Application of machine learning to ultrasonic nondestructive evaluation

Richard Pyle

Department of Mechanical Engineering,
UNIVERSITY OF BRISTOL

A dissertation submitted to the University of Bristol in accordance with the requirements for award of the degree of ENGINEERING DOCTORATE in the Faculty of Engineering.

January 2023

Word count: 33947

Abstract

Machine learning (ML) techniques have the potential to provide automated data analysis for nondestructive evaluation (NDE) applications with human-level accuracy. This is of great value as the data gathered by NDE inspection is, increasingly, large and complex, making manual data analysis expensive, slow and sensitive to operator variability. However, there are three major barriers to the application of ML models to industrial NDE: sourcing useful training data, choosing informative features, and building trust in the model's predictions. This thesis investigates how these barriers can be overcome by deep learning with simulated training sets, domain adaptation, uncertainty quantification, and improved interpretability. An example NDE use case is considered: defect sizing for ultrasonic inline pipe inspection. An inspection configuration is devised to closely match the conditions found in inline inspection of oil pipelines, resulting in ultrasonic plane wave images of surface breaking defects. These ultrasonic images are used as input to ML models to predict the size of the defects.

A convolutional neural network (CNN) is trained to size defects, using a simulated data set, and applied to previously unseen experimental data. As the CNN takes ultrasonic images as input there is no need to manually select informative features. The CNN is compared to a traditional NDE sizing method, 6 dB drop, and demonstrates significantly better sizing accuracy. Further sizing accuracy improvements are achieved through the inclusion of a small amount of experimental data in the training procedure. This additional training data is included with the aim of reducing the effect that differences in simulated and experimental data have on sizing performance. An adversarial-based domain adaptation technique is found to be the optimal way to leverage small amounts of experimental training data.

Building trust in the prediction of ML models is essential for qualifying them for use in NDE industry. Uncertainty quantification (UQ) is a significant part of this, as it is essential to the decision making for any automated data analysis. This thesis investigates two modern UQ techniques, finding deep ensembles to be an effective way to quantify the uncertainty of sizing predictions. Further trust is built by improving the interpretability and explainability of ML for NDE. This is achieved with a novel dimensionality reduction method: Gaussian feature approximation (GFA). GFA involves fitting a 2D gaussian to an ultrasonic image and storing the resulting seven parameters that describe it. These parameters can be used as input features for a ML model. As individual GFA features are meaningful to a human (unlike pixel intensities) the resulting model is implicitly more interpretable than one trained on raw images. Shapley additive explanations are used to indicate how each feature contributes to a crack size prediction. The results presented in this thesis indicate that it is possible to use ML to achieve automated data analysis for real-world industrial NDE applications.

Dedication and acknowledgements

During this project I have received outstanding support and encouragement. Firstly, I would like to thank my academic supervisors: Professor Paul Wilcox, Dr. Rhodri Bevan and Dr. Robert Hughes, without whose guidance and expertise, none of this work would have been possible.

This work was in part sponsored by the Engineering and Physical Sciences Research Council (EPSRC) under Grant EP/L015587/1, via the Research Centre for Non-Destructive Evaluation (RCNDE). I would like to thank both the staff and students within the RCNDE for fostering such a supportive organization.

Baker Hughes, Cramlington, U.K also sponsored part of this work. I thank everyone in the team for their help on this project. In particular, I would like to thank Dr. Amine Ait Si Ali, Dr. Giovanni Cannì and Dr. Martin Spies for their supervision and guidance.

I am grateful to all of my past and present colleagues in the Ultrasonics and Nondestructive Testing group in Bristol for their friendship and advice. Special thanks go to Dr. Rosen Rachev, Dr. Nicolas Budyn and Dr. Jessica McKee for helping me settle in and find my feet at the start of this project.

Finally, I am forever indebted to my partner Vic, my parents Roland and Jane, and all the friends and family who have transformed the long and lonely process of doctoral studies into some of the best years of my life.

Author’s declaration

I declare that the work in this dissertation was carried out in accordance with the requirements of the University's *Regulations and Code of Practice for Research Degree Programmes* and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, the work is the candidate's own work. Work done in collaboration with, or with the assistance of, others, is indicated as such. Any views expressed in the dissertation are those of the author.

SIGNED: ..... DATE: .....

Nomenclature

Subscripts are used in this thesis to indicate sub-groups. For example, c_S and c_L are shear and longitudinal sound speeds respectively. Definition of symbol subscripts are not included in this section.

Symbol	Meaning	Section Introduced
ϕ	Plane wave angle in fluid	2.1
ψ	Plane wave angle in pipe material	2.1
ς	Standoff	2.1
Γ	Pipe material thickness	2.1
P	Horizontal distance between defect and array centre	2.1
L	Defect length	2.1
θ	Defect angle	2.1
D	Vertical extent of defect	2.1
X	Horizontal axis, origin at array centre	2.2
Z	Vertical axis, origin at array centre	2.2
$I(x,z)$	PWI image intensity at position x,z	2.2
γ	Modality of return ray path	2.2
$h(t)$	Complex, filtered A-scan at time t	2.2
c	Sound speed	2.2
λ	Wavelength	2.2
n	Number of transducers in the array	2.3
β	Sound speed multiplier	3.3
$\mathcal{L}$	Loss function	4.5
N	Number experimental data in training set	4.5
M	Number simulated data in training set	4.5
κ	Dimensionality of feature space	4.5
δ	Tolerance on label difference	4.5
y	Data label	4.5
$\hat{y}$	Network output	4.5
α	Scaling factor for loss function	4.5
T	Input and output training data	5.1
m	Number of networks in ensemble	5.5
μ	Mean of ensemble predictions	5.5
σ	Standard deviation	5.5
K	Lipschitz constants	5.5
η	Spectral norm	5.5
p	Dropout probability	5.5
Δ	Mean absolute change in uncertainty	5.6
u	Uncertainty	5.6
R	Correlation coefficient	5.6

$A, x_0, z_0, \sigma_X, \sigma_Z, \Theta, B$	Parameters of a 2D gaussian: amplitude, X and Z position, X and Z standard deviation, angle and offset respectively	6.3
φ	Shapley additive explanation (SHAP) values	6.3
z'	Simplified, binary inputs	6.3
k	Weighting kernel	6.3
s	Number of ones in z'	6.3
a	Number of binary mask iterations	6.3
b	Number of samples of training data	6.3
Λ	Mean absolute difference in SHAP values	6.3
F	Mean absolute SHAP value	6.4

Author's publications

R. J. Pyle, R. L. T. Bevan, R. R. Hughes, R. K. Rachev, A. Ait Si Ali, and P. D. Wilcox, "Deep Learning for Ultrasonic Crack Characterization in NDE," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 68, no. 5, pp. 1854–1865, 2020, doi: 10.1109/TUFFC.2020.3045847.

R. J. Pyle, R. L. T. Bevan, R. R. Hughes, A. A. S. Ali, and P. D. Wilcox, "Domain Adapted Deep-Learning for Improved Ultrasonic Crack Characterization Using Limited Experimental Data," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 69, no. 4, pp. 1485–1496, 2022, doi: 10.1109/TUFFC.2022.3151397.

R. J. Pyle, R. R. Hughes, A. A. S. Ali, and P. D. Wilcox, "Uncertainty Quantification for Deep Learning in Ultrasonic Crack Characterization," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 69, no. 7, pp. 2339–2351, 2022, doi: 10.1109/TUFFC.2022.3176926.

R. J. Pyle, R. R. Hughes, and P. D. Wilcox, "Interpretable & Explainable Machine Learning for Ultrasonic Defect Sizing," *in review for IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*

Contents

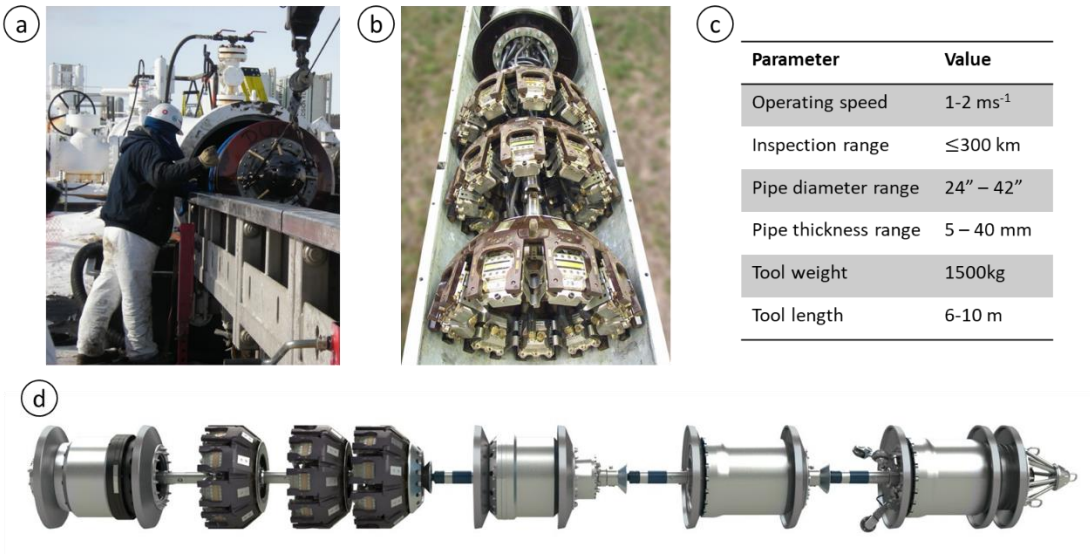
Chapter 1. Introduction

This thesis is concerned with improving the automated data analysis of ultrasonic nondestructive evaluation data (NDE) through the application of machine learning (ML) techniques. This introductory chapter describes the industrial application this work is applied to and background information for ML in NDE.

1.1. Ultrasonic pipeline inspection

NDE techniques aim to evaluate the health of a component without damaging it. These techniques can be applied both during component manufacture and throughout its in-service life. The basic principle of most NDE techniques involves the application of a stimulus to the component (such as ultrasound, x-ray or eddy currents), the recording of its response, and the analysis of that response to infer integrity. A large-scale ongoing industrial application of NDE is the inspection of oil and gas pipelines using tools which travel in the flow of product, assessing the integrity of the surrounding pipe. This task is commonly termed 'inline pipe inspection'. The sizing of defects from ultrasonic inline pipe inspection data is used as an example industrial application for research presented in this thesis.

NDE is an important part of the oil and gas industry due to its scale and the damage done when defects are not found soon enough. The worldwide oil and gas pipeline network is over 2 million km in length, with many individual pipelines running over hundreds of kilometres, and a few over 3000 km [1]. Environmental factors and in-service stresses can cause corrosion and cracking, which without detection and repair, can cause spills. For example, in 2020 the Colonial pipeline oil spill, caused by a crack in the pipewall, lead to at least 1.4 million gallons of spilt oil, major damage to the surrounding environment, and an ongoing clean-up that cost more than $55 million [2].



Parameter	Value
Operating speed	1-2 ms^{-1}
Inspection range	≤300 km
Pipe diameter range	24" – 42"
Pipe thickness range	5 – 40 mm
Tool weight	1500kg
Tool length	6-10 m

Fig. 1. Pictures of an UltraScan™ Duo pig a) being launched and b) phased array arrangement, c) its key specifications and d) a computer-aided design render of the full tool.

Inline inspection is a major part of avoiding oil and gas leaks. The oldest inline inspection technique, used since the 1960s [3], is magnetic flux leakage (MFL), in which a strong magnetic field is applied to a ferromagnetic material, and changes in the field can be monitored to detect discontinuities. In more recent years, ultrasound has seen increasing use in inline pipe inspection. While the requirement for couplant limits its application to pipelines carrying fluid product, such as oil, ultrasound's sensitivity to a broad range of defects, and ability to accurately size defects makes it extremely useful [4], [5]. For larger pipelines, inspection is carried out by a cylindrical device, commonly known as a 'pig', which is carried along in the flow of product, while inspecting the surrounding pipe. The specific tool considered in this thesis, as pictured in Fig. 1, is the UltraScan™ Duo (Baker Hughes (formerly PII Pipeline Solutions), Stutensee, Germany). Using this device typically requires inspection every 2 mm, and to keep oil flow rate high the pig must travel at $\sim 2\,\mathrm{ms}^{-1}$ [5]. This high measurement rate, and the computational limitations of the device, severely limit the number of measurements that can be taken by each array, at each axial location. To maximise the region inspected, a circumferential ring of arrays is mounted on the pig. Each array fires 3 plane waves, at different angles, at every axial location, with all transducers used individually on reception. Despite only firing a handful of plane waves, the rate at which it occurs along the pipe means this quickly generates a vast amount of data. Each plane wave fired produces ~ 140 kB of relevant time domain data, meaning that firing 3 plane waves produces ~ 270 MBs^{-1} from each array. Onboard detection procedures mean that data relating to all 'clean' pipe need not be stored, but data storage size and computational speed limitations are still significant issues. Historically, this has motivated discarding most of the received signals, keeping only one peak amplitude per plane wave, upon detecting a defect. This allows for basic amplitude-based sizing only [5]–[7]. This is usually achieved by building an empirical, nonlinear relation between amplitude and size, using a reference data set, gathered from samples manufactured to mimic real defects. This approach struggles to accurately size angled defects as their amplitude response is affected significantly even by small changes in angle [8]. However, through advances in compression software, data storage, and computational hardware, it is becoming increasingly possible to store full time-traces. This enables offline (i.e., after the pig has been retrieved from the pipe) plane wave imaging (PWI) [9], [10] and the use of image-based sizing algorithms. These can be based on physics (e.g., 6 dB drop [11]) or learnt from data.

1.2. Machine learning for NDE

NDE data analysis has traditionally been achieved by a skilled operator. As these data are increasingly becoming very high-dimensional and most inspections must be carried out many times, manual data interpretation becomes slow, expensive, and prone to human error. This can be accelerated by multiple operators working in parallel, but the results then become even more inconsistent. The cost, complexity, speed, and inconsistency of human NDE data analysis motivates the use of automated methods.

There are many analytical automated NDE data analysis methods in use that are based on physics, 'rules of thumb', or thresholds set by experimental testing. For example, to size defects from images, the 6 dB drop method is often used. The 6 dB drop method estimates the extent of a defect as the size of the smallest box that can enclose the top -6 dB of the image. Traditional approaches such as this one function well when there are relatively simple relationships between the data and the quantity to be estimated. However, much of the information in NDE data is too complex to be used by these simple methods, so is discarded. However, the analysis of even very complex, high dimensional NDE data, is essentially a pattern recognition task, so ML is well suited. Also, as the field starts to make use of cyber-physical systems and internet of things (a revolution termed 'NDE 4.0' [12]), ML may well become the only viable route for processing the volume of data produced [13]–[15].

ML refers to a broad range of techniques which aim to learn functions from data. It has repeatedly been shown to produce human-level data interpretation performance in NDE [16]–[24] as well as in related fields such as computer vision [25] and medical imaging [26], [27]. The ML literature relevant to the research in this thesis is reviewed within each of the main chapters (Chapter 3-6). The subset of ML relevant to this thesis is 'supervised' learning. The term 'supervised' refers to learning from labelled data. In this application this means that all training data is associated with defects of known size. The predictive model to be learnt is one that can take a new set of ultrasonic images, from a previously unseen defect, and predict its size. Note that to avoid confusion, the term *model* is used exclusively in this thesis to describe any algorithm learnt from data, for example, a neural network, while physics-based approaches to approximating real data are described as *simulations*.

As well as categorising ML as supervised or unsupervised, another distinction is 'deep' or 'shallow' learning. The exact definition of these terms is not always agreed upon. The definition used in this thesis is these terms refer to whether features are hand selected: shallow learning gives a prediction based on a subset of features, chosen by a human, followed by statistical analysis, whereas deep learning is an end-to-end method that calculates the desired result directly from the raw data. Shallow learning in NDE dates back to at least 1991 with the use of decision trees to detect defects based on the loss in amplitude between ultrasonic wall reflections [28], and has continued to be an active area of research [29]. For example, neural networks have been applied to traditional features of ultrasonic measurements to estimate material properties [17], [18] and classify defects [16], [22], support vector machines used to size cracks from eddy current field peaks [30] and random forest used to detect defects from features of fluorescent penetrant inspection images [24]. The bulk of NDE research has focused on shallow learning as the reduction in input data dimensionality leads to a lower requirement in model complexity. This in turn reduces the size of training data required, which is an attractive property in a field such as NDE, where training data is scarce. However, the success of shallow learning is heavily reliant on selecting the correct features and a lot of the information in the raw data is discarded.

Deep learning can make use of all information in the raw data so has the potential to offer more accurate results [19], [20]. In medical imaging the shift to deep learning is well underway [27], but application to NDE has been hindered by the cost of generating experimental training sets and the difficulty of qualifying uninterpretable, 'black box' algorithms. Data augmentation (i.e., cropping, translating, zooming etc. in the context of photographic image analysis) can be used to expand training set size but care must be taken to ensure these methods create realistic examples. In some NDE applications, by good fortune or large expense, a large enough number of defects are available to form a training set for deep learning [31], [32]. Also, if the scope and parameter space of the ML task can be heavily restricted, a smaller training set can be used [33]. However, neither of these approaches supply a solid, general-purpose solution to creating large NDE training sets, or answer the worries surrounding qualifying the use of 'black box' algorithms.

1.3. Objectives and outline

Despite the high number of publications using ML for NDE data analysis in recent years there are very few industrial implementations in use today. As described in [29] there are three main barriers to the application of ML to NDE industry: feature engineering, the black box problem, and shortage of data. Feature engineering is the task of distilling high dimensional data (e.g., time series or images) into a form that is easier to analyse. This usually involves dimensionality reduction, and to create a useful model, the features must retain the information that is informative to the task at hand. As the interaction between defects and NDE data can be complex, it is not always captured well by simple, traditional features. The black box problem refers to the fact that it is usually difficult to provide human-interpretable explanations for why a ML model makes a certain prediction for a given input. This makes it difficult to prove that the model will always operate as expected, even in unlikely 'edge cases', and building enough trust to qualify the model for industrial use, without clear understanding of how it operates, is challenging. The third barrier to the application of ML to NDE is the shortage of data. ML algorithms require large quantities of data to train, and most NDE applications require this data to relate to defects. Real defects are rare, and synthetic ones are expensive to manufacture, so it is difficult to create enough data to train complex ML models without extreme financial cost. Providing solutions to these three challenges is the main objective of the work in this thesis. Table 1 illustrates which challenge(s) each of the main chapters in this thesis is associated with. Building on [9], which demonstrated the usefulness of PWI for inline pipe inspection, this thesis considers the same industrial application, showing how ML can be leveraged to size defects from PWI images more accurately than traditional NDE approaches.

The rest of this thesis is structured as follows. Chapter 2 outlines the inspection set up used to approximate inline inspection conditions, the simulation and experimental methodologies, and the data sets they are used to produce. Chapter 3-6 provide the main research outputs and are each based on a journal paper. The associated three papers for Chapter 3-5 are published. The paper associated with

Chapter 6 is in review. Chapter 3 describes how deep learning can provide accurate crack sizing without the need for human feature engineering and simulation used to create the necessary training data. Chapter 4 compares different domain adaptation methods for the purposes of optimally leveraging a small quantity of experimental training data to improve sizing accuracy. Chapter 5 aims to increase trust in ML predictions by quantifying their uncertainty. Chapter 6 proposes a dimensionality reduction methodology that, in conjunction with game theory results, can be used to create a more interpretable and explainable ML framework. Finally, Chapter 7 provides chapter summaries and suggestions for future work.

Table 1. Description of which challenges the main content chapters of this thesis address.

Challenge \ Chapter	Chapter 3: Deep learning for crack sizing	Chapter 4: Domain adaptation	Chapter 5: Uncertainty quantification	Chapter 6: Interpretability and explainability
Feature engineering	✓			✓
Black box problem			✓	✓
Data shortage	✓	✓		

Chapter 2. Inspection, simulation and data sets

This chapter describes the inspection setup, simulation and experimental methodologies, imaging technique used, and resulting data sets. The information in this chapter is applicable to all following chapters as they use broadly the same data sets. The small data set changes made between chapters result in minor differences in data set size and distribution. These differences are outlined in this chapter, and the exact distribution of data sets is further detailed in each corresponding chapter. The content in this chapter is reproduced from the author's published work [34].

2.1. Inspection setup

A major objective in inline pipe inspection is to detect and size the cracks that might occur on the outer or inner surfaces of the pipe. These are usually caused by manufacturing flaws such as weld toe cracks or lack of fusion, or in-service mechanisms such as stress corrosion cracking or thermal cycling fatigue and most commonly occur at the outer surface of the pipe. Sizing these surface-breaking defects, at the outer surface, with detection assumed already complete, is the focus of this thesis. With access to a real pipeline not available for this work, an inspection setup is devised to match in-service conditions as closely as possible. The resulting inspection configuration, and its relation to an oil pipeline, is shown in Fig. 2a,b.

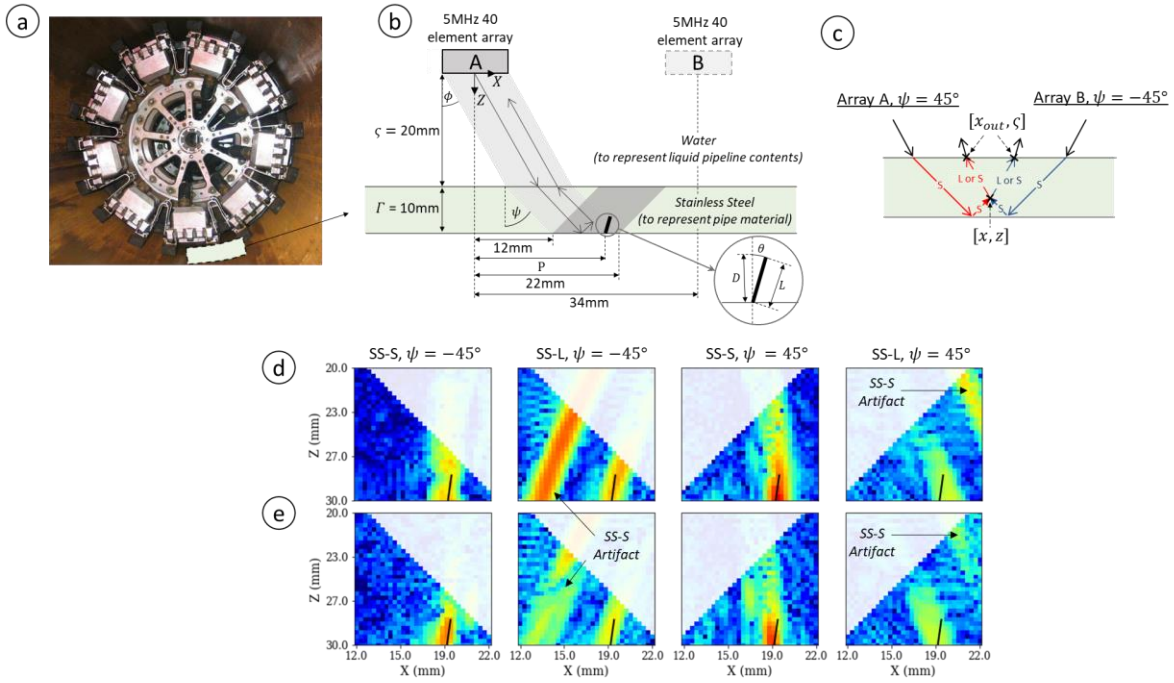


Fig. 2. a) A picture of an UltraScan^{TM} Duo pig in an empty oil pipeline, b) a diagram of the inspection scenario using a plane wave at angle ψ to the sample normal transmitted in the sample with a standoff and thickness of ς and Γ where L, θ and P represent the crack length, angle and position respectively, c) all half-skip shear (S) and longitudinal (L) mode ray-paths used in this thesis where x, z are the co-ordinates of the imaging point and x_{out}, ς the co-ordinates of the returning ray on the front wall, d) an example set of simulated images for a defect with $P = 19$ mm, $L = 2$ mm and $\theta = 8°$ and e) a fully experimental set of images for a defect of the same parameters. Note that the black lines show the true extent of the defects, and all images are on the same dB colour scale, normalised to the maximum intensity in the experimental set. The areas displayed with more transparency are outside the region insonified by the incident plane wave. Figure reproduced from [34].

As pigs are typically used for pipes of least 40 cm diameter the effect of curvature is ignored, and a flat, 10 mm thick stainless-steel plate sample is used to represent the pipe wall material. Most pipelines for this application are made of carbon-steel and contain oil. Stainless-steel has been chosen for these samples to avoid corrosion and water used instead of oil to reduce cost. These replacements are acceptable as sound speeds, attenuation and levels of grain noise are comparable. A commercially available 5 MHz, 0.3 mm pitch, 40 element standard phased array is used. Plane waves are fired at $\phi = 0°$ and $\pm 19°$ to the vertical, in water, to create longitudinal waves at $\psi = 0°$ and shear waves at $\pm 45°$ inside the stainless-steel sample. All 40 elements receive individual A-Scans on reception, forming plane wave capture (PWC) data. The two angled waves are used to characterise defects while the 0° wave is only used to calculate standoff (ς) and thickness (Γ). Note that as $\phi = 19°$ is beyond the first critical angle there are no longitudinal waves transmitted into the stainless-steel. Sound speeds in the steel are calculated using a calibration sample of known thickness (10 mm) and standoff (20 mm) giving longitudinal speed to be 5759 m/s, a shear speed of 3165 m/s and water speed to be 1480 m/s. PWC data is collected from either side of the defect to replicate the use of a pair of arrays from the circumferential ring of arrays on the pig (as shown in Fig. 2a).

All experimental defects are made using electrical discharge machining (EDM) to create 0.3 mm wide notches, described by their angle from the vertical (θ), length (L) and horizontal distance from the array centre (P), as indicated in Fig. 2b. While EDM notches are simpler in shape and texture than most cracks found in in-service pipelines they allow for very accurate true length measurement. Research into the effect of using the presented methods with more realistic cracks is left to future work. Note that in Chapter 3, both L and θ are predicted, while in Chapter 4-6 only the vertical extent of the defect ($D = L\cos(\theta)$) is predicted. This simplification was made as D is the parameter most relevant to predicting the component's remaining surface life.

2.2. Imaging

PWC data is first filtered using a Gaussian filter centred at 5 MHz with a -40 dB half width of 4.5 MHz. Then, the filtered PWC data is focused on reception, with the overall process termed PWI [10]. When multiple ray paths are considered, the images are termed 'views', and are described by the modality(s) of their transmit and receive legs (L for longitudinal, S for shear) separated by a hyphen to indicate reflection from the image point. In this application, half-skip shear ray-paths in transmission and direct shear or longitudinal ray-paths in reception have been found to provide the strongest signal response and clearest images of the defect, hence the views SS-L and SS-S and are used throughout. These ray-paths are illustrated in Fig. 2c. Imaging occurs in the XZ plane, with the origin placed at the centre of the array, as shown in Fig. 2b. The imaged region is defined as the insonified part of the backwall ($12 \leq X \leq 22$ mm) and the full depth of the thickness plate ($20 \leq Z \leq 30$ mm). The intensity of the PWI

image I for view SS-γ (where γ is L or S) at position x, z due to the plane wave at angle ψ in the sample is defined by

$$I_{\gamma,\psi}(x,z) = \left| \sum_j h_{j,\psi}(t_\psi^T + t_{j,\psi,\gamma}^R) \right| \tag{1}$$

where $h_{j,\psi}(t)$ is the complex, filtered A-Scan for receiving transducer j, and the ultrasonic transit times between the array and image point in transmission, t_ψ^T, and between the imaging point and receiving transducer, $t_{j,\psi,\gamma}^R$, are calculated using

$$t_\psi^T(x,z) = \frac{\varsigma}{c_c \cos(\phi)} + \frac{\Gamma}{c_S \cos(\psi)} + \frac{\varsigma + \Gamma - z}{c_S \cos(\psi)} \tag{2}$$

$$t_{j,\psi,\gamma}^R(x,z) = \frac{\sqrt{(x - x_{out})^2 + (z - \varsigma)^2}}{c_\gamma} + \frac{\sqrt{(x_{out} - x_j)^2 + (\varsigma)^2}}{c_c} \tag{3}$$

where ς is standoff, Γ is thickness, x_{out} is the position of the exiting ray on the front wall (as described in Fig. 2b), c_c is the speed of sound in the couplant, c_S is the shear speed in the sample and c_γ is the speed of the return ray. Note that x_{out} must be found using an iterative method such as Newton-Raphson [35] to minimise the time of flight between the imaging point and receiving transducer. The array is assumed to be parallel to the X-axis and positioned at $z = 0$. In this thesis a resolution of half a shear wavelength at the centre frequency is used (pixel size = $\frac{\lambda_S}{2} = 0.317$ mm) for imaging to minimise the data volume while preserving all information above the diffraction limit. This results in a set of four 32×32 images or each defect. Example sets of simulated and experimental half-skip PWI images are given in Fig. 2c,d. In these images, indications at the expected location of the defect are caused by corner reflections and tip diffractions while the 'artefacts' at other locations are due to these same effects but from a ray-path other than the one being imaged. For example, in Fig. 2d, an artefact from the SS-S ray-path from the defect is very clear in the SS-L image. Regions not significantly insonified by the fired plane waves are ignored, these regions are indicated by transparency in Fig. 2d,e.

2.3. Simulation

To simulate large training sets in a reasonable time, an efficient method is needed. To this end, a mixture of finite element (FE) simulations in the defect locality and ray-based simulations for the whole region of interest are used. Structural and grain noise are included by summation of the simulated data with data collected from a defect-free sample. A flow chart describing this process is given in Fig. 3. This overall simulation method matches the approach titled 'finite element' in [9].

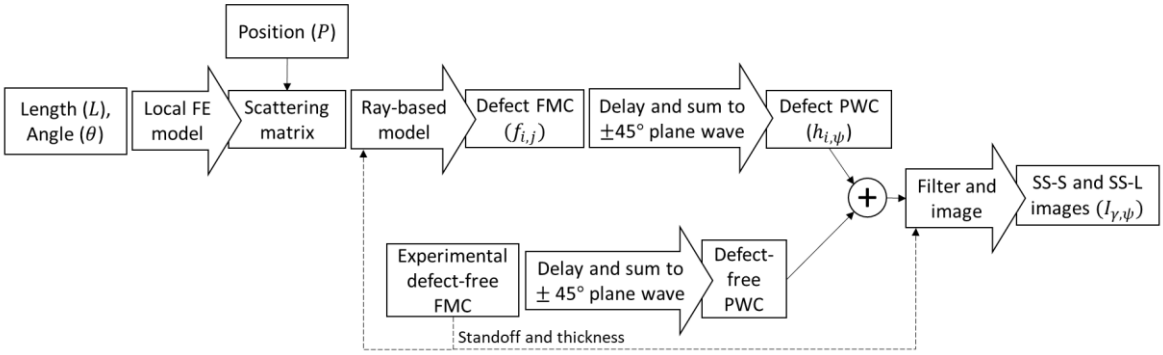


Fig. 3. A flow chart describing the method used to create a simulated set of PWI images given a defect's length (L), angle (θ) and position (P). Figure reproduced from [34].

The local FE simulation functions by exciting a scatterer with a uni-modal plane wave and recording the angle-dependent scattered wave amplitude to calculate its scattering matrix [36]. In this thesis, the scatterers are surface-breaking cracks represented as 0.3 mm wide perfect reflectors with flat tips. The local FE simulation can be conducted independently of the ray-based simulation, so each defect length and angle combination need only to be simulated once, no matter where in the image the defect lies. This is significantly more efficient than using a fully FE approach, which would demand a simulation encompassing the whole region of interest to be run for all positions, as well as all lengths and angles. It should be noted that this method assumes that the receiving element is in the far field of the defect, where the scattered field decays monotonically with distance. In the immersion set-up considered here, the approach has been found to be applicable for defects of up to ~4 mm in length. For longer defects, the far-field assumption is not satisfied, and their simulated image responses are noticeably distorted.

A ray-based simulation is used to create Full Matrix Capture (FMC) data, tracing all relevant paths (direct and half skip) from the array to the defect, and using scattering matrices to calculate the phase and amplitude of its reflections [37]. The FMC dataset, $f_{i,j}$, is used to generate PWC data, $h_{j,\psi}$, with

$$h_{j,\psi}(t) = \sum_i f_{i,j}\left(t - \tau_{j,\psi}\right) \tag{4}$$

by summation over transmitting transducers, i, where the appropriate delay, τ, is given by

$$\tau_{j,\psi} = (x_j - x_r)\frac{\sin(\psi)}{c_S} \tag{5}$$

where x_r is the x position of an arbitrary reference element in the array, chosen to be the central element in this thesis. Note that scattering induced attenuation has not been included in the simulation. This is because through the 10mm thick stainless-steel sample its effect is minimal.

Accurately representing structural and grain noise in the training data is achieved efficiently by collecting experimental FMC datasets from a defect-free sample and combining them with the simulated defect data [38]. This is implemented here by choosing, at random, one of 36 FMC data sets obtained from a defect-free stainless-steel plate. To ensure the arrival times in these data sets match accurately with the simulated data, the standoff (ς) and sample thickness (Γ) used both in imaging and the ray-based simulation, are calculated using the experimental 0° PWC data ($h_{j,0}$). These are calculated with

$$\varsigma = \frac{\sum_j t_{j,F}}{2nc_w} \quad , \quad \Gamma = \frac{\sum_j (t_{j,B} - t_{j,BFB})}{2nc_L} \tag{6}$$

where $t_{j,\mathrm{F}}$ and $t_{j,\mathrm{B}}$ are the arrival times of the front and backwall reflections, n is the number of transducers in the array, and $t_{j,\mathrm{BFB}}$ is the arrival time of the first reverberation inside the sample. Amplitudes are ensured to be on the same scale by normalizing both defect and defect-free sets to a backwall reflection in the 0° PWC data set.

The resulting PWC data now contains signals due to grain noise, front and back wall reflections, and all direct and half skip ray-paths from the defect. A Gaussian filter centred at 5 MHz with a -40 dB half width of 4.5 MHz is applied to the PWC data to remove data outside the frequency range of the transducer. Finally, PWI using Eq. (1) is used to create SS-S and SS-L images from the arrays on each side of the defect.

2.4. Data set summary

The defects of interest are surface-breaking cracks between 1 and 5 mm in length, angled at most 20° from the vertical. The root of the defect can be positioned anywhere between $P = 13$ mm and $P = 21$ mm which corresponds to all bar 2 mm of the insonified backwall region. The simulated set covers lengths and angles beyond that of the experimental set to ensure the resulting network learns across the full parameter space. Defects larger than 5 mm in length are not included as they are well above the critical crack size for this application. The parameter space of the simulated data sets used is summarised in Tables 2 & 3. As highlighted in blue/red, Chapter 4 onwards uses a larger step between crack positions. This change was made to produce a more even coverage of the L, P, θ parameter space and represent the effect of changing P more efficiently by bringing the step up to the diffraction limit ($\frac{\lambda_S}{2} = 0.317$ mm).

Table 2. Simulated data summary for Chapter 3

Parameter	*Range*	*Step*	*Count*
Crack Length, L (mm)	0.2 to 5	0.2	25
Crack Position, P (mm)	13 to 21	0.2	41
Crack Angle, θ (°)	-24 to 24	2	25
Non-Defect Scan	-	-	36
Total = 25×41×25 = **25,625 image sets**			

Table 3. Simulated data summary for Chapter 4-6
Differences between this data set and the set presented in Table 2 are highlighted in red.

Parameter	*Range*	*Step*	*Count*
Crack Length, L (mm)	0.2 to 5	0.2	25
Crack Position, P (mm)	13 to 21	0.3	27
Crack Angle, θ (°)	-24 to 24	2	25
Non-Defect Scan	-	-	36
Total = 25×27×25 = **16,875 image sets**			

Table 4. Experimental data summary for Chapter 3

Crack Angle, θ (°) \ *Crack Length, L (mm)*	1	2	3	4	5
0	✓	✓	✓	✓	✓
±2		✓✓	✓✓	✓✓	
±5		✓✓	✓✓	✓✓	
±8			✓✓	✓✓	
±12			✓✓	✓✓	
±15		✓✓	✓✓	✓✓	
±20		✓✓	✓✓	✓✓	

	Range	*Step*	*Count*
Crack Position, P (mm)	13 to 21	0.2	41
Total = $N_{\theta,L} \times N_P$ = 37×41 = **999 image sets**			

Table 5. Experimental data summary for Chapter 4-6
Differences between this data set and the set presented in Table 4 are highlighted in red.

Crack Angle, θ (°) \ *Crack Length, L (mm)*	1	2	3	4	5
0	✓	✓	✓	✓	✓
±2	✓✓	✓✓	✓✓	✓✓	✓✓
±5	✓✓	✓✓	✓✓	✓✓	✓✓
±8	✓✓	✓✓	✓✓	✓✓	✓✓
±12					
±15	✓✓	✓✓	✓✓	✓✓	✓✓
±20	✓✓	✓✓	✓✓	✓✓	✓✓

	Range	*Step*	*Count*
Crack Position, P (mm)	13 to 21	0.3	27
Total = $N_{\theta,L} \times N_P$ = 55×27 = **1,485 image sets**			

Experimentally, changes in P are achieved by movement of the array relative to the defects. The samples are rotated 180° to obtain data from defects with both positive and negative angles. The parameter space coverage of the experimental data is summarised in Tables 4 & 5. The research presented in Chapter 3 was implemented using the experimental defects available at the time, as indicated by the ticks in Table 4. Chapter 4 includes experimental data in the training set, so good coverage of the L, θ parameter space coverage was required. More defects were manufactured to achieve this, as indicated by the ticks in Table 5. $|\theta| = 12°$ defects are not used beyond Chapter 3 as they were found to cause reverberations from ray paths not considered in the simulation. These reverberations cause differences between experimental and simulated data significant enough to affect the sizing accuracy. As this is an effect specific to this data set, these defects were excluded, simplifying analysis and making the results applicable to a wider NDE audience. This is further discussed in Section 3.4.1. As with the simulated data, the step in P was increased to 0.3 mm after Chapter 3. The maximum intensity of defect indications in the simulated set is found to have a Mean Absolute Error (MAE) relative to the experimental set of 0.97%. Along with the visual similarity of the images, such as in Fig. 2d,e, this low level of MAE is considered to validate the simulation.

Chapter 3. Deep learning for crack sizing

This chapter demonstrates how deep learning and physics-based simulations can be used to achieve automated data analysis without experimental training data or hand-engineered features. A deep learning network is trained on simulated data and shown to generalise to experimental data it has never seen before. While this approach could be applied to any inspection scenario, here the application considered is sizing surface-breaking cracks in ultrasonic inline pipe inspection. The content in this chapter is drawn from the author's published work [34].

3.1. Introduction

When a defect is flagged by ultrasonic inline pipe inspection the data from that position is compressed and stored for offline analysis, often involving significant operator input. This chapter is concerned with automating this offline analysis using deep learning and a simulated training set. The orientation and length of each defect is predicted from four distinct images. While imaging provides a large amount of data size reduction relative to the raw time-domain data acquired by the receiving arrays, learning from four images directly is still a very high dimensional problem. convolutional neural networks (CNNs) [39] are a natural answer to this as they connect only nearby pixels at each layer, vastly reducing the complexity of the network. They have also seen widespread success with natural [40], medical [41]–[43] and NDE [31], [32], [44]–[46] images in the past. There are many well-known CNN architectures for image characterization such as LeNet, DenseNet, Inception, AlexNet and ResNet [47]. The broad structure for the network used throughout this thesis takes inspiration from networks such as AlexNet and VGG-19 and makes use of advances such as dropout, ReLu activations and max pooling [47] to assist in generalizing to experimental data after training on simulated data.

Typically, medical and NDE deep learning papers use 500-10,000 examples in their training sets. However, in the wider machine learning community sets such as ImageNet are being used that have more than 10,000,000 examples. It is generally accepted that the power of a deep learning network hinges heavily on the size of its training set. In NDE, useful training data usually requires real or manufactured flaws, but as this is expensive to produce, there is a shortage of experimental training data. This chapter intends to show how by using simulations to create training sets, the NDE community can begin to unlock the power of the state-of-the-art deep learning being used elsewhere. The techniques described in Chapter 2 are used to generate 25,625 simulated image sets that train the sizing network, and 999 purely experimental sets from samples containing notches, made using electrical discharge machining (EDM), are used to evaluate its accuracy. These data sets are generated using the methodologies described in Chapter 2 and their distribution is outlined in Tables 2 & 4.

The rest of this chapter is structured as follows. Section 3.2 outlines the deep learning method used to characterise defects, as well as a more traditional sizing technique, the 6 dB drop method. Section 3.3 describes the method used to approximate material sound speed variation. Section 3.4 presents results

for the accuracy of the methods in sizing experimental defects and demonstrates the adaptability of the deep learning approach, by sizing defects imaged with incorrectly estimated sound speed. Overall success is judged by comparison in sizing error to the 6 dB drop method.

3.2. Defect characterization algorithms

In this section, the process for implementing the 6 dB drop sizing method will be explained, the CNN architecture used will be described, and the training method outlined.

3.2.1. 6 dB drop method

The 6 dB drop method is a common way to size defects in ultrasonic images and is presented here as a comparison for the deep learning approach. The 6 dB drop method is based upon the idea that if a defect is the strongest indicator in an image the region of the image that is within 6 dB of the peak value can be used as a good approximation of the size of the defect. This is implemented by calculating the minimum area of a rectangular box that encloses all pixels within 6 dB of the peak value and taking the crack length and angle as those of the major axis of the enclosing box [11]. Pixels above -6 dB must be within a certain distance of each other to be considered part of the same defect. In this chapter the maximum distance is set at 4 pixels (1.27 mm). 6 dB drop is deemed to be the most appropriate traditional sizing technique as amplitude-based methods for large surface breaking defects suffer from constant amplitude corner reflections [48], [49], tip diffraction signals are not consistently strong enough to enable temporal based techniques and the restricted range of incident and reflected angles means that scattering matrices [50] cannot be calculated. The reader is directed to [49] for a comprehensive review of traditional NDE sizing techniques.

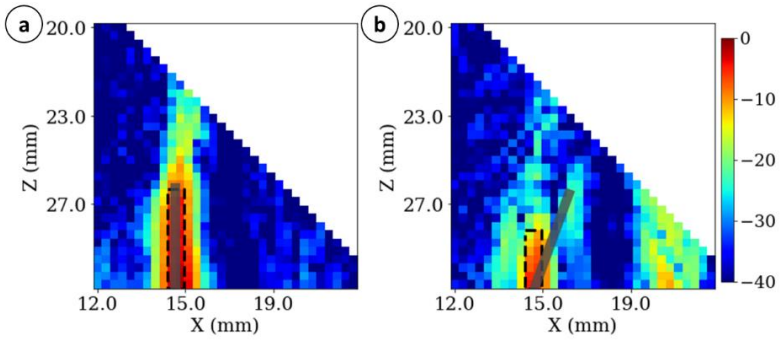


Fig. 4. Experimental SS-S images for a) a defect with $L = 4$ mm, $\theta = 0°$ and $P = 15$ mm, predicted by the 6 dB drop method to have $L = 3.5$ mm and $\theta = 0°$ and b) a defect with $L = 4$ mm, $\theta = 20°$ and $P = 15$ mm, predicted to have $L = 1.9$ mm and $\theta = 0°$. Black dashed lines indicate the box fit by the 6 dB drop method and grey solid lines the true extent of the defect. Note that the images are in dB, each image is normalised to its own maximum value.

This method has a few advantages over a machine learnt one. For example, it requires no tuning other than setting the range at which indications are considered to be from different defects, it is simple to execute, and is not a 'black box' method (a common criticism of deep learning). It can be argued to be a physics-based approach in that a single transducer above a large, planar defect will return half the

amplitude (i.e., 6 dB) when 'half on, half off' the defect compared to a measurement from directly above it [49], [51]. As this occurs at the edges of the defect the indication in a simple B-Scan should be described by a 6 dB drop. Fig. 4a shows an experimental example where this works well, with the 6 dB box describing the extent of the defect quite accurately, undersizing by only 0.5 mm. However, this method performs poorly in more complex scenarios. For example, Fig. 4b shows a more angled defect from which the specular and tip reflections are well below -6 dB so only the corner indication is picked up, resulting in undersizing by 2.1 mm. Importantly, it is also difficult to make use of information from more than one image using 6 dB drop. In this application, defects of $|\theta| > 12°$ are much more accurately visualised in the SS-L view than SS-S, however, effectively deciding which one to use without prior knowledge of the defect is challenging. The SS-S view has been used for 6 dB sizing throughout this work as on average it gives a more accurate result.

3.2.2. Deep learning

3.2.2.1. Network architecture

The deep learning architecture used here is convolutional and loosely based on image recognition architectures such as AlexNet and VGG-19 due to their widespread success in image classification and regression [52]. Similar to these architectures, sets of convolutional and max pooling layers with ReLu activation functions are used to achieve feature extraction and are followed by fully connected layers to predict the output. However, all hyperparameters have been tuned for this application. Directly using a well-known architecture 'off-the-shelf' is not possible as the images they are designed for are much larger in size than those used in this thesis. It also cannot be assumed that the most successful architecture for natural images will be the best choice for NDE images as their content is significantly different in structure. In this thesis, as shown in Fig. 5a, dropout before each fully connected layer is used to minimise overfitting to the training set, 10% is chosen as this was found to be a good trade-off between train time needed to converge and decrease in validation set loss (indicating reduced overfitting). To make use of all four images they are stacked at the input (akin to how natural image CNNs treat red, green and blue channels) producing a 32x32x4 input. This input is then fed into one network that predicts crack length and another network that predicts angle. These networks are decoupled to allow them to learn the image features that are most useful in predicting each property. Note that the outputs can take on any real value, making this a regression, rather than classification network.

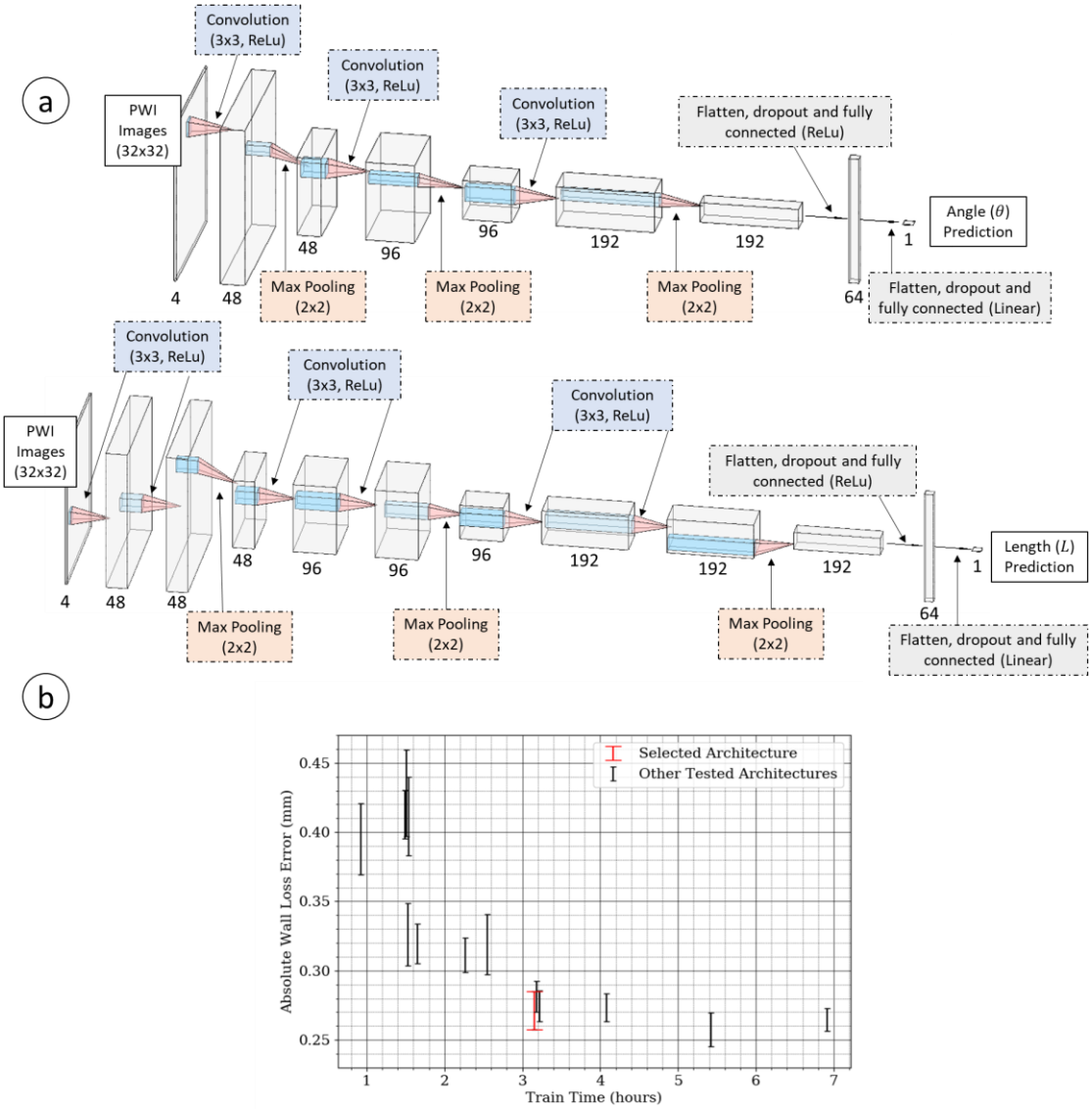


Fig. 5. a) An illustration of the chosen architecture and b) absolute experimental validation set wall loss error for all tested architectures where error bars represent ± standard deviation over 10 independent initializations.

Table 6. Hyperparameters for all tested architectures, as shown in Fig. 5. The selected architecture is highlighted in green.

Train time (hrs)	Mean absolute wall loss error (mm)	Convolutional filter number per layer	Filter size	Neurons in dense layer
0.93	0.39	12, 12, 24, 24, 48, 48	3	128
1.49	0.41	24, 24, 48, 48, 96, 96	2	64
1.51	0.43	24, 24, 48, 48, 96, 96	2	128
1.53	0.33	24, 24, 48, 48, 96, 96	3	128
1.54	0.41	24, 24, 48, 48, 96, 96	2	126
1.65	0.32	24, 24, 48, 48, 96, 96, 96, 96	3	128
2.27	0.31	24, 24, 48, 48, 96, 96	4	128
2.55	0.32	24, 24, 48, 48, 96, 96	5	128
3.15	0.27	48, 48, 96, 96, 192, 192	3	128
3.18	0.28	48, 48, 96, 96, 192, 192	3	32
3.23	0.27	48, 48, 96, 96, 192, 192	3	128
4.08	0.27	64, 64, 128, 128, 256, 256	3	128
5.43	0.26	72, 72, 144, 144, 288, 288	3	96
6.92	0.26	96, 96, 192, 192, 384, 384	3	128

The route to arriving at the final networks shown in Fig. 5a is by trialling different numbers of layers, filters and filter sizes to increase complexity (and therefore train time) until the improvement in accuracy is minimal. To simplify this analysis, length and angle predictions are combined into 'wall loss,' defined as $L\cos(\theta)$, which is usually the metric of interest when deciding upon the safety of a pipeline. The result of this study, using the experimental validation set to calculate wall loss, is shown in Fig. 5b with error bars representing the standard deviation in results over 10 independent initializations of each architecture using different starting weights and train/test/validate shuffles. Fig. 5b shows that there is a diminishing return in adding complexity to the network. The architecture selected is chosen as further increasing network complexity offered no statistically significant improvements in sizing accuracy.

3.2.2.2. Data sets and training

The simulation and experimental methodologies used to generate data for this chapter are those described in Chapter 2 and Tables 2 & 4. For use in machine learning, the simulated and experimental sets are each further split into two more:

Simulated, train: 85% (21781) of simulated data, used to iteratively update the weights and biases of the network.

Simulated, validation: 15% (3843) of simulated data, automatically analysed to implement the training stop condition, minimizing overfitting to the simulation.

Experimental, validation: 75% (749) of experimental data, used during research and design stages to ensure the network is not overfitting to the simulated images and to implement the training stop condition.

Experimental, test: 25% (250) of experimental data, used to evaluate the performance of the trained network on previously unseen data.

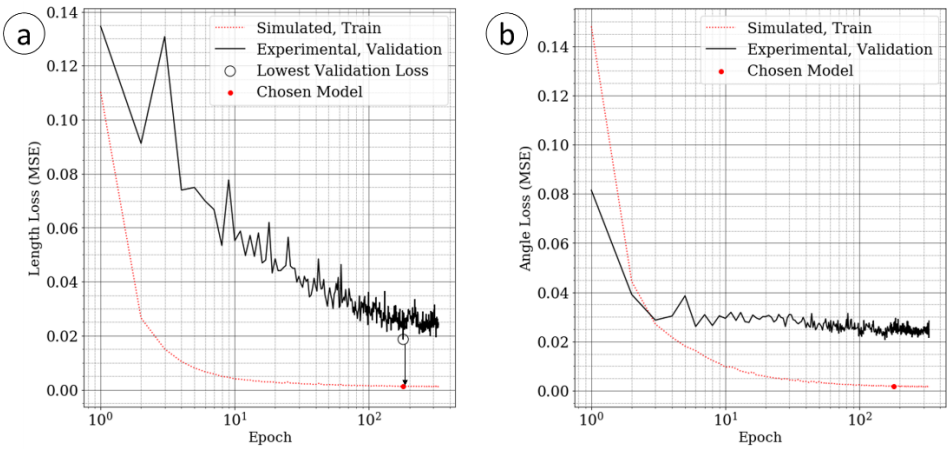


Fig. 6. An example pair of training progress graphs for a) L and b) θ predictions.

The CNN is implemented in Python using TensorFlow, trained using a Mean Square Error (MSE) loss function and the state-of-the-art 'Adam' optimiser [53] with a learning rate of 0.001, in mini-batches of 128 for a maximum of 400 epochs with a patience of 150. For machine learning terminology and definitions see [54]. This learning rate was selected by raising it from a low value until any significant instability in simulated training set loss appeared. The mini-batch size of 128 is selected to reduce overfitting without causing train time to increase dramatically as while a small batch size gives a regularization effect it increases train time [55]. After training, the weights and biases chosen are from the point of minimum experimental validation loss. An example training progress graph is shown in Fig. 6 where it can be seen that a minima in experimental validation loss is reached at 180 epochs, past which point validation loss begins to increase due to overfitting to the simulation. On a workstation GPU (NVIDIA Quadro K620) training 400 epochs takes ~3 hours.

3.3. Sound speed variation

With machine learning, creating a network that can cope with expected variations in inspection conditions is achieved by including these variations in the training set. Here the case of inaccurate knowledge of sound speeds is considered as an example. In practice, the variation would be in the physical measurements and the image reconstruction sound speeds would be fixed. However, because it is not readily possible to obtain a large amount of experimental data from physical systems with different sound speeds, the sound speeds used for image reconstruction are varied instead. Varying the reconstruction sound speed is not directly equivalent to varying the specimen sound speed, as the latter causes changes in physical quantities such as the crack length to wavelength ratio. However, in terms final image distortion, these are second order effects compared to a mismatch between the specimen and reconstruction sound speeds.

It is assumed that a sensor is available to get an accurate reading of temperature in the couplant from which its speed of sound can be estimated from previously acquired speed vs. temperature calibration data. Because in practice the pipeline product acts as the couplant, there will be some uncertainty in its sound speed due to uncontrolled variables, such as the exact composition of the product. Shear and longitudinal speeds in the steel pipe have larger potential uncertainty caused by effects such as variation in material composition, and temperature change due to the external environment. To include these variations in sound speed, random uniform multipliers are used at the imaging stage. These are

$$0.99 < \beta_W < 1.01 \tag{7}$$

$$0.9 < \beta_S < 1.1 \tag{8}$$

$$0.9 < \beta_L < 1.1 \tag{9}$$

where β_W, β_S and β_L are multipliers for the water speed, c_w, shear speed, c_S, and longitudinal speed, c_L, used in Eqs. (7-9). These values are larger than the true variation in material sound speed is likely to be; for example, carbon steel experiences less than a 10% variation in sound speed [56] across the full temperature range an inline pipe inspection tool is able to operate in (-10 to 50°C [57]). These large values of β are chosen to demonstrate the effectiveness of this method even under extreme conditions. As is evident from Eq. (6), the calculated thickness and standoff will change proportionally to longitudinal and water speeds, respectively. The coordinates of the imaging mesh are moved to consistently sit at the predicted position of the plane wave aperture on the backwall. An example of images produced with the most severe set of errors is given in Fig. 7b where it can be seen that the sound speed errors have caused significant spatial movement of defect responses, total loss of co-registration and a change in indication amplitude and size for some cases.

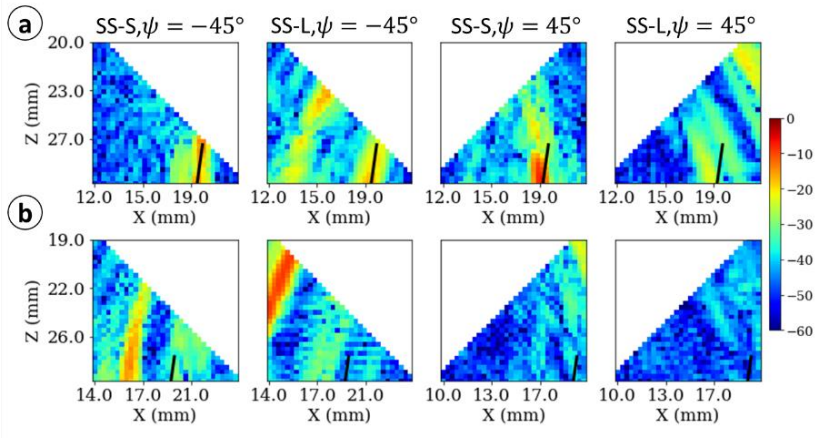


Fig. 7. a) An experimental image set for a defect with $L = 3$ mm, $\theta = 8°$, $P = 19.2$ mm and no sound speed variation and b) the same PWC data imaged with $\beta_S = 1.1, \beta_L = 0.9$ and $\beta_W = 0.99$. All images are on the same color scale in dB, normalised to the maximum intensity in the experimental set. The black lines show the true extent of the defect.

3.4. Results and discussion

As outlined in Section 3.2.2.2, the weights within the CNN are initialised with a random seed. In addition, the assignment of a particular dataset to the train, test and validation sets is also random to avoid potential bias. The first consideration is therefore repeatability of the trained CNN. This section will also present and discuss length and angle prediction accuracy of the 6 dB drop method in comparison to that of the CNN both with and without errors in sound speed estimation.

3.4.1. Deep learning repeatability

It is important to know the amount of variation in accuracy over different network initializations as large scatter could suggest poor generalization. This is because in a wide distribution of test results the lower errors may be caused by fortuitous train/test splits rather than better networks. With low scatter, a higher level of confidence can be placed in the model's success not being due to overfitting. To test this, 80 networks are trained from different starting seeds and the spread of their results for three

example defects are shown in Fig. 8. The low standard deviations in these results suggests that there is a big enough training set and enough network complexity for the network to generalise and the training to be satisfactorily independent of initial weights and train/test/validation data partitioning. This means that the final network can be picked at random from these 80 realizations.

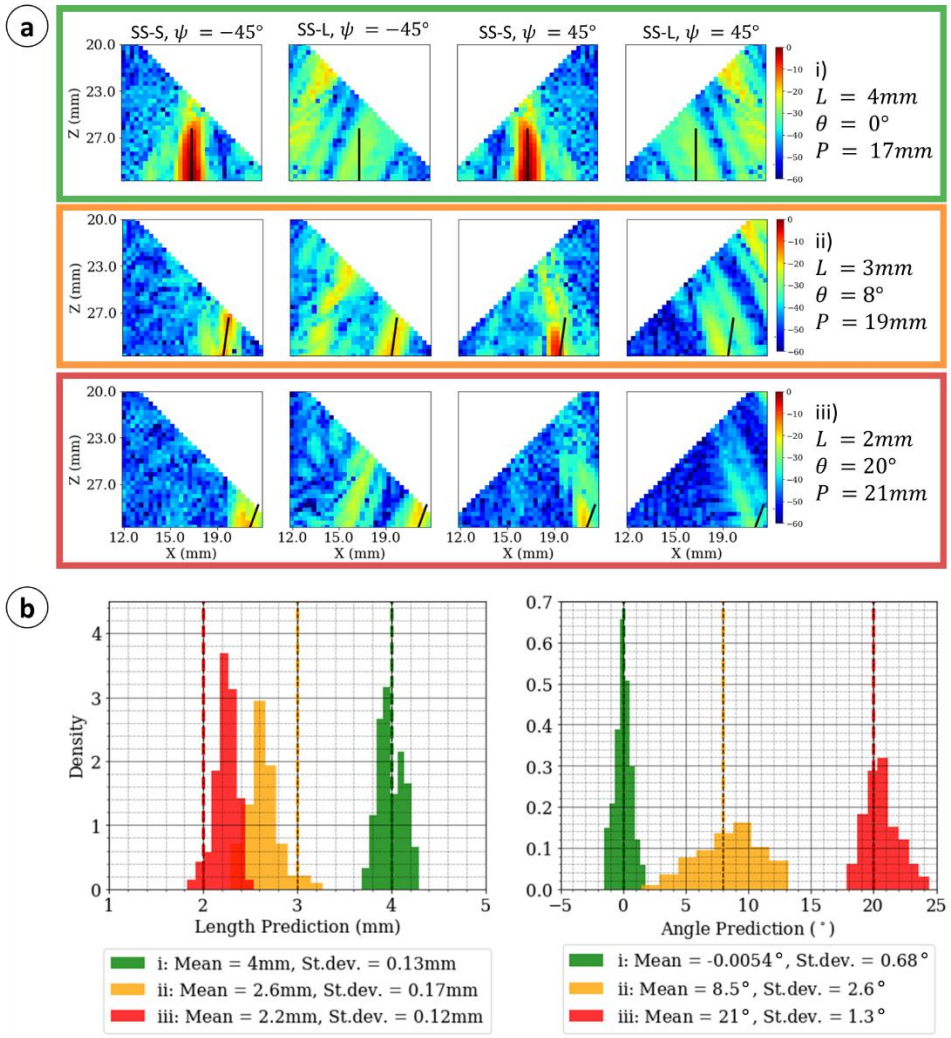


Fig. 8. a) Three experimental image sets with black lines indicating the true defect extent (all images are on the same colour scale in dB, normalised to the maximum intensity in the experimental set) and b) a histogram showing the length and angle CNN predictions for these defects from 80 different training attempts. Dashed vertical lines represent true length and angle.

The larger standard deviation in error for defect ii) compared to iii) is unintuitive as defect ii) has a higher Signal to Noise Ratio (SNR) and its indications better match its true size. Investigation into this found that experimental defects of $8 \leq |\theta| \leq 15$ cause weak reverberations from ray paths not considered in the simulation. These are very low in amplitude relative to the SS-S and SS-L views but cause an average SNR drop of 2 dB across these angles. While this is a small value for high SNR data like this (~30 dB) it is hypothesised to be the cause of the larger spread in error for defect ii). This finding highlights the importance of an accurate simulation. Further research into the cause of these reverberations will allow them to be modelled in the future.

3.4.2. Deep learning vs 6 dB drop sizing accuracy

Fig. 9a shows the error in characterizing the experimental test set using the 6 dB drop method, a CNN trained without any variation in training set sound speeds and a CNN trained with the sound speed variation described in Section 3.3. Table 7 gives the mean and standard deviation of these prediction errors. In terms of length prediction, the 6 dB drop method shows a non-negligible mean prediction error of -0.86 mm so is on average under-sizing the cracks. It also has a significant standard deviation of 1.1 mm. Both the standard CNN and the CNN trained with speed variation outperform this with near-zero mean error and standard deviations of 0.39 mm and 0.59 mm, respectively. The results for angle follow a similar pattern. The most successful method for this test set is the standard CNN that has 95% confidence intervals of $\pm$0.77 mm and $\pm$8.0°.

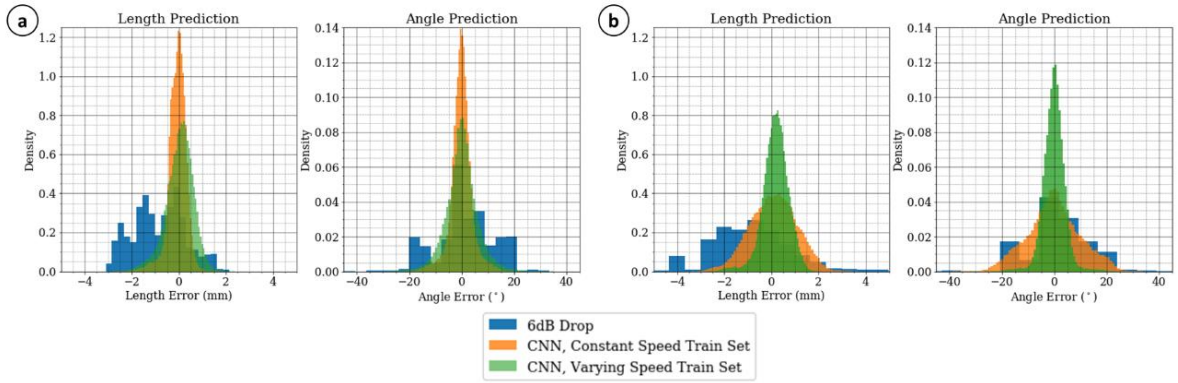


Fig. 9. Histograms of length and angle prediction error of methods applied to a) the standard experimental test set and b) the experimental test set with sound speed variation applied. Note that for CNN results all predictions from 80 independent initializations are shown.

Table 7. Mean and standard deviation of prediction errors for the experimental test set.

		Characterization Algorithm					
		6 dB Drop		CNN (Standard Train Set)		CNN (Speed Varying Train Set)	
		Mean	*St.Dev.*	*Mean*	*St.Dev.*	*Mean*	*St.Dev.*
Test Set, Quantity	Speed Constant, Length (mm)	-0.86	1.1	-0.063	0.39	0.03	0.59
	Speed Varying, Length (mm)	-0.78	1.8	0.088	0.98	0.18	0.56
	Standard, Angle (°)	1.4	12	-0.13	4.1	0.062	4.1
	Speed Varying, Angle (°)	-2	20	-0.15	10	-0.048	4.2

Fig. 9b shows the performance of the same methods on the experimental test set with sound speed variation included. As shown in Table 7, the standard deviations in length and angle prediction for the 6 dB drop method rise by 64% and 67%, respectively, compared to results on the standard test set. The standard CNN more than doubles in standard deviation. However, while adding sound speed to the training set increases errors for the standard test set the increased generality it creates means that adding speed variation to the test set decreases standard deviation by 5% for length and 31% for angle. This

results in a network with 95% confidence intervals of ± 1.1 mm and $\pm 8.2°$ even with uncertainties in material sound speed up to 10%.

Whilst the results presented are for a relatively coarse imaging grid (pixel size = $\frac{\lambda_S}{2} = 0.317$ mm), a finer grid (pixel size = $\frac{\lambda_S}{6}$) provided negligible improvement for either the CNN or dB drop methodologies. For the 6 dB drop, this is because the limitation is accuracy rather than precision, evidenced by 81% of absolute length errors in Fig. 9a being larger than the coarse image pixel size. The standard CNN sizes much fewer defects with errors larger than a pixel (34% in Fig. 9a), but its prediction is not intrinsically based on distances in the image so is harder to relate to the pixel size. Furthermore, as the chosen resolution is already at the diffraction limit ($\frac{\lambda_S}{2}$) reducing it does not provide any further information about the defect to the network.

3.4.3. Discussion

This chapter shows, once again, that avoiding overfitting is key to the success of deep learning. While this is common knowledge within the machine learning community, its importance cannot be overstated. This is of even more importance when training on simulated data, as for the network to be useful it must be able to operate on real data, despite any simplifications or assumptions the simulation may make. Use of dropout, analysis of validation data and careful training set creation is essential. It must also be ensured that the training set contains all significant variation that is expected to occur in the real inspection. This is demonstrated here with sound speed variation in the training set, but the principle extends to many other properties such as variable attenuation, standoff, surface roughness and array alignment. It is worth noting that finding which simulation inaccuracies cause significant errors in experimental sizing is difficult and not always intuitive. This is exampled in Section 3.4.1 where reverberations not included in the simulation, despite being weak relative to the defect's half skip response, cause non-negligible decreases in angle prediction accuracy. Ultimately, the main limitation of this method is the breadth and accuracy of the training set. Including the correct variation, or somehow accounting for the deficiencies of the simulation, is key to creating a network that is applicable to real data.

Due to its simplicity the 6 dB drop method is computationally inexpensive. However, it is shown to give far less accurate predictions than the CNN. A large factor in this is the quantity of information available to each sizing algorithm. While the 6 dB drop method must size a crack solely from its shape in one image the CNN is able to take information from the amplitude and shape of indications and artefacts in multiple images. This could be further capitalised upon if applied to situations with more views such as multi-mode Total Focusing Method (TFM) [39]. However, deep learning is not without its drawbacks as it is often perceived to be a 'black box' method. This makes it difficult to directly relate its predictions to their cause. For a conservative field like NDE where historically, inspections have

been qualified using physics-based reasoning this is a big drawback, but one that could be overcome by ongoing work in techniques such as activation and feature visualization [58]–[60] which provide mechanisms to understand the rationale behind the final sizing choice. How deep learning is integrated into the workplace must also be done carefully. Making use of its predictions without introducing unwanted bias or degrading human skills through overreliance are issues demanding thought and care. However, as current deep learning application to safety-critical problems such as self-driving cars has proved, this is certainly achievable. Therefore, the significant increase in characterization accuracy compared to current methods that this chapter has presented are a strong motivation for application and further research of deep learning for NDE.

3.5. Conclusion

This chapter has demonstrated how a simulation approach can generate the large training datasets which enable deep learning for crack characterization. The resulting CNN sizes 97% of the tested experimental defects of length 1 to 5 mm within ± 1 mm while the 6 dB drop method only achieves 48%. Even with a maximum of 10% uncertainty in material sound speed the CNN still achieves 91% sizing in the ± 1 mm range, while the 6 dB method drops to 40%. Future research should be carried out in testing the adaptability and limits of this method by characterizing a wider range of defects such as branching cracks, corrosion and cracks at welds. The network could also be improved by exploring methods to add an output that indicates a level of confidence in its characterization of each defect. The deep learning characterization approach identified in this chapter is demonstrated to be successful for in-line pipe inspection and is readily applicable to other ultrasonic NDE inspections.

Chapter 4. Domain adaptation

This chapter investigates how a small amount of experimental data can optimally be added to a large, simulated training set, to improve the performance of a deep learning architecture for ultrasonic defect sizing. If achieved effectively, this can reduce the chance of sizing errors caused by domain shift between simulated and experimental data. The content in this chapter is drawn from the author's published work [61].

4.1. Introduction

ML for NDE has seen a large number of successes, demonstrating human-level NDE data interpretation [16]–[22], [24], [30]. However, these successes rely on the availability of sufficient data, and that this data closely matches real inspection conditions. One solution to the data shortage problem, as demonstrated in Chapter 3, is to use a physics-based simulation to create the training set. However, while simulating a training set is an attractive approach, simulated NDE data can never perfectly match real data as it invariably contains simplifications and assumptions. This means that a model trained only with simulated data may not accurately size experimental data. This chapter looks to solve this problem by including a small pool of experimental data in the training process. This is a 'Transfer Learning' (TL) [62] problem in that it aims to train a network using data from a 'source' domain (i.e., simulation), that is intended to perform a task in a different, but related, 'target' domain (i.e., experiment). TL for problems with the same task in both domains, as in this chapter, is called Domain Adaptation (DA).

In this chapter, three DA approaches are presented and compared against two baseline cases in their ability to improve the sizing accuracy of a CNN by adding a small amount of experimental data to the simulated training set. Building on the work presented in Chapter 3, the same CNN architecture, inspection set up, simulation methodology and imaging protocol are used here. Also, as with Chapter 3, the DA methods presented are applicable to any NDE application and modality but their effectiveness is demonstrated by considering inline pipe inspection. As described in Chapter 2, the pig considered uses a ring of ultrasonic arrays to induce plane waves in the pipe that travel at both 45° and -45° to the surface. From the received data, four distinct ultrasonic array images are created for each surface breaking defect and used as input to the CNN to predict the through thickness extent of the defect (from here on referred to as 'crack depth'). The effectiveness of the baseline and DA methods to improve the CNN's sizing accuracy is explored in this chapter by training with a simulated training set size of 14,343 and a varying size of experimental training set (54-729, from measurements on 1-14 physical defect samples). The sizing accuracy of the resulting network is assessed using an experimental test set formed of 756 image sets from 15 physical defect samples not included in the training set.

The rest of this chapter is structured as follows. Section 4.2 outlines previous, relevant research, Section 4.3 describes the inspection setup and data sets, Section 4.4 details the deep-learning architecture,

Section 4.5 describes the DA methods used, Section 4.6 provides results and discussion and Section 4.7 the conclusion.

4.2. Relevant research

Outside of NDE, TL has found success in a broad range of applications such as multilingual text classification, WiFi-based localization, speech recognition across different speakers, object recognition across different cameras, human motion parsing from videos, facial recognition and 3D pose estimation [62]–[64]. A major reason for this widespread usage of TL in recent years is the availability of large, free to access, source domain data, such as ImageNet [65] and CIFAR-10 [66] for natural image classification, IMdB reviews [67] and WordNet [68] for natural language processing, and LibriSpeech [69] for English speech recognition. For NDE there is a small, but insufficient, amount of work towards creating an equivalent data set [70]. But where source data is available, promising results with TL for NDE have been found. For example, a database of NDE X-ray images [71] has been used to train a CNN for inclusion detection in composites and unsupervised (i.e., without labelled target data) DA using the Case Western Reserve University bearing data set has been used to train a CNN for bearing inspections across different rotation speeds and load conditions. However, for most NDE applications, a training set large enough to function as source data for deep-learning is not available. Shallow-learning methods (i.e., predicting on hand selected features) require much less training data than deep-learning and have been used in structural health monitoring to train a hidden Markov model with source and target data from different transducer placements [72] and a K-Nearest Neighbours (KNN) method used to detect defects with source and target data from different carbon fibre composite samples [73]. A KNN model has also been used for structural health monitoring of buildings from the first three natural frequencies trained on source data from an analytical beam-bending model [74].

To find the most effective DA methods for use with labelled target data, as used in this thesis, research was conducted into popular deep learning DA methods proposed in recent published papers. During initial testing, some of the methods [75], [76] were found to produce lower sizing accuracy than networks trained without any target data at all, and are not presented here. It is the author's belief that the poor performance of these methods is largely due to the fact that they are optimised for the 'semi-supervised' case where there is both unlabelled and labelled target data. Research specifically into supervised DA (i.e., where all data is labelled) methods has attracted little recent attention as most modern DA applications are motivated by lack of labelled data [64]. It is believed that there are only two recently published methods specifically designed for supervised DA. These are Regression and Contrastive Semantic Alignment (RCSA) and Adversarial. RCSA uses an extra loss function to encourage proximity in the embedding space (the output of the convolutional layers) for data of the same label [77] while Adversarial optimally confuses a domain classifier to force the embedding space to be domain independent [78], [79]. These two DA methods are presented in the current chapter along with a simpler DA approach, MixedSet, where training is performed with a mixed

experimental/simulated set with sample weightings used to make up for the lack of experimental data. As noted in [80] most DA research has focused on 'classification' tasks where the desired parameter is a discrete label. RCSA and Adversarial as originally presented in [77], [79] are consistent with this observation as they do not function with continuous labels. Because of this they have been adapted for the regression setting in this work; this is explained further in Section 4.5. To the author's knowledge the only prior work in using simulated NDE data as a source domain for domain adapted deep-learning is [81] in which phased array data generated using a finite element model is used as source data to locate and size defects in an aluminium block. The authors of [81] use a basic DA approach in which they train on simulated, then experimental data. This method is similar to MixedSet in terms of its effect on the network.

4.3. Data sets

While the target is the extent of the defect perpendicular to the surface, $D = L\cos\theta$, the parameter space of defects considered is defined by L, P and θ. The experimental and simulated data sets used in this chapter are generated using the methodologies described in Chapter 2 and their distribution is outlined in Tables 3 & 5. For machine learning purposes the data sets are split into a further four categories:

Simulated, training: 85% (14,343) of simulated data used as 'source' data to iteratively update the weights and biases of the network.

Simulated, validation: 15% (2,532) of simulated data used to qualitatively ensure the network is not overfitting to the training set.

Experimental, training: 3% to 49% (54-729) of experimental data used as 'target' in the DA methods to iteratively update the weights and biases of the network. The size of this set varies to investigate the effect on network accuracy.

Experimental, testing: 51% (756) of experimental data used to measure the sizing accuracy of the resulting network on previously unseen data.

The split of data used for testing is fixed for all methods, meaning that this data is never used by any method during the training stage. As this work is motivated by creating an accurate sizing network with a minimum amount of NDE samples, the effect of the amount of experimental training data is explored. This requires a way of systematically increasing the size of the experimental training set in a way that optimally covers the parameter space. To achieve this, the 5x6 parameter space of L and θ is considered as a Cartesian grid of potential data points with axes normalised to span the range [0,1]. The first training point is added at (1,1). Additional training data points are progressively added to the vacant sites in the grid, with each new training data point added at the vacant site that has the maximum Euclidean distance to the nearest existing training data point in the normalised axes. This method is referred to as 'uniform

sampling' in this chapter. The resulting sampling regime is given in Table 8, where Tr_i relates to i_{th} point added. The remaining 15 points are used as the test set. This method has the added benefit of ensuring that all data relating to any given defect is placed in either the training or test set and cannot be spread across both. Because of this, any test set accuracy gained from the DA methods should generalise across the $\{L, \theta\}$ space and is not due to parameters covered by the experimental training data. The simulated training/validation split is achieved by drawing samples randomly in P, L, θ space.

Table 8. Experimental training/testing data distribution for Chapter 4

The experimental test set contains all of the L/θ combinations marked 'Test' while the experimental training set a variable number of those marked 'Tri'.

		Crack Length, L (mm)				
		1	2	3	4	5
Crack Angle, θ (°)	0	**Tr_2**	Test	**Tr_6**	Test	**Tr_3**
	±2	Test	**Tr_{11}**	Test	**Tr_{10}**	Test
	±5	Test	Test	Test	Test	**Tr_7**
	±8	**Tr_8**	**Tr_{14}**	**Tr_5**	**Tr_{15}**	Test
	±15	Test	**Tr_{12}**	Test	**Tr_{13}**	Test
	±20	**Tr_4**	Test	**Tr_9**	Test	**Tr_1**

	Range	*Step*	*Count*
Crack Position, P (mm)	13 to 21	0.3	27

All Training = $N_{\theta,L} \times N_P$ = 27×27 = **729 image sets**

Testing = $N_{\theta,L} \times N_P$ = 28×27 = **756 image sets**

4.4. Network architecture

The CNN architecture that was designed for this data set and presented in Chapter 3 is also used in this chapter. Small architecture changes have been made between the two separate networks defined in Section 3.2.2.1 (that predicted L and θ individually), and the current chapter, where only a single network is required to predict D. The single network used here, as illustrated in Fig. 10a, matches the structure of the L prediction network in Fig. 5a, other than an increase in dropout rate from 0.1 to 0.3 which results in ~4% better prediction accuracy on the validation set at the cost of needing ~200 more epochs to reach convergence. The Adversarial DA method requires an additional domain classifier network, which is illustrated in Fig. 10b and comprises a single hidden layer of 128 neurons. This design was also obtained by adding layers until accuracy improvement was minimal. The purpose of the domain classifier is explained further in Section 4.5.5.

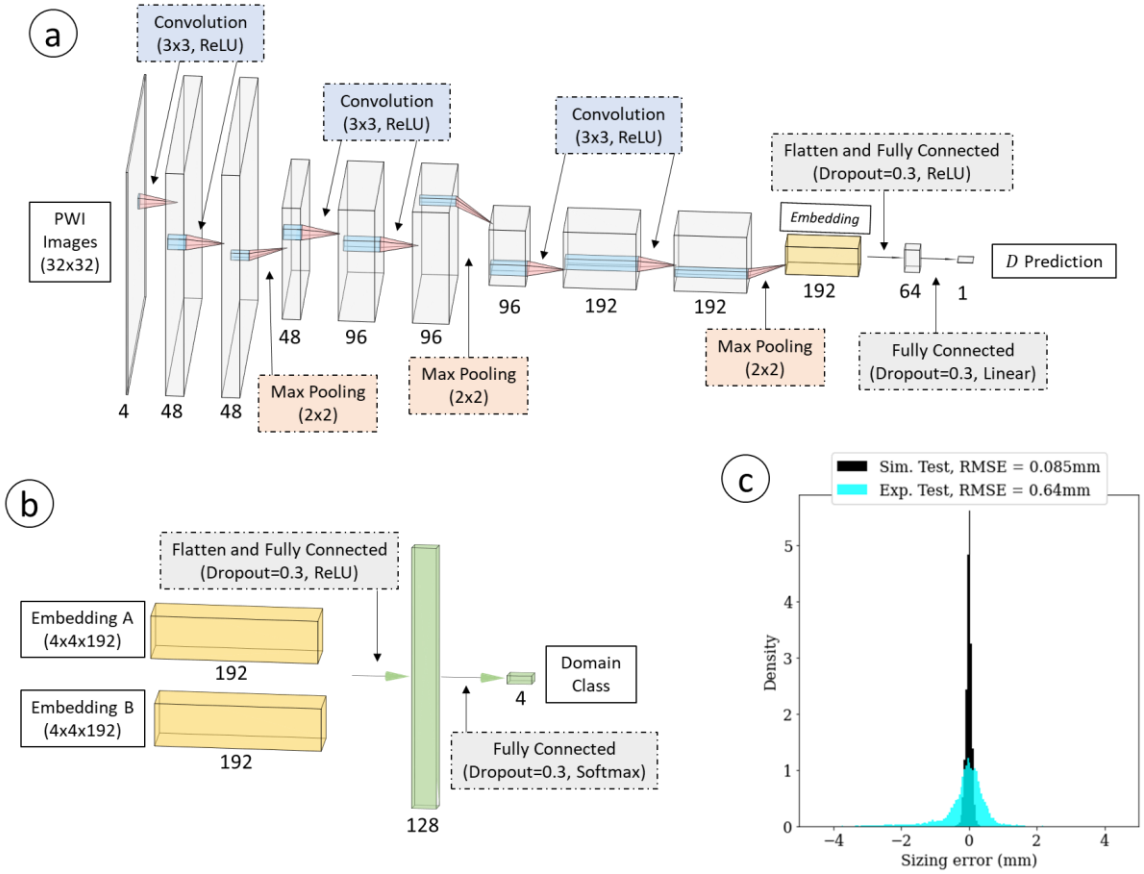


Fig. 10. Illustrations of a) the sizing CNN and b) the domain classifier used by Adversarial and c) the aggregated sizing error results for 20 initialisations of SimOnly applied to the experimental and simulated test sets.

Training the sizing network with all methods presented in this chapter is achieved using the state-of-the-art Adam optimiser [53]. A learning rate of 1×10^{-3} is used unless otherwise stated in Section 4.5. This value is used as increasing it created instabilities during training and decreasing it did not improve the performance of the converged network. A mini-batch size of 64 is used unless the training set size is less than 64, in which case the entire set is processed at once. This is the case for ExpOnly, RCSA and Adversarial when the experimental training set contains only one defect. The number of epochs the network is trained for varies for each method and has been set to ensure convergence of the validation loss. Experimental and simulated D sizing errors for this network, trained only with simulated data, are illustrated in Fig. 10c. While a RMSE of 0.64 mm is significantly better than the performance of 6 dB drop on this data set (as presented in Section 3.4.2), the ~7.5 times lower simulated RMSE again motivates the need for DA when training using primarily simulated data.

4.5. Domain adaptation and baseline methods

This section describes the two baseline cases and three DA methodologies compared in this chapter. As described in Section 4.2, these are the only DA methods found in the literature that are specifically designed for the supervised setting. The baseline approaches train the CNN on the available data sets in isolation while the DA methodologies make use of both sets. The simplest of the DA approaches

(MixedSet) trains on a mixture of the weighted experimental and simulated data while RCSA and Adversarial use different mechanisms to find an embedding space where the distributions of the data sets appear similar. As the output of the convolutional layers is the best approximation to the 'features' that the network is using to determine its final output [82], [83], this is selected as the embedding space.

4.5.1. Simulated data only (SimOnly)

This method makes no use of experimental data, except for testing. The network is trained with only the simulated training set, for 600 epochs, using Mean Square Error (MSE) as the loss function ($\mathcal{L}_R^S$).

4.5.2. Experimental data only (ExpOnly)

This method makes no use of simulated data, training the network with only the experimental training set, for 600 epochs, using MSE as the loss function ($\mathcal{L}_R^e$).

4.5.3. Mixture of experimental and simulated data (MixedSet)

The training set for MixedSet is formed by shuffling together the M experimental and N simulated training image sets. The experimental data's contribution to the loss function is weighted by $\frac{N+M}{2M}$ and the simulated by $\frac{N+M}{2N}$ to ensure the large size of the simulated set does not swamp the effect of the experimental data [84]. The sizing network is trained on the combined set, using MSE as the loss function ($\mathcal{L}_R^{e,S}$), for 600 epochs

4.5.4. Regression and contrastive semantic alignment (RCSA) [77]

RCSA combines the standard 'Regression' loss (MSE in this thesis) with a 'Contrastive Semantic Alignment' loss that aims to force data with the same label (equivalent to the value of D in this thesis) to be close in the embedding space, regardless of the domain. If this is achieved effectively it ensures that the features used by the fully connected layers to predict D, are domain independent. This means prediction accuracy learnt from simulated data should generalise well to experimental data, even if the particular $\{L, \theta\}$ combination tested was not present in the experimental training set.

RCSA functions by training a pair of networks with shared weights, one of which takes source domain data and the other target domain data. The distance metric used to define nearness in the embedding space must be selected. For this chapter this has been set as the mean L_1 distance as lower orders of L_n caused instabilities in training and higher orders produced worse results. The Contrastive Semantic Alignment (CSA) loss was originally presented in [77] for classification of data with discrete labels where it is logical to cluster the same-label data into groups. Because of this, the CSA loss is formulated in [77] by penalizing distance between samples with the same label and rewarding distance between samples with different labels. To facilitate regression, it is more logical to have embedding distance be proportional to label difference. To this end the loss has been reformulated in this chapter to encourage

the distance between samples in the embedding space to scale with absolute difference in D. The L_1 norm is chosen to define the embedding space distance as it usually performs better than higher order norms for high-dimensional data [85]. The new CSA loss ($\mathcal{L}_{CSA}$) is therefore described by

$$\mathcal{L}_{CSA} = \frac{1}{N}\sum_{i=1}^{N}\left\{|D_i^s - D_i^e| - \frac{\sum_{j=1}^{\kappa}|E_{i,j}^s - E_{i,j}^e|}{\kappa}\right\} \tag{10}$$

where N is the size of the training set, D_i^s and D_i^e the simulated and experimental crack depths of the i_{th} image set, E_i^s and E_i^e the simulated and experimental embedding activations and κ the dimensionality of the embedding ($\kappa = 4 \times 4 \times 192 = 3072$ in this chapter). The full RCSA loss ($\mathcal{L}_{RCSA}$) is given by

$$\mathcal{L}_{RCSA} = \mathcal{L}_R^s + \mathcal{L}_R^e + \alpha_{CSA}\mathcal{L}_{CSA} \tag{11}$$

where $\mathcal{L}_R^s$ and $\mathcal{L}_R^e$ are the regression losses (i.e., MSE) for the simulated and experimental data respectively and α_{CSA} is a tunable parameter that adjusts the relative importance of $\mathcal{L}_{CSA}$. The performance of the resulting network was found to be insensitive to the choice of α_{CSA} for the values tested (between 0.05 and 20) so, for simplicity, it is set to 1 in this work.

The training set for this method is formed by randomly pairing the experimental data with a sample of the simulated data meaning that one epoch contains iterations equal to the size of the experimental training set. Both the pairings and the simulated data chosen are shuffled every 5 epochs to stop the network overfitting to any particular combination/subset. Training instabilities due to this overfitting occurred without implementing shuffling, but the resulting validation set accuracy was found to be insensitive to the choice of the frequency of shuffling provided it was <100 epochs. The network is trained for 5,000 epochs in total. Many more epochs are required to achieve convergence than for SimOnly as each epoch only contains a small subset of the simulated data and an even smaller subset of all possible pairings.

4.5.5. Adversarial domain classifier (Adversarial) [78], [79]

A potentially impactful issue for RCSA is that in very high dimensional space, conventional distance metrics find the distance between most points to be similar; this is a product of the 'curse of dimensionality' [86]. Adversarial DA bypasses the problem of finding a useful distance metric by training a separate neural network which aims to infer the domain of the data from embedding space activations (i.e., a domain discriminator). Once this is achieved, domain independent embeddings are achieved by maximally confusing the domain classifier.

As stated in [79], training a two-class domain discriminator with very little target data is difficult. The task is made easier by distinguishing between four cases: 1) Same label, same domain; 2) different label, same domain; 3) same label, different domain; and 4) different label, different domain). This approach

does not have a natural reformulation for regression as the definition of 'same' and 'different' labels for continuous values is not clear. The equivalent proposed here is to say that if $|D^s - D^e| \leq \delta$ then the labels are the same, where δ is a tolerance that depends on the application and availability of data. Here $\delta = 1$ mm is used, as this is the smallest value that can form 'same label, same domain' cases for the experimental data.

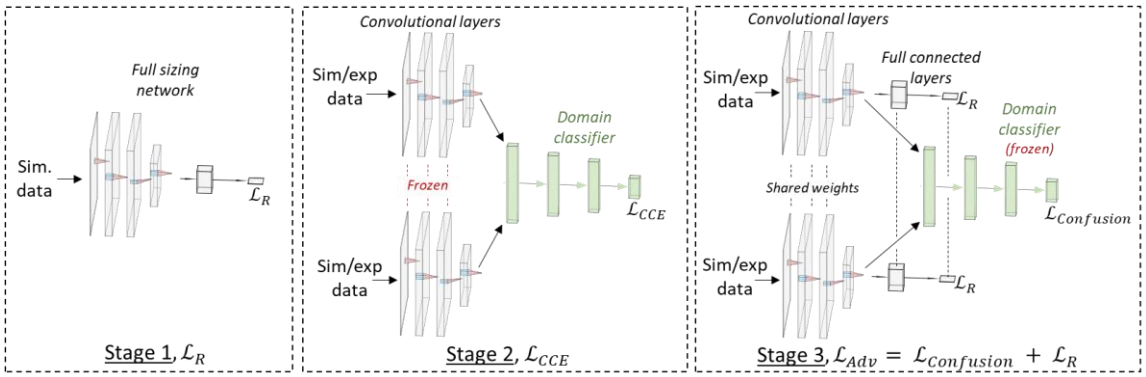


Fig. 11. An illustration of the three stages of training used in the Adversarial DA method.

The training process for Adversarial can be broken into three stages. These are illustrated in Fig. 11 and described in the following:

1. Train the sizing network with only the simulated data, minimizing MSE. As with the baseline methods, this is run for 600 epochs.
2. Form a weight shared pair of the convolutional blocks from stage 1. These convolutional blocks output into a domain classifier to predict which of the four groups a pair of data belong in. The architecture for the classifier is shown in Fig. 10b. This classifier is trained by freezing the weights and biases of the convolutional layers and minimizing the Categorical Cross Entropy ($\mathcal{L}_{CCE}$) which is described by

$$\mathcal{L}_{CCE} = -\frac{1}{N}\sum_{i=1}^{N} y_{i,1}\log\hat{y}_{i,1} + y_{i,2}\log\hat{y}_{i,2} + y_{i,3}\log\hat{y}_{i,3} + y_{i,4}\log\hat{y}_{i,4} \tag{12}$$

   where $y_{i,j}$ is the binary class label for the i_{th} image set and j_{th} class and $\hat{y}_{i,j}$ the output of the domain classifier. This is run for 2400 epochs.
3. Both the convolutional and dense layers of the sizing network are trained whilst confusing the domain classifier with the weights and biases of the domain classifier frozen. The confusion loss ($\mathcal{L}_{Confusion}$)

$$\mathcal{L}_{Confusion} = -\frac{1}{N}\sum_{i=1}^{N} y_{i,1}\log\hat{y}_{i,3} + y_{i,2}\log\hat{y}_{i,4} + y_{i,3}\log\hat{y}_{i,1} + y_{i,4}\log\hat{y}_{i,2} \tag{13}$$

   means that any changes made to the convolutional layers must maintain the domain classifiers label prediction accuracy whilst decreasing its domain prediction accuracy. The full Adversarial loss ($\mathcal{L}_{Adv}$) is a trade-off between accurate sizing and domain independent embeddings and is defined by

$$\mathcal{L}_{Adv} = \mathcal{L}_R^s + \mathcal{L}_R^e + \alpha_{Confusion}\mathcal{L}_{Confusion} \tag{14}$$

where $\alpha_{Confusion}$ is a tunable parameter that adjusts the relative importance of $\mathcal{L}_{Confusion}$. The performance of the resulting network was found to be insensitive to the choice of $\alpha_{Confusion}$ for the values tested (between 0.05 and 20) so, for simplicity, it is set to 1 in this work. This is run for 2400 epochs.

The training set for stages two and three are formed in a similar fashion to RCSA, with pairs of experimental and simulated data. However, while for RCSA all data pairs are from different domains, for Adversarial some must be from the same domain so after pairing the sets they are shuffled across the domains. As with RCSA this pairing and shuffling is redone each 5 epochs. Learning rate for stage 3 is reduced to 0.2×10^{-3} to avoid gradient 'explosion' instabilities during training.

4.6. Results and discussion

The success of both the baseline and DA methods is measured by the sizing accuracy of the resulting networks on the unseen experimental test set. The mean error, and standard deviation of error (STDE) for varying experimental training set sizes is given in Fig. 12. The graphics at the top of Fig. 12 represent the $\{L, \theta\}$ space covered by the experimental training set. As the final network is affected by the initialisation of the weights and the train/validation shuffles, every point has error bars representing $\pm$ one standard deviation, based on results from 20 initialisations. For SimOnly, RCSA and Adversarial these error bars are shown as variable width lines for visual clarity. SimOnly produces networks with a STDE of 0.63 ± 0.04 mm and a small negative mean of -0.10 ± 0.06 mm, indicating a slight bias towards undersizing. As SimOnly makes no use of experimental training data these results are displayed as a constant grey band across Fig. 12.

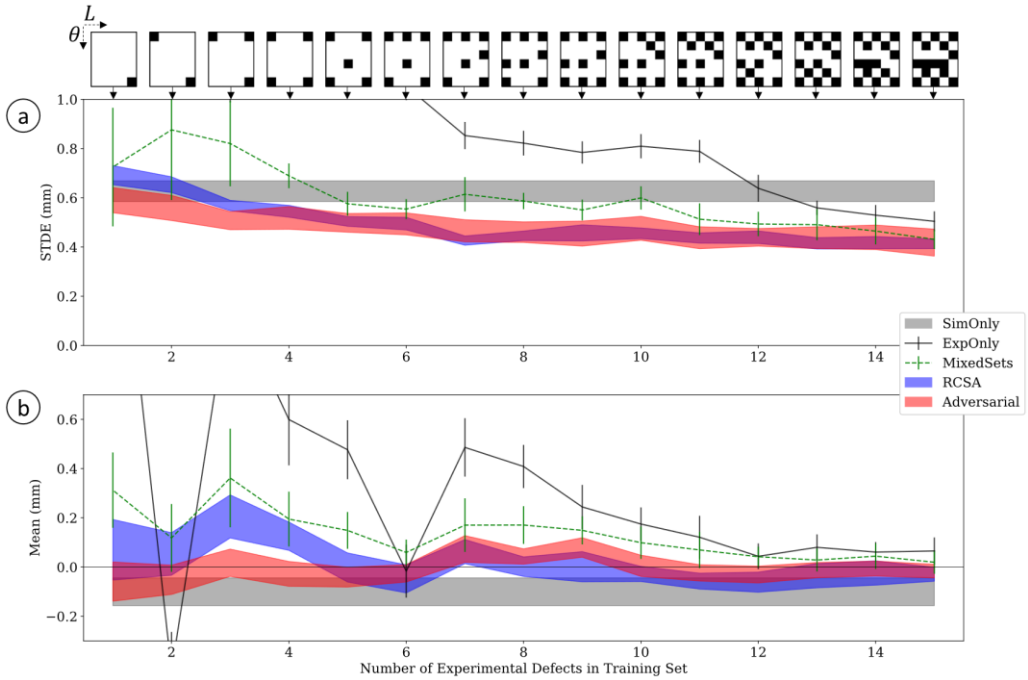


Fig. 12. a) The standard deviation in error (STDE) and b) the mean of the sizing error for the experimental test set across varying sizes of experimental training set. The error bars represent $\pm$ standard deviation over 20 independent initialisations. The graphics above the plots represent the $\{L, \theta\}$ coverage of the experimental training set.

The second baseline method, ExpOnly, is heavily reliant on having a large experimental training set. While it demonstrates greater accuracy than SimOnly with 13 or more defects in the training set, below this point, the STDE and mean increase quickly due to the network overfitting to the small set of training data. Overfitting, rather than more generalised learning, can be demonstrated by considering ExpOnly networks' performance on simulated data across the same $\{L, \theta\}$ space as the experimental test set. When trained with all 15 experimental defects, ExpOnly has a STDE of 1.02 mm on simulated data, whereas the STDE of SimOnly on the experimental test set is 0.65 mm. This asymmetry shows that while SimOnly can generalise reasonably well across the domain shift from simulated to experimental data, ExpOnly cannot do the reverse, and as a result, is unlikely to generalise well to even minor changes in inspection conditions (e.g., slight array movement, sound speed changes or crack roughness). This overfitting is likely caused by the significantly smaller training set available to ExpOnly compared to SimOnly.

MixedSet outperforms both baseline methods with 5 or more defects in the experimental training set, but still suffers from inaccuracies due to overfitting when experimental data is scarce. The two other DA methods are given the same training data as MixedSet but perform better at all points. In terms of STDE, RCSA performs slightly worse than SimOnly with only one experimental training defect but at every other point outperforms both baseline methods and MixedSet. Adversarial gives the lowest STDE of all methods with 5 or less experimental training defects and has similar performance to RCSA above this point. The absolute mean for RCSA and Adversarial is negligible in most cases, becoming slightly larger for RCSA with low numbers of experimental training defects. This is likely due to uneven coverage of the $\{L, \theta\}$ parameter space.

It is clear from Fig. 12 that the two DA methods: RCSA and Adversarial, make better use of limited experimental data than MixedSet. This can be explained by their differing objectives. Rather than aiming for accurate experimental sizing directly, which is difficult with limited data, RCSA and Adversarial focus on extracting domain independent embeddings. This is an easier task to achieve with limited data. Also, if domain invariant embeddings are found between the $\{L, \theta\}$ examples in the experimental training set and the full simulated training set they are likely to generalise to all $\{L, \theta\}$ of interest as these are all present in the simulated training set.

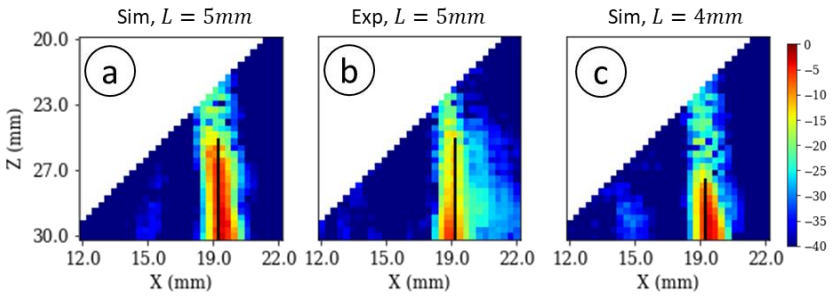


Fig. 13. a) Simulated and b) experimental SS-S PWI images for a defect with $P = 19mm, L = 5mm$ and $\theta = 0°$, and c) a simulated SS-S PWI image for a defect with with $P = 19mm, L = 4mm$ and $\theta = 0°$.

The negative mean for SimOnly is caused by undersizing of 5 mm defects. This is because the far-field assumption of the simulation is inaccurate for defects larger than 4 mm as their tips enter the array's near-field. This inaccuracy is exampled in Fig. 13a where it can be seen that the simulation overestimates the amplitude of the tip reflection in comparison to the experimental data in Fig. 13b. SimOnly sizes the PWI data from the $D = 5$ mm defect shown in Fig. 13b to be of $\widehat{D} = 4.4$ mm which makes intuitive sense as, visually, the image appears closer to the simulated $D = 4$ mm defect in Fig. 13c (which SimOnly sizes as $\widehat{D} = 4.0$ mm) than the simulated $D = 5$ mm defect in Fig. 13a. This kind of simulation deficiency is a good example of the need for DA.

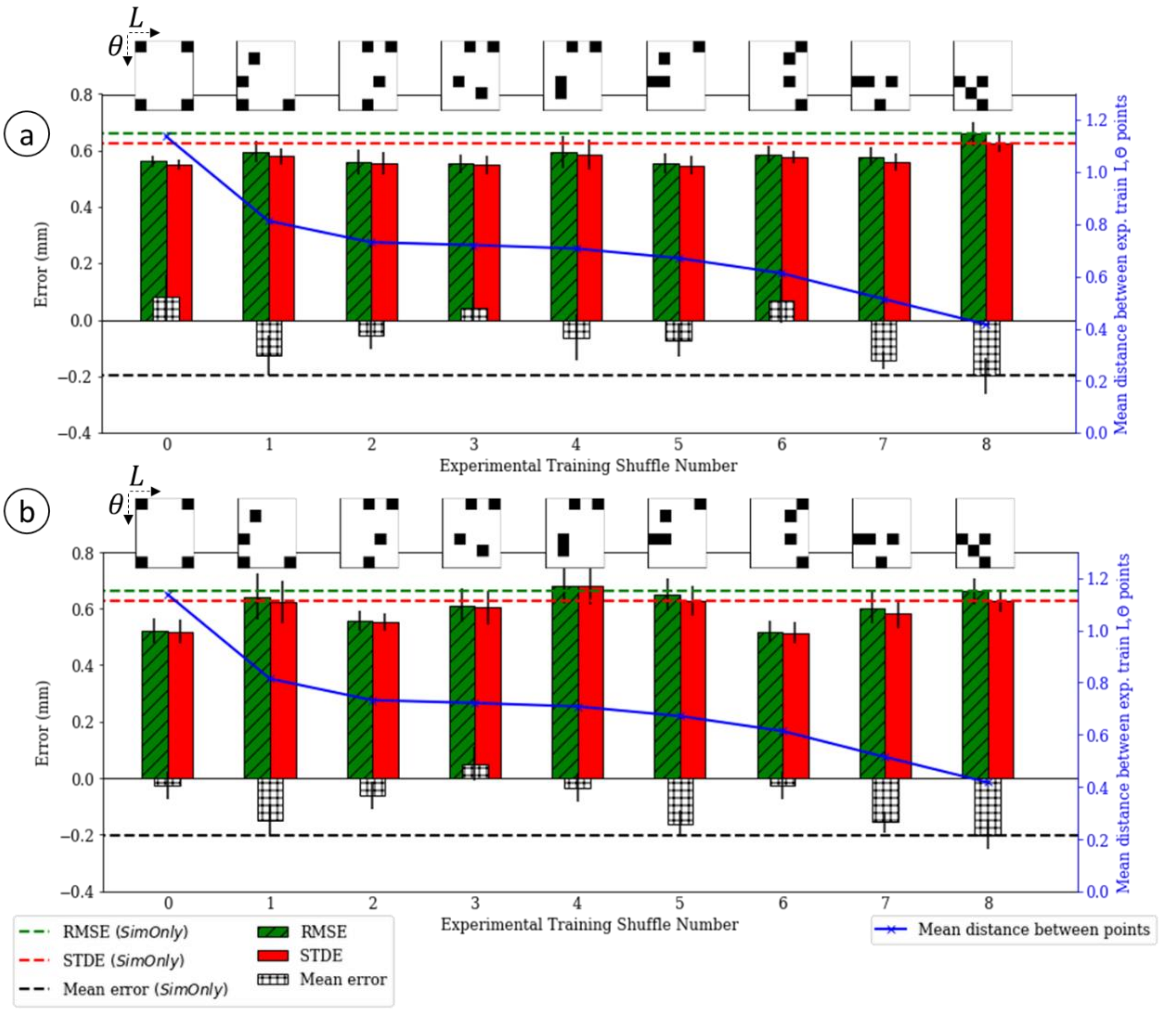


Fig. 14. The Root Mean Square Error (RMSE), mean error (μ), standard deviation of error (STDE) and mean Euclidean distance between points in normalised $\{L, \theta\}$ space for a) RCSA and b) Adversarial methods with varying choices of experimental training set. The graphics above the plots represent the $\{L, \theta\}$ coverage of the four experimental defects.

The effect of the position of the training data points in $\{L, \theta\}$ space is investigated by using RCSA and Adversarial with four experimental training defects but rather than using uniform sampling to optimally choose the $\{L, \theta\}$ combination, they are picked at random. The mean Euclidean distance between the training set examples in terms of normalised $\{L, \theta\}$ is used as an indication of how well sampled the

parameter space is. The random selection of the training data points is repeated 8 times with different random number generator seeds (training shuffle number = 1-8). The results of this experiment are shown in Fig. 14. Root Mean Squared Error (RMSE) is reported alongside STDE and mean error to provide a single metric with which sizing accuracy can be easily compared across shuffle numbers.

In Fig. 14, the results are presented in order of decreasing mean distance between points in $\{L, \theta\}$ space as a measure of parameter space coverage. A reduction in parameter space coverage might intuitively be expected to lead to increased RMSE, however, as shown in Fig. 14a, the prediction accuracy of RCSA does not change significantly across the training shuffle cases. Adversarial shows some variation across shuffle cases (Fig. 14b), but this effect is not correlated to parameter space coverage. However, both methods produce the lowest errors in the uniformly sampled case (training shuffle number = 0), compared to other possibilities and the most poorly sampled case (training shuffle number = 8) offered no accuracy increase over SimOnly. This demonstrates the importance of sampling the defect's parameter space as evenly as possible with the available experimental training data.

4.7. Conclusions

This chapter has demonstrated the ability of modern DA methods to improve the accuracy of deep networks for defect sizing, trained on simulated data, with even a very limited amount of experimental data. The key metrics for comparison of the methods considered are illustrated in Fig. 15. Adversarial and RCSA produced the most accurate networks for all sizes of experimental training set with Adversarial outperforming RCSA with less than 6 experimental training defects. With only 4 experimental defects RCSA and Adversarial reduced STDE on the experimental test set by 13% and 17% respectively, compared to SimOnly. However, RCSA is the easier method to implement as it only introduces one extra tuneable parameter (loss function scaling factor, α_{CSA}) while Adversarial requires tuning of $\alpha_{Confusion}$, design of the architecture for the domain classifier, and takes almost $\sim$10 times longer to train than RCSA when the experimental training set is small. The success of both modern DA methods was shown to be sensitive to coverage of the $\{L, \theta\}$ parameter space by the experimental training set. The results of this chapter suggest that uniform sampling, starting at the corners of the parameter space, is an effective way of designing a small experimental training set. Optimal sampling for higher dimensional parameter spaces needs further investigation.

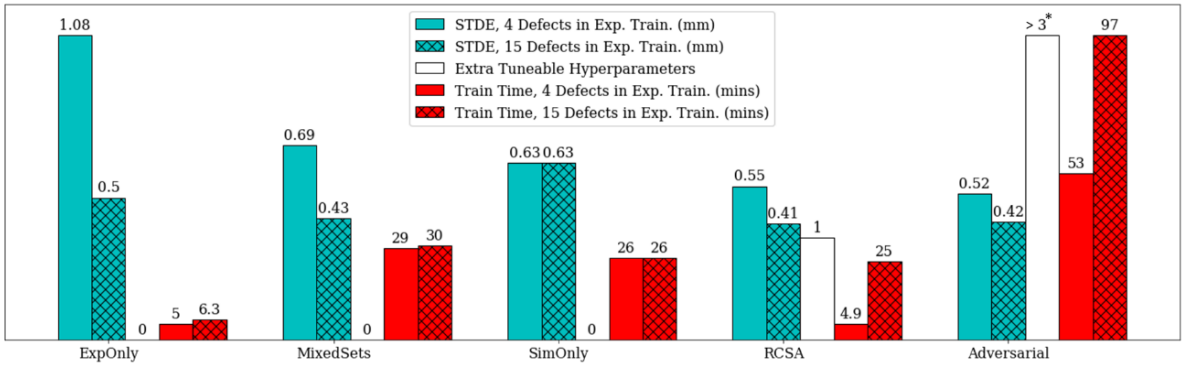


Fig. 15. Summary of the key properties of the methods investigated in this chapter.
*This includes full design of the domain classifier architecture.

Future research should be carried out to investigate the impact of larger gaps in the distributions of source and target domains. For example, testing if ultrasonic data from a different inspection or even natural images would be useful source domains. The possibility for using NDE specific data augmentation alongside the DA methods presented here to further increase the usefulness of small pools of experimental data should also be investigated. Another major improvement would be to use probabilistic methods to add values of uncertainty to the predictions of the deep learning network. The modern DA methods presented in this chapter are shown to be successful for improving the accuracy of deep learning for in-line pipe inspection and, as they are agnostic to the structure of the data, are expected to be applicable to other NDE inspections and modalities.

Chapter 5. Uncertainty quantification

This chapter investigates the effectiveness of two modern uncertainty quantification (UQ) methods: Monte Carlo (MC) dropout and deep ensembles. Success is judged by the techniques' ability to predict the expected sizing error of a CNN applied to surface breaking defects as well as detect anomalous inputs. The content in this chapter is drawn from the author's published work [87].

5.1. Introduction

Due to the safety-critical nature of NDE, UQ is an essential part of inspection qualification [88] and decision making, for any automated data analysis. This is because undersizing of defects can result in unexpected part failures, causing damage to structures and/or people. Effective UQ can signal to the operator when there is high uncertainty in the defect size prediction so the data can be referred to a human for further analysis and possibly the use of additional NDE measurements. This chapter focusses on how to quantify uncertainty for deep learning in the context of crack sizing in ultrasonic inline pipe inspection. Automatic defect detection occurs online, and in this thesis is assumed to have already been performed, hence the task is to characterise and size a defect given data that contains an indication of a defect. Defect sizing occurs offline and is traditionally carried out by skilled human operators. In this chapter deep learning is applied to the defect characterisation and sizing task with the aim of investigating how the uncertainty of that operation can be assessed

Evaluating the success of UQ methods is challenging as there is no 'ground-truth' for uncertainty. This chapter uses two criteria to analyse the success of the UQ methods. The first is for the UQ method to be 'well calibrated' [89]. For regression tasks, such as the one considered in this thesis, this means that predicted uncertainty is equal to (or at least proportional to) the expected error (i.e., the difference between the crack depth predicted by the network and the true crack depth). This is tested using both a simulated and experimental test set of surface breaking cracks. The second metric is the predicted uncertainty for out-of-distribution (OOD) data, testing if the network 'knows what it knows.' As the network is trained on surface-breaking cracks, OOD data from experimental embedded side-drilled holes (SDHs) and simulated embedded cracks are used for this purpose. The OOD data set ($N_{OOD} = 76$) contains examples of defects not included in the training data and therefore an effective UQ method should assign them high uncertainty

In practice, as in this chapter, UQ typically produces a single metric, e.g., standard deviation of the probability density function, $P(\hat{y}|\hat{x}, T)$, where $\hat{x}$, $\hat{y}$ are the network's input and output for test data and T is the input and output training data. The methods described in this chapter achieve UQ by sampling from the space of all possible trained networks (parameterised by their weights, W) and taking the standard deviation of their predictions as an estimate of uncertainty. In more rigorous terms, all UQ methods function by approximating the intractable posterior distribution of weights given the labelled

training data, $P(W|T)$, with which inference on the uncertainty associated with new test data, $P(\hat{y}|\hat{x}, T)$, can be calculated. The two most common modern methods for estimating the uncertainty of the CNN's predictions are investigated for this chapter: deep ensembles (DE) [90] and Monte Carlo (MC) dropout [91]. The intuition for these approaches to posterior approximation is that if the sampled networks are sufficiently diverse, they should produce diverse predictions for inputs far from the training data, indicating high uncertainty. DE achieves this by training multiple networks from different initializations, while MC dropout produces predictions by using dropout (traditionally used at train time to reduce overfitting [92]) at test time.

The structure of the rest of this chapter is as follows. Relevant literature is discussed in Section 5.2, datasets and associated sources of uncertainty are described in Section 5.3, the network architecture used is illustrated in Section 5.4, the UQ methods presented in this chapter are outlined in Section 5.5, results are presented in Section 5.6, methods for efficient use of computational resources are discussed in Section 5.7 and conclusions are given in Section 5.8.

5.2. Relevant literature

UQ is a relatively new and active area of research in deep learning [93]. Because of this, there are few applications to NDE in the literature. The only examples of UQ for deep learning in NDE found are the following: MC dropout used to estimate uncertainty for defect detection in a heat exchanger with eddy-current measurements [94] as well as for defect categorization and localization in visual inspection of bridges [95]. A mixture density network [96] has been used to estimate aleatoric uncertainty for guided-wave-based defect localization in simulated data of structural plates [97]. Deep ensembles have been used to increase the accuracy of deep learnt predictions in NDE [98]–[100], but there has been little investigation into leveraging their ability to quantify uncertainty.

While this chapter focusses primarily on DE and MC dropout, two other commonly used UQ methods were also investigated: a CNN/Gaussian process (CNN-GP) hybrid [101], [102], and variational inference (VI) [103], [104]. These methods take a more 'Bayesian' rather than 'Frequentist' approach to approximation of the posterior. CNN-GP makes use of the natural probabilistic inference of the Gaussian process combined with the expressive powers of convolutional layers. Following the implementation described in [102], the fully connected layers of a CNN were replaced with a sparse Gaussian process approximation based on variational inducing points [105] for the current application. This method was found to produce no correlation between uncertainty and magnitude of error on the experimental test set. VI approximates the posterior by casting it as an optimization problem: reducing the Kullback-Leibler divergence [106] between the true posterior and that produced by the network. For the application described in the current chapter, VI was implemented using a reparameterization estimator [107]. However, VI proved to be unstable in training and converged either to a network predicting the mean of the training set or one with poor predictive accuracy (sizing defects with a root

mean square error ≈ 0.4 times their true length). There have also been recent publications that question the quality of VI's posterior approximation [45]–[47]. As these methods require a lot of hyperparameter tuning and, despite this, were found to produce poor UQ, they are not investigated further in this chapter.

5.3. Data sets

This chapter focusses mainly on quantifying uncertainty for sizing surface-breaking cracks but data from other defects is also tested to analyse the predicted uncertainty for OOD defects. All of the data used in this chapter and their main sources of uncertainty are described in this section.

5.3.1. Surface breaking cracks

The main experimental and simulated data sets used in this chapter are generated using the methodologies described in Chapter 2 and their distribution is outlined in Tables 3 & 5. For machine learning purposes the data sets are split into a further four categories:

Simulated, training: 85% (14,343) of simulated data used to iteratively update the weights and biases of the network.

Simulated, validation: 7.5% (1,266) of simulated data used during research and design stages to qualitatively ensure the network is not overfitting to the training set.

Simulated, testing: 7.5% (1,266) of simulated data used to test the calibration of UQ on previously unseen data.

Experimental, validation: 15% (216) of experimental data used during research and design stages to ensure the network is not overfitting to the simulated data and to implement the training stop condition.

Experimental, testing: 85% (1,269) of experimental data used to test the network's sizing accuracy and calibration of UQ on previously unseen data.

The training/validation/testing split for simulated data is drawn randomly, from a uniform distribution, across all image sets (i.e., across all $\{L, \theta, P\}$), but, as shown in Table 9, the experimental validation/testing split is drawn randomly in $\{L, \theta\}$ space. This is to guarantee that no data from the same physical defect is split across sets, ensuring test set performance generalises past the L, θ combinations used to implement the stop condition. The aim of these surface breaking defect test sets is to analyse the calibration between uncertainty and D prediction error.

Table 9. Experimental testing/validation data distribution for Chapter 5

The experimental test set contains all of the L/θ combinations marked 'Test' while the experimental validation set all those marked 'Val.'

		Crack Length, L (mm)				
		1	2	3	4	5
Crack Angle, θ (°)	0	Test	Test	Test	Test	Test
	±2	Test	**Val**	Test	Test	Test
	±5	**Val**	Test	Test	Test	Test
	±8	Test	Test	Test	**Val**	Test
	±15	Test	Test	Test	Test	Test
	±20	Test	Test	**Val**	Test	Test

	Range	*Step*	*Count*
Crack Position, P (mm)	13 to 21	0.3	27

Validation = $N_{\theta,L} \times N_P$ = 8×27 = **216 image sets**

Test = $N_{\theta,L} \times N_P$ = 47×27 = **1269 image sets**

5.3.2. Defects outside of training set

To test whether the UQ methods can detect data drawn from distributions significantly different to the training set, defect types not included in the training set are tested. As exampled in Fig. 16, this group of data includes two experimental side drilled holes (SDHs) and two simulated embedded (rather than surface-breaking) cracks. This data is gathered using the same experimental and simulation procedures as described in Chapter 2. These four defect classes are imaged at 14 X-locations, equally spaced across the same range of horizontal positions as the surface breaking cracks (13 mm $\leq P \leq$ 21 mm). 20 examples of experimental defect free data are also tested, forming a total of $N_{OOD} = 4 \times 14 + 20 = 76$ image sets.

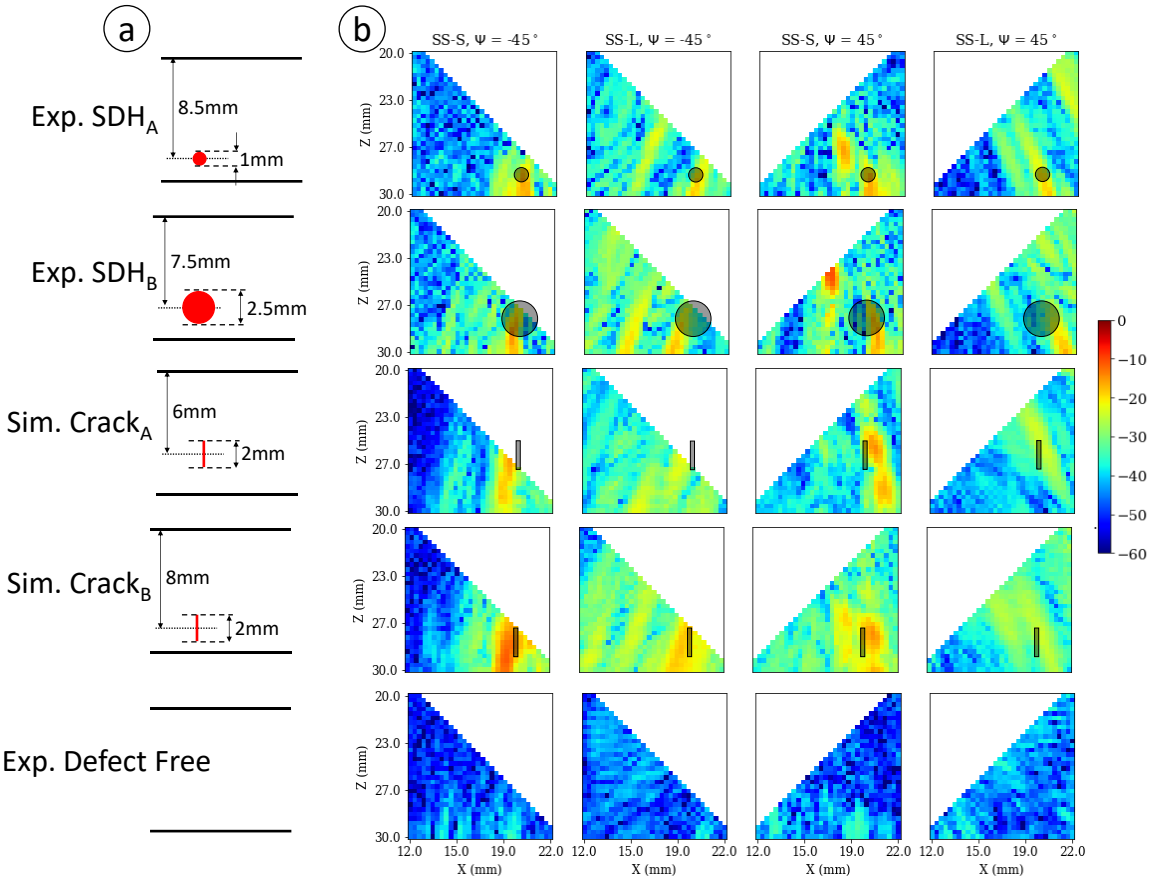


Fig. 16. a) Diagrams and b) sets of example PWI images of defects outside of the training set. The black circles and rectangles in b) show the true size and placement of the defects. All images are on the same dB colour scale, normalised to the maximum intensity in the experimental set.

5.3.3. Sources of uncertainty

Sources of uncertainty can broadly be broken down into two categories; aleatoric and epistemic. Aleatoric or 'data' uncertainty stems from noise inherent to the data generation process and cannot be reduced by adding training data. Epistemic uncertainty is caused by ignorance in how the data is generated, creating uncertainty in the network's parameters, and can be minimised by adding appropriate training data as long as the training data chosen matches the test data distribution well. It should be highlighted that if there is a significant domain shift between training and test domains (e.g., when using a numerical simulation to approximate reality) adding training data can never fully minimise epistemic uncertainty.

In sizing defects from PWI images the two main sources of aleatoric uncertainty are noise and poor correlation between indication and defect size. Noise is caused by reflections from grains and structural features (such as front and back walls), as well as "artifacts" at locations away from the defect, due to ray paths other than the one being imaged. Poor angular coverage of a defect from incident and received ray paths blurs indications in images but as PIGs for inline pipe inspection travel at ~2 m/s, capturing data every 1-10 mm, there is too little time to remedy this by firing more than ~3 plane waves per array, per location. However, aleatoric uncertainty is deemed to be negligible in comparison to epistemic

uncertainty for this application. This is due to both sources of aleatoric uncertainty being relatively small. Firstly, the data has a large signal to noise ratio (SNR) of ~30 dB. Secondly, as shown in Chapter 3, classical sizing methods (such as 6 dB drop) suffer due to the weak link between indication size and defect length, however, a CNN can make predictions on more complex features, reducing the need for good angular coverage. If aleatoric uncertainty is not constant across different input samples (i.e., heteroscedastic) it can be estimated by using negative log likelihood as the loss function [108] but this was found to predict values of ~3% of the total uncertainty, supporting the hypothesis of low aleatoric uncertainty. For simplicity, mean squared error (MSE) is used as the loss function in this chapter, omitting aleatoric uncertainty from the UQ.

Epistemic uncertainty is the main cause of errors in this application. This is evidenced by the gap in simulated (RMSE = 0.095 mm) and experimental (RMSE = 0.63 mm) test set sizing accuracy of a CNN trained on simulated data. This performance discrepancy is caused by inaccuracies in the simulation such as those given in Table 10. Epistemic uncertainty could be reduced by adding experimental data to the training set or using a more accurate simulation. However, these approaches are financially or computationally expensive respectively.

Table 10. Example sources of epistemic uncertainty for the application in this thesis.

Variations in inspection conditions	Defect geometry simplifications	Wave modelling simplifications
Array mispositioning	Defects modelled as rectangular, perfect reflectors while test set defects have some roughness and rounded tips	Ray paths with more than three legs
Sound speed variation	Surface roughness not modelled	Surface waves
Inconsistency in array element performance	Array assumed to be in far-field of defect in model, but array is partially in defect near field for $L >= 4$ mm	Non-linear effects

5.4. Network architecture

The CNN architecture that was designed for this data set and presented in Section 3.2.2 is also used in this chapter. There are three minor architecture changes from Chapter 3 to this chapter. Firstly, only a single network is needed to predict D. This matches the structure of the L network in Fig. 5a. Secondly, dropout is increased to 0.3, which resulted in slightly better experimental validation set accuracy at the cost of needing ~50 more epochs to converge. Thirdly, when using the DE-ResSpec UQ method (described in Section **Error! Reference source not found.**), residual connections are added. The resulting network is illustrated in Fig. 17 and residual connections are further described in Section **Error! Reference source not found.**.

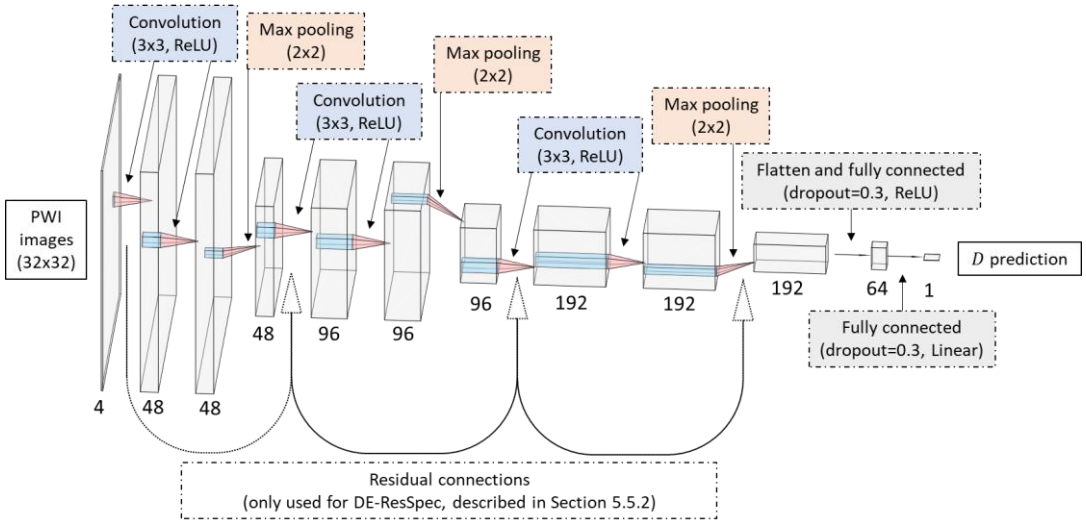


Fig. 17. An illustration of the CNN architecture used in this chapter.

5.5. Uncertainty quantification methods

To achieve UQ the posterior distribution over the network's weights and biases (W) must be calculated or approximated. Using Bayes' theorem this can be written as

$$P(W|T) = \frac{likelihood \times prior}{evidence} = \frac{P(T|W)P(W)}{P(T)} = \frac{P(T|W)P(W)}{\int P(T|W)P(W)dW} \tag{15}$$

where T is the training data inputs and outputs. With this, inference for a given input $\hat{x}$ can be calculated by

$$P(\hat{y}|\hat{x},T) = \int P(\hat{y}|\hat{x},W)P(W|T)dW \tag{16}$$

where $\hat{y}$ is the predicted output. However, the posterior is computationally intractable due to the difficulty of evaluating the normalization constant, $P(T) = \int P(T|W)P(W)dW$ due to the high dimensionality of both T and W and the fact that the likelihood, $P(T_i|W)$ and the prior, $P(W)$ are 'nonconjugate' i.e., do not take the same form in relation to W [109]. Approximating this distribution as closely as possible to produce accurate inference of the posterior is the aim of the methods presented in this section.

For all methods considered in this chapter the likelihood of the output is considered to be Gaussian

$$P(\hat{y}|\hat{x},W) = \mathcal{N}(\mu,\sigma) \tag{17}$$

where both mean, μ and standard deviation, σ are a function of the network's parameters. Because of this assumption, the UQ methods described in this chapter can be said to be 'well calibrated' if they demonstrate a 1:1 relationship between predicted uncertainty and σ. Other approaches such as mixture density networks (MDNs) can be used to avoid this assumption, but it is commonly used in deep learning UQ literature and is considered sufficient for this application.

5.5.1. Deep ensemble [90]

Ensembling of machine learning networks has long been recognised as a way to improve accuracy [110], [111], but more recently it has also become a popular UQ method, commonly termed 'deep ensembles' (DE) [90]. DE functions by training m networks, usually of the same architecture (as is the case in this chapter), to produce a diverse ensemble of predictors. Diversity in the ensemble can be encouraged by training each member with a subset of the full training set, sampled with replacement, this is commonly called bagging or bootstrapping. However, it has been observed that the randomness in network initialisation is sufficient [90], [112] so bagging is not used.

The ensemble's overall prediction is represented by a mean (μ) and standard deviation (σ) of the individual member's predictions

$$\mu = \frac{1}{m}\sum_{i=1}^{m} y_i \tag{18}$$

$$\sigma = \sqrt{\frac{\sum_{i=1}^{m}(y_i - \mu)^2}{m}} \tag{19}$$

where y_i is the output of the i_{th} member of the ensemble, σ is taken as the measure of uncertainty in all methods presented in this chapter. The intuition for DE as a UQ method is that different members of the ensemble will tend to output similar values when the inputs are similar to the training data, because each network, even if different, is optimised for that data. But when inputs are less alike to the training data, the networks are more affected by the specificities of the sub-optimal solution reached, producing higher variance results. This can be thought of in a 'loss landscape' perspective as members of the ensemble, due to their different initializations, ending up at local minima, that all accurately predict on the training data, but behave diversely on anomalous data [113]. Prediction error for a specific defect is calculated using

$$Error_j = \mu_j - D_j \tag{20}$$

where j is the index of the defect and D_j is true depth. Error over a full test can be summarised by root mean squared error (RMSE)

$$RMSE = \sqrt{\frac{1}{N}\sum_{j=1}^{N}(\mu_j - D_j)^2} \tag{21}$$

where N is the size of the test set.

5.5.2. Deep Ensemble with residual connections [114] and spectral normalization [115]

Neural networks can suffer from an effect called 'feature-collapse' where distances in the input space are not correlated with distances in the feature space [102]. This means that inputs far from the training data may be mapped close to training set features, erroneously assigning them low uncertainty. It has been shown that feature collapse can be avoided by enforcing 'smoothness' and 'sensitivity' [116]. Smoothness means that small input changes cannot cause large output changes, and sensitivity requires input changes to always change the feature space representation. These properties can be described mathematically by bi-Lipshitz continuity

$$K_1\|x_1 - x_2\|_2 \leq \|f(x_1) - f(x_2)\|_2 \leq K_2\|x_1 - x_2\|_2 \tag{22}$$

where K_1 and K_2 are the Lipschitz constants of function $f(x)$, and $\|\,.\,\|_2$ represents the L_2 norm. In this chapter, the feature extractor (convolutional layers) is encouraged to be bi-Lipshitz continuous by spectral normalization [115] and residual connections [114] which create smoothness and sensitivity respectively. Residual networks with spectral normalization have been shown to be 'distance-aware' (i.e., the ability to assess test data's distance from training data distribution) [117] and capture uncertainty effectively [102], [118]. These properties are used to improve the UQ capability of deep ensembles for NDE in this chapter.

Residual connections create a connection between the input, and layers deeper into a neural network. They were originally proposed to ease the optimization of very deep networks [115] but in doing so they also make the network's activations more sensitive to the input, motivating their use in UQ. As shown in Fig. 17, residual connections take information and shortcut the next few layers by summation with their output. This shortcut should be as close to an identity mapping as possible. As the number of filters changes and max pooling reduces image size by 2 in both width and height, a 1x1 convolutional layer with a stride of 2 and no activation function is used for the residual connections in this chapter.

Spectral normalization is equivalent to regularizing the largest singular value of a layer's weight matrix. It has been popularised recently as a way to improve generalization of generative adversarial networks (GANs) [115]. Following [117] and the implementation in [119] the spectral norm, η, is estimated at every training iteration, for every layer, using the power iteration method. Weights are normalised by multiplication with a scaling constant divided by the spectral norm, $\frac{\alpha_{spectral}}{\eta}$. This approach has two hyperparameters, the number of power iterations and the scaling constant ($\alpha_{spectral} > 0$). As in [117], one power iteration was found sufficient so is used here and $\alpha_{spectral}$ was set by a grid search for the smallest value that does not reduce the validation set accuracy of network, this was found to be $\alpha_{spectral} = 1.2$. This method will be referred to as DE-ResSpec from this point onwards.

5.5.3. Monte Carlo dropout [91]

Dropout was originally proposed as a technique for reducing overfitting by setting the output of individual neurons to 0 during training, with probability p, at each iteration [92]. It has later been shown that implementing dropout at both training and test time, before every weight layer, is a close approximation of a deep Gaussian Process [91] and has been termed 'Monte Carlo (MC) dropout'. The intuition for MC dropout as an UQ method is that each initialisation of dropout at test time is acting as a member of an ensemble. As such, μ and σ are calculated using Eqs. (18) & (19) with m equal to the number of dropout initialisations run at test time, $m_{dropout}$. This is set to 200 in this work as μ and σ were found to change negligibly for $m_{dropout}$ larger than this. Dropout probability, p, is set to 0.3 as larger values significantly increased time to convergence, without improving UQ.

Due to its simplicity, MC dropout has been used in a lot of UQ literature [93] but has also received criticism by [113] in which it is shown to produce significantly less diverse predictors in comparison to DE. This is exampled in [120] where a simple single-hidden layer ReLU network with MC dropout fails to produce high uncertainty between clusters of 2D data. However, the same work also shows that deeper (>= 2 hidden layers) neural networks with MC dropout should theoretically approximate the posterior accurately.

5.6. Results

This section presents results relating to the quality of UQ from the methods presented in the previous section.

5.6.1. Number of networks in ensemble

When originally proposed in [90] it is suggested that five networks are sufficient for effective UQ using DE. However, because neither training nor test time computational resources are limited in this application a larger ensemble can be used. To determine the optimal size of the ensemble, the effect of iteratively adding a network to the ensemble was measured in terms of the mean absolute change in uncertainty

$$\Delta_m = \frac{1}{N}\sum_{i=1}^{N}\left|u_{m,i} - u_{m+1,i}\right| \tag{23}$$

where $u_{m,i}$ is the uncertainty for the i_{th} sample of the experimental validation set predicted by an ensemble of m networks and N the size of the data set (216 for experimental validation). As shown in Fig. 18a, Δ_m decreases as m increases, indicating a diminishing effect of increasing ensemble size on UQ. 60 networks are used for DE as $\Delta_{60} \approx 1 \times 10^{-3}\ mm$. This is deemed to be low enough to assume that the ensemble predictions have mostly converged and adding more networks will only minorly change the results.

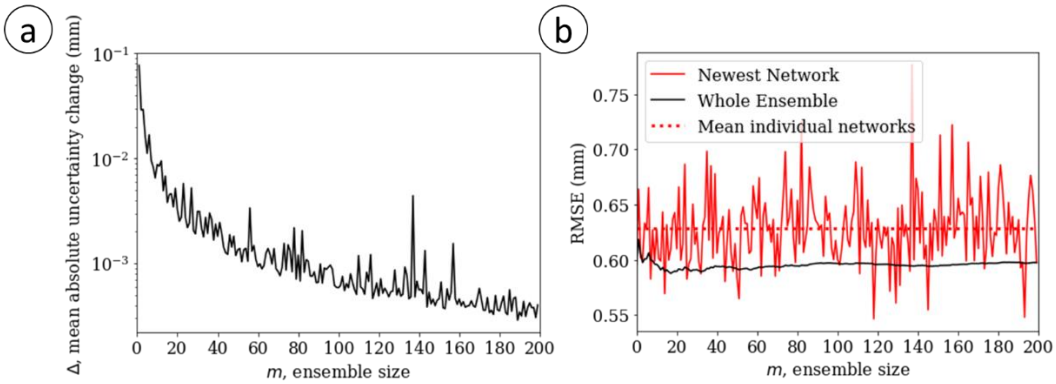


Fig. 18. Mean absolute change in uncertainty of the experimental validation set and b) RMSE of the experimental test set for both the whole ensemble and the newest member for increasing ensemble size.

It should also be noted that while prediction accuracy is not the focus of this chapter, ensembling does provide a slight reduction in defect sizing error. This can be seen in Fig. 18b where the experimental test set RMSE of an ensemble with $m > 10$ (solid black line) is ~0.035 mm lower than the mean RMSE of the 200 networks when used independently (dotted red line).

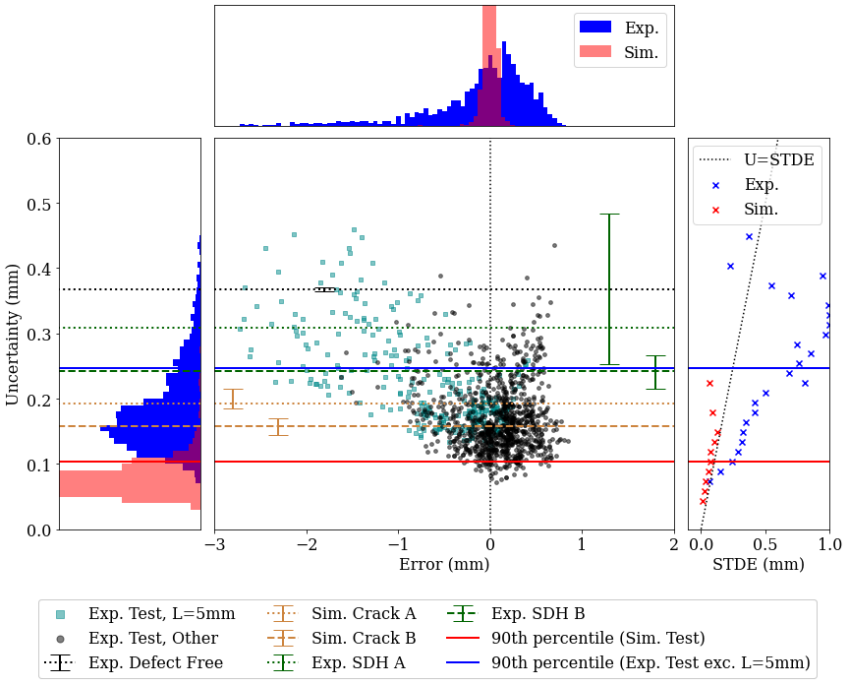


Fig. 19. Deep ensemble (DE) uncertainty predictions for both in and out of distribution test sets. Experimental test set RMSE = 0.592 mm

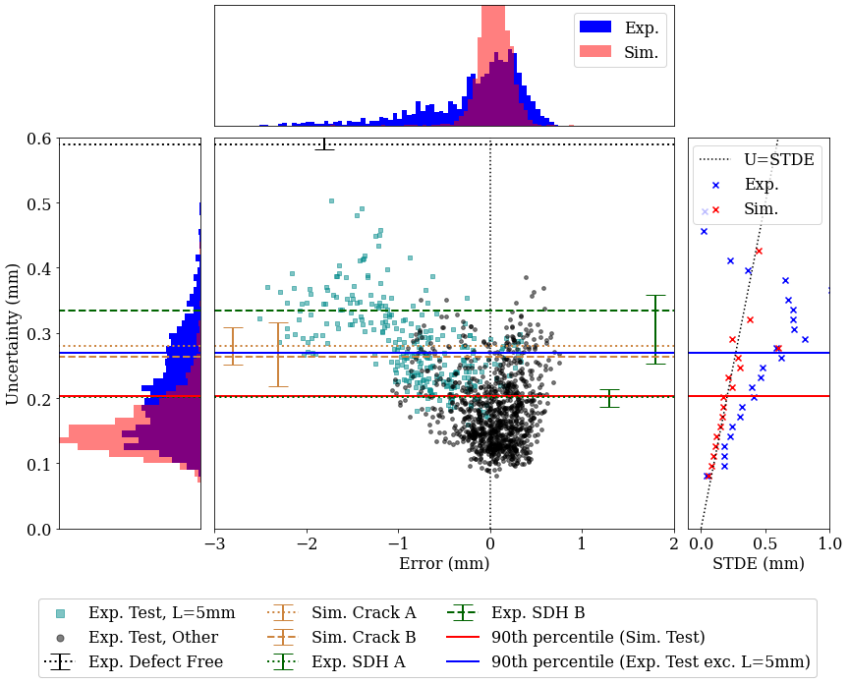


Fig. 20. Deep ensemble with residual connections and spectral normalization (DE-ResSpec) uncertainty predictions for both in and out of distribution test sets. Experimental test set RMSE = 0.5831 mm.

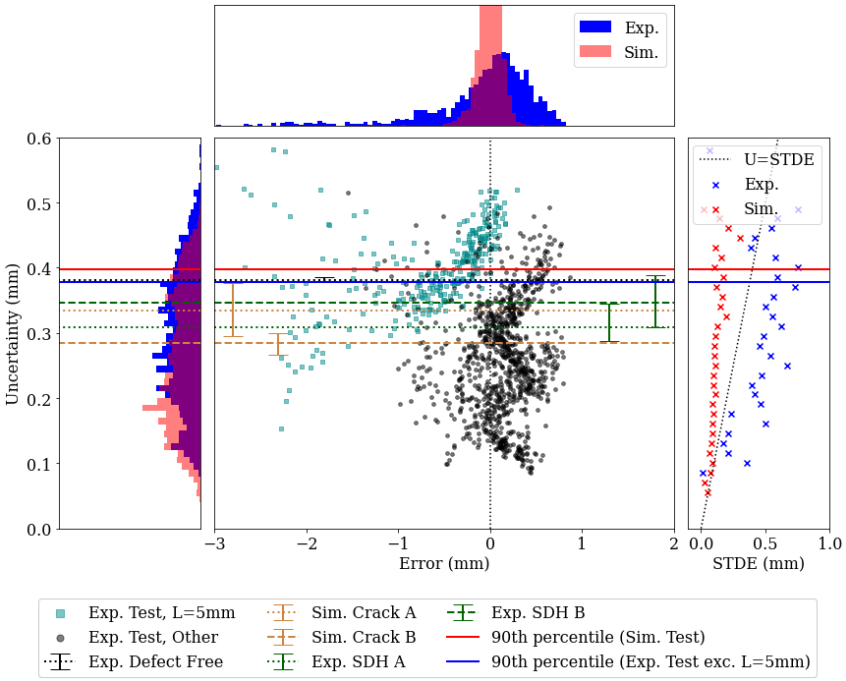


Fig. 21. Monte Carlo (MC) dropout uncertainty predictions for both in and out of distribution test sets. Experimental test set RMSE over 30 initialisations = 0.673±0.05 mm.

5.6.2. Calibration

The uncertainty quantification (Eq. 19) and prediction error (Eq. 20) of the methods described in Section 5.5 are illustrated in Fig. 19-Fig. 21. The predictions for uncertainty and crack depth (D) for DE and DE-ResSpec (Fig. 19 & Fig. 20) are formed from 60 independently trained networks. For MC dropout (Fig. 21) inference uses the output of one network with 200 forward passes, assigning a new random

seed to the dropout realizations each time. The main scatter plots in these figures show predicted uncertainty vs. sizing error for each defect in the experimental test set. Effective UQ for in distribution data in this plot appears as a zero-mean distribution of error that widens as uncertainty increases. Also shown in main scatter plot, as horizontal dotted and dashed lines, are the calculated uncertainties for OOD datasets. These do not have associated error as there is no equivalent 'true' value. Error bars for this data show the 25th and 75th percentiles of uncertainty across the full range of X-positions for each defect type. Ideally, these data should be assigned higher uncertainty values than most in-distribution data. Blue bars in the histograms plotted above and to left show the uncertainty and sizing error distributions for experimental and simulated test datasets based on bins of width of 0.05 mm (above) and 0.01 mm (left). For visual clarity, the red simulated test data histograms are not shown in the main scatter plot. Solid horizontal lines indicate the 90th percentile of the test sets' UQ. Graphs plotted to the right show aggregated uncertainty vs standard deviation of error (STDE). These are calculated by splitting the uncertainty predictions into equally spaced bins of height 0.015 mm and calculating the STDE in each bin containing more than one defect. The black dotted line is uncertainty = STDE which is the ideal result for in-distribution data as points on this line indicate predicted uncertainty is close to σ (as defined in (3)). Table 11 gives correlation coefficient, R for the linear fits to the data in the right-most graphs as well as the mean difference between STDE and predicted uncertainty. While, for the methods described in this chapter, the relationship between STDE and uncertainty is expected to be monotonic, there is no guarantee it will be 1:1, or even linear. Therefore, the following sections describe the observed trends in calibration of uncertainty to error for the experimental and simulated test sets.

Table 11. Metrics regarding linear fit of STDE to uncertainty below 90th percentile of uncertainty predictions for simulated and experimental test sets

	Simulated Test Set		Experimental Test Set	
	R	mean(U - STDE) (mm)	R	mean(U - STDE) (mm)
DE	0.98	0.032	0.95	-0.24
DE-ResSpec	0.99	0.015	0.98	-0.15
MC Dropout	0.84	0.11	0.84	-0.21

5.6.2.1. Simulated

Below the 90th percentile of sim test, DE (Fig. 19) and DE-ResSpec (Fig. 20) have a strong linear relationship between uncertainty and STDE. This is quantified by the high correlation coefficient of linear fits, ($R_{DE,Sim} = 0.99$, $R_{DE-RS,Sim} = 0.98$). The lines fit to this data have a slope of ~1 for both methods with low mean differences between uncertainty and STDE of 0.032 mm for DE and 0.015 mm for DE-ResSpec. In the upper tail of the uncertainty distribution (upper 10th percentile of sim test) both methods show increased scatter in STDE. This is likely due to the low amount of data in the STDE bins. MC dropout (Fig. 21) produces a linear fit for the simulated test set ($R_{MC,Sim}$ = 0.84) but its slope is 2.3, severely underestimating error for larger uncertainty values.

5.6.2.2. Experimental

In the upper tail of the uncertainty distribution (upper 10th percentile of exp. test) both DE and DE-ResSpec underestimate error significantly. While this is likely contributed to by insufficient ensemble diversity it is mainly due to inaccuracies in the simulation of the $L = 5$ mm defects. This is because the simulation used to create the training set assumes that the receiving transducer array elements are in the far-field of the defect. As discussed in Section 4.6, this is not the case for the $L = 5$ mm defects, noticeably affecting their PWI images. As shown in Fig. 19-Fig. 21, the experimental defects of length $L = 5$ mm are significantly undersized because of this domain shift. However, with DE and DE-ResSpec they are also assigned higher uncertainty. DE-ResSpec achieves this most effectively, assigning a mean uncertainty to $L = 5$ mm defects higher than 92% of the rest of the experimental test set. Even without knowing the true size of the defects this would highlight to the operator that they are somehow seen as anomalous by the networks. However, these uncertainty values are still low in comparison to their absolute error. This is because the difference in simulated and experimental $L = 5$ mm defects creates a systematic undersizing in all members of the ensemble. As this change in the predictions has a non-zero mean across the ensemble the increased uncertainty is not fully captured in the ensemble's overall variance (Eq. 19). This is an example of domain shift negatively affecting the quality of UQ, a known issue [121].

Experimental test set uncertainty below the 90th percentile increases monotonically with STDE for both DE and DE-ResSpec ($R_{DE,Exp} = 0.95$, $R_{DE-RS,Exp} = 0.98$) whereas MC dropout shows more significant scatter ($R_{MC,Exp} = 0.84$). All three of these trends have a slope <1, indicating that UQ is significantly underestimating error. The consequence of this for implementation of these methods is that if uncertainty predictions are to be used as an estimate of expected sizing error on a new experimental sample, an experimental validation set is needed to calculate the slope. This method is commonly called 'temperature scaling' [122]. However, even without temperature scaling, the strong linear fit means that higher uncertainty is a strong indicator of higher error for the DE based approaches.

5.6.3. Anomaly detection

Effective UQ should detect test cases drawn from distributions significantly far away from that of the training set. As the network has little to no prior information about these cases, it should assign them high uncertainty. As described in Section 5.3.2 this is primarily tested here using defect types not included in the training set. All three methods assign higher uncertainty to the OOD defects than the bulk of the experimental test set but for MC dropout it is also almost all below the 90th percentile of simulated data, demonstrating poor anomaly detection. DE-ResSpec demonstrates the best anomaly detection; assigning uncertainty above 90% of the non $L = 5$ mm experimental test set to 60% of the OOD cases. Exp. SDH_A is assigned the lowest uncertainty by DE-ResSpec. This makes intuitive sense, as of all the OOD defects, it is the smallest and nearest the back wall, and therefore produces PWI

images that most closely resemble a surface breaking crack. This is exampled in Fig. 16b where the set of PWI images for Exp. SDH_A is the only set where the main indication in each image are co-registered on the backwall, as occurs with a corner reflection from a surface breaking defect.

5.6.4. Choosing an uncertainty threshold

In implementing these UQ methods for industry, test cases with uncertainty above a certain threshold can be dealt with separately. This may mean inspection by a human operator, further data acquisition, use of traditional sizing methods or a combination of these approaches. To do this, a value for the uncertainty threshold must be decided upon. Ideally, this would be done through the use of an experimental validation set that represents the true inspection conditions well. However, in the absence of such data, using the simulated validation set could be an effective approach. The left and top panels of Fig. 20 show that this works well for DE-ResSpec as both the simulated and experimental 'in distribution' test cases are assigned similar uncertainty distributions, meaning that almost all high sizing error (>1 mm) and OOD cases are above the 90^{th} percentile of the simulated validation set. In contrast, in Fig. 19, DE demonstrates limited overlap between the UQ distributions for simulated and experimental test sets. This means that using a cut-off defined by only simulated data will find almost all experimental data anomalous with DE. It is hypothesised that the regularization of the spectral norm is responsible for DE-ResSpec demonstrating better simulated and experimental overlap than DE. MC-Dropout has good overlap but does not distinguish either of these sets from OOD data.

5.7. Making efficient use of resources

In the application considered in this thesis the computational resources at training and test time are not a barrier for implementation of DE. Both training and testing computation is trivially parallelizable, but even with multiple GPUs, some applications require more computational efficiency. This section discusses ways that training and inference time for DE can be reduced.

5.7.1. Training resources

As the architecture used here has a relatively low number of parameters (842,000) each epoch takes ~3.5s using an NVIDIA GeForce GTX 1070 Ti, so training a full ensemble of 60 networks can be completed in ~6hrs. If a more complex network was used (e.g., VGG 19 with 138 million) training an ensemble could take multiple weeks, making the development cycle very slow. Alongside its simplicity, MC dropout has also gained popularity as an UQ method because it only requires the training of one network so is a good candidate for reducing training time. Another approach is 'snapshot ensembles' [123] in which the members of an ensemble can be captured from one initialisation, using a cyclic learning rate. For this application snapshot ensembles were found to provide significantly worse UQ than DE. It is hypothesised that this is because the local optima found by snapshot ensembles are not as diverse as that found by re-initializing the network's parameters.

5.7.2. Test resources

Inference with the 60-network ensemble used in this chapter takes $\sim 8ms$ per image set which for most applications is quick enough to be considered 'realtime.' However, if realtime inference was required on lightweight hardware and/or using a more complex network the test time resources would need to be managed more efficiently. This could be achieved by pruning the weights of the individual networks [124], using a smaller number of networks in the ensemble by optimizing which members are used [125] or distilling the ensemble down to a single 'multi-headed' network with one set of common convolutional layers and multiple sets of fully connected layers [126].

5.8. Conclusions

This chapter has investigated the performance of UQ using DE, DE-ResSpec and MC Dropout for modern deep learning in application to inline pipe inspection when using a simulated training set and experimental test data. The success of these methods is judged by their calibration and anomaly detection performance. MC Dropout demonstrates only slightly raised uncertainty values for OOD samples and poorly calibrated uncertainty estimates. DE-ResSpec produced the best calibration on simulated test data, created the largest gap between in-distribution and out-of-distribution data and is the most reliable method in terms of assigning high uncertainty to high error test cases. However, while both DE and DE-ResSpec show a strong linear fit between experimental data error and uncertainty, the gradient of this fit is << 1, meaning that uncertainty significantly underestimates error. The implication of this for industrial applications is that an experimental validation set for scaling is needed if uncertainty values are used to infer expected prediction error. However, as the monotonic relationship between uncertainty and error is strong, even without an experimental validation set, predicted uncertainty can be used to compare relative error between test cases and detect anomalies. Therefore, DE-ResSpec is recommended for UQ when using deep learning for NDE.

One of the biggest unknowns in the field of data science for NDE is how data-driven NDE inspections are to be qualified. Within the current industrial framework, physics-based data analysis is qualified on a small pool of test samples and generalization assured by the interpretability of the method. However, in the future, the high levels of accuracy demonstrated by 'black-box' methods may well create a drive to qualify them by rigorous testing on a large range of test samples. For this to be realised, UQ methods such as the ones presented in this chapter, are going to be essential. As presented in this chapter, DE and DE-ResSpec are suitable for application to approximating uncertainty of deep learning for NDE. Improvements could be made by research into producing better calibrated UQ on experimental test data, despite the domain shift from the simulated training set. Domain adaptation methods or techniques for increasing the diversity within the ensemble are promising candidates for this problem.

Chapter 6. Interpretability and explainability

This chapter presents a novel dimensionality reduction method termed Gaussian feature approximation (GFA). GFA aims to improve the interpretability and explainability of ML models trained by providing a meaningful feature space. A fully connected neural network is trained to predict defect size from GFA features, and Shapley additive explanations are used to calculate how each feature contributes to the prediction of an individual defect's length. The content in this chapter is drawn from the author's work [127], which is currently in review.

6.1. Introduction

Building trust in ML for safety critical applications such as pipeline inspection is a challenge due to the 'black-box' of the algorithms. This chapter aims to tackle this issue by improving both the interpretability and local (i.e., for a specific test sample) explainability of ML based models for ultrasonic defect sizing. The precise definitions for *interpretability* and *explainability* are disagreed upon both between and within research fields. This work follows the definitions laid out in [128]. *Interpretability* is a domain specific notion, but in general it is the ability for a human to understand the link between cause and effect without anything other than the model itself. An explanation is an approximation of a model that aims to describe the cause of a local prediction. The term *explainability* is used here to follow convention but, as pointed out in [128], "summaries of predictions," "summary statistics," or "trends" are more truthful descriptors as the fact that "explanations" are an approximation to the complex internal calculations within a model is often overlooked.

Explanations for ML based on images are commonly provided by saliency maps which describe the locations in the input data that most significantly impact the prediction. There are many methods for creating saliency maps, such as gradient-weighted class activation mapping (grad-CAM, [129]), local interpretable model-agnostic explanations (LIME, [130]), deep learning important features (DeepLIFT, [131], [132]) and layer-wise relevance propagation (LRP, [133]). Shapley additive explanations (SHAP, [134]) provide a unified view of these methods, giving model-agnostic feature importance values for any type of input data and any type of model. However, as pointed out in [128] a saliency map does not show what about that location in the image is important (e.g., texture/amplitude/colour). This means saliency maps are of little use for most ML based defect classification where images or time domain signals are used as input, as highlighting the defect's indication in the data is of little use when it is usually already clear where the indication is. In other words, the challenge is interpreting how properties of an indication inform the prediction, rather than explaining which parts of an image led to the prediction. The root cause of this problem is a lack of interpretability in the model, due to the complex nature of the input data.

While interpretable ML is a relatively new field it has attracted a lot of research attention from the computer science community in recent years, due to its potential to address the 'black box' nature of

ML [135]. However, within NDE there have been only a small number of publications with a focus on either explainable or interpretable ML. Saliency-map based explanations have been produced using LIME, for ultrasonic defect detection [136]. Text-based explanations have been used with a human-designed decision tree for crack characterization [137], an effective approach when the decision-making process of the model is simple enough to be explained in a small number of sentences. Improving the interpretability of ML methods for ultrasonic NDE data has been achieved by replacing the trainable convolutional filters of a CNN with filters matched to the shape of Lamb waves [138] in application to localizing damage in aluminium plate using guided waves. Another published approach is to use well-known dimensionality reduction methods such as principal component analysis (PCA) to reduce the complexity of input data. This has been used with a support vector machine (SVM) to detect damage in carbon fibre reinforced polymer plate using ultrasonic guided wave data [139].

As discussed in [140] it is important to consider what constitutes useful interpretability for the relevant domain when applying ML, as it can vary a lot between applications. In NDE, useful interpretability usually stems from the ability to relate a model's inner workings to the reasoning of a skilled human operator or a physics-based approach. Ensuring input data is of a reasonably low dimensionality is also essential for achieving this goal, as humans are not able to process high-dimensional data effectively. To achieve improved interpretability for NDE data analysis, this chapter proposes a novel dimensionality reduction method, optimised for ultrasonic NDE images, called Gaussian feature approximation (GFA). GFA reduces ultrasonic images to a small number of meaningful descriptors of defect indications, making models trained on these descriptors interpretable and explainable, while still providing accurate defect sizing. GFA operates by fitting a 2D elliptical Gaussian to defect indications in ultrasonic images. Predictions of a ML model trained on GFA features are interpretable because GFA features are based on properties of the defect indication, which are meaningful to a human operator. Local explanations are enabled by GFA as methods such as SHAP can be used to indicate how individual properties of a defect indication contribute to the defect size prediction.

To allow comparison of sizing accuracy using GFA, two other well-known methods are applied to create reduced dimensionality feature spaces: principal component analysis (PCA) and the parameters of 6 dB drop boxes fitted around defect indications. Defect sizing is achieved by training a dense neural network on PCA, 6 dB drop and GFA features as well as a convolutional neural network (CNN) [141] on the raw ultrasonic images. CNNs are state of the art for learning from images and their ability to provide accurate sizing for the application considered in this thesis is demonstrated in Chapter 3. CNNs are used in this chapter as a high accuracy, low interpretability, baseline method to compare against.

All three dimensionality reduction methods and their corresponding sizing algorithms are applied to ultrasonic plane wave imaging (PWI) images. Detection is already considered complete, so the target is to size the defects of interest (surface breaking cracks) from the PWI images. The simulation and

experimental set-ups are designed to closely approximate the conditions in the example application of this thesis: ultrasonic inline pipe inspection. The usefulness of GFA, coupled with a neural network for sizing, and kernel SHAP to produce local explanations, is judged by interpretability, explainability and sizing accuracy. The rest of the chapter is structured as follows. Section 6.2 describes the data sets used in this chapter, Section 6.3 describes all relevant data pre-processing and analysis methods, Section 6.4 the sizing accuracy and explainability results and Section 6.5 conclusions.

6.2. Data Sets

The application considered in this thesis requires the sizing of defects, after their detection. The target is therefore the extent of the defect perpendicular to the surface, $D = L\cos(\theta)$. The parameter space of surface breaking defects considered is defined by P, L, θ. All experimental and simulated data used in this chapter are generated using the methodologies described in Chapter 2 and distributed as described in Section 5.3.1. This results in a total of 16,875 simulated PWI image sets which is split 85/7.5/7.5 for training, validation and testing respectively and 1,485 experimental PWI image sets, from the 30 manufactured defects, split 15/85 for validation and testing respectively.

6.3. Data processing and analysis methods

This section describes the data processing and analysis methods used in this chapter: an initial image windowing step, the dimensionality reduction used to improve interpretability, the neural network architectures used to predict defect size, and kernel SHAP: the technique used to produce local explanations.

6.3.1. Windowing images

As exampled in Fig. 2d,e, the PWI image sets in this thesis often contain artefacts caused by views other than the one being imaged. This can cause difficulties for sizing algorithms, especially those using transform-coding-based features (such as PCA), as information about the artefact can become ‘entangled’ with information about the imaged mode. To avoid this, the PWI images are windowed around the defect location before implementing any dimensionality reduction or sizing in this chapter. This step is not a fundamental requirement for any of the presented methods, but is a useful pre-processing step, as it forces the model to focus on the location of the defect, and removes unhelpful information, improving sizing accuracy and simplifying explanations. It is also simple to execute for this data set as surface-breaking defects are easy to locate due to their strong corner reflections when insonified at $\psi = \pm 45°$.

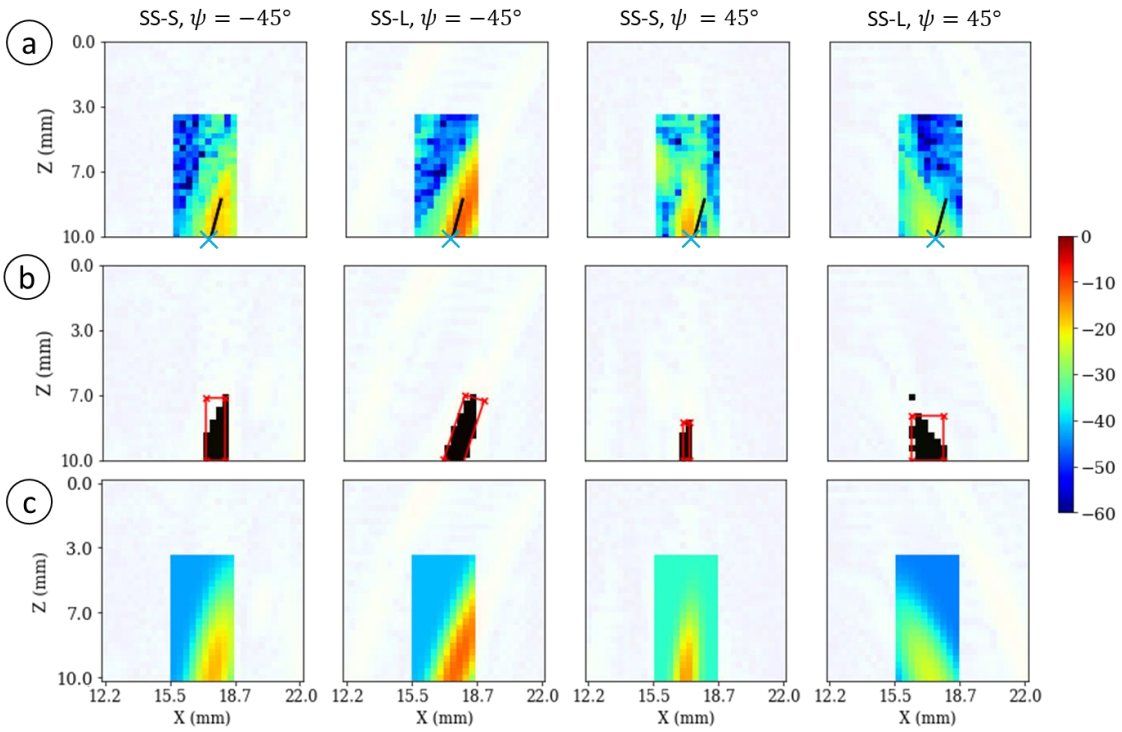


Fig. 22. a) An example of an experimental, windowed PWI image set, from a defect with $P = 17.4\ mm, L = 2\ mm$ and $\theta = 15°$ with the calculated location of the defect on the backwall (x_W) shown as a blue cross and the true extent of the defect shown in black, b) the top 6 dB of the windowed PWI images (in black) and their 6 dB bounding boxes (in red) and c) 2D elliptical Gaussians fit to the windowed PWI images using GFA. In all images the full, unwindowed image, is displayed in the background.

Locating a defect on the backwall is implemented by summation of the four associated PWI images (e.g., Fig. 22a). The X-location is then found using the maxima in the resulting 32x32 composite image. Using this method on all experimental and simulated data in this chapter produces a maximum X-location error of 0.56 mm (1.76 pixels). A window is then applied to the PWI images around the calculated backwall location of the defect (x_W) to isolate the correct indication. In this chapter, the window size is set to be 3.15 mm (10 pixels) in X and 6.30 mm (20 pixels) in Z. This window size is selected to be large enough to cover indications from all possible defects within the domain of operation, with minimal contributions from artefacts. An example of a set of windowed PWI images is given in Fig. 22a.

6.3.2. Dimensionality reduction methods

Three different dimensionality reduction methods are applied to the windowed PWI images in this chapter; PCA, 6 dB drop and GFA. The first two of these methods are well-known and presented for comparisons to GFA.

6.3.2.1. Principal component analysis

PCA is the process of finding the sequence of orthogonal vectors that best explain the variance of sets of high dimensional data [142]. PCA is often used to find a reduced set of features, with minimal loss of information, for use in ML [143]. In this chapter, the principal components are calculated using the

windowed, simulated, training set PWI images. The four images per defect are handled individually, using different PCA transforms, to preserve the separation of information between images. M_P different principal components are kept for each 10×20 image. To make the reduced dimensionality consistent with that of with GFA (described in Section 6.3.2.3), $\kappa_{PCA} = 7$ in this chapter. As shown in Fig. 23, $\kappa_{PCA} = 7$ describes 96% of the variance in the simulated training set, showing that the majority of the information in the images has been captured.

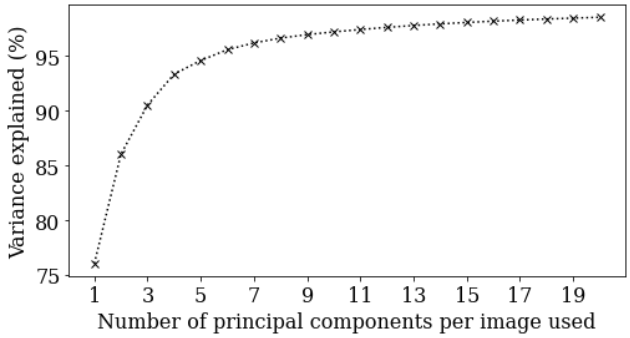


Fig. 23. The variance of the training set captured by different numbers of PCA components.

6.3.2.2. 6 dB Drop

6 dB drop is a well-established defect sizing method in NDE. It is based upon the idea that if a defect is the strongest indicator in an image, the image region within 6 dB of the peak value can be used as a good approximation of the true size of the defect. Traditionally, crack-like defects are sized using the longest edge of a rectangular bounding box that encloses all pixels within 6 dB of the peak [11].

In this chapter, the 6 dB drop bounding box is obtained by finding the rectangle with the minimum area that can fit the relevant pixels. The relevant pixels are selected by picking the region of conjoined pixels above -6 dB with the largest total amplitude. Picking the highest conjoined region of high amplitude in this way reduces the chance of noise expanding the size of the box. The SS-L, $\psi = 45°$ image in Fig. 22b shows an example of high amplitude noise excluded by this approach. The 6 dB drop bounding box calculated with this method is used both for dimensionality reduction and as a direct sizing method in this chapter. Calculating the parameters of the bounding box (X-position, Z-position, orientation, width and height), results in 5 features per image. These features carry no information directly related to the indication's amplitude. GFA features do contain amplitude information, so for a fair comparison, when sizing from 6 dB box features using the neural networks presented in Section 6.3.3.2, two additional features are used, resulting in $\kappa_{6dB} = 7$ features. These two additional features are chosen to be the maxima and root mean square (RMS) of all pixels within the bounding box, above -6 dB (i.e., the black pixels in Fig. 22b). Direct, traditional sizing with 6 dB drop is also considered, and is implemented by taking the mean of the longest edges of the boxes fitted to each image.

6.3.2.3. Gaussian Feature Approximation

GFA is a novel dimensionality reduction method presented in this chapter with the aim of creating a feature space that is informative (i.e., retains the information needed for accurate defect sizing), interpretable (i.e., meaningful to NDE operators) and improves the quality of local explanations. GFA is performed by fitting a 2D elliptical Gaussian function to each PWI image and using the parameters that define that Gaussian as the features of the image. GFA features describes a defect indication in a similar fashion to 6 dB drop features, but with a more robust fitting procedure that is not dependent on selecting a threshold value, and avoids the need for pre-processing to deal with conjoined pixels. It is also a richer feature space, containing more information about the indications shape, as well as the background noise level. As shown in Section 6.4.1, these differences make sizing on GFA features significantly more accurate than sizing on 6 dB drop features.

GFA is motivated by the importance of a defect indication's amplitude, spatial size and location in traditional NDE sizing techniques. These underlying features are encoded within PWI images, but not in a form that allows for interpretable models to be trained on them. Fitting an appropriate shape to a PWI image disentangles properties of the defect indication from each other, as well as from information relating to noise and artefacts. The shape used for fitting in GFA is a 2D elliptical Gaussian, this can be described by amplitude at position in the X and Z direction (x, z), given by

$$f_{x,z}(A, x_0, z_0, \sigma_X, \sigma_Z, \Theta, B) = Ae^{-a(x-x_0)^2 - b(x-x_0)(z-z_0) - c(z-z_0)^2} + B \tag{24}$$

$$a = \frac{\cos^2(\Theta)}{2{\sigma_X}^2} + \frac{\sin^2(\Theta)}{2{\sigma_Z}^2},$$

$$b = \frac{\sin(2\Theta)}{2{\sigma_X}^2} - \frac{\sin(2\Theta)}{2{\sigma_Z}^2}, \tag{25}$$

$$c = \frac{\sin^2(\Theta)}{2{\sigma_X}^2} + \frac{\cos^2(\Theta)}{2{\sigma_Z}^2}$$

using seven GFA features: amplitude (A), X-position (x_0), Z-position (z_0), X-sigma (σ_X), Z-sigma (σ_Z), angle (Θ) and offset (B). Finding the optimum set of parameters is achieved by minimizing

$$\mathcal{L}(A, x_0, z_0, \sigma_X, \sigma_Z, \Theta, B) = \sum_x \sum_z (f_{x,z} - I_{x,z})^2 \tag{26}$$

where $I_{x,z}$ is the windowed PWI image and the summations are over the windowed region only. This optimization problem is solved in this chapter by using SciPy's curve fitting function [144] with the trust region reflective minimization algorithm [145] as it is particularly suitable for large, bounded problems such as this one. The bounds and initial guess for the seven parameters that define $f_{x,z}$ are described in Table 12. It is important to note that bounding x_0 and z_0 within the window is necessary as $\mathcal{L}$ has zero gradient when the Gaussian's centre is far away from the window. Also, constraining

$-\frac{\pi}{4} < \Theta < \frac{\pi}{4}$ is necessary to ensure there are not two equivalent solutions with σ_X and σ_Z values swapped.

Table 12. Initial guess and Bounds for GFA Features.

Lower and upper bounds are inclusive.
$\max(I_{x,z})$ refers to the maxima in the current image for which GFA features are being calculated.
x_W is the centre of the 10×20-pixel window and δ is the image resolution ($\delta = \frac{\lambda_S}{2} = 0.317$ mm)
η is calculated by the root mean square of an experimental PWI image set from a defect free sample.

Parameter	Amplitude, A	X-position, x_0 (mm)	Z-position, z_0 (mm)	X-sigma, σ_X (mm)	Z-sigma, σ_Z (mm)	Angle, Θ (rad)	Offset, B
Initial guess	$\max(I_{x,z})$	$\underset{x}{\operatorname{argmax}}(I_{x,z})$	$\underset{z}{\operatorname{argmax}}(I_{x,z})$	0.5δ	2δ	0	0
Lower bound	0	$x_W - 5\delta$	0	0	0	$-\frac{\pi}{4}$	0
Upper bound	$\max(I_{x,z})$	$x_W + 5\delta$	20δ	10δ	20δ	$\frac{\pi}{4}$	20η

In principle, more than one Gaussian could be fit to each image. However, for the application presented in this chapter, adding a second Gaussian per image and sizing using a neural network (as presented in Section 6.3.3.2) was not found to increase sizing accuracy. This is likely because most information useful to the sizing process can be captured by one Gaussian. This is further evidenced by the root mean square error (RMSE) for GFA based sizing only being 23% higher than sizing from the original image (detailed sizing accuracy results are presented in Section 6.4.1). It should be noted that if fitting more than one Gaussian is deemed necessary it should be done in series (i.e., fit the second Gaussian, $f_{x,z}^2$, to $I_{x,z} - f_{x,z}^1$) rather than in parallel. This is to ensure the ordering of the GFA features is meaningful to the sizing algorithm. More complexity could also be added to $f_{x,z}$ by using more complex-shaped fitting functions with more parameters, such as skewness or properties of background noise, but this would reduce the interpretability of the feature space, so should not be done without certainty that the extra features are informative for the task at hand.

GFA, as introduced in this section, creates a feature space that is implicitly more interpretable than the raw PWI images and enables useful local explanations. GFA features are interpretable as they each uniquely describe a property of the defect indication which is meaningful to an NDE practitioner. They are also only minorly affected by background noise and artefacts, meaning sizing on GFA features is guaranteed to be informed by the defect indication, and not overfitted to other confounding features. Local explanations are made more useful by GFA as they can ascribe importance to specific aspects of a defect indication with GFA features instead of a saliency map in real space. Explainability is further discussed in Sections 6.3.4 and 6.4.2.

6.3.3. Neural network architectures for defect sizing

6.3.3.1. Convolutional neural network

As a baseline approach with high accuracy and low interpretability the raw PWI images are sized using the CNN designed for this data set, as presented in Section 3.2.2.1. CNNs are state-of-the-art for image classification tasks due to the power of convolutional layers to map structured, high-dimensional data to informative feature spaces [25]. The CNN architecture used here is illustrated in Fig. 24a. The input is composed of the four 32×32 PWI images stacked in the third dimension, akin to how natural image CNNs treat red, green and blue channels. The general structure is made up of repeated blocks of convolutional and max-pooling layers for feature extraction, followed by fully connected layers for regression. Rectified linear unit (ReLU) activation is used throughout. Ten percent dropout is applied to the fully connected layer inputs for regularization. The state-of-the-art Adam optimiser [53] is used to train the CNN with a learning rate of 0.001, in mini-batches of 128, with a patience of 150 epochs (i.e., until 150 epochs with no reduction in experimental validation set loss). The network hyperparameters (depth, filter size and number, dropout rate, neuron number etc.) have been selected to optimise experimental validation set accuracy. More details on this design process can be found in Section 3.2.2.1.

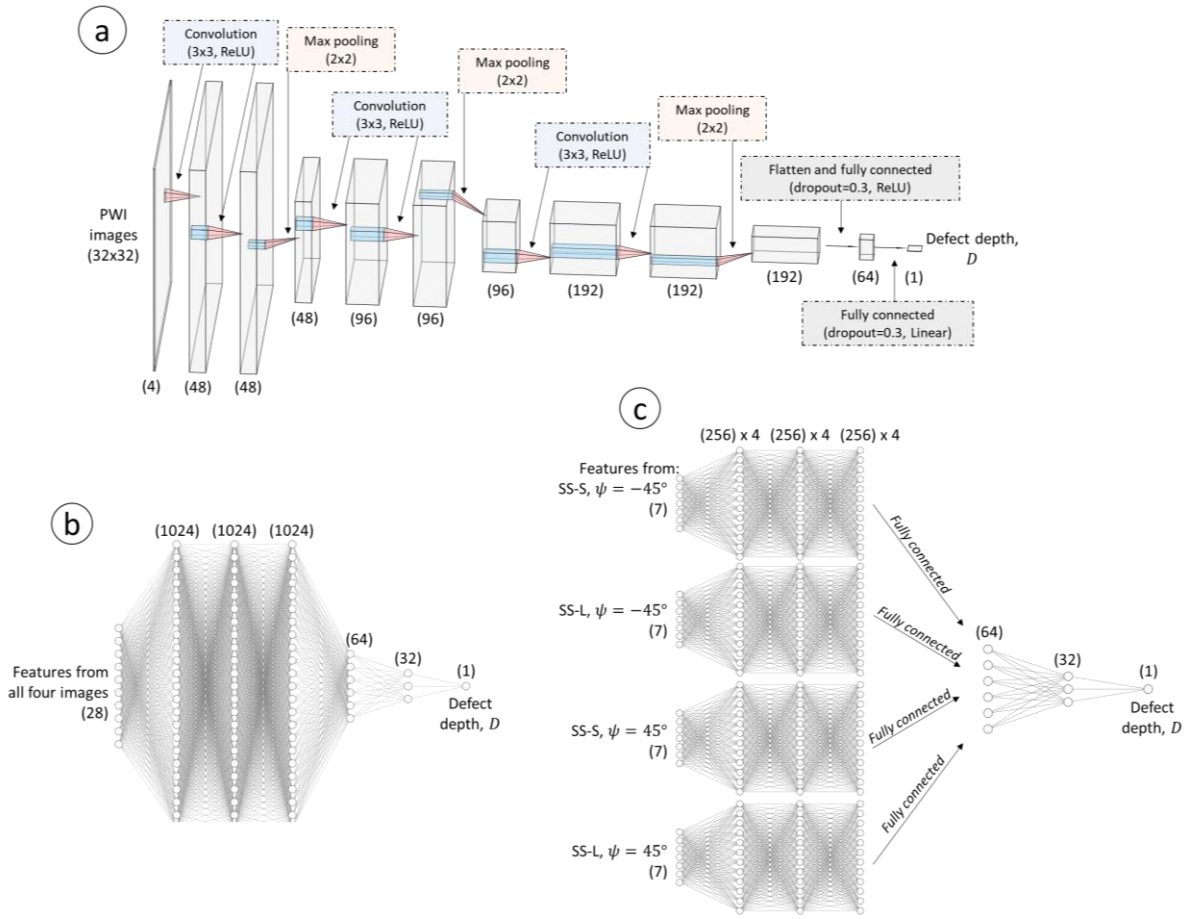


Fig. 24. Neural network architectures used in this chapter, as described in Section 6.3.3: a) CNN, b) NN-Single, c) NN-Split.

There are three minor changes to the implementation of the CNN between Chapter 3 and this chapter. Firstly, only a single network is needed to predict D. The network used matches the structure of the L network in Chapter 3. Secondly, dropout is increased to 0.3, which resulted in slightly better experimental validation set accuracy (~4%) at the cost of needing ~50 more epochs to converge. Thirdly, the windowing of the PWI images described in Section 6.3.1 must be accounted for. For computational efficiency this could, in principle, be done by reducing the input layer size to 10×20×4 and concatenating the X-position with flattened features before the dense layers. However, as the purpose of including CNN-based sizing in this chapter is as a baseline for sizing accuracy, computational efficiency is not of major concern. Therefore, the images are simply zero-padded to their original 32×32×4 size before being input into the CNN. This offers a simple way to encode X-position without drastically altering the CNN design and potentially reducing sizing accuracy.

6.3.3.2. Dense neural network

Training a sizing algorithm from a set of unstructured numerical features such as those produced by GFA, PCA and 6 dB drop can be done with many ML algorithms (e.g., random forest, support vector machine and k-nearest neighbours). In this chapter, sizing from the reduced feature sets is done using a dense neural network, i.e., layers of neurons that are fully connected to preceding layers. This gives a natural comparison with the CNN as both algorithms operate in a similar fashion and have the capability to represent complex, non-linear functions.

To match the CNN, the dense neural networks in this chapter are trained with the Adam optimizer and use ReLU activation functions on all layers except the input and output. As with CNNs these are also common design choices for dense neural networks. All other hyperparameters are selected via the same design process as used to design the CNN (Section 3.2.2.1); grid search, with selection made using the lowest GFA experimental validation set RMSE. The optimal learning rate was found to be 1×10^{-4}. Application of dropout and L2 regularization were tested but found to increase validation set error, suggesting that they are unnecessary for this reduced dimensionality input data, and so are not used.

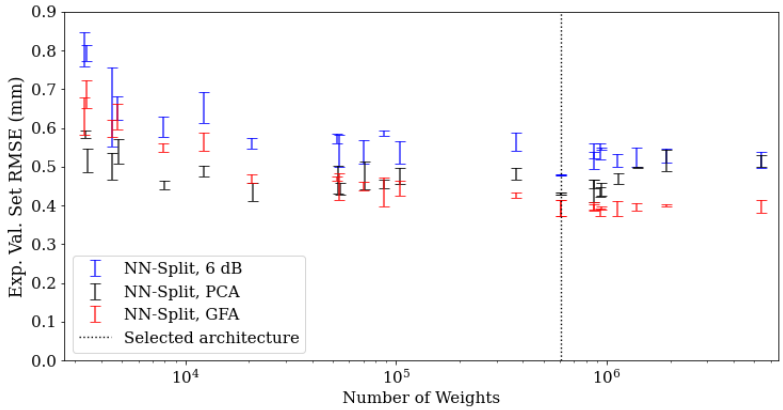


Fig. 25. Experimental validation set RMSE for the NN-Split architecture applied to GFA features, with different hyperparameters. Details of the exact hyperparameters tested are given Table 13. The error bars represent ± standard deviation over five independent initializations.

Table 13. Hyperparameters for all tested NN-Split architectures, as shown in Fig. 25. The selected architecture is highlighted in green.

Number of weights	RMSE (mm)	Neurons in each layer (*before* full connection)	Neurons in each layer (*after* full connection)
3329	0.64	1024, 1024, 1024, 1024	64, 32
3409	0.69	16, 16	64, 32
4481	0.61	16	64, 32
4769	0.62	32, 32	64, 32
7873	0.55	64, 64	64, 32
12225	0.55	64, 64, 64, 64, 64, 64	64, 32
20609	0.47	256	64, 32
52321	0.50	512, 256, 128, 64, 32	64, 32
53889	0.46	256, 256, 256	64, 32
70529	0.45	256, 256, 256	64, 32
87169	0.44	256, 256, 256, 256, 256	64, 32
103809	0.45	256, 256, 256, 256, 256, 256, 0, 0	64, 32
369281	0.42	512, 512, 512, 512, 512, 512	64, 32
602241	0.39	1024, 1024, 1024	64, 32
865409	0.40	1024, 1024, 1024, 1024	64, 32
865921	0.40	1024, 1024, 1024, 1025	64, 32, 16
933057	0.39	1024, 1024, 1024, 1027	128, 32
939265	0.40	1024, 1024, 1024, 1026	128, 64, 32
1128577	0.40	1024, 1024, 1024, 1024, 1024	64, 32
1391745	0.40	1024, 1024, 1024, 1024, 1024, 1024	64, 32
1918081	0.39	1024, 1024, 1024, 1024, 1024, 1024, 1024, 1024	64, 32
5402753	0.39	2048, 2048, 2048, 2048, 2048, 2048	64, 32

In the initial design process for the number of neurons in each layer (i.e., width) and number of layers (i.e., depth), the dense neural network was set to follow a common structure: a sequential set of fully connected layers of reducing width. This architecture is illustrated in Fig. 24b and referred to as NN-Single from here onwards. However, in following iterative design stages it was found that fixing the number of neurons but removing connections between the features from different image modes improved performance. This produces a structure of four dense neural networks, fully connected in the final few layers. This architecture is illustrated in Fig. 24c and referred to as NN-Split from here onwards. It is the author's belief that NN-Split outperforms NN-Single, even with the same number of neurons, because it allows the initial layers to compose the features from an individual image into a more expressive form without immediately entangling them with features from other images. The experimental validation set RMSE for all NN-Split widths and depths tested are given in Fig. 25, and their hyperparameters described in Table 13. As found in Section 3.2.2.1 for the design of the CNN, NN-Split with GFA and 6 dB features shows a 'diminishing returns' relationship between the number of weights (here used as a proxy for complexity) and sizing accuracy. PCA features provide good sizing accuracy even with the lowest complexity networks tested. The architecture selected (indicated by a dashed line in Fig. 25, and illustrated in Fig. 24c) is deemed to be a good trade-off between computational complexity and performance for GFA features, and provides good sizing accuracy with all three feature types. It is interesting to note that despite the input data dimensionality reduction of 96.5% (i.e., from $10 \times 20 \times 4$ to 7×4) the number of weights in NN-Split are only 76% lower than in

CNN. This suggests that the relationship between both 6 dB drop and GFA features, and crack size is still very complex and non-linear, despite the dimensionality reduction. Understanding why PCA features require a significantly less complex neural network to achieve good sizing accuracy requires further research.

6.3.4. Local explanations using kernel SHAP

There are many popular methods for creating local explanations for ML predictions. A unifying view for these methods, termed SHAP, has been presented in [134]. SHAP aims to produce game theory results (i.e., Shapley values [146], [147]) in a computationally efficient manner and unifies most modern model explanation methods (LIME [130], DeepLIFT [131], [132], LRP [133] and classic Shapley estimation methods [148]–[150]) as different versions of the same framework. The underlying logic behind SHAP is to approximate the output of the original prediction model, given the current input, $f(x)$, with a linear explanation model, given a set of simplified inputs (e.g., bag of words for text features or saliency maps for images),

$$f(x) \approx g(z') = \varphi_0 + \sum_{i=1}^{\kappa} \varphi_i z_i' \tag{27}$$

where $z' \in \{0,1\}^{\kappa}$, κ is the number of simplified input features and φ_i the importance of each feature (i.e., the SHAP values). φ_0 is set to be the mean of each feature in the model's training set in this chapter, as is common in most published implementations. φ_i is a function of the current input, x.

If features are assumed to be independent when approximating conditional expectations, as in LIME and DeepLIFT, then SHAP values can be estimated directly using the Shapley sampling values method [150]. This involves uniformly sampling permutations of z_i' . Note that for most applications, setting a feature to 0 does not effectively represent the absence of that feature ($z_i' = 0$), so instead, that feature is set to a value sampled from the training set. The issue with the Shapley sampling values method is that sampling enough to get an accurate explanation is slow to compute for large numbers of inputs. Kernel SHAP [134] is a more computationally efficient sampling method as it jointly estimates all φ_i using a linear regression formulation, leading to fewer required samples for accurate estimation of Shapley values. This is achieved by weighting samples of z' by a kernel,

$$k(z') = \frac{\kappa - 1}{\binom{\kappa}{s} s(\kappa - s)} \tag{28}$$

where s is the number of ones in z' and $\binom{\kappa}{s}$ represent a binomial coefficient. This is a very similar approach to that of LIME, but removes the need to select a loss function, weighting kernel, or regulariser, while guaranteeing local accuracy, missingness and consistency (as defined in [134]) in the explanation. The only two hyperparameters for kernel SHAP are the number of binary mask iterations

(a) and samples of training data in the background data set (b). The number of iterations to calculate SHAP values is $a \times b$.

To select an a and b that ensures sufficient sampling, without excessive computation, a grid search is carried out. For each combination of a and b tested the SHAP values of five random experimental test set image sets, for a NN-Split model using GFA features, are calculated. The mean absolute difference between these SHAP values and those calculated with a large number of samples ($a = b = 5000$) is calculated,

$$\Lambda = \sum_{i=0}^{5} \sum_{j=0}^{28} \left| \varphi_{i,j,a,b} - \varphi_{i,j,5000,5000} \right| \tag{29}$$

where $\varphi_{i,j,a,b}$ represents the SHAP value for data set i and feature j. The results of this for grid search are displayed in Fig. 26. $\Lambda = 0.01$ is considered to indicate sufficient convergence, so $a = 750$, $b = 450$ are selected for use in the rest of this chapter as this is the minimum number of kernel SHAP samples necessary to achieve $\Lambda = 0.01$.

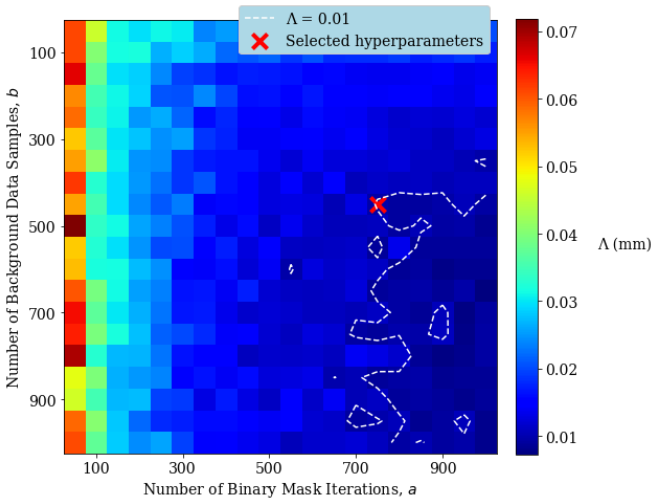


Fig. 26. The change in mean absolute SHAP values ($\Lambda = \sum_{i=0}^{5} \sum_{j=0}^{28} \left| \varphi_{i,j,a,b} - \varphi_{i,j,5000,5000} \right|$) for NN-Split, sizing from GFA features, for different numbers of binary mask iteration (a) and background data samples (b). $\Lambda = 0.01$ is deemed to be sufficiently low so a contour at this level (as described by the white dotted line) is used to select a and b.

6.4. Results

This section gives a comparison of sizing accuracy when using the dimensionality reduction techniques presented in Section 6.3.2 with the ML architectures presented in Section 6.3.3. The interpretability of the presented sizing networks is discussed and local explanations of predictions using GFA with NN-Split are presented.

6.4.1. Sizing accuracy

While sizing accuracy is not the main focus of this chapter, an interpretable defect sizing algorithm that cannot size reasonably accurately is not of any use. As shown in Fig. 27, the most accurate sizing method tested is a CNN sizing from raw images, providing a RMSE of 0.58 mm. Note that this is 29% lower than the same architecture trained on unwindowed images, proving the value of removing information unrelated to the task at hand. For all the ML based sizing methods thirty independently trained networks are trained, with the bars in Fig. 27 displaying the mean RMSE and ± one standard deviation plotted as error bars.

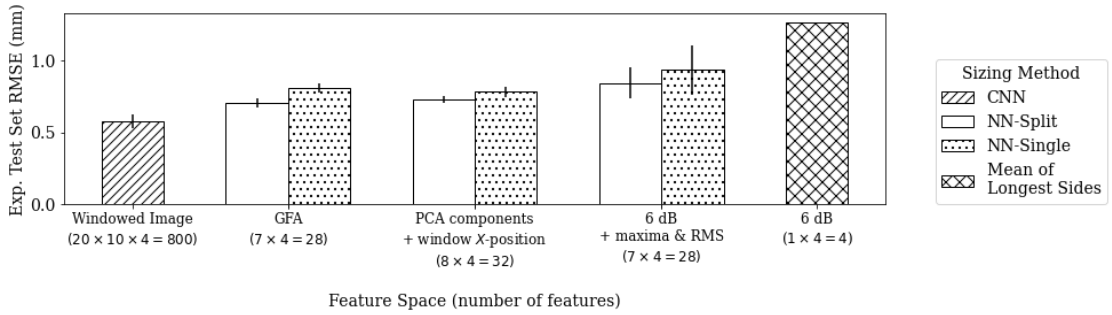


Fig. 27. Experimental test set sizing RMSE for all dimensionality reduction methods and associated sizing methods discussed in this chapter. The error bars represent over thirty independent initializations. The numbers in brackets indicate the dimensionality of the input data.

The least accurate sizing is provided by 6 dB drop. Both training a NN-Split on the 7×4 parameters of the 6 dB drop box (including maxima and RMS amplitude information) and directly using the mean of the longest sides (as is the traditional method) for sizing produces poor sizing with an experimental test set RMSE of 0.843 mm and 1.26 mm respectively. The high sizing error when using 6 dB drop features is likely because they do not carry enough information relevant to sizing the defects. The next most accurate sizing technique is NN-Split, using 7 PCA components, concatenated with the window's X-position. This gives a RMSE of 0.73 mm. GFA with NN-Split offers the closest sizing accuracy to the CNN with a RMSE of 0.71 mm, despite having only 7 features per image. NN-Single predictions on GFA data are 14% higher than NN-Split predictions on the same data. This motivates the use of NN-Split as it offers better sizing accuracy despite containing 73% less weights.

6.4.2. Interpretability and Explainability

As discussed in Section 6.1, training a CNN on raw ultrasonic images is a 'black box' approach, as it is not interpretable, and local explanations are limited to saliency maps. PCA provides a lower dimensionality input data but neither the magnitude nor form of the components has physical meaning, so are not explainable or interpretable. Sizing based on GFA and 6 dB drop box features is interpretable as the features are simple descriptors of the defect indication. Also, as these dimensionality reduction methods are only minorly affected by background noise and artefacts, the operator can be confident that sizing predictions are informed only by the defect indication. Despite the similar levels of

interpretability, GFA is significantly more useful than 6 dB drop due to the significantly lower sizing error, as presented in Section 6.4.1.

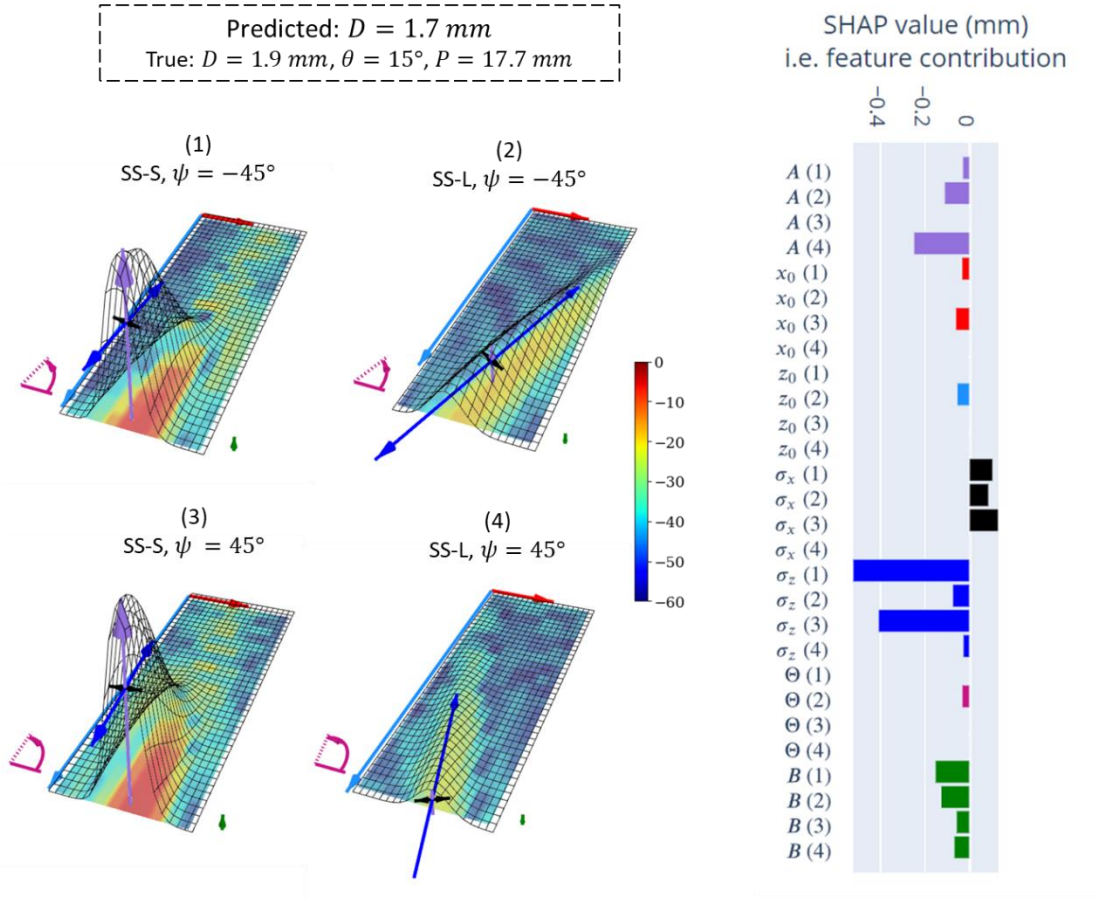


Fig. 28. An example explanation visualization for a sizing prediction from an experimental test set example with $P = 17.7\ mm, L = 2\ mm$ and $\theta = 15°$. The sizing is achieved using NN-Split and GFA features and the feature contributions calculated using kernel SHAP. The 3D plots display the original PWI image as a colourmap, a visualization of the GFA fit as a wireframe surface and the GFA features drawn with arrows coloured in relation to the SHAP bar chart. An interactive version of this figure can be found at *https://richardp1234.github.io/GFA-Vis/index4.html*.

As well as the implicit interpretability provided by sizing with GFA features, useful local explanations can be created with them. As described in Section 6.3.4, SHAP values can be calculated to indicate the importance of each feature to the sizing prediction for a specific defect. An example of how this could be visualised for an operator is given in Fig. 28. The magnitude of the SHAP values indicate how important each feature is to the prediction and their sign (i.e., positive or negative) shows whether that feature is pushing the prediction higher or lower from φ_0. For the example in Fig. 28, the most impactful features are the σ_Z of the SS-S views. This makes intuitive sense as the defects of interest in this thesis are oriented roughly in the Z-direction and in these two views there is high amplitude specular reflections from the full extent of the defect.

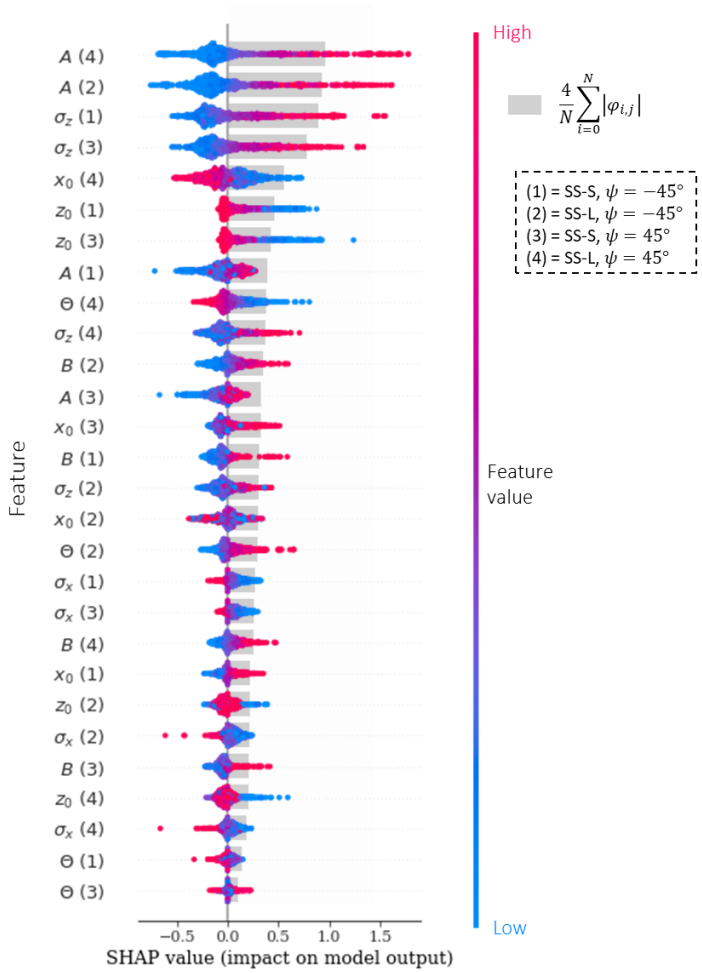


Fig. 29. A bee-swarm plot of the SHAP values for the experimental test set with sizing achieved using NN-Split and GFA features. (n) represents the n_{th} imaging mode as indicated in Fig. 28. The features are sorted by $F_j = \frac{1}{N}\sum_{i=0}^{N}|\varphi_{i,j}|$ where i represents the index of the sample in the test set and j the feature. $4 \times F$ is plotted as gray bars, this has been scaled by 4 in this plot for ease of visual comparison between features.

To analyse overall feature importance for the trained NN-Split model, SHAP values are calculated for every defect in the experimental test set. These SHAP values are then visualised in a 'bee-swarm' plot [151] in Fig. 29. In this plot each defect is represented as 28 dots, one for each feature. A dot's colour represents the normalised magnitude of the feature and its X-position is determined by SHAP value. Dots placed at the same X-position are plotted with different vertical positions to avoid covering each other. The features are sorted by their mean absolute SHAP value

$$F_j = \frac{1}{N}\sum_{i=0}^{N}|\varphi_{i,j}| \tag{30}$$

where i represents the index of the sample in the test set and j the feature. F for the top four features in Fig. 29 is significantly larger than for others, indicating that these are the features that impact crack size prediction the most. The fact that σ_Z for the two SS-S views is considered important by the network builds trust in its predictions. This is because the SS-S view produces the indications most closely

matching the true extent of the defect and σ_Z is similar to the parameter used by traditional 6 dB sizing. The fact that the SS-L view amplitudes are the top two most impactful features is an interesting result. It is the author's belief that this may be where the network is inferring θ, to account for the variations in amplitude and σ_Z it causes. This hypothesis is based on the high correlation between true crack angle and the angle of the indication (i.e., θ and Θ) in these views (Spearman's correlation = 0.56). Fig. 29 also shows that the features not used by classical NDE sizing techniques are assigned low F. These include, for example, the width of the indication (σ_X) and the background noise level (B).

The correlations between the value of features (colour of the dots in Fig. 29) and their impact on network output (X-position of the dots in Fig. 29) can also be in inspected, and here too similar behaviours as found in classic physics-based NDE sizing methods are found:

- A and σ_Z values are positively correlated with defect size.
- The nearer a defect is to the array (low x_0 for $\psi = 45°$, high x_0 for $\psi = -45°$), the larger the SHAP value. This positively contributes to the D prediction, correcting for the fact that defects near to the array appear smaller, because they are not insonified over their full extent.
- Defects angled towards the array (-ve Θ for $\psi = 45°$, +ve Θ for $\psi = -45$) have larger SHAP values. This positively contributes towards the D prediction, correcting for the fact that these indications have significantly lower amplitude in SS-L modes.

There are more complex interactions happening with z_0, σ_x and B that are harder to draw conclusions from in Fig. 29, but they are also lower in average feature importance (F) and have less significance for physics-based defect sizing.

6.5. Conclusions

This chapter has presented GFA, a novel dimensionality reduction method, aimed at improving the interpretability and explainability of ML for ultrasonic NDE. Defect sizing with a neural network (NN-Split), trained on simulated GFA features, tested on experimental data, has been shown to produce a RMSE only 23% higher than a CNN applied to the full PWI image sets, despite the dimensionality reduction from $10 \times 20 \times 4 = 400$ to $7 \times 4 = 28$. The other dimensionality reduction methods tested all provided comparable or worse sizing performance. In terms of interpretability, GFA improves upon both the original images and PCA. 6 dB drop features have comparable interpretability to GFA but provide significantly less accurate sizing.

GFA provides improved interpretability to models that use the features as, unlike individual pixel values, their values are meaningful to NDE operators. Also, as GFA features are only minorly affected by artefacts and background noise, sizing on GFA features is guaranteed to be informed by the defect indication, and not overfitted to other confounding features. GFA also enables useful local explanations as methods such as kernel SHAP can be used to inform the operator which features are important to the sizing of a specific defect and if that feature contributes positively or negatively to the prediction. It should be noted that SHAP values are a simplification of the model, assuming a linear combination of

independent features, so cannot fully explain global model behaviour. However, they can still be of great use in building trust in a model's predictions, by comparison with the decision-making process of expert intuition or physics-based sizing approaches.

If further interpretability was desired beyond that of GFA, as described in this chapter, the feature with the lowest average SHAP value (i.e., lowest F_j) could be removed from the training set and the network retrained using only the remaining features. This could be done iteratively until validation set RMSE became unacceptably large, or the feature space was deemed to be small enough to have satisfactory interpretability. This iterative training approach could also be used in applications where it is useful to discover which GFA feature is the most impactful to the task at hand, akin to the aim of sparse identification of nonlinear dynamics (SINDy) [152].

GFA, as presented in this chapter, is readily applicable to all ultrasonic NDE image analysis. If windowing around one defect indication per image is not possible, the iterative fitting of more Gaussians can be used to better capture the useful information. In general, fitting functions to NDE data to reduce its dimensionality is an approach that is generalizable to other modalities and data structures (e.g., electromagnetic NDE data and ultrasonic B-Scans) and increasingly, a computationally tractable task. 2D elliptical Gaussians, as used in GFA, are effective for ultrasonic images, but using a different function will be necessary when defect indications are a significantly different shape.

Chapter 7. Conclusion

7.1. Review of thesis

This thesis has investigated several solutions to the major barriers obstructing the application of ML to NDE. Defect sizing for inline pipe inspection using ultrasonic PWI images has been used as an example industrial application throughout.

First, Chapter 3 demonstrated how deep learning with simulated training sets can bypass the requirement for feature engineering and large data sets of defect data, respectively. An efficient, hybrid FE and ray-based simulation was used to train a CNN to characterise real defects. The CNN uses four plane wave images from two arrays and was applied to the characterization of cracks of length 1-5 mm and inclined at angles of up to 20° from the vertical. A standard image-based sizing technique, the 6 dB drop method, was used as a comparison point. For the 6 dB drop method the average absolute error in length and angle prediction is $\pm$1.1 mm, $\pm$8.6° while the CNN is almost four times more accurate at $\pm$0.29 mm, $\pm$2.9°. To demonstrate the adaptability of the deep-learning approach, an error in sound speed estimation was included in the training and test set. With a maximum error of 10% in shear and longitudinal sound speed the 6 dB drop method has an average error of $\pm$1.5 mm, $\pm$12° while the CNN has ±0.45 mm, ±3.0°. This demonstrates far superior crack characterization accuracy by using deep learning rather than traditional image-based sizing. However, a simulation can neither be completely accurate, nor capture all variability present in the real inspection. This means that the experimental and simulated data will be from different (but related) distributions, leading to possible sizing errors. Chapter 4 investigated how to tackle this problem through the use of Domain Adaptation (DA). Three DA methods across varying sizes of experimental training data (using between one and fifteen notches) were compared to two non-DA methods as a baseline. Of the DA methods investigated, an adversarial approach was found to be the most effective way to use the limited experimental training data. With this method, and only three notches, the resulting network's RMSE was improved by 23% compared to using only simulated data and 67% compared to using only experimental data.

Chapter 5 investigated how UQ can best be achieved for deep learning in ultrasonic crack sizing. This is essential for qualifying NDE inspections and building trust in their predictions. Two modern UQ methods were used: deep ensembles and MC dropout. Successful UQ was judged by calibration and anomaly detection, which refer to whether in-domain model error is proportional to uncertainty and if out of training domain data is assigned high uncertainty, respectively. Calibration was tested using simulated and experimental images of surface breaking cracks, while anomaly detection was tested using experimental side drilled holes and simulated embedded cracks. MC dropout demonstrated poor uncertainty quantification with little separation between in and out-of-distribution data and a weak linear fit ($R = 0.84$) between experimental root mean squared error and uncertainty. Deep ensembles improved upon MC dropout in both calibration (R=0.95) and anomaly detection. Adding spectral

normalization and residual connections to deep ensembles slightly improved calibration ($R = 0.98$) and significantly improved the reliability of assigning high uncertainty to out-of-distribution samples.

Chapter 6 also focused on building trust in the predictions of ML models. Improved interpretability and explainability were achieved through the use of GFA, a novel dimensionality reduction method. GFA involves fitting a 2D elliptical Gaussian function to an ultrasonic image and storing the seven parameters that describe each Gaussian. These seven parameters can then be used as inputs to data analysis methods such as the defect sizing neural network presented in Chapter 6. This approach was compared to sizing with the same neural network and two other dimensionality reduction methods (the parameters of 6 dB drop boxes and PCA), as well as a CNN applied to raw ultrasonic images. Of the dimensionality reduction methods tested, GFA features produced the closest sizing accuracy to sizing from the raw images, with only a 23% increase in RMSE, despite a 96.5% reduction in the dimensionality of the input data. Implementing ML with GFA features as inputs is implicitly more interpretable than using raw images or PCA components as inputs and gives significantly more sizing accuracy than 6 dB drop boxes. Explainability was achieved by using Shapley additive explanations (SHAP) to calculate how each feature contributes to the prediction of an individual defect's length. Analysis of SHAP values demonstrated that the GFA-based neural network proposed displays many of the same relationships between the properties of a defect indication and its predicted size as occur in traditional NDE sizing methods

7.2. Summary of findings

7.2.1. CNN based crack sizing

Automated defect sizing for ultrasonic inline pipe inspection can be achieved using a CNN with PWI images as input. This approach is significantly more accurate than the traditional 6 dB drop method but relies heavily on the size and appropriateness of the training set. Known variations in inspection conditions, such as in changes in sound speeds, can be accounted for by including them in the training set.

7.2.2. Sources of training data

Using a hybrid FE/analytical simulation is an effective way to efficiently generate the large training sets required by deep learning for NDE.

If even a small amount of experimental data is available for use in training, this can significantly reduce the effects of domain shift, and thus improve accuracy. An adversarial training approach makes best use of the available experimental data.

7.2.3. Deep ensembles for uncertainty quantification

Deep ensembles are an effective method for adding UQ to the predictions of ML for NDE. The resulting UQ is calibrated for in distribution samples and can effectively detect anomalous inputs. In the presence of domain shift (e.g., differences between a simulated training set and experimental test samples) deep ensemble's UQ is proportional to expected error but not with a 1:1 mapping. This means that relative uncertainty between test samples can be calculated. 'Temperature scaling' (i.e., calibration), using a validation set, is required to achieve quantification of expected error.

7.2.4. Opening the machine learning 'black box'

Deep learning that uses image or time domain NDE data as input is uninterpretable, and explaining predictions using saliency maps usually unhelpful. Dimensionality reduction can provide increased interpretability. Using properties of a defect indication that are relevant to an NDE inspector as input features offers significant interpretability and explainability benefits. This can be achieved with GFA. Defect sizing using GFA features input to a dense neural network is almost as accurate as using raw ultrasonic images and a CNN, and is significantly more accurate than traditional NDE sizing methods.

7.3. Suggestions for future work

7.3.1. Learning interpretable features

Fitting a shape to ultrasonic images (as GFA does) provides a general-purpose way to extract interpretable and informative features, but still discards a lot of the information present in the raw images, and high-performing data analysis is reliant on choosing an appropriate shape. A CNN learns optimal features from the training set but is not interpretable. Ideally, performative and interpretable features could be learnt from the training set. One route to achieving this would be to fit a large number of shapes to the training data and enforce activation sparsity on the first layer of the sizing neural network. This is akin to the aim of sparse identification of nonlinear dynamics (SINDy) [152]. Another approach worth investigating would be to build a CNN with restrictions/regularisation applied to the filters, ensuring that they could only apply interpretable transformations. A framework to achieve this kind of regularisation does not currently exist, as it is challenging to mathematically define the interpretability of a transformation.

7.3.2. Standardised training sets for NDE

For computer vision there are many publicly available data sets such as ImageNet [65] and CIFAR-10 [66] which can be downloaded and used to develop and evaluate ML algorithms. This lowers the barrier to research that comes with having to gather and label data sets, makes results easy to compare with other work that uses the same data set, and unlocks the ability to use pre-trained networks and transfer learning. There are a small number of large, publicly available data sets for NDE, such as GDXray [71] for X-ray testing and USimgAIST [70] for ultrasonic wave propagation, but these are both quite limited

in scope and small in size comparative to most computer vision data sets. Due to the diversity of NDE inspections it is likely impossible to form a 'one size fits all' data set, but large publicly available data sets for the main inspection modalities would be of great value to the field. Data fusion across modalities could also be achieved with standardization of data set's metadata (i.e., component geometries, inspection conditions etc.).

7.3.3. Incorporating NDE knowledge into ML

There is growing research interest in incorporating domain specific knowledge into ML using physics informed neural networks (PINNs) [153]. To date, the application of PINNs has been limited to problems that can be expressed as solutions to partial differential equations [154], [155]. This has been implemented for ultrasonic NDE crack characterisation by solving for sound speed in the wave equation, using surface wave measurements [156]. However, there is much more information and knowledge associated with every inspection than just the wave equation (e.g., structural geometry, material properties, likely defect types, inspection conditions). An interesting research goal would be a general purpose NDE-ML approach that could learn to solve any NDE inverse problem, while making use of all prior knowledge about the inspection, understanding of physics, and large sets of varied training data. This seems like a 'moon-shot' with today's ML technology, but if achieved it would be transformative for automated data analysis in NDE.

7.3.4. Communication with end-users and standards boards

As well as technical advancements, another key factor in enabling the application of ML to NDE industry is communication between researchers, end-users and standards boards. Research must ensure that it is tackling real problems, rather than 'low hanging fruit' such as lab condition accuracy gains that cannot be replicated reliably in actual inspections, and research will never be implemented if not communicated to the people who could use it.

References

[1] GlobalData Energy, "North America has the highest oil and gas pipeline length globally," 2019. https://www.offshore-technology.com/comment/north-america-has-the-highest-oil-and-gas-pipeline-length-globally/ (accessed Aug. 19, 2022).

[2] North Carolina Dept. of Environmental Quality, "Colonial Pipeline Spill Information - Huntersville, N.C," Jul. 2022. https://deq.nc.gov/about/divisions/waste-management/underground-storage-tanks-section/colonial-pipeline-spill-information-huntersville-nc (accessed Aug. 23, 2022).

[3] R. C. Ireland and C. R. Torres, "Finite element modelling of a circumferential magnetiser," *Sens Actuators A Phys*, vol. 129, no. 1–2, pp. 197–202, 2006.

[4] X. Zhao, V. K. Varma, G. Mei, B. Ayhan, and C. Kwan, "In-line nondestructive inspection of mechanical dents on pipelines with guided shear horizontal wave electromagnetic acoustic transducers," 2005.

[5] K. Reber, M. Beller, H. Willems, and O. A. Barbian, "A new generation of ultrasonic in-line inspection tools for detecting, sizing and locating metal loss and cracks in transmission pipelines," in *2002 IEEE Ultrasonics Symposium, 2002. Proceedings.*, 2002, vol. 1, pp. 665–671.

[6] A. Atto, M. Grigat, and J. Voss, "Continuous Depth Sizing of ILI Ultrasonic Crack Detection," in *International Pipeline Conference*, 2016, vol. 50251, p. V001T03A064.

[7] K. Korol, Y. Hubert, G. Fredine, P. Senf, and S.-A. Koon Koon, "A Study of Crack Detection Ultrasonic Attributes to Manage Leak Threats Associated With Short Crack-Like Flaws in ERW Pipelines," in *International Pipeline Conference*, 2016, vol. 50251, p. V001T03A073.

[8] M. G. Lozev, R. L. Spencer, and D. Hodgkinson, "Optimized inspection of thin-walled pipe welds using advanced ultrasonic techniques," 2005.

[9] R. Rachev, "Advanced ultrasonic array processing for pipeline inline inspection," University of Bristol, 2021.

[10] L. le Jeune, S. Robert, E. L. Villaverde, and C. Prada, "Plane Wave Imaging for ultrasonic non-destructive testing: Generalization to multimodal imaging," *Ultrasonics*, vol. 64, pp. 128–138, 2016.

[11] J. Zhang, B. W. Drinkwater, and P. D. Wilcox, "The use of ultrasonic arrays to characterize crack-like defects," *J Nondestr Eval*, vol. 29, no. 4, pp. 222–232, 2010.

[12] N. G. Meyendorf *et al.*, "NDE 4.0–NDE for the 21st century–the internet of things and cyber physical systems will revolutionize NDE," in *15th Asia Pacific conference for non-destructive testing (APCNDT 2017), Singapore*, 2017.

[13] J. Vrana and R. Singh, "NDE 4.0—a design thinking perspective," *J Nondestr Eval*, vol. 40, no. 1, pp. 1–24, 2021.

[14] M. Bertovic and I. Virkkunen, "NDE 4.0: new paradigm for the NDE inspection personnel," *Handbook of Nondestructive Evaluation 4.0*, pp. 1–31, 2021.

[15] N. Brierley *et al.*, "Advances in the UK Toward NDE 4.0," *Research in Nondestructive Evaluation*, vol. 31, no. 5–6, pp. 306–324, 2020.

[16] L. Udpa and S. S. Udpa, "Neural networks for the classification of nondestructive evaluation signals," *IEE Proceedings, Part F: Radar and Signal Processing*, vol. 138, no. 1, pp. 41–45, 1991, doi: 10.1049/ip-f-2.1991.0007.

[17] N. Amiri, G. H. Farrahi, K. R. Kashyzadeh, and M. Chizari, "Applications of ultrasonic testing and machine learning methods to predict the static & fatigue behavior of spot-welded joints," *J Manuf Process*, vol. 52, pp. 26–34, Apr. 2020, doi: 10.1016/j.jmapro.2020.01.047.

[18] M. Mishra, A. S. Bhatia, and D. Maity, "Predicting the compressive strength of unreinforced brick masonry using machine learning techniques validated on a case study of a museum through nondestructive testing," *J Civ Struct Health Monit*, pp. 1–15, Mar. 2020, doi: 10.1007/s13349-020-00391-7.

[19] Z. Lin, H. Pan, G. Gui, and C. Yan, "Data-driven structural diagnosis and conditional assessment: from shallow to deep learning," in *Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems 2018*, Mar. 2018, vol. 10598, p. 38. doi: 10.1117/12.2296964.

[20] J. Ye, S. Ito, and N. Toyama, "Computerized ultrasonic imaging inspection: From shallow to deep learning," *Sensors (Switzerland)*, vol. 18, no. 11, Nov. 2018, doi: 10.3390/s18113820.

[21] I. Virkkunen, T. Koskinen, O. Jessen-Juhler, and J. Rinta-Aho, "Augmented ultrasonic data for machine learning," *J Nondestr Eval*, vol. 40, no. 1, pp. 1–11, 2021.

[22] S. Sambath, P. Nagaraj, and N. Selvakumar, "Automatic defect classification in ultrasonic NDT using artificial intelligence," *J Nondestr Eval*, vol. 30, no. 1, pp. 20–28, Mar. 2011, doi: 10.1007/s10921-010-0086-0.

[23] X. L. Travassos, S. L. Avila, and N. Ida, "Artificial neural networks and machine learning techniques applied to ground penetrating radar: A review," *Applied Computing and Informatics*, 2020.

[24] N. J. Shipway, T. J. Barden, P. Huthwaite, and M. J. S. Lowe, "Automated defect detection for fluorescent penetrant inspection using random forest," *NDT & E International*, vol. 101, pp. 113–123, 2019.

[25] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep learning for computer vision: A brief review," *Comput Intell Neurosci*, vol. 2018, 2018.

[26] A. S. Lundervold and A. Lundervold, "An overview of deep learning in medical imaging focusing on MRI," *Z Med Phys*, vol. 29, no. 2, pp. 102–127, 2019.

[27] K. Suzuki, "Overview of deep learning in medical imaging," *Radiological Physics and Technology*, vol. 10, no. 3. Springer Tokyo, pp. 257–273, Sep. 01, 2017. doi: 10.1007/s12194-017-0406-5.

[28] P. O'Rorke, S. Morris, M. Amirfathi, W. Bond, and D. St. Clair, "Machine Learning for Nondestructive Evaluation," in *Machine Learning Proceedings 1991*, Elsevier, 1991, pp. 620–624. doi: 10.1016/b978-1-55860-200-7.50126-4.

[29] J. B. Harley and D. Sparkman, "Machine learning and NDE: Past, present, and future," in *AIP Conference Proceedings*, May 2019, vol. 2102, no. 1, p. 090001. doi: 10.1063/1.5099819.

[30] A. Bernieri, L. Ferrigno, M. Laracca, and M. Molinara, "Crack shape reconstruction in Eddy current testing using machine learning systems for regression," *IEEE Trans Instrum Meas*, vol. 57, no. 9, pp. 1958–1968, 2008, doi: 10.1109/TIM.2008.919011.

[31] J. Feng, F. Li, S. Lu, J. Liu, and D. Ma, "Injurious or noninjurious defect identification from MFL images in pipeline inspection using convolutional neural network," *IEEE Trans Instrum Meas*, vol. 66, no. 7, pp. 1883–1892, 2017.

[32] Y.-J. Cha, W. Choi, and O. Büyüköztürk, "Deep Learning-Based Crack Damage Detection Using Convolutional Neural Networks," *Computer-Aided Civil and Infrastructure Engineering*, vol. 32, no. 5, pp. 361–378, May 2017, doi: 10.1111/mice.12263.

[33] M. Meng, Y. J. Chua, E. Wouterson, and C. P. K. Ong, "Ultrasonic signal classification and imaging system for composite materials via deep convolutional neural networks," *Neurocomputing*, vol. 257, pp. 128–135, 2017.

[34] R. J. Pyle, R. L. T. Bevan, R. R. Hughes, R. K. Rachev, A. Ait Si Ali, and P. D. Wilcox, "Deep Learning for Ultrasonic Crack Characterization in NDE," *IEEE Trans Ultrason Ferroelectr Freq Control*, vol. 68, no. 5, pp. 1854–1865, 2020, doi: 10.1109/TUFFC.2020.3045847.

[35] R. McGill and P. Kenneth, "Solution of variational problems by means of a generalized Newton-Raphson operator," *Aiaa Journal*, vol. 2, no. 10, pp. 1761–1766, 1964.

[36] P. D. Wilcox and A. Velichko, "Efficient frequency-domain finite element modeling of two-dimensional elastodynamic scattering," *J Acoust Soc Am*, vol. 127, no. 1, pp. 155–165, Jan. 2010, doi: 10.1121/1.3270390.

[37] L. W. Schmerr, *Fundamentals of ultrasonic nondestructive evaluation*. Springer, 2016.

[38] H. A. Bloxham, A. Velichko, and P. D. Wilcox, "Combining simulated and experimental data to simulate ultrasonic array data from defects in materials with high structural noise," *IEEE Trans Ultrason Ferroelectr Freq Control*, vol. 63, no. 12, pp. 2198–2206, 2016.

[39] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A comprehensive review," *Neural Computation*, vol. 29, no. 9. MIT Press Journals, pp. 2352–2449, Sep. 01, 2017. doi: 10.1162/NECO_a_00990.

[40] A. Bhandare, M. Bhide, P. Gokhale, and R. Chandavarkar, "Applications of convolutional neural networks," *International Journal of Computer Science and Information Technologies*, vol. 7, no. 5, pp. 2206–2215, 2016.

[41] B. Kayalibay, G. Jensen, and P. van der Smagt, "CNN-based segmentation of medical imaging data," *arXiv preprint arXiv:1701.03056*, 2017.

[42] Q. Li, W. Cai, X. Wang, Y. Zhou, D. D. Feng, and M. Chen, "Medical image classification with convolutional neural network," in *2014 13th international conference on control automation robotics & vision (ICARCV)*, 2014, pp. 844–848.

[43] H.-C. Shin *et al.*, "Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning," *IEEE Trans Med Imaging*, vol. 35, no. 5, pp. 1285–1298, 2016.

[44] H. wei Huang, Q. tong Li, and D. ming Zhang, "Deep learning based image recognition for crack and leakage defects of metro shield tunnel," *Tunnelling and Underground Space Technology*, vol. 77, pp. 166–176, Jul. 2018, doi: 10.1016/j.tust.2018.04.002.

[45] J. C. Aldrin and D. S. Forsyth, "Demonstration of using signal feature extraction and deep learning neural networks with ultrasonic data for detecting challenging discontinuities in

composite panels," in *AIP Conference Proceedings*, May 2019, vol. 2102, no. 1. doi: 10.1063/1.5099716.

[46] N. J. Shipway, P. Huthwaite, M. J. S. Lowe, and T. J. Barden, "Using ResNets to perform automated defect detection for Fluorescent Penetrant Inspection," *NDT & E International*, vol. 119, p. 102400, 2021.

[47] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," *arXiv preprint arXiv:1901.06032*, 2019.

[48] M. V. Felice, "Ultrasonic array inspections for complex defects." University of Bristol, 2015.

[49] M. V Felice and Z. Fan, "Sizing of flaws using ultrasonic bulk wave testing: A review," *Ultrasonics*, vol. 88, pp. 26–42, 2018.

[50] L. Bai, A. Velichko, and B. W. Drinkwater, "Ultrasonic characterization of crack-like defects using scattering matrix similarity metrics," *IEEE Trans Ultrason Ferroelectr Freq Control*, vol. 62, no. 3, pp. 545–559, 2015.

[51] J. Krautkrämer and H. Krautkrämer, "Ultrasonic testing by determination of material properties," in *Ultrasonic Testing of Materials*, 4th ed., Springer, 1990, pp. 319–326.

[52] M. Z. Alom *et al.*, "The History Began from AlexNet: A Comprehensive Survey on Deep Learning Approaches," Mar. 2018.

[53] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, Dec. 2015.

[54] Google, "Machine Learning Glossary | Google Developers," *Machine Learning Crash Course*. https://developers.google.com/machine-learning/glossary (accessed Jun. 10, 2020).

[55] D. R. Wilson and T. R. Martinez, "The general inefficiency of batch training for gradient descent learning," *Neural networks*, vol. 16, no. 10, pp. 1429–1451, 2003.

[56] Z. Liu, J. Zhao, B. Wu, and C. He, "Temperature dependence of ultrasonic longitudinal guided wave propagation in long range steel strands," *Chinese Journal of Mechanical Engineering (English Edition)*, vol. 24, no. 3, pp. 487–494, May 2011, doi: 10.3901/CJME.2011.03.487.

[57] N. I. Uzelac, K. Reber, M. Belter, and O. A. Barbian, "Ultrasonic In-Line Inspection of Pipelines, New Generation of Tools," in *Rio Pipeline Conference*, 2003.

[58] Q. Zhang, Y. Nian Wu, and S.-C. Zhu, "Interpretable convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8827–8836.

[59] C. Olah, A. Mordvintsev, and L. Schubert, "Feature Visualization," *Distill*, vol. 2, no. 11, p. e7, Nov. 2017, doi: 10.23915/distill.00007.

[60] S. Carter, Z. Armstrong, L. Schubert, I. Johnson, and C. Olah, "Activation Atlas," *Distill*, vol. 4, no. 3, p. e15, Mar. 2019, doi: 10.23915/distill.00015.

[61] R. J. Pyle, R. L. T. Bevan, R. R. Hughes, A. A. S. Ali, and P. D. Wilcox, "Domain Adapted Deep-Learning for Improved Ultrasonic Crack Characterization Using Limited Experimental Data," *IEEE Trans Ultrason Ferroelectr Freq Control*, vol. 69, no. 4, pp. 1485–1496, 2022, doi: 10.1109/TUFFC.2022.3151397.

[62] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans Knowl Data Eng*, vol. 22, no. 10, pp. 1345–1359, 2009.

[63] M. Wang and W. Deng, "Deep visual domain adaptation: A survey," *Neurocomputing*, vol. 312, pp. 135–153, 2018.

[64] G. Csurka, "Domain adaptation for visual applications: A comprehensive survey," *arXiv preprint arXiv:1702.05374*, 2017.

[65] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, 2009, pp. 248–255.

[66] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," 2009.

[67] A. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, 2011, pp. 142–150.

[68] G. A. Miller, *WordNet: An electronic lexical database*. MIT press, 1998.

[69] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, 2015, pp. 5206–5210.

[70] J. Ye and N. Toyama, "Benchmarking Deep Learning Models for Automatic Ultrasonic Imaging Inspection," *IEEE Access*, vol. 9, pp. 36986–36994, 2021.

[71] D. Mery *et al.*, "GDXray: The database of X-ray images for nondestructive testing," *J Nondestr Eval*, vol. 34, no. 4, p. 42, 2015.

[72] D. Chakraborty, N. Kovvali, B. Chakraborty, A. Papandreou-Suppappola, and A. Chattopadhyay, "Structural damage detection with insufficient data using transfer learning techniques," in *Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems 2011*, 2011, vol. 7981, p. 798147.

[73] P. Gardner *et al.*, "Machine learning at the interface of structural health monitoring and non-destructive evaluation," *Philosophical Transactions of the Royal Society A*, vol. 378, no. 2182, p. 20190581, 2020.

[74] P. Gardner, X. Liu, and K. Worden, "On the application of domain adaptation in structural health monitoring," *Mech Syst Signal Process*, vol. 138, p. 106550, 2020.

[75] H. Daumé III, A. Kumar, and A. Saha, "Frustratingly easy semi-supervised domain adaptation," in *Proceedings of the 2010 Workshop on Domain Adaptation for Natural Language Processing*, 2010, pp. 53–59.

[76] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Trans Neural Netw*, vol. 22, no. 2, pp. 199–210, 2010.

[77] S. Motiian, M. Piccirilli, D. A. Adjeroh, and G. Doretto, "Unified deep supervised domain adaptation and generalization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 5715–5725.

[78] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7167–7176.

[79] S. Motiian, Q. Jones, S. Iranmanesh, and G. Doretto, "Few-shot adversarial domain adaptation," in *Advances in Neural Information Processing Systems*, 2017, pp. 6670–6680.

[80] A. Singh and S. Chakraborty, "Deep Domain Adaptation for Regression," in *Development and Analysis of Deep Learning Architectures*, W. Pedrycz and S.-M. Chen, Eds. Cham: Springer International Publishing, 2020, pp. 91–115. doi: 10.1007/978-3-030-31764-5_4.

[81] T. Latête, B. Gauthier, and P. Belanger, "Towards using convolutional neural network to locate, identify and size defects in phased array ultrasonic testing," *Ultrasonics*, vol. 115, p. 106436, 2021.

[82] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," *arXiv preprint arXiv:1405.3531*, 2014.

[83] J. Donahue *et al.*, "Decaf: A deep convolutional activation feature for generic visual recognition," in *International conference on machine learning*, 2014, pp. 647–655.

[84] scikit-learn, "sklearn.utils.class_weight.compute_class_weight," 2020. https://scikit-learn.org/stable/modules/generated/sklearn.utils.class_weight.compute_class_weight.html (accessed May 06, 2021).

[85] C. C. Aggarwal, A. Hinneburg, and D. A. Keim, "On the surprising behavior of distance metrics in high dimensional space," in *International conference on database theory*, 2001, pp. 420–434.

[86] M. Köppen, "The curse of dimensionality," in *5th Online World Conference on Soft Computing in Industrial Applications (WSC5)*, 2000, vol. 1, pp. 4–8.

[87] R. J. Pyle, R. R. Hughes, A. A. S. Ali, and P. D. Wilcox, "Uncertainty Quantification for Deep Learning in Ultrasonic Crack Characterization," *IEEE Trans Ultrason Ferroelectr Freq Control*, vol. 69, no. 7, pp. 2339–2351, 2022, doi: 10.1109/TUFFC.2022.3176926.

[88] M.-H. DOD, "Department of Defense Handbook: Nondestructive Evaluation System Reliability Assessment," *Department of Defense, Washington, DC*, 2009.

[89] A. P. Dawid, "The well-calibrated Bayesian," *J Am Stat Assoc*, vol. 77, no. 379, pp. 605–610, 1982.

[90] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," *arXiv preprint arXiv:1612.01474*, 2016.

[91] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *international conference on machine learning*, 2016, pp. 1050–1059.

[92] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[93] M. Abdar *et al.*, "A review of uncertainty quantification in deep learning: Techniques, applications and challenges," *Information Fusion*, 2021.

[94] P. Zhu, Y. Cheng, P. Banerjee, A. Tamburrino, and Y. Deng, "A novel machine learning model for eddy current testing with uncertainty," *NDT & E International*, vol. 101, pp. 104–112, 2019.

[95] S. O. Sajedi and X. Liang, "Uncertainty-assisted deep vision structural health monitoring," *Computer-Aided Civil and Infrastructure Engineering*, vol. 36, no. 2, pp. 126–142, 2021.

[96] F. Richter, "Mixture Density Networks," *Kombination Künstlicher Neuronaler Netze*, pp. 171–198, 2003, doi: 10.1007/978-3-322-81570-5_8.

[97] I. D. Khurjekar and J. B. Harley, "Uncertainty aware deep neural network for multistatic localization with application to ultrasonic structural health monitoring," *arXiv preprint arXiv:2007.06814*, 2020.

[98] W. Chen, Y. Gao, L. Gao, and X. Li, "A new ensemble approach based on deep convolutional neural networks for steel surface defect classification," *Procedia CIRP*, vol. 72, pp. 1069–1072, 2018.

[99] M. Marino, K. Virupakshappa, and E. Oruklu, "A Stacked Ensemble Neural Network Classifier for Ultrasonic Non-Destructive Evaluation Applications," in *2020 IEEE International Ultrasonics Symposium (IUS)*, 2020, pp. 1–4.

[100] F. Chang, M. Liu, M. Dong, and Y. Duan, "A mobile vision inspection system for tiny defect detection on smooth car-body surfaces based on deep ensemble learning," *Meas Sci Technol*, vol. 30, no. 12, p. 125905, 2019.

[101] J. Bradshaw, A. G. de G. Matthews, and Z. Ghahramani, "Adversarial examples, uncertainty, and transfer testing robustness in gaussian process hybrid deep networks," *arXiv preprint arXiv:1707.02476*, 2017.

[102] J. van Amersfoort, L. Smith, A. Jesson, O. Key, and Y. Gal, "On Feature Collapse and Deep Kernel Learning for Single Forward Pass Uncertainty," *arXiv preprint arXiv:2102.11409*, 2021.

[103] G. E. Hinton and D. van Camp, "Keeping the neural networks simple by minimizing the description length of the weights," in *Proceedings of the sixth annual conference on Computational learning theory*, 1993, pp. 5–13.

[104] A. Graves, "Practical variational inference for neural networks," *Adv Neural Inf Process Syst*, vol. 24, 2011.

[105] J. Hensman, A. Matthews, and Z. Ghahramani, "Scalable variational Gaussian process classification," in *Artificial Intelligence and Statistics*, 2015, pp. 351–360.

[106] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[107] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[108] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?," in *Advances in neural information processing systems*, 2017, pp. 5574–5584.

[109] C. M. Bishop, “Pattern recognition,” *Mach Learn*, vol. 128, no. 9, 2006.

[110] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[111] T. G. Dietterich, “Ensemble methods in machine learning,” in *International workshop on multiple classifier systems*, 2000, pp. 1–15.

[112] S. Lee, S. Purushwalkam, M. Cogswell, D. Crandall, and D. Batra, “Why M heads are better than one: Training a diverse ensemble of deep networks,” *arXiv preprint arXiv:1511.06314*, 2015.

[113] S. Fort, H. Hu, and B. Lakshminarayanan, “Deep ensembles: A loss landscape perspective,” *arXiv preprint arXiv:1912.02757*, 2019.

[114] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[115] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, “Spectral normalization for generative adversarial networks,” *arXiv preprint arXiv:1802.05957*, 2018.

[116] J. van Amersfoort, L. Smith, Y. W. Teh, and Y. Gal, “Uncertainty estimation using a single deep deterministic neural network,” in *International Conference on Machine Learning*, 2020, pp. 9690–9700.

[117] J. Z. Liu, Z. Lin, S. Padhy, D. Tran, T. Bedrax-Weiss, and B. Lakshminarayanan, “Simple and principled uncertainty estimation with deterministic deep learning via distance awareness,” *arXiv preprint arXiv:2006.10108*, 2020.

[118] J. Mukhoti, A. Kirsch, J. van Amersfoort, P. H. S. Torr, and Y. Gal, “Deterministic neural networks with appropriate inductive biases capture epistemic and aleatoric uncertainty,” *arXiv preprint arXiv:2102.11582*, 2021.

[119] Tensorflow, “tfa.layers.SpectralNormalization,” 2021. https://www.tensorflow.org/addons/api_docs/python/tfa/layers/SpectralNormalization (accessed Sep. 08, 2021).

[120] A. Y. K. Foong, D. R. Burt, Y. Li, and R. E. Turner, “On the expressiveness of approximate inference in bayesian neural networks,” *arXiv preprint arXiv:1909.00719*, 2019.

[121] Y. Ovadia *et al.*, “Can you trust your model’s uncertainty? Evaluating predictive uncertainty under dataset shift,” *arXiv preprint arXiv:1906.02530*, 2019.

[122] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *International Conference on Machine Learning*, 2017, pp. 1321–1330.

[123] G. Huang, Y. Li, G. Pleiss, Z. Liu, J. E. Hopcroft, and K. Q. Weinberger, "Snapshot ensembles: Train 1, get m for free," *arXiv preprint arXiv:1704.00109*, 2017.

[124] U. Dorjsembe, J. H. Lee, B. Choi, and J. W. Song, "Sparsity Increases Uncertainty Estimation in Deep Ensemble," *Computers*, vol. 10, no. 4, p. 54, 2021.

[125] R. Hu, Q. Huang, S. Chang, H. Wang, and J. He, "The MBPEP: a deep ensemble pruning algorithm providing high quality uncertainty prediction," *Applied Intelligence*, vol. 49, no. 8, pp. 2942–2955, 2019.

[126] L. Tran *et al.*, "Hydra: Preserving ensemble diversity for model distillation," *arXiv preprint arXiv:2001.04694*, 2020.

[127] R. J. Pyle, R. R. Hughes, and P. D. Wilcox, "Interpretable & Explainable Machine Learning for Ultrasonic Defect Sizing," *In review for IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, 2022.

[128] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nat Mach Intell*, vol. 1, no. 5, pp. 206–215, 2019.

[129] 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," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 618–626.

[130] M. T. Ribeiro, S. Singh, and C. Guestrin, "" Why should i trust you?' Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1135–1144.

[131] A. Shrikumar, P. Greenside, A. Shcherbina, and A. Kundaje, "Not just a black box: Learning important features through propagating activation differences," *arXiv preprint arXiv:1605.01713*, 2016.

[132] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *International conference on machine learning*, 2017, pp. 3145–3153.

[133] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," *PLoS One*, vol. 10, no. 7, p. e0130140, 2015.

[134] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Adv Neural Inf Process Syst*, vol. 30, 2017.

[135] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal, "Explaining explanations: An overview of interpretability of machine learning," in *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*, 2018, pp. 80–89.

[136] A. Karthikeyan, A. Tiwari, Y. Zhong, and S. T. S. Bukkapatnam, "Explainable AI-infused ultrasonic inspection for internal defect detection," *CIRP Annals*, 2022.

[137] L. Fradkin, S. Uskuplu Altinbasak, and M. Darmon, "Towards Explainable Augmented Intelligence (AI) for Crack Characterization," *Applied Sciences*, vol. 11, no. 22, p. 10867, 2021.

[138] H. Zhang, J. Lin, J. Hua, and T. Tong, "Interpretable convolutional sparse coding method of Lamb waves for damage identification and localization," *Struct Health Monit*, p. 14759217211044806, 2021.

[139] C. Schnur *et al.*, "Towards interpretable machine learning for automated damage detection based on ultrasonic guided waves," *Sensors*, vol. 22, no. 1, p. 406, 2022.

[140] A. Zytek, I. Arnaldo, D. Liu, L. Berti-Equille, and K. Veeramachaneni, "The Need for Interpretable Features," *ACM SIGKDD Explorations Newsletter*, vol. 24, no. 1, pp. 1–13, 2022, doi: 10.1145/3544903.3544905.

[141] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[142] K. Pearson, "LIII. On lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, vol. 2, no. 11, pp. 559–572, 1901.

[143] F. Kherif and A. Latypova, "Principal component analysis," in *Machine Learning*, Elsevier, 2020, pp. 209–225.

[144] SciPy, "scipy.optimize.curve_fit," 2022. https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.least_squares.html (accessed Jul. 12, 2022).

[145] M. A. Branch, T. F. Coleman, and Y. Li, "A subspace, interior, and conjugate gradient method for large-scale bound-constrained minimization problems," *SIAM Journal on Scientific Computing*, vol. 21, no. 1, pp. 1–23, 1999.

[146] L. Shapley, "Quota solutions op n-person games1," *Edited by Emil Artin and Marston Morse*, p. 343, 1953.

[147] A. E. Roth, *The Shapley value: essays in honor of Lloyd S. Shapley*. Cambridge University Press, 1988.

[148] A. Datta, S. Sen, and Y. Zick, "Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems," in *2016 IEEE symposium on security and privacy (SP)*, 2016, pp. 598–617.

[149] S. Lipovetsky and M. Conklin, "Analysis of regression in game theory approach," *Appl Stoch Models Bus Ind*, vol. 17, no. 4, pp. 319–330, 2001.

[150] E. Štrumbelj and I. Kononenko, "Explaining prediction models and individual predictions with feature contributions," *Knowl Inf Syst*, vol. 41, no. 3, pp. 647–665, 2014.

[151] S. Lundberg, "beeswarm plot," 2018. https://shap.readthedocs.io/en/latest/example_notebooks/api_examples/plots/beeswarm.html (accessed Aug. 12, 2022).

[152] S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Discovering governing equations from data by sparse identification of nonlinear dynamical systems," *Proceedings of the national academy of sciences*, vol. 113, no. 15, pp. 3932–3937, 2016.

[153] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *J Comput Phys*, vol. 378, pp. 686–707, 2019, doi: 10.1016/j.jcp.2018.10.045.

[154] S. Cai, Z. Mao, Z. Wang, M. Yin, and G. E. Karniadakis, "Physics-informed neural networks (PINNs) for fluid mechanics: A review," *Acta Mechanica Sinica*, pp. 1–12, 2022.

[155] S. Cai, Z. Wang, S. Wang, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks for heat transfer problems," *J Heat Transfer*, vol. 143, no. 6, 2021.

[156] K. Shukla, P. C. di Leoni, J. Blackshire, D. Sparkman, and G. E. Karniadakis, "Physics-informed neural network for ultrasound nondestructive quantification of surface breaking cracks," *J Nondestr Eval*, vol. 39, no. 3, pp. 1–20, 2020.