defects with high elongation and low sphericity, i.e. shrinkage cavities, are automatically categorized and saved as references.
2. Defects with intermediate sphericity, compactness and elongation are randomly paired with a sample of defects from the first category.
3. The pairs are sent to the trained SNN model to decide whether they are similar

Table VI.2: Inference results of SNN model on a test vector of 5478 negative pairs and 7302 positive pairs. The model achieved high F1-score of 99.63%.

SNN Confusion Matrix

	Negative Pairs	Positive Pairs
Negative Pairs	5438	40
Positive Pairs	0	7302

or not. For example, if an intermediate defect has a higher average probability of similarity when paired with shrinkage cavities, it is categorized accordingly.

Conclusion

This chapter focused on the geometrical properties of defects as well as the defect categories. Porosities and shrinkage cavities have different morphologies in terms of sphericity, compactness and elongation, as the former are caused by gas bubbles and the latter by metal contraction and lack of material during solidification of the casting.

The field of few-shot learning was explored and the potential of siamese neural networks in classifying pairs of defects as similar (two porosities or two shrinkage cavities) or different. With this model, embedded defects can be classified as porosities or shrinkage cavities in the binary output of the approach of the last chapter by forming pairs with defects of clear morphologies, considered as references.

Conclusion

Industrial CT images segmentation using classical image processing techniques requires the adjustment of several parameters, since the contrast resolution changes with the thickness of the specimen and since CT is prone to different types of artifacts that depend on the tomography system. This thesis covered an automatic detection approach that was developed to isolate casting defects in CT volumes of aluminium castings without requiring user intervention. This approach consists of the following steps:

Deep Segmentation: Segmentation of the CT slices carries two major risks: (1) under-segmentation, which leads to genuine defects being overlooked, or (2) over-segmentation, which leads to irrelevant discontinuities being highlighted. Since the first case leads to skipping defects that could be crucial and harmful, we decided to train the state-of-the-art U-Net model with a slightly over-segmented database to reduce the number of false negatives in its output, even if at the expense of false positives.

Deep Classification: To eliminate false positives and recognize true defects, a classifier must be coupled with U-Net to remove irrelevant discontinuities. CT-Casting-Net, a new convolutional neural network, was trained in a supervised manner on a two-class dataset containing greyscale zones with defects (true alarms) and zones with artifacts, noise or homogeneous material (false alarms).

Segmentation Followed by Classification: To develop an automatic defect detection approach, U-Net and CT-Casting-Net are coupled together to process industrial CT volumes. U-Net over-segments each slice of the model making up a binary volume with 3D objects representing real defects or false alarms. By isolating each object, three ortho-images are cropped around the centre-of-mass and sent to CT-Casting-Net to decide whether to keep or remove the object. The final output is a binary volume containing the embedded discontinuities inside the greyscale CT input classified as defects.

Defects Characterization and Categorization: Defects occur during the solidification of the molten aluminium alloy as it takes the final shape of the mould. These defects can represent gas bubbles, metal contractions or turbulence in the molten metal. By studying the geometrical properties of the embedded defects, we can categorize them into porosities or shrinkage cavities based on their sphericity, elongation and compactness, or decide whether two defects belong to the same category or whether they are different using a siamese neural network that we have

trained with very many examples.

When applied to a new inference of size $512 \times 512 \times 100$ for example, this approach requires only 20 seconds of processing, including the over-segmentation that yields an excess of discontinuities, the automatic classification of these discontinuities into true defects or false alarms, and finally the categorization of the true defects into porosities or shrinkage cavities.

Building a training dataset for each model, finding the optimal architecture and the right set of hyperparameters, and validating against new unseen inferences required tedious work and a long series of trial-error attempts. The global processing methodology is now ready to be integrated into a computer vision software at the industrial sites to help metal manufacturers who use CT as a non-destructive method to inspect their casting volumes, locate casting defects and investigate their impact on the lifetime of the component.

Future Work

Extending the database: the available data have a spatial resolution between 150 μm and 450 μm. This database needs to be expanded in the future with unseen volumes with more unique contrast and spatial resolutions to refine the models and increase their generalization on new inferences derived from new tomography systems. In addition, the database can be expanded to include castings made from magnesium and titanium alloys, not just aluminium.

Extending Current Models: the siamese network has only been trained with the most common and frequent defect types, porosities and shrinkage cavities. The 3D defect database needs to be expanded to include surface cracks and cold fills, which are rare, and the siamese model can be re-trained with the four categories instead of two to increase its reliability.

Résumé en français

Détection automatique des défauts dans des volumes tomographiques des pièces de fonderie en alliage d'aluminium

1 Introduction : contexte et objectives

En France et en Europe occidentale, des fonderies se sont équipées avec des tomographes industriels pour le développement et la mise au point des pièces mais aussi pour le contrôle de production. Dans ce dernier cas, le but est d'avoir des informations pertinentes sur le suivi de la qualité de la production afin d'anticiper des dérives éventuelles. Dans le secteur automobile, toutes les pièces ne peuvent pas être contrôlées en tomographie et la tendance est de privilégier les temps d'acquisition plutôt que la recherche de résolution, ce qui réduit la qualité des images. D'ailleurs, la tomographie est très prône aux artefacts, ce qui nécessite un logiciel de traitement d'images pour interpréter ses volumes d'une façon automatique et isoler les vrais défauts. Afin de répondre à ce besoin, le Centre Technique des Industries de la Fonderie (CTIF) a lancé ce projet de recherche en collaboration avec le Laboratoire Vibrations et Acoustique (LVA) de l'INSA de Lyon et avec des groupes industriels : RENAULT, SAB, EUROCAST, MONTUPET et CONSTELLIUM.

L'objectif de ce travail porte sur le développement d'algorithmes de traitement des données tomographiques des pièces de fonderie en alliages d'aluminium. Afin de pouvoir automatiser entièrement le traitement, plusieurs types d'approches d'intelligence artificielle ont été employées, pour (1) localiser et segmenter des indications pouvant être des défauts, mais aussi des artefacts ; (2) classer ces indications en vrais défauts ou fausses alarmes ; (3) identifier la nature des vrais défauts (retassure ou porosité).

La partie segmentation des volumes tomographiques repose sur un apprentissage à partir d'une base de données de 3000 images 2D en niveaux de gris, extraites de

20 volumes tomographiques fournis par les industriels du projet. Les 3000 images binaires "cibles" correspondantes ont été obtenues par un algorithme de traitement d'image classique nécessitant un ajustement manuel du seuil de détection. D'autre part, la partie classification a été développée grâce à une base d'images 2D de petite taille (64×64) en niveaux de gris, constituée de 5500 images de défauts, et 8500 images de zones normales (y inclus des artefacts). Ces images ont été elles aussi extraites des 20 volumes industriels, mais également de la base de données interne du CTIF. Les performances de la chaîne globale de traitement (segmentation + classification) ont été mesurées sur 6 volumes tomographiques supplémentaires fournis par les industriels et par l'INSA.

Enfin, la partie identification de la nature de défaut a nécessité une base d'images 3D de défauts dont la nature a été labélisée. 462 volumes (320 retassures, 142 porosités) ont été fournis, soit segmentés à partir de la base interne du CTIF, soit embarqués dans les volumes fournis par les industriels. Il est important de noter ici que l'identification fine du défaut nécessite le volume complet de celui-ci, ce qui explique le fait que le nombre d'images est plus réduit que lorsque la base de données est constituée d'images 2D.

2 Le principe de la tomographie

Comme la radiographie (en deux dimensions), la tomographie est basée sur l'absorption différentielle des rayons X en fonction de la densité de matière, mais elle exploite un plus grand nombre de vues réalisées suivant différents angles par rotation de l'objet observé comme illustré dans la Figure VI.13. Les différentes vues permettent de déterminer l'absorption de chaque élément de volume appelé "voxel" et ainsi de reconstituer l'objet en trois dimensions. Il est alors possible d'obtenir plusieurs représentations du volume de l'objet, dont la visualisation se fait sous forme de coupes virtuelles. Cette représentation est la plus conventionnelle et la plus pratique pour déterminer des taux de porosités ou mesurer des discontinuités internes à la pièce. Pour examiner la totalité du volume, il faut faire défiler à l'écran les coupes virtuelles 2D ou utiliser un algorithme permettant une représentation en 3D du volume de l'objet.

En tomographie, deux systèmes correspondant à deux géométries d'acquisition existent :

- Les systèmes dits "Cône-Beam" pour lesquels, le cône de rayonnement du tube à rayons X est exploité pour examiner l'objet à partir de vues 2D obtenues sur un détecteur plan (ou panneau plat). Ce sont les équipements principalement utilisés pour la tomographie industrielle.
- Les systèmes dits "Fan-Beam" pour lesquels, un fin faisceau de rayonnement est envoyé à travers l'objet vers un détecteur linéaire. Ces matériels sont plutôt réservés aux hautes énergies lorsque le rayonnement diffusé devient trop important.

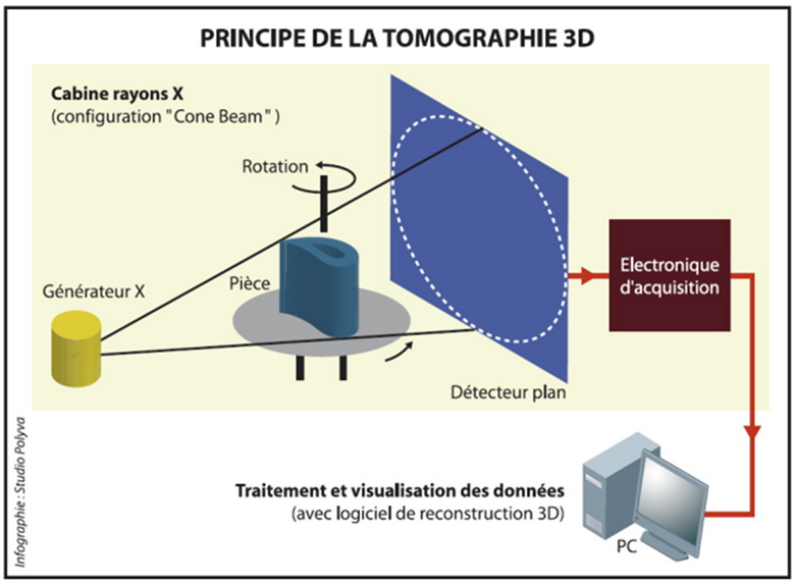


Figure VI.13: Principe de la tomographie en configuration "cône-beam".

Dans le cadre de ce projet, les données fournies par les partenaires industriels sont issues de systèmes "cone beam" essentiellement. A noter que la résolution spatiale du système dépend du besoin et résulte d'un compromis avec le temps d'inspection. Par conséquent, les images fournies pour ce travail ont des tailles de voxels pouvant aller de 150 à 450 µm.

3 Le besoin industriel vis-à-vis de la tomographie

Pour le secteur industriel de la fonderie qui produit des pièces métalliques aux formes "tourmentées" avec souvent des conduits internes, la tomographie est un outil d'analyse de la santé interne de ces pièces. Par rapport à la radiographie, qui produit des images du volume projeté sur un plan, la tomographie permet d'examiner la matière en coupant le volume selon certaine direction, générant une série des coupes. Cela évite d'être gêné par les nombreuses variations d'épaisseurs ou par les projections de parois qui sont caractéristiques de la radiographie de pièces de fonderie, et ainsi la reconnaissance de la nature des discontinuités présentes est grandement facilitée. Une illustration de la puissance de la tomographie par rapport à la radiographie est l'examen des structures alvéolaires (structures lattices) métalliques, issues de fonderie ou de fabrication additive est dans la Figure VI.14. Cette figure montre la différence entre l'image radiographique et une coupe tomographique d'une mousse métallique produite par fonderie: en radiographie, l'image est floue, les porosités se superposent et il est impossible de déterminer la dimension des pores, tandis que les coupes tomographiques donnent des images nettes qui permettent d'accéder à des données précises sur la dimension des pores et leur distribution.

En tomographie, le positionnement de la discontinuité dans l'épaisseur devient possible ainsi que sa visualisation suivant plusieurs orientations. Cela permet d'envisager

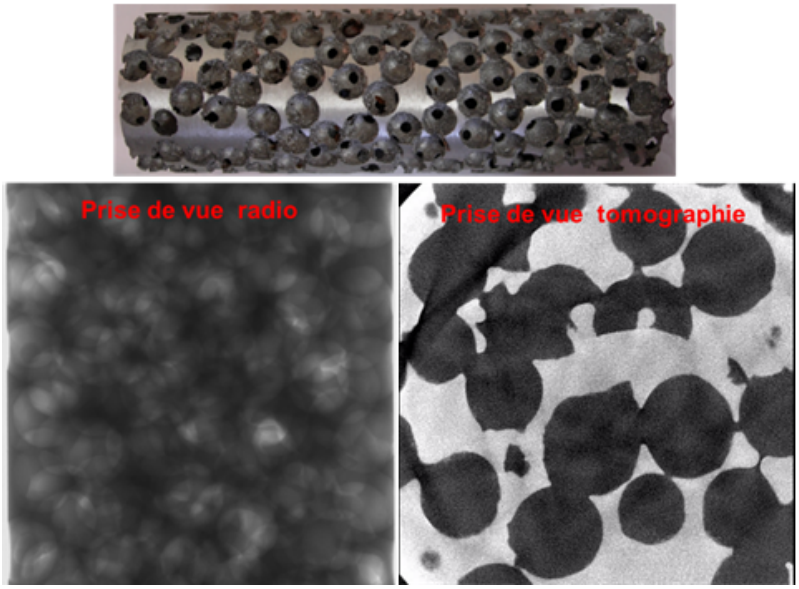


Figure VI.14: Illustration des résultats de prise de vue en radiographie (à gauche) et en tomographie (à droite) d'une mousse métallique produite par fonderie (les vides sous forme de sphères ont ici des dimensions de l'ordre de 5 à 8 mm).

des critères plus pertinents en fonction de la position de la discontinuité par rapport à la surface ou par rapport à une zone critique très sollicitée. Actuellement, la conformité de la santé interne des pièces de fonderie est réalisée en comparant des images radiographiques 2D avec des images de référence disponibles auprès de l'ASTM (American Society for Testing and Materials). L'inconvénient est que cela peut conduire à rebuter une pièce alors que la discontinuité se situe dans la fibre neutre et n'aura pas d'influence sur sa durée de vie en service. A l'inverse, de petites indications situées en zone superficielle critique peuvent être conformes aux cahiers des charges et néanmoins conduire à des ruptures prématurées de pièces. Avec la tomographie, il est possible de savoir si les discontinuités vont partir à l'usinage ou déboucher en surface ou si elles se situent dans une zone désignée dangereuse pour la tenue mécanique de la pièce.

D'autre part, le volume des données tomographiques est très important, aussi une exploitation manuelle coupe par coupe est fastidieuse. L'automatisation du traitement d'image est donc nécessaire pour une utilisation en contrôle de production. Dans le cadre de cette thèse, nous avons souhaité développer l'ensemble des traitements de façon indépendante de tout logiciel afin de maîtriser complètement la chaîne de traitement. Le développement des algorithmes a été réalisé en Python 3.6, Matlab et C++17, notamment les traitements d'images dits "classiques", et les algorithmes de deep learning.

4 La chaîne globale de traitement : segmentation et classification

Le processus de traitement des données issues de coupes tomographiques comprend plusieurs étapes résumées sur Figure VI.15:

- La sur-segmentation qui a pour but de séparer les zones d'irrégularités du reste de la pièce : celle-ci se fait coupe par coupe, puis les coupes sont empilées pour former un volume binaire, et les indications sont labélisées.
- La classification des irrégularités, ou indications, pour séparer les vrais défauts des faux positifs. Les faux positifs sont les indications relevées qui ne sont pas des discontinuités de matière, mais plutôt des artefacts de tomographie. La classification se fait sur la base de trois images perpendiculaires entre elles qui sont découpées autour du centre-de-masse de chaque indication labélisée à l'étape 1. Une indication est classée comme un défaut si les trois indications présentent à la fois des zones défectives.
- Le "nettoyage" du volume binaire en éliminant les indications classées comme fausses alarmes.
- Et finalement le classement des défauts validés selon 2 types : dans notre cas, retassures ou porosités (cette étape n'est pas représentée dans la Figure VI.15).

4.1 L'étape de segmentation par deep learning

La segmentation s'applique aux coupes tomographiques donc en 2D. Cette segmentation est basée sur les différences de niveaux de gris entre les défauts et la matrice. Les défauts étant en fonderie le plus souvent des manques de matière, les niveaux de gris de ces indications sont plus sombres (atténuation plus faible). Dans ce cas, la segmentation nécessite une opération de seuillage où une limite en niveau de gris doit être définie telle qu'en-deçà de cette valeur tous les voxels sont considérés comme faisant partie du défaut et tous les autres comme étant de la matière saine. On obtient ensuite une image binaire avec les défauts en blanc et le reste de la pièce en noir. Bien évidemment, il faut auparavant avoir défini et fermé les contours de la pièce, ce qui est souvent délicat. Ce processus de segmentation par seuillage est difficilement automatisable et nécessite un certain niveau d'expertise pour bien placer le curseur au bon endroit, au risque sinon de donner des résultats en volume ou densité de défauts qui seraient incohérents avec la réalité. En outre, la quantité d'images en tomographie est très élevée et il est indispensable que les systèmes de détection des indications soient autonomes. De plus, les images tomographiques comportent un grand nombre d'artefacts : des cercles concentriques "rings" ou des marques dues au durcissement du faisceau de rayons X lorsqu'ils traversent beaucoup de matière, comme illustré dans la Figure VI.16). Ces artefacts peuvent être considérés comme des défauts dans le cas d'une segmentation par seuillage global.

Pour améliorer l'efficacité de la détection, et surtout obtenir une méthode entièrement automatique, sans intervention de l'opérateur, et fonctionnant pour toutes les

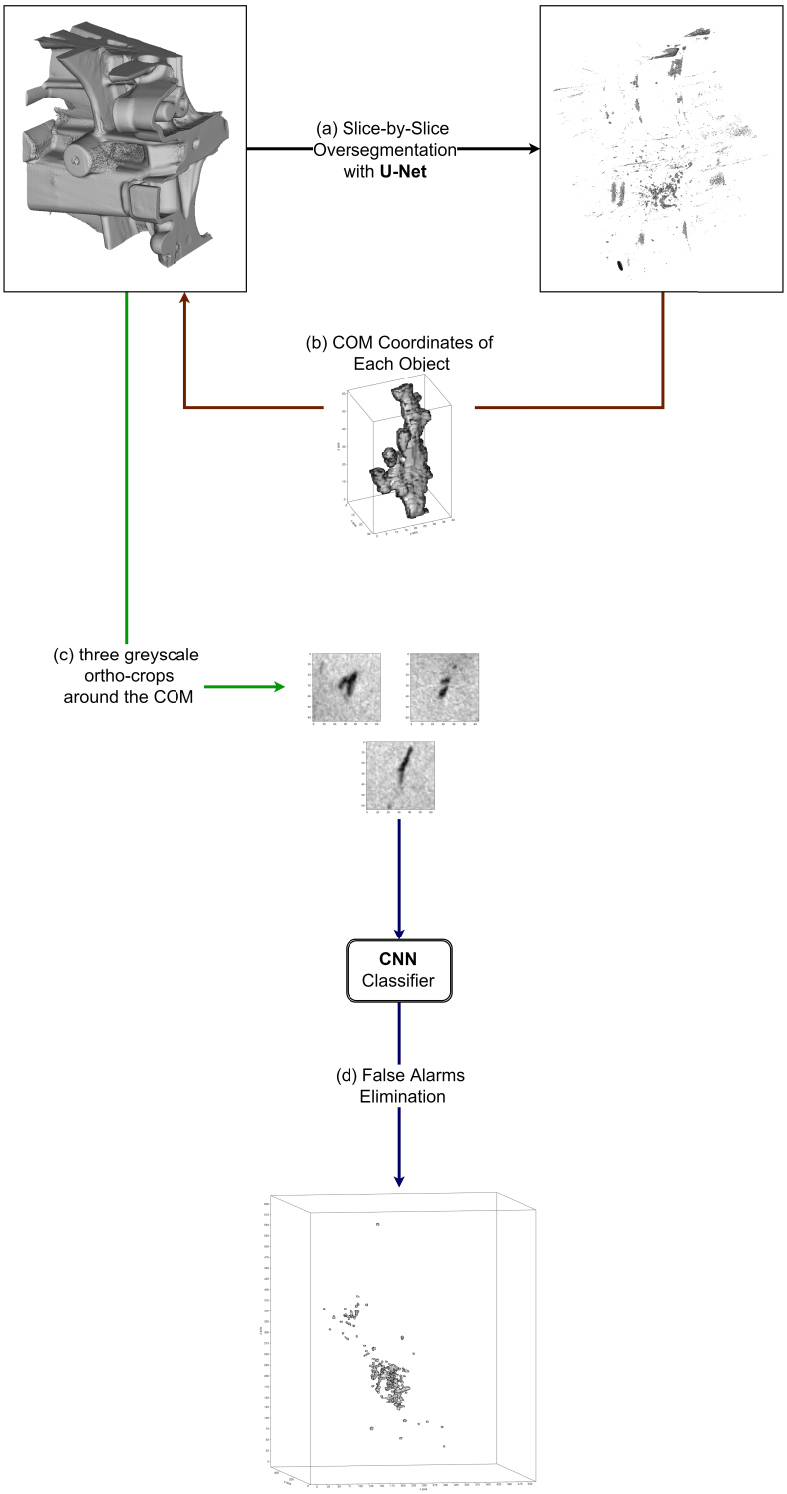


Figure VI.15: Résumé de la chaîne globale de détection et classification des indications en vrais défauts ou fausses alarmes.

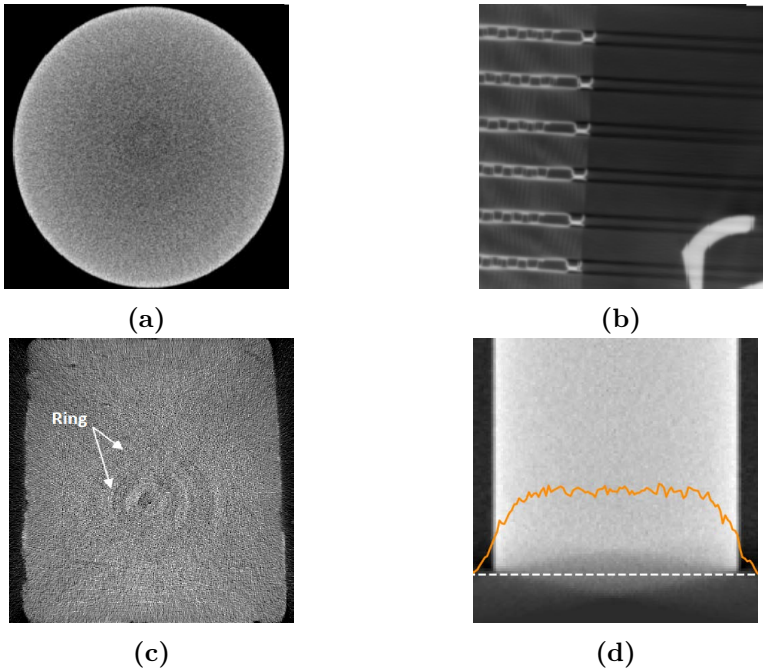


Figure VI.16: Different types d'artefacts dans la tomographie industrielle : (a) beam hardening, (b) streaking, (c) ring artifact, and (d) Feldkamp artefact.

pièces fournies par les industriels, nous avons décidé d'utiliser les algorithmes de type deep Learning avec une approche en deux temps : segmentation puis classification.

Pour la segmentation, nous avons sélectionné des réseaux de neurones convolutifs type U-Net, cf. Figure VI.17. Ce type de réseau de neurones a été développé en 2015 pour la segmentation d'images biomédicales au département d'informatique de l'université de Fribourg en Allemagne et a donné des résultats très performants, au-delà de tous les réseaux équivalents. Il s'agit d'un réseau entièrement convolutif et sous forme de "U", qui a l'avantage de pouvoir fonctionner avec relativement peu d'images d'entrainement et de permettre une segmentation précise. Il fait partie de la famille des auto-encodeurs : la partie encodage (à gauche dans la Figure VI.17) consiste à obtenir une représentation "latente" de l'image d'entrée sous forme de caractéristiques, tandis que la partie décodage permet de reconstruire une image de sortie qui doit ressembler le plus possible à la version binarisée de l'image initiale. Il faut noter que la sortie n'est pas directement une image binaire mais une image dont chaque voxel représente la probabilité d'appartenir à la classe discontinuité si sa valeur est proche de 1, ou bien la classe arrière-plan si sa valeur est proche de zéro.

4.1.1 Base de données d'entraînement

Un ensemble de 3000 images de taille (512×512) a été extrait des 20 volumes tomographiques fournis par les industriels. Nous avons volontairement mélangé des images provenant de différents procédés de fonderie (sous pression et gravité), de différents tomographes, de façon à avoir une base riche en termes de diversité

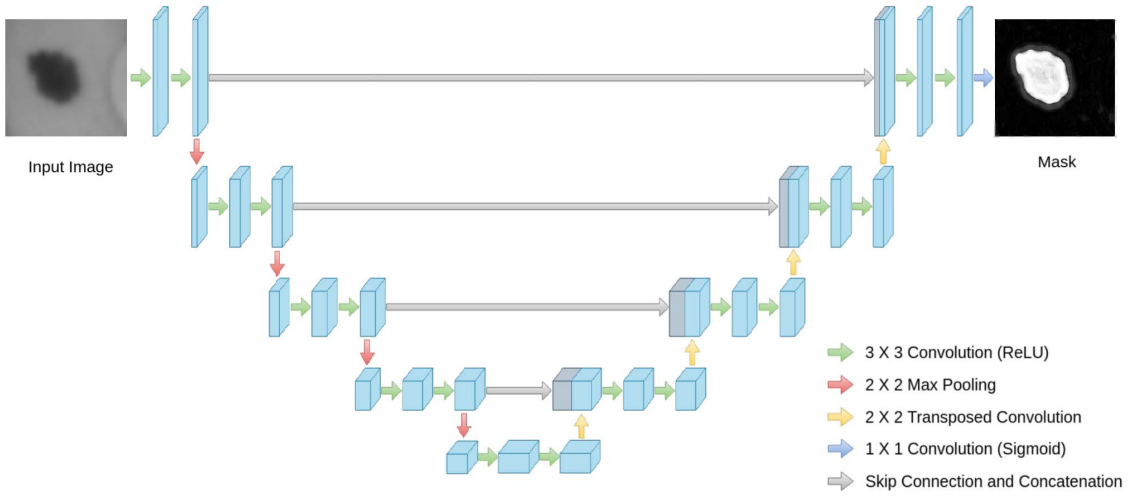


Figure VI.17: Architecture type U-NET : le modèle consiste en une partie encodage à gauche, et décodage à droite, chaque couche étant connectée à sa correspondante par une concaténation des vecteurs de caractéristiques. La sortie du modèle est une image dont les valeurs sont comprises entre 0 et 1 et correspondant à la version la plus proche de la version binaire de l'image d'entrée.

d'artefacts, de contraste, et de résolution spatiale. Le découpage en 512 permet à la fois de conserver une taille non négligeable permettant d'avoir une segmentation non seulement des défauts mais aussi des zones normales comme les bords de pièce, ou les artefacts, et pour préserver la vitesse de traitement. Dans la mesure où on souhaite développer un réseau de type segmentation supervisée, il est nécessaire que chaque image en niveau de gris soit accompagnée de sa version binaire "vérité terrain" (ground-truth). Cette image binaire a été obtenue par un algorithme de traitement d'image automatique développé pendant la thèse, utilisant une approche classique de débruitage, puis détection de contours et seuillage par la méthode d'Otsu. Le seuil calculé par l'algorithme d'Otsu a été modifié par un facteur de réglage qui permet d'ajuster la sévérité de la segmentation. Etant donnée que les volumes tomographiques sont très gros, et les pièces sont d'épaisseur variable, le contraste varie fortement au sein d'un même volume, ce qui empêche de trouver un seuil optimal unique pour un volume entier. C'est pourquoi nous avons d'une part recherché une méthode complètement automatique de type U-Net, et d'autre part, utilisé ce réglage manuel pour générer la base de données de vérité terrain. A noter qu'il n'était pas possible d'obtenir cette vérité terrain directement de la part des industriels car chacun aurait réalisé une segmentation sans doute différente, et certains ne sont pas équipés de système de traitement automatique. La Figure VI.18 montre une série d'images exemples tirées de la base ainsi constituée : dans la colonne de gauche, le choix du facteur de sévérité est réalisé pour n'avoir que les défauts dans la vérité terrain, alors que dans la colonne de droite, on choisit de détecter un excès de discontinuités.

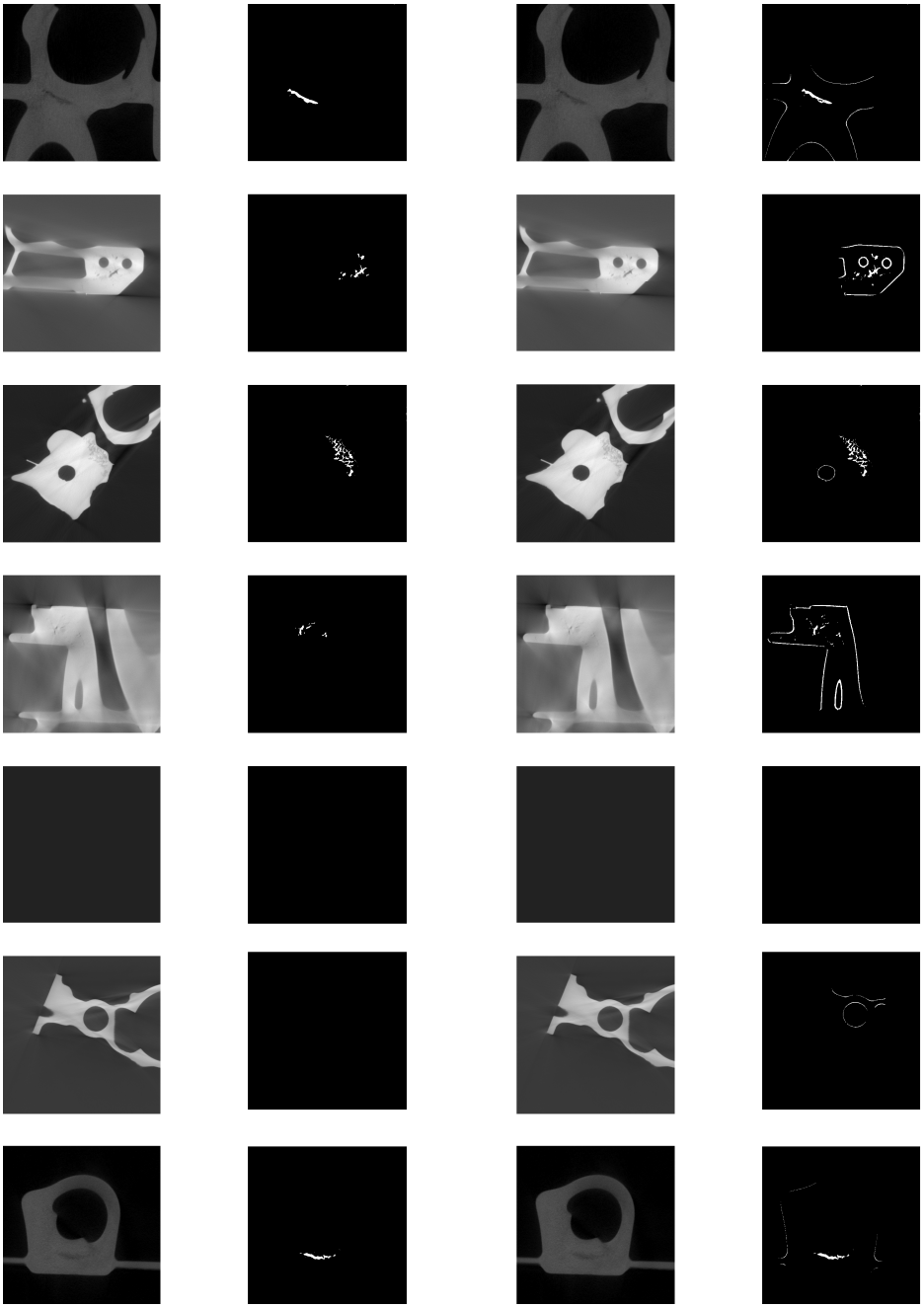


Figure VI.18: Exemples d'images en niveau de gris et leur image binaire correspondante : à gauche version initiale de la base contenant uniquement les défauts segmentés, et à droite version modifiée de la base avec une sur-segmentation.

4.1.2 Résultats d'entrainement

Un premier essai du réseau U-Net a été effectué avec la base de données d'entraînement dont les images binaires ont été segmentées de façon à ne conserver que les défauts (réglage sévère du facteur d'ajustement correspondant à la colonne de gauche sur la Figure VI.18. Le résultat n'a pas été probant car le réseau n'a pas été capable de généraliser l'entraînement sur la base de test : en effet, les images en niveau de gris contiennent des artefacts dont le contraste est aussi élevé que les vrais défauts comme montré dans la Figure VI.19.

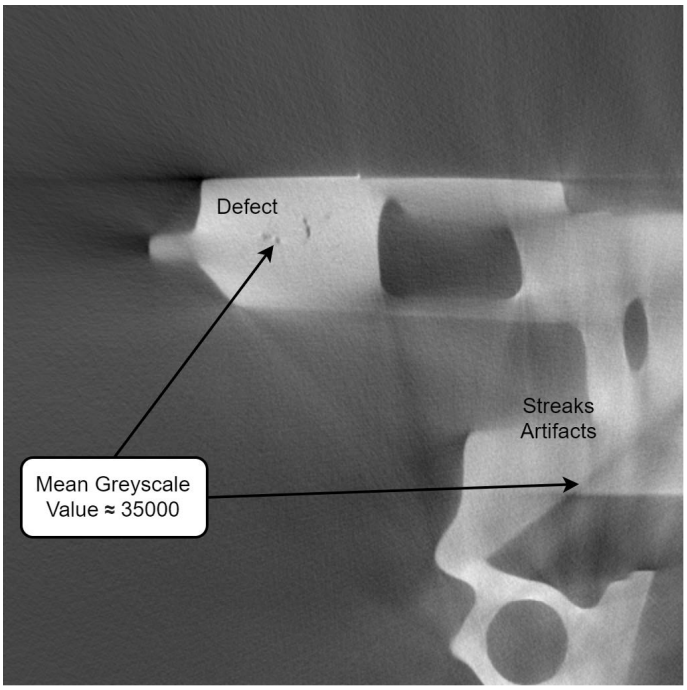


Figure VI.19: Exemple d'image contenant des défauts et des artefacts, de même contraste.

Par la suite, nous avons donc opté pour une méthode de sur-segmentation : la base de données a été modifiée en réglant le seuil de sévérité de façon à conserver un excès de discontinuités : les défauts, mais également les bords de pièce et les artefacts (colonne de droite dans la Figure VI.18). L'intérêt est que le risque de rater des défauts est dans ce cas très faible, voire nul. Néanmoins, la contrepartie est que les indications doivent ensuite être triées pour ne garder que les vrais défauts par un classifieur entraîné.

4.2 L'étape de classification des indications par deep learning

La sur-segmentation entraîne, bien entendu, de nombreux artefacts (faux positifs) qui sont traités au moyen d'un autre réseau de neurones pour trier les vraies discontinuités des faux positifs. Pour cette deuxième étape cruciale pour l'efficacité de la détection, nous avons privilégié de conserver l'information en niveaux de gris,

en faisant le pari que les réseaux de neurones seront capables de séparer les vrais défauts par rapport aux artefacts grâce à ces niveaux de gris. Pour cela, une nouvelle base de données a été créée en découpant des zones avec et sans défauts. Ces zones de taille (64×64) ont été prélevées dans les volumes tomographiques fournis par les industriels, mais aussi dans la base d'images interne du CTIF. La taille de 64 a été sélectionnée pour pouvoir contenir les plus gros défauts quelle que soit la taille des voxels (entre 150 et 450 µm), sans être trop grande non plus pour accélérer le traitement. Au total une base de 5500 images de défauts, et 8500 zones sans défauts (zones normales homogènes, ou artefacts). La Figure VI.20a montre comment les images sont découpées, et la Figure VI.20b montre un extrait de la base.

Nous avons élaboré un réseau peu profond (seulement 10 couches) avec une architecture composée de plusieurs couches convolutives. Nous avons privilégié une architecture simple dans la mesure où nos images d'entrée sont beaucoup moins complexes que les images naturelles telles que celles de la base de données ImageNet sur laquelle de nombreux réseaux disponibles sont optimisés. Le Table VI.3 montre les performances atteintes avec ce nouveau réseau appelé CT-Casting-Net, en comparaison avec plusieurs réseaux de la littérature.

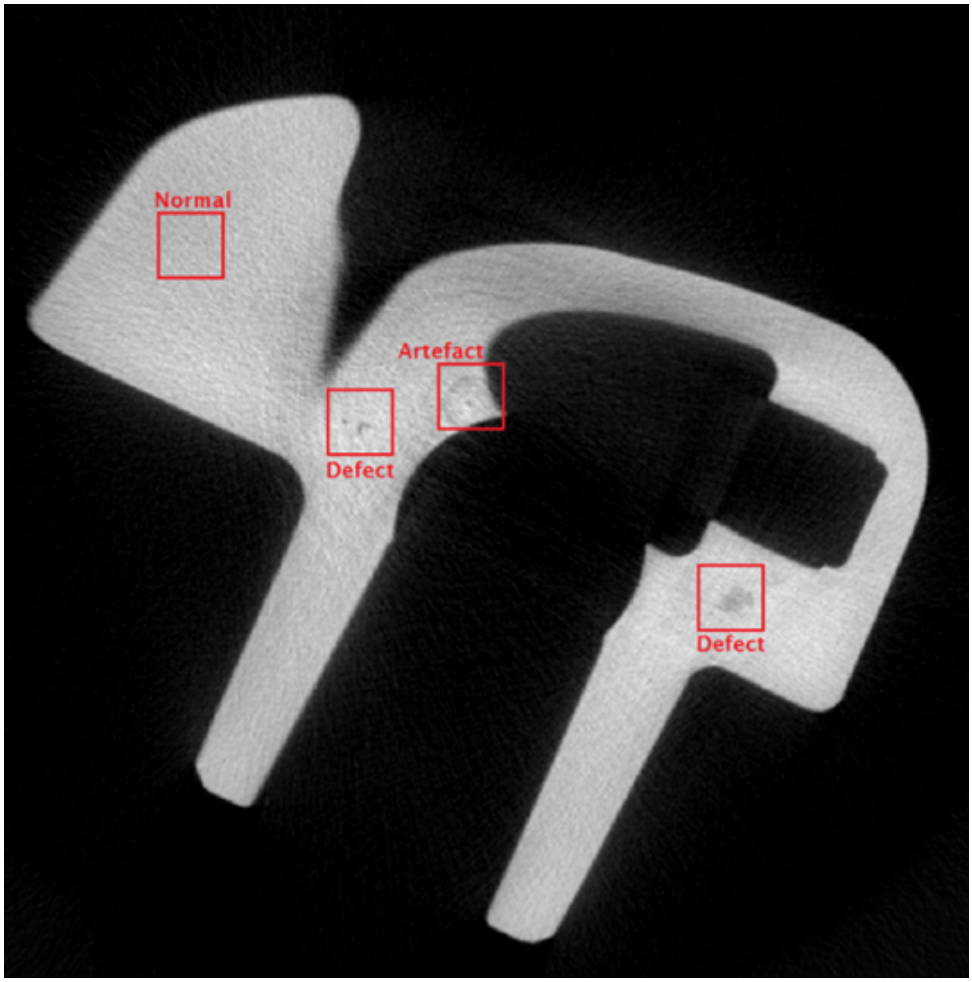
Table VI.3: Performances du réseau CT-Casting-Net en comparaison avec d'autres réseaux de la littérature, lors de l'apprentissage sur la base de 14000 images (5500 avec défauts, 8500 sans défauts).

Model	Depth	Training Loss	Training Accuracy	Validation Loss	Validation accuracy
CT-Casting-Net	**10**	**0.0364**	**98.80%**	**0.0488**	**98.30%**
VGG19	22	0.4764	77.74%	0.4775	78.44%
ResNet50	50	0.1481	94.58%	0.1278	95.47%
ResNet101	101	0.2367	91.68%	0.2620	91.55%
Xception	126	0.3596	88.46%	0.3025	90.01%
InceptionV3	159	0.3588	81.67%	0.3694	82.88%

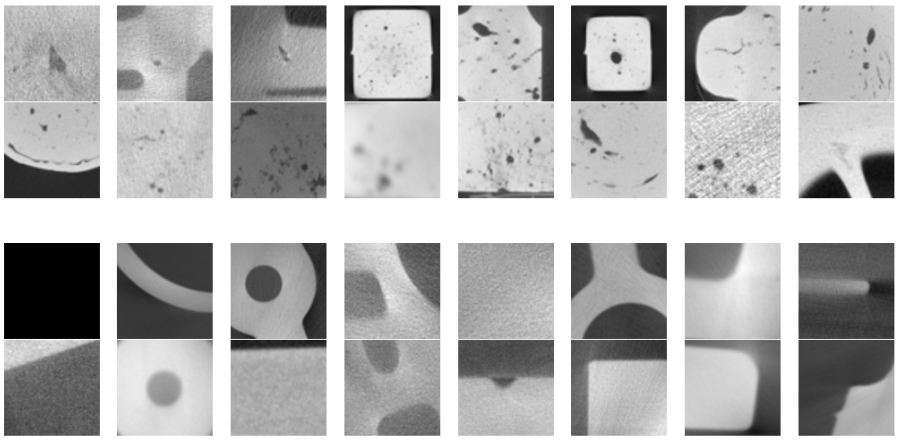
Après l'apprentissage, le réseau est testé sur un ensemble de 700 images nouvelles ne faisant pas partie de la base des 14000 images d'apprentissage. La matrice de confusion obtenue est donnée par le Table VI.4. Sur les 350 zones avec défauts, 12 ne sont pas détectées ce qui correspond à un pourcentage de détection "recall" de $338/350 = 96,5\%$. Par ailleurs, 10 zones sans défauts sont classées par erreur comme des défauts, ce qui correspond à une précision de $338/(338 + 10) = 97,1\%$.

4.3 Validation de l'approche de détection + classification

Comme montré dans la Figure VI.15, pour chaque indication 3D identifiée par le réseau U-Net, le centre de masse sert à découper une zone de (64×64) dans le volume tomographique en niveau de gris, selon 3 plans perpendiculaires. Pour chaque indication on a donc 3 résultats du réseau de classification, et seules les in-



(a)



(b)

Figure VI.20: (a) Exemple de coupe tomographique et zones découpées pour former la base ; (b) Exemples d'images de la base de données de classification : les deux lignes du haut pour les défauts, et les deux lignes du bas pour les zones normales (artefacts, bords de pièce, zones homogènes).

Table VI.4: Matrice de confusion obtenue avec le réseau CT-Casting-Net sur un lot de 700 images nouvelles ne faisant pas partie de l'ensemble d'apprentissage.

	CT-Casting-Net		Metrics
False Alarm	TN= 340	FP= 10	Recall= 96.57%
True Alarm	FN= 12	TP= 338	Precision= 97.12%
	False Alarm	True Alarm	

dications dont les 3 coupes présentent à la fois des zones défectives sont gardées pour la suite du traitement. Cette approche a été validée sur un ensemble de 6 volumes tomographiques n'ayant pas servi à l'apprentissage. Les performances obtenues sont détaillées dans le Table VI.5.

On distingue les performances obtenues en termes d'objets détectés dans le volume 3D pouvant être des défauts ou des fausses alarmes (FA), et les performances obtenues au niveau des voxels, mesurées par le paramètre IoU "intersection over union". Ce dernier paramètre mesure l'intersection entre le volume segmenté obtenu par le traitement automatique, et le volume de référence "vérité terrain", et le divise par l'union des deux. Il est donc lié non seulement à la bonne détection des défauts, mais aussi à la précision de leur dimensionnement. D'un point de vue industriel, les performances de détection au niveau objet ont plus d'importance que l'IoU au niveau voxel, d'où l'intérêt de montrer la probabilité de détection POD, qui représente le taux de défauts détectés dans le volume donné par l'approche sur le nombre total de défauts dans le volume binaire ground-truth.

Table VI.5: Performances obtenues sur 6 volumes tomographiques n'ayant pas servi à l'apprentissage.

Test Volume	**Size**	**Nb Obj Before Classification**	**Nb Obj After Classification**	**FA**	**POD %**	**IoU %**	**Processing Time (s)**
V1	$512 \times 512 \times 100$	504	24	12	100	57.25	17.9
V2	$512 \times 512 \times 100$	676	30	0	100	72.18	18.3
V3	$512 \times 512 \times 96$	587	45	14	100	59.68	16.0
V4	$512 \times 512 \times 100$	220	19	0	100	68.79	12.2
V5	$512 \times 512 \times 86$	167	40	0	100	68.77	11.8
V6	$512 \times 512 \times 100$	354	24	0	94.23	55.28	13.8
			Average:	**4.33**	**99.03**	**63.66**	**15**

4.4 La reconnaissance du type de défauts

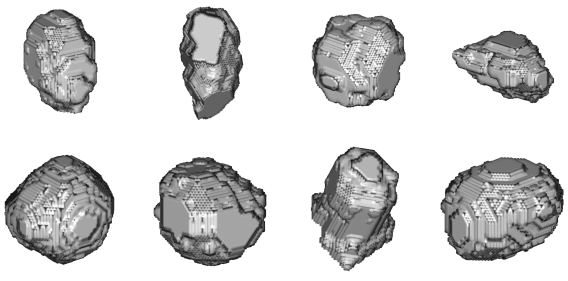
A l'issue de la classification (voir Figure VI.15) chaque indication validée comme défaut est disponible dans sa version binaire en 3D, et la Figure VI.21 montre des exemples de défauts des deux types à différencier. Pour pouvoir classifier les défauts entre porosités ou retassures, un nouveau modèle était entraîné sur une base de

données contenant 263 retassures et 101 porosités pour la fonderie sous pression, et 57 retassures et 41 porosités pour la fonderie gravité, soit 462 défauts au total labélisé manuellement. Dans le métier du CND, ce nombre de défaut peut être considéré élevé, mais en termes d'apprentissage il reste très faible.

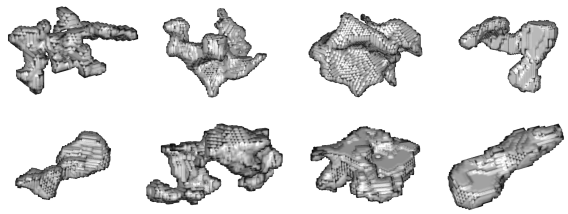
A partir des volumes de discontinuités reconstruites, celles-ci ont été caractérisées par leur taille (volume et surface), ainsi que trois caractéristiques de forme, comme la sphéricité, l'allongement et la compacité. Les porosités sont globalement plutôt sphériques et plutôt compactes alors que les retassures présentent un facteur d'allongement élevé et une forme assez rugueuse, "chaotique". Le diagramme de Figure VI.22 représente les paramètres de forme des deux populations d'indications, selon la procédure de fonderie.

Etant donné que les deux types de procédés de fonderie donnent des défauts ayant les mêmes gammes de caractéristiques, nous avons décidé de rassembler tous les défauts en une seule base pour entraîner un nouveau classifieur à les catégoriser automatiquement. La première approche a été de tester un SVM (Support Vector Machine) entraîné avec 80% de la base totale de défauts, et utilisant les trois caractéristiques de forme en entrée. Le résultat obtenu n'était pas suffisant, avec un taux de détection de 10% des porosités et 28% des retassures. Ensuite, une quatrième caractéristique a été ajouté (le volume du défaut) ce qui a amélioré les performances avec un taux de détection de 40% des porosités et 35% des retassures, mais cela reste insuffisant pour un déploiement industriel.

Par la suite, nous nous sommes tournés vers des méthodes d'apprentissage qui ne demandent que peu de données d'apprentissage, le domaine de "Few-Shot Learning".



(a) Porosities

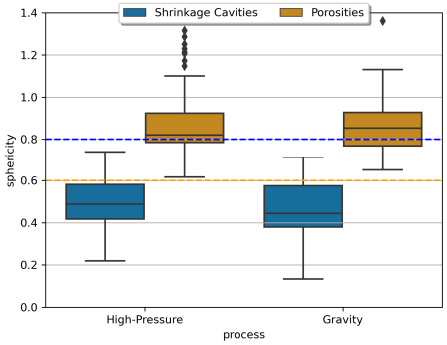


(b) Shrinkage Cavities

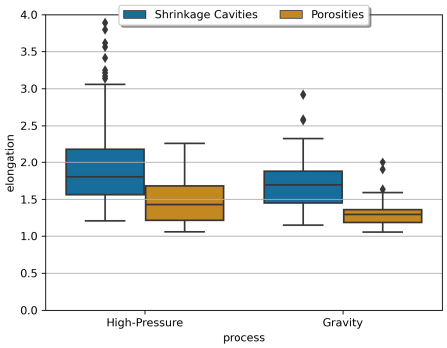
Figure VI.21: Reconstruction en 3 dimensions de défauts de type cavité-retassure et porosités. (sans échelle)

En effet, notre base de défauts contient peu d'instances (462 au total). Par ailleurs, la spécificité de nos défauts est que certains sont très faciles à séparer (ceux dont la sphéricité est respectivement très faible ou très forte) : pour ceux-là un simple seuil sur la sphéricité peut les discriminer. La difficulté de séparation concerne donc seulement sur certains défauts de la base.

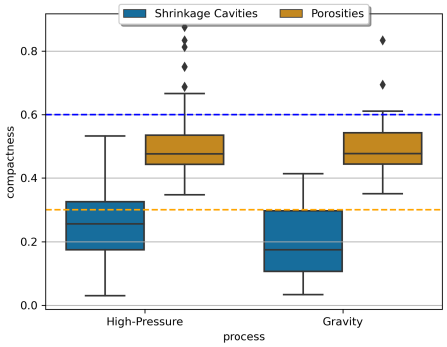
Les réseaux de neurones de type "siamois" (ou SNN Siamese Neural Networks) sont



(a)



(b)



(c)

Figure VI.22: Propriétés géométriques des discontinuités selon le procédé de fonderie –sous pression ou gravité–, avec en bleu les retassures et en orange les porosités : (a) sphéricité, (b) allongement, (c) compacité.

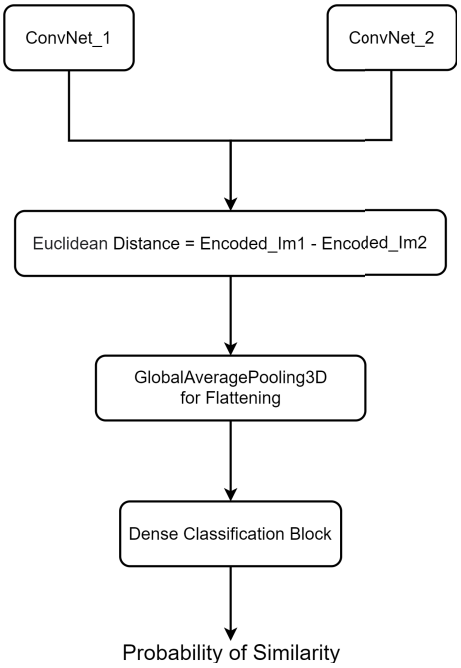


Figure VI.23: Architecture du réseau siamois. Un réseau siamois consiste en deux réseaux convolutifs qui permettent d'obtenir une représentation latente de chaque image d'entrée. La distance euclidienne entre ces deux représentations est alors calculée pour donner la probabilité de similarité grâce à une fonction sigmoïde.

des réseaux dont la finalité est de donner en sortie la probabilité de similarité entre deux ou plusieurs images d'entrée, comme illustré dans la Figure VI.23. Ils sont constitués de deux sous-réseaux de même architecture, paramètres et poids. Lors de l'entraînement, la mise à jour des poids de chaque sous-réseau entraîne la mise à jour de l'autre sous-réseau. L'apprentissage consiste à fournir au SNN un ensemble de paires d'objets similaires (label 1 = paire positive) ou non (label 0 = paire négative). Dans notre cas, les paires sont les volumes 3D représentant les défauts de différents types. La difficulté est que toutes les images d'entrée doivent avoir la même dimension, ce qui n'est pas le cas de nos défauts. Nous avons adopté le prétraitement suivant : (1) étant donné que chaque défaut est représenté par sa boîte englobante, les plus petites dimensions sont "remplies" de valeurs zéro jusqu'à obtenir une boîte cubique; (2) Chaque boîte cubique est zoomée par interpolation spline jusqu'à obtenir une taille de $40 \times 40 \times 40$ (le zoom peut être "in" ou "out" selon la taille initiale des défauts).

La technique d'augmentation de données a été utilisée pour compenser le faible nombre de défauts. Pour préserver la forme des objets, nous avons sélectionné uniquement des transformations de retournement selon un des trois axes et rotation entre 0 et +/-180°. L'entraînement du réseau consiste à fournir en entrée toutes les paires possibles positives (défauts de même nature) ou négatives (défauts de nature différente). Compte tenu de notre nombre initial de $N = 462$ défauts, cela nous donne $N' = (N-1) * N/2 = 106,491$ paires possibles. 80% de ces paires sont

consacrées à l'apprentissage, et 20% sont conservées pour la validation.

Après plusieurs essais d'optimisation en modifiant les hyperparamètres des deux sous-réseaux convolutifs, Une architecture de 8 couches a été adopté. A la sortie de chaque sous réseau, la représentation latente de chaque image d'entrée est obtenue; La distance euclidienne (racine de la somme de la différence carrée des deux vecteurs) est calculée; les matrices de distances sont transformées en un vecteur 1D à l'aide une couche de "pooling"; et finalement un bloc de trois couches "denses" et enfin une fonction sigmoïde pour obtenir une probabilité entre 0 et 1. Un seuil de 0.5 est appliqué pour la décision finale (probabilité de similarité inférieure à 0.5 = paire identique, ou supérieure = paire différente).

Pour chaque itération (epoch) l'entraînement et la validation ont été suivis par le calcul de la fonction de coût mesurée par l'entropie binaire croisée et la précision (Figure VI.24). Comme le nombre de paires est élevé, la taille du lot a été fixée à 100, ce qui signifie que l'optimiseur (SGD) met à jour les poids du réseau après avoir vu seulement 100 paires, sans attendre la fin de l'itération, ce qui permet d'accélérer la convergence. La Figure VI.24 montre que la convergence est obtenue après une vingtaine d'itération, sans manifester de signe de sur ou sous-apprentissage.

La matrice de confusion obtenue sur les paires de la validation (20% du nombre initial de paires de défauts, soit 7302 paires identiques et 5478 paires différentes) est montrée dans le Table VI.6. Le score F1 correspondant est de 99.6% avec seulement

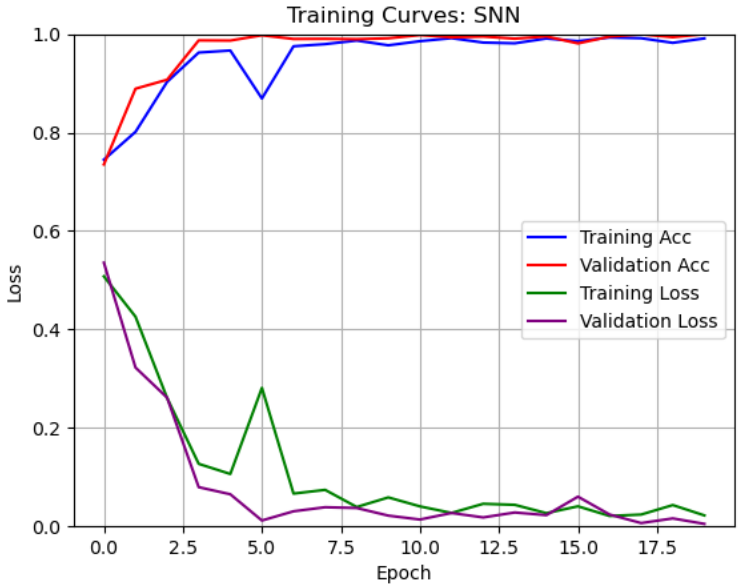


Figure VI.24: Courbes de coût et précision obtenues lors de l'apprentissage du réseau siamois développé pour la reconnaissance de la nature du défaut.

Table VI.6: Matrice de confusion obtenue sur un vecteur test de 5478 paires négatives et 7302 paires positives. Le score F1 est de 99.63%.

SNN Confusion Matrix		
Negative Pairs	5438	40
Positive Pairs	0	7302
	Negative Pairs	Positive Pairs

40 paires différentes classées comme identiques.

Conclusion

Les travaux développés dans cette thèse ont permis de démontrer que les modèles d'intelligence artificielle rendent possible un traitement entièrement automatisé des données tomographiques. Ce traitement intègre la reconnaissance du type de discontinuité de matière, et a été ici évalué et validé sur des alliages d'aluminium produits en fonderie gravité et en injection sous-pression. Les modèles développés au cours de la thèse seront implémentés dans un logiciel industriel. L'enjeu étant à terme par ces solutions de fiabiliser les contrôles en production des produits métalliques par tomographie, et de mettre à la disposition des industriels producteurs et utilisateurs de ces produits contrôlables par tomographie, de nouvelles solutions d'automatisation des contrôles sur la base de solutions qualifiées.

En fonction du type de défauts et à partir de leur taille, de leur localisation par rapport à la surface, ou par rapport à une zone désignée, des critères pertinents pourront être établis par les donneurs d'ordre en fonction du cahier-des-charges appliqué à la pièce.

Bibliography

[1] I. Boromei, L. Ceschini, A. Morri, G. Nicoletto, and E. Riva, "Influence of the solidification microstructure and porosity on the fatigue strength of Al-Si-Mg casting alloys," *Metallurgical Science and Tecnology*, vol. 28, no. 2, 2010.

[2] G. F. Vander Voort, *Metallography, principles and practice*. ASM International, 1999.

[3] J. Schache and O. Brunke, "Developments in 2D and CT X-Ray Inspection of Light Alloy Castings for Production," in *The 14th International Conference of the Slovenian Society for Non-Destructive Testing, September 4-6, 2017, Bernardin, Slovenia. NDT.net Issue: 2018-06*, 2017.

[4] D. M. Stefanescu, "Computer simulation of shrinkage related defects in metal castings – a review," *http://dx.doi.org/10.1179/136404605225023018*, vol. 18, no. 3, pp. 129–143, 2013.

[5] L. De Chiffre, S. Carmignato, J. P. Kruth, R. Schmitt, and A. Weckenmann, "Industrial applications of computed tomography," *CIRP Annals*, vol. 63, pp. 655–677, jan 2014.

[6] F. Zanini and S. Carmignato, "X-Ray Computed Tomography for Dimensional Metrology," pp. 1–48, 2019.

[7] J. P. Kruth, M. Bartscher, S. Carmignato, R. Schmitt, L. De Chiffre, and A. Weckenmann, "Computed tomography for dimensional metrology," *CIRP Annals*, vol. 60, pp. 821–842, jan 2011.

[8] S. Carmignato, "Traceability of dimensional measurements in computed tomography," in *Proc. 8th A.I.Te.M. Conf., Montecatini, Italy*, 2007.

[9] M. Defrise, F. Noo, and H. Kudo, "A solution to the long-object problem in helical cone-beam tomography," *Physics in Medicine \& Biology*, vol. 45, p. 623, mar 2000.

[10] S. Gorham and P. C. Brennan, "Impact of focal spot size on radiologic image quality: A visual grading analysis," *Radiography*, vol. 16, pp. 304–313, nov 2010.

[11] F. R. Verdun, D. Racine, J. G. Ott, M. J. Tapiovaara, P. Toroi, F. O. Bochud, W. J. Veldkamp, A. Schegerer, R. W. Bouwman, I. Hernandez-Giron, N. W.

Marshall, and S. Edyvean, "Image quality in CT: From physical measurements to model observers," *Physica Medica*, vol. 31, pp. 823–843, dec 2015.

[12] Z. Rui, Y. Jili, W. Jinjie, Y. Yang, L. Zhenyu, and Z. Zhenjie, "Study on the inverse square law of X-ray radiation field," *Chinese Journal of Nuclear Science and Engineering*, vol. 37, no. 3, pp. 482–486, 2017.

[13] "EN AW-2014. Aluminium Material Data Sheet EN AW-2014, EN AW-Al Cu4SiMg. 2011,"

[14] T. Zikmund, *Possibilities of state of the art lab-based CT systems for industrial part inspection.* Ceitec, 2018.

[15] D. Mihailidis, "Computed Tomography From Photon Statistics to Modern Cone-Beam CT," *Medical Physics*, vol. 36, no. 8, p. 3858, 2009.

[16] J. D. O'Sullivan, J. Behnsen, T. Starborg, A. S. MacDonald, A. T. Phythian-Adams, K. J. Else, S. M. Cruickshank, and P. J. Withers, "X-ray micro-computed tomography (μCT): an emerging opportunity in parasite imaging.," *Parasitology*, vol. 145, pp. 848–854, nov 2017.

[17] U. Khan, A. U. Yasin, M. Abid, I. S. Awan, and S. A. Khan, "A Methodological Review of 3D Reconstruction Techniques in Tomographic Imaging," *Journal of Medical Systems*, vol. 42, pp. 1–12, oct 2018.

[18] G. Wang, T. H. Lin, P. C. Cheng, and D. M. Shinozaki, "A General Cone-Beam Reconstruction Algorithm," *IEEE Transactions on Medical Imaging*, vol. 12, no. 3, pp. 486–496, 1993.

[19] W. Sun, S. B. Brown, and R. K. Leach, "An overview of industrial X-ray computed tomography.," 2012.

[20] J. F. Barrett and N. Keat, "Artifacts in CT: Recognition and avoidance," *Radiographics*, vol. 24, nov 2004.

[21] J. Kastner and C. Heinzl, "X-Ray Tomography," in *Handbook of Advanced Non-Destructive Evaluation*, pp. 1–72, Cham: Springer International Publishing, 2018.

[22] A. A. Malcolm, T. Liu, I. Kee, B. Ng, W. Y. Teng, T. Tung, P. Wan, and C. J. Kong, "A Large Scale Multiple Source X-ray CT System for Aerospace Applications," pp. 13–15, 2013.

[23] ASTM International, "ASTM E155-20, Standard Reference Radiographs for Inspection of Aluminum and Magnesium Castings, ASTM International, West Conshohocken, PA, 2020, www.astm.org,"

[24] ASTM International, "ASTM E505-15, Standard Reference Radiographs for Inspection of Aluminum and Magnesium Die Castings, ASTM International, West Conshohocken, PA, 2015, www.astm.org,"

[25] ASTM International, "ASTM E2422-17, Standard Digital Reference Images for Inspection of Aluminum Castings, ASTM International, West Conshohocken, PA, 2017, www.astm.org,"

[26] ASTM International, "ASTM E2973-15, Standard Digital Reference Images for Inspection of Aluminum and Magnesium Die Castings, ASTM International, West Conshohocken, PA, 2015, www.astm.org,"

[27] S. Shukla, "Study of Porosity Defect in Aluminum Die Castings and its Evaluation and Control for Automotive Applications," *International Research Journal of Engineering and Technology*, 2020.

[28] S. Gondrom, S. Gondrom, and M. Maisl, "3D reconstructions of micro-systems using x-ray tomographic methods," in *16th World Conference on Nondestructive Testing 2004. CD of proceedings*, (Montreal), p. TS5.3.1, 2004.

[29] D. Mery, · . V. Riffo, U. Zscherpel, G. Mondragón, I. Lillo, I. Zuccar, H. Lobel, and M. Carrasco, "GDXray: The Database of X-ray Images for Nondestructive Testing," *J Nondestruct Eval*, vol. 34, p. 42, 2015.

[30] D. Mery, T. Jaeger, and D. Filbert, "A review of methods for automated recognition of casting defects," tech. rep., 2002.

[31] S. Arita, H. Takimoto, H. Yamauchi, and A. Kanagawa, "Automatic Detection Method for Casting Defects based on Gradient Features," 2014.

[32] B. Wu, J. Zhou, X. Ji, Y. Yin, and X. Shen, "Research on Approaches for Computer Aided Detection of Casting Defects in X-ray Images with Feature Engineering and Machine Learning," *Procedia Manufacturing*, vol. 37, pp. 394–401, jan 2019.

[33] R. B. Tokime, X. Maldague, and L. Perron, "Automatic Defect Detection for X-Ray inspection: Semantic segmentation with deep convolutional network," in *International Symposium on Digital Industrial Radiology and Computed Tomography (DIR 2019)*, (Furth, Germany), 2019.

[34] W. Du, H. Shen, J. Fu, G. Zhang, and Q. He, "Approaches for improvement of the X-ray image defect detection of automobile casting aluminum parts based on deep learning," *NDT & E International*, vol. 107, p. 102144, oct 2019.

[35] D. Mery, "Aluminum Casting Inspection Using Deep Learning: A Method Based on Convolutional Neural Networks," *undefined*, vol. 39, mar 2020.

[36] L. Duan, K. Yang, and L. Ruan, "Research on Automatic Recognition of Casting Defects Based on Deep Learning," *IEEE Access*, vol. 9, pp. 12209–12216, 2021.

[37] A. du Plessis, S. G. le Roux, and A. Guelpa, "Comparison of medical and industrial X-ray computed tomography for non-destructive testing," *Case Studies in Nondestructive Testing and Evaluation*, vol. 6, pp. 17–25, nov 2016.

[38] S. Kumar, S. Rani, and K. R. Laxmi, "Artificial Intelligence and Machine Learning in 2D/3D Medical Image Processing," *Artificial Intelligence and Machine Learning in 2D/3D Medical Image Processing*, dec 2020.

[39] M. Hadwiger, L. Fritz, C. Rezk-Salama, T. Höllt, G. Geier, and T. Pabel, "Interactive volume exploration for feature detection and quantification in industrial CT data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, pp. 1507–1514, nov 2008.

[40] I. Szabo, J. Sun, G. Feng, J. Kanfoud, T. H. Gan, and C. Selcuk, "Automated Defect Recognition as a Critical Element of a Three Dimensional X-ray Computed Tomography Imaging-Based Smart Non-Destructive Testing Technique in Additive Manufacturing of Near Net-Shape Parts," *Applied Sciences 2017, Vol. 7, Page 1156*, vol. 7, p. 1156, nov 2017.

[41] J. Chen, J. Benesty, Y. Huang, and S. Doclo, "New insights into the noise reduction Wiener filter," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, no. 4, pp. 1218–1233, 2006.

[42] S. Gondrom-Linke, "Physics and Special Technology of Very Fast or Even Inline Industrial 3D-CT," in *15th Asia Pacific Conference for Non-Destructive Testing*, (Singapore), NDT.net Issue - 2018-03, 2017.

[43] M. Rieter, C. Gusenbauer, R. Huemer, and J. Kastner, "At-line X-ray computed tomography of serial parts optimized by numerical simulations," in *International Symposium on Digital Industrial Radiology and Computed Tomograph (DIR))*, (Fürth, Germany), NDT.net Issue - 2019-11, 2019.

[44] F. Zhao, P. R. Mendonça, J. Yu, and R. Kaucic, "Learning-based automatic defect recognition with computed tomographic imaging," *2013 IEEE International Conference on Image Processing*, pp. 2762–2766, 2013.

[45] S. E. Grigorescu, N. Petkov, and P. Kruizinga, "Comparison of texture features based on Gabor filters," *Ieee transactions on image processing*, vol. 11, pp. 1160–1167, oct 2002.

[46] P. Vincent, H. Larochelle, Y. Bengio, and P. A. Manzagol, "Extracting and composing robust features with denoising autoencoders," *Proceedings of the 25th International Conference on Machine Learning*, pp. 1096–1103, 2008.

[47] P. R. Mendonça, R. Bhotika, S. A. Sirohey, W. D. Turner, J. V. Miller, and R. S. Avila, "Model-based analysis of local shape for lesion detection in CT scans," *Medical image computing and computer-assisted intervention : MICCAI ... International Conference on Medical Image Computing and Computer-Assisted Intervention*, vol. 8, no. Pt 1, pp. 688–695, 2005.

[48] A. Osman, V. Kaftandjian, and U. Hassler, "Automatic classification of 3D segmented CT data using data fusion and support vector machine," in *Tenth*

International Conference on Quality Control by Artificial Vision, vol. 8000, p. 80000F, SPIE, jun 2011.

[49] A. Osman, V. Kaftandjian, and U. Hassler, "Improvement of X-ray castings inspection reliability by using Dempster–Shafer data fusion theory," *Pattern Recognition Letters*, vol. 32, pp. 168–180, jan 2011.

[50] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, pp. 861–874, jun 2006.

[51] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep Learning for Computer Vision: A Brief Review," *Computational Intelligence and Neuroscience*, vol. 2018, 2018.

[52] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-Decem, pp. 770–778, dec 2015.

[53] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 07-12-June, pp. 1–9, oct 2015.

[54] K. He, X. Zhang, S. Ren, and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," in *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1026–1034, IEEE, dec 2015.

[55] J. Maier, S. Sawall, M. Kachelriess, and Y. Berker, "Deep scatter estimation (DSE): feasibility of using a deep convolutional neural network for real-time x-ray scatter prediction in cone-beam CT," *SPIE*, vol. 10573, p. 105731L, mar 2018.

[56] M. L. Giger, "Machine Learning in Medical Imaging," *Journal of the American College of Radiology*, vol. 15, pp. 512–520, mar 2018.

[57] T. Higaki, Y. Nakamura, F. Tatsugami, T. Nakaura, and K. Awai, "Improvement of image quality at CT and MRI using deep learning," *Japanese Journal of Radiology 2018 37:1*, vol. 37, pp. 73–80, nov 2018.

[58] K. Liang, L. Zhang, H. Yang, Y. Yang, Z. Chen, and Y. Xing, "Metal artifact reduction for practical dental computed tomography by improving interpolation-based reconstruction with deep learning," *Medical Physics*, vol. 46, pp. e823–e834, dec 2019.

[59] M. Schlotterbeck, L. Schulte, W. Alkhaldi, M. Krenkel, E. Toeppe, S. Tschechne, and C. Wojek, "Automated defect detection for fast evaluation of real inline CT scans," *https://doi.org/10.1080/10589759.2020.1785446*, vol. 35, pp. 266–275, jul 2020.

[60] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9901 LNCS, pp. 424–432, jun 2016.

[61] G. Mooij, I. Bagulho, and H. Huisman, "Automatic segmentation of prostate zones," 2018.

[62] P. Fuchs, T. Kröger, and C. S. Garbe, "Defect detection in CT scans of cast aluminum parts: A machine vision perspective," *Neurocomputing*, vol. 453, pp. 85–96, sep 2021.

[63] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," may 2015.

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

[65] M. D. Zeiler and R. Fergus, "Visualizing and Understanding Convolutional Networks," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8689 LNCS, pp. 818–833, nov 2013.

[66] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," mar 2016.

[67] T. Rashid, *Make Your First GAN With PyTorch*. Independently Published, 2020.

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

[69] F. and others Chollet, "Keras," 2015.

[70] R. Gordon, "Three-Dimensional Reconstruction from Projections: A Review of Algorithms," *International Review of Cytology*, vol. 38, pp. 111–151, jan 1974.

[71] Rahul T Patil, Veena S Metri, and Shubhangi S Tambore, "Causes of Casting Defects with Remedies," *International Journal of Engineering Research and*, vol. V4, nov 2015.

[72] J. Hsieh, *Computed Tomography: Principles, Design, Artifacts, and Recent Advances*. SPIE PRESS, 2015.

[73] C. W. Kang, M. B. Ramzan, B. Sarkar, and M. Imran, "Effect of inspection performance in smart manufacturing system based on human quality control system," *The International Journal of Advanced Manufacturing Technology 2017 94:9*, vol. 94, pp. 4351–4364, oct 2017.

[74] T. S. Newman, "Survey of automated visual inspection," *Computer Vision and Image Understanding*, vol. 61, no. 2, pp. 231–262, 1995.

[75] C. Stolojescu-Crişan and t. Holban, "A Comparison of X-ray image segmentation techniques," *Advances in Electrical and Computer Engineering*, vol. 13, no. 3, pp. 85–92, 2013.

[76] R. C. Gonzalez and R. E. Woods, *Digital image processing, 4th Edition*. Pearson Education, 2018.

[77] A. Alazzawi, "EDGE DETECTION-APPLICATION OF (FIRST AND SECOND) ORDER DERIVATIVE IN IMAGE PROCESSING," *Diyala Journal of Engineering Sciences*, vol. 8, pp. 430–440, dec 2015.

[78] H. P. Narkhede, "Review of Image Segmentation Techniques," *International Journal of Science and Modern Engineering (IJISME)*, no. 8, pp. 2319–6386, 2013.

[79] C. Glasbey, "An Analysis of Histogram-Based Thresholding Algorithms," *CVGIP: Graphical Models and Image Processing*, vol. 55, pp. 532–537, nov 1993.

[80] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," *IEEE Trans Syst Man Cybern*, vol. SMC-9, no. 1, pp. 62–66, 1979.

[81] P. Getreuer, "A Survey of Gaussian Convolution Algorithms," *Image Processing On Line*, vol. 3, pp. 286–310, dec 2013.

[82] N. Senthilkumaran and R. Rajesh, "Edge Detection Techniques for Image Segmentation-A Survey of Soft Computing Approaches," *INFORMATION PAPER International Journal of Recent Trends in Engineering*, vol. 1, no. 2, p. 250, 2009.

[83] Vipin Tyagi, *Understanding Digital Image Processing*. CRC PRESS, 2018.

[84] A. Alam, Zain-Ul-Abdin, and B. Svensson, "Parallelization of the estimation algorithm of the 3D structure tensor," *2012 International Conference on Reconfigurable Computing and FPGAs, ReConFig 2012*, 2012.

[85] Z. Zhou, M. M. Rahman Siddiquee, N. Tajbakhsh, and J. Liang, "Unet++: A nested u-net architecture for medical image segmentation," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11045 LNCS, pp. 3–11, Springer Verlag, 2018.

[86] A. N. J. Raj, H. Zhu, A. Khan, Z. Zhuang, Z. Yang, G. V. Mahesh, and G. Karthik, "ADID-UNET—a segmentation model for COVID-19 infection from lung CT scans," *PeerJ Computer Science*, vol. 7, pp. 1–34, jan 2021.

[87] T. M. Mitchell, *Machine Learning*. New York: McGraw-Hill, 1997.

[88] Sorensen and T. A., "A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on Danish commons," *Biol. Skar.*, vol. 5, pp. 1–34, 1948.

[89] S. Jadon, "A survey of loss functions for semantic segmentation," in *2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology, CIBCB 2020*, 2020.

[90] C. H. Sudre, W. Li, T. Vercauteren, S. Ourselin, and M. J. Cardoso, "Generalised Dice overlap as a deep learning loss function for highly unbalanced segmentations," 2017.

[91] V. P. Plagianakos, G. D. Magoulas, and M. N. Vrahatis, "Learning Rate Adaptation in Stochastic Gradient Descent," pp. 433–444, 2001.

[92] F. Sherwani, B. S. Ibrahim, and M. M. Asad, "Hybridized classification algorithms for data classification applications: A review," *Egyptian Informatics Journal*, vol. 22, pp. 185–192, jul 2021.

[93] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, dec 2014.

[94] V. Nekrasov, J. Ju, and J. Choi, "Global deconvolutional networks for semantic segmentation," in *British Machine Vision Conference 2016, BMVC 2016*, vol. 2016-Septe, pp. 124.1–124.14, 2016.

[95] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[96] J. Lee, B. K. Iwana, S. Ide, and S. Uchida, "Globally Optimal Object Tracking with Fully Convolutional Networks," dec 2016.

[97] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," tech. rep.

[98] F. Milletari, N. Navab, and S.-A. Ahmadi, "V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation," *Proceedings - 2016 4th International Conference on 3D Vision, 3DV 2016*, pp. 565–571, jun 2016.

[99] Y. Xia, E. K. Fishman Johns Hopkins Medicine, Q. Yu, L. Xie, E. K. Fishman, and A. L. Yuille, "Thickened 2D Networks for 3D Medical Image Segmentation," tech. rep., 2019.

[100] F. Isensee, P. Jaeger, P. M. Full, I. Wolf, S. Engelhardt, and K. H. Maier-Hein, "Automatic Cardiac Disease Assessment on cine-MRI via Time-Series Segmentation and Domain Specific Features," vol. 10663, jul 2017.

[101] D. C. Cireşan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, "Deep neural networks segment neuronal membranes in electron microscopy images," in *Advances in Neural Information Processing Systems*, vol. 4, pp. 2843–2851, 2012.

[102] O. Rukundo, "Effects of Image Size on Deep Learning," jan 2021.

[103] Y. Moe, A. Groendahl, O. Tomic, E. Dale, E. Malinen, and C. Futsaether, "Deep learning-based auto-delineation of gross tumour volumes and involved nodes in PET/CT images of head and neck cancer patients," *European journal of nuclear medicine and molecular imaging*, vol. 48, pp. 2782–2792, aug 2021.

[104] Z. Tian, J. Chen, and Q. Niu, "DropFilter: Dropout for Convolutions," oct 2018.

[105] N. Srivastava, G. Hinton, A. Krizhevsky, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," Tech. Rep. 56, 2014.

[106] L. Perez and J. Wang, "The Effectiveness of Data Augmentation in Image Classification using Deep Learning," dec 2017.

[107] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, pp. 84–90, jun 2017.

[108] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," *32nd International Conference on Machine Learning, ICML 2015*, vol. 1, pp. 448–456, feb 2015.

[109] D. M. W. Powers, "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation," oct 2011.

[110] P. Harrington, *Machine Learning in Action*, vol. 37. 2012.

[111] S. Marsland, *Machine learning : an algorithmic perspective*. CRC Press, 2009.

[112] S. Ruder, "An overview of gradient descent optimization algorithms," sep 2016.

[113] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *MM 2014 - Proceedings of the 2014 ACM Conference on Multimedia*, pp. 675–678, Association for Computing Machinery, Inc, nov 2014.

[114] N. Qian, "On the momentum term in gradient descent learning algorithms," *Neural Networks*, vol. 12, pp. 145–151, jan 1999.

[115] J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," pp. 248–255, mar 2010.

[116] H.-C. Shin, H. R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R. M. Summers, "Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning," *IEEE Transactions on Medical Imaging*, vol. 35, pp. 1285–1298, feb 2016.

[117] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 2015.

[118] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization," *International Journal of Computer Vision*, vol. 128, pp. 336–359, oct 2016.

[119] D. A. Turner, "An intuitive approach to receiver operating characteristic curve analysis.," *Journal of nuclear medicine : official publication, Society of Nuclear Medicine*, vol. 19, pp. 213–220, feb 1978.

[120] L. A. Jeni, J. F. Cohn, and F. De La Torre, "Facing imbalanced data - Recommendations for the use of performance metrics," *Proceedings - 2013 Humaine Association Conference on Affective Computing and Intelligent Interaction, ACII 2013*, pp. 245–251, 2013.

[121] J. Davis and M. Goadrich, "The relationship between precision-recall and ROC curves," *ACM International Conference Proceeding Series*, vol. 148, pp. 233–240, 2006.

[122] T. Saito and M. Rehmsmeier, "The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets," *PLOS ONE*, vol. 10, p. e0118432, mar 2015.

[123] Y. J. Zhang, "A survey on evaluation methods for image segmentation," *Pattern Recognition*, vol. 29, pp. 1335–1346, aug 1996.

[124] M. Everingham, S. M. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The Pascal Visual Object Classes Challenge: A Retrospective," *International Journal of Computer Vision*, vol. 1, pp. 98–136, jan 2015.

[125] J. Yu, J. Xu, Y. Chen, W. Li, Q. Wang, B. I. Yoo, and J.-J. Han, "Learning Generalized Intersection Over Union for Dense Pixelwise Prediction," jul 2021.

[126] I. Virkkunen, T. Koskinen, S. Papula, T. Sarikka, and H. Hänninen, "Comparison of â Versus a and Hit/Miss POD-Estimation Methods: A European Viewpoint," *Journal of Nondestructive Evaluation*, vol. 38, pp. 1–13, dec 2019.

[127] J. H. Kurz, A. Jüngert, S. Dugan, and G. Dobmann, "Probability of Detection (POD) Determination Using Ultrasound Phased Array for Considering NDT in Probabilistic Damage Assessments," in *World Conference on Nondestructive Testing (WCNDT)*, 2012.

[128] G. Dour, *Aide-mémoire - Fonderie - Livre Mécanique et matériaux*. Dunod, 2e ed., 2016.

[129] J. Lindblad and I. Nyström, "Surface area estimation of digitized 3D objects using local computations," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 2301, pp. 267–278, 2002.

[130] W. Lorensen and H. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," *ACM SIGGRAPH Computer Graphics*, vol. 21, pp. 163–169, aug 1987.

[131] H. Wadell, "Volume, Shape, and Roundness of Rock Particles," *The Journal of Geology*, vol. 40, pp. 443–451, jul 1932.

[132] W. Pabst and E. Gregorová, "Characterization of particles and particle systems," in *ICT*, (Prague), 2007.

[133] J. Cervantes, F. Garcia-Lamont, L. Rodríguez-Mazahua, and A. Lopez, "A comprehensive survey on support vector machine classification: Applications, challenges and trends," *Neurocomputing*, vol. 408, pp. 189–215, sep 2020.

[134] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Generalizing from a Few Examples: A Survey on Few-Shot Learning," *ACM Computing Surveys*, vol. 53, apr 2019.

[135] J. Bromley, I. Guyon, Y. Lecun, E. Sickinger, R. Shah, A. Bell, and L. Holmdel, "Signature Verification using a "Siamese" Time Delay Neural Network," *Advances in Neural Information Processing Systems*, vol. 6, 1993.

[136] Y. Sugawara, S. Shiota, and H. Kiya, "Checkerboard artifacts free convolutional neural networks," *APSIPA Transactions on Signal and Information Processing*, vol. 8, pp. 1–9, 2019.

[137] C. A. Hall and W. W. Meyer, "Optimal error bounds for cubic spline interpolation," *Journal of Approximation Theory*, vol. 16, pp. 105–122, feb 1976.

FOLIO ADMINISTRATIF

THESE DE L'UNIVERSITE DE LYON OPEREE AU SEIN DE L'INSA LYON

NOM : DAKAK **DATE DE SOUTENANCE** : 09/03/2022

PRÉNOM : Abdel Rahman

TITRE : Automatic Defect Detection in Industrial CT Volumes of Castings /
Détection Automatique des Défauts dans des Volumes Tomographiques des Pièces de Fonderie

NATURE : Doctorat **NUMÉRO D'ORDRE** : 2022LYSEI013

ÉCOLE DOCTORALE : Electronique, Electrotechnique, automatique (EEA)

SPÉCIALITÉ : Traitement du Signal et de l'Image

RESUMÉ :

Industrial X-ray computed tomography (CT) has proven its value as a non-destructive method for inspecting light metal castings. The CT volume generated enables the internal and external geometry of the specimen to be measured, casting defects to be localized and their statistical properties to be investigated. On the other hand, CT volumes are very prone to artifacts that can be mistaken for defects by conventional segmentation algorithms. Based on CT data of aluminium alloy castings provided by industrial partners, we have developed an automatic approach to analyze discontinuities inside CT volumes based on a three-step pipeline: (1) 2D segmentation of CT slices with automatic deep segmentation to detect suspicious greyscale discontinuities; (2) classification of these discontinuities into true alarms (defects) or false alarms (artifacts and noise), using a trained convolutional neural network classifier; (3) localization of the validated defects in 3D to investigate their statistical properties such as sphericity, elongation and compactness. Based on this, the validated 3D defects are then classified into porosities or shrinkage cavities using an SVM classifier and a siamese neural network. The choice of each model and the training results are presented and discussed, as well as the efficiency of the approach as an automatic defect detection algorithm for industrial CT volumes.

MOTS-CLÉS : Casting, CT, X-Ray, Defect Detection, Machine Learning, Deep Learning, Few-Shot Learning

LABORATOIRE DE RECHERCHE : Laboratoire Vibrations et Acoustique (LVA)

DIRECTEUR DE THÈSE : KAFTANDJIAN Valérie

PRÉSIDENT DE JURY : M. OSMAN Ahmad

COMPOSITION DU JURY : OSMAN Ahmad (président), ROMBAUT Michèle (rapporteur de thèse), RUAN Su (rapporteur de thèse), MERY Domingo (examinateur), DUFFNER Stefan (examinateur), KAFTANDJIAN Valérie (directrice de thèse), DUVAUCHELLE Philippe (co-encadrant).