Ultrasonics 152 (2025) 107639

Contents lists available at ScienceDirect

Ultrasonics

journal homepage: www.elsevier.com/locate/ultras

On the use of a Transformer Neural Network to deconvolve ultrasonic signals

T. Sendra *, P. Belanger

Department of Mechanics, Ecole de Technologie Superieure, 1100 Notre-Dame Street West, Montreal, H3C 1K3, QC, Canada

ARTICLE INFO

Keywords:
Convolutional neural network
Corrosion
Deconvolution
Non-Destructive Evaluation
Time-of-flight
Thickness measurement
Transformer Neural Network
Ultrasonic testing

ABSTRACT

Pulse-echo ultrasonic techniques play a crucial role in assessing wall thickness deterioration in safety-critical industries. Current approaches face limitations with low signal-to-noise ratios, weak echoes, or vague echo patterns typical of heavily corroded profiles. This study proposes a novel combination of Convolution Neural Networks (CNN) and Transformer Neural Networks (TNN) to improve thickness gauging accuracy for complex geometries and echo patterns. Recognizing the strength of TNN in language processing and speech recognition, the proposed network comprises three modules: 1. pre-processing CNN, 2. a Transformer model and 3. a post-processing CNN. Two datasets, one being simulation-generated, and the other, experimentally gathered from a corroded carbon steel staircase specimen, support the training and testing processes. Results indicate that the proposed model outperforms other AI architectures and traditional methods, providing a 5.45% improvement over CNN architectures from NDE literature, a 1.81% improvement over ResNet-50, and a 17.5% improvement compared to conventional thresholding techniques in accurately detecting depths with a precision under 0.5λ.

1. Introduction

Pulse-echo ultrasonic approaches have been widely adopted for the monitoring of wall thickness deterioration due to aging and corrosion in various sectors, such as the oil and gas [1,2], offshore [3], and energy industries [4]. Accurate and reliable wall thickness measurements facilitate informed decisions for the safe operation of critical assets. For thickness evaluation, an emitted ultrasonic pulse is transmitted by an ultrasonic transducer, and propagates through the material. The wave reflects on the backwall or any other sudden acoustic impedance changes along the propagation path. The reflected waves are then captured by the transducer. Often referred to as an A-scan, the recorded signal informs users of pertinent features within the inspected material. Meanwhile, computing the time difference between emission and reception – commonly known as the Time-of-Flight (ToF) – allows to calculate the wall thickness [5] if the speed of sound is known. Therefore, accurately determining the ToF in order to properly assess the safety of pipeline thickness, for instance, is crucial. However, corrosion profiles may produce complex backwall geometries, leading to variable echo amplitudes and patterns resulting from overlapping reflections coming from different depths. Constructive and destructive interferences add a layer of complication by obscuring true distances within unwanted information. These highly complex geometries thus introduce challenges, such as unknown echo patterns or overlapping echoes, as shown in Fig. 1. Accurately identifying the correct ToF to calculate the minimal distance to the probe remains a critical unsolved problem in nondestructive evaluation (NDE).

Overlapping echoes have been extensively scrutinized in NDE. A first approach used to tackle them consists in increasing the frequency of the transducer [6] to increase the resolution, but attenuation follows the same trend, leading to a reduced amplitude for a given sample thickness [7,8]. Increasing the amplitude is hardly a solution because coherent noise will maintain the overall signal-to-noise ratio (SNR) [8].

Over the years, cross-correlation [9], threshold-crossing [10] or sparse and blind deconvolution [11,12] algorithms, among others, have been proposed to improve the resolution of ultrasonic thickness measurements. While those methods are effective in most cases, they become limited with low SNR, attenuated echoes or unknown echo signatures.

Sparse signal representation methods, also known as dictionary-based methods, such as matching-pursuit [13], orthogonal matching-pursuit [14], and most recently, support matching pursuit [15], are other possible approaches for tackling the problem. Matching echo patterns to decompose the signals effectively separates the echoes and allows a global understanding of the A-scan's composition. However, as stated before, irregular corrosion profiles lead to highly variable echo patterns. There is therefore the need for a large dictionary of such patterns, which reduces efficiency and performance.

Advances in computational capabilities – specifically, boosts in GPU and CPU capacities and speeds – have rendered deep learning a feasible industrial solution for complicated tasks such as image classification [16], face recognition [17] and image-to-text generation [18].

* Corresponding author.
E-mail addresses: thibault.sendra.1@ens.etsmtl.ca (T. Sendra), Pierre.Belanger@etsmtl.ca (P. Belanger).

https://doi.org/10.1016/j.ultras.2025.107639
Received 15 November 2024; Received in revised form 3 March 2025; Accepted 12 March 2025
Available online 21 March 2025
0041-624X/© 2025 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

Consequently, deep learning, particularly through convolutional neural network (CNN), has successfully made its way into NDE for defect detection and characterization [19,20]. Indeed, CNN allows to reduce high-dimensional problems and capture local features with its filters using shared weights. Its ability to detect complex patterns have thus been proven through the years in ultrasonic NDE image classification [21].

The ability of CNN to perform inverse operations through its primary convolution blocks is at the root of recent advances in ultrasonic NDE, including signal deconvolution and thickness measurement. For instance, Chapon et al. [22] used a simple CNN architecture consisting of two convolutive layers to effectively deconvolve two overlapping echoes, distinguishing flat-bottom holes up to a depth of 0.5 λ. However, their database lacks diversity, and only allows the signal signature of a single probe to be recognized, with one frequency and a relatively high SNR. Furthermore, an individual A-scan can be composed of an unknown number of superposed echoes, whereas their CNN was solely trained to identify two. Shpigler et al. [23] refined the model by adding distinction layers and augmenting the database with highly variable signals, varying SNRs, and an unknown number of overlapping echoes. The architecture was evaluated and compared on two phantoms with matching pursuit algorithms and was proven to have a higher detection accuracy in high overlap conditions, where the layer thickness can reach 0.25 λ. Yet, the tests were realized on flat geometries and the architecture was not proven to be effective on corroded profiles, where signal signatures can be modified from echo to echo within the same A-scan. As such, Cantero-Chinchilla et al. [24] proposed an improved database with simulated A-scans on randomly generated corroded profiles. An optimal CNN model was developed to capture the minimal and mean wall thicknesses under the transducer, and was proven to provide a four-fold root mean square error improvement relative to the peak of the envelope technique. Nonetheless, the lack of real data hindered efficient and precise wall thickness determination on experimental data acquired on equally difficult geometries.

Due to the limited scope of kernels, CNNs require the superposition of multiple convolutional layers to capture a global context and physical meaning. Unfortunately, this multiplication hampers their initial computational advantage [25]. To tackle this drawback, the Transformer Neural Network (TNN), based on the attention mechanism, was devised, initially for machine translation and language processing [26]. Subsequently, TNNs were broadened to cover signal processing with speech recognition [25,27,28], electrocardiogram signal classification [29,30] and noise reduction [31,32]. This innovative approach outperforms previous architectures in the mentioned disciplines by offering affordable long-range dimensions and interactions to AI models.

More recent research has explored the combination of CNNs with TNNs, yielding significant performance improvements over traditional Transformer architectures. Wu et al. successfully enhanced vision Transformer models by embedding convolutional tokens before the Transformer blocks and incorporating convolutional projections within the Transformer framework [33]. Similarly, Wang et al. took inspiration from encoder–decoder frameworks to propose a CNN encoder–decoder architecture combined with a Transformer block. This approach achieved superior results for monaural speech enhancement in the time domain by effectively extracting both local and global contextual information [32]. Gulati et al. further demonstrated the effectiveness of integrating CNNs with TNNs by incorporating a CNN block within the TNN framework and adding a convolutional subsampling layer prior to the modified Transformer architecture. This approach achieved state-of-the-art accuracy in speech recognition tasks [25]. These successes highlight the benefits of CNN-augmented Transformer architectures, where locality guidance and the added distance limitation to the self-attention mechanism effectively improve convergence and performance with smaller datasets [34].

In this paper, the integration of a TNN for wall thickness measurements was investigated. Inspired by previous research, it was hypothesized that bringing together global and local interactions through a hybrid CNN-TNN architecture will enable accurate and robust wall thickness estimation, regardless of the complexity of the underlying geometry. The proposed architecture was trained with a unique dataset comprising simulations and experimentally measured A-scans on a corroded carbon steel sample. This unique dataset was used to gain valuable insights on the performance of the proposed architecture.

The paper begins with an overview of the materials and methods, and provides details on the generation of the related datasets and architecture. The results are then presented and compared with other architectures and conventional methods available in the literature. Finally, the results are discussed and conclusions are drawn.

2. Materials and methods

2.1. Problem formulation

The fundamental nature of the deconvolution predicament is generally outlined as follows. Given the received signal $y(n)$ as an A-scan, it can be modeled as the mathematical product of $h(n)$, the impulse response, convoluted with $x(n)$, representing the ultrasonic waveform, and contaminated by a zero-mean additive band-limited Gaussian noise $e(n)$: $y(n) = h(n) * x(n) + e(n)$, where $*$ denotes the linear convolution operation [35].

To grasp the form of the impulse response, one can represent it mathematically by a series of Dirac delta functions $\delta(n)$ having diverse amplitudes h_i at distinct moments τ_i, hence forming a “sparse spike train” [36].

$$h(n) = \sum_{i=1}^{\infty} (h_i \cdot \delta(n - \tau_i)) \tag{1}$$

Thus, locating any pair of consecutive τ_i values would allow to measure the ToF accurately. While effective on flat surfaces, it becomes nonviable on rough surfaces where thickness variations follow a continuous function.

In typical NDE inspections, the goal is to establish the minimum thickness under the transducer. To achieve this, the machine learning (ML) architecture will be designed to identify the ultrasonic waveform corresponding to this minimum thickness (Fig. 1). This simplifies the problem to locating the τ_i pairs associated with the ultrasound reflections from the minimal thickness beneath the probe. The resulting subset of the deconvolved signal will be denoted as h'.

This paper's approach, therefore, focuses on approximating the function $f(y) = h'$ by training a CNN-TNN architecture using a learning database of deconvolved A-scans on minimum thicknesses.

2.2. Database generation

At the core of machine learning lies the quality of the database. Having a qualitative and abundant dataset is paramount to successfully train the different AI models. In this paper, simulated and experimental datasets were generated.

2.2.1. Simulation database

Finite element simulations using GPU-accelerated Pogo FEA [37] were performed to obtain a large amount of training data. A 2D plane strain corroded steel block was modeled with a depth $L_z \in [|3;20|]$ mm and a width $L_x = 45$ mm, using 20 linear square elements per the shortest wavelength λ_{shear}, as shown in Fig. 2. The isotropic and homogeneous steel material properties were $E = 205$ GPa, $\nu = 0.29$ and $\rho = 7870$ kg/m^3, implying a longitudinal velocity $V_l = 5842.5$ m/s. To ensure time marching stability, a time step $dt = \frac{\lambda_{shear}}{2\cdot 20\cdot V_l}$ was used.

For a given signal and block generated, Pogo FEA simulates the emission including the beam spread, the propagation, and the reflections of ultrasonic waves throughout the block . However, attenuation and dispersion were not considered, as the experiments were conducted on an isotropic steel bar of a relatively small thickness. The simulation

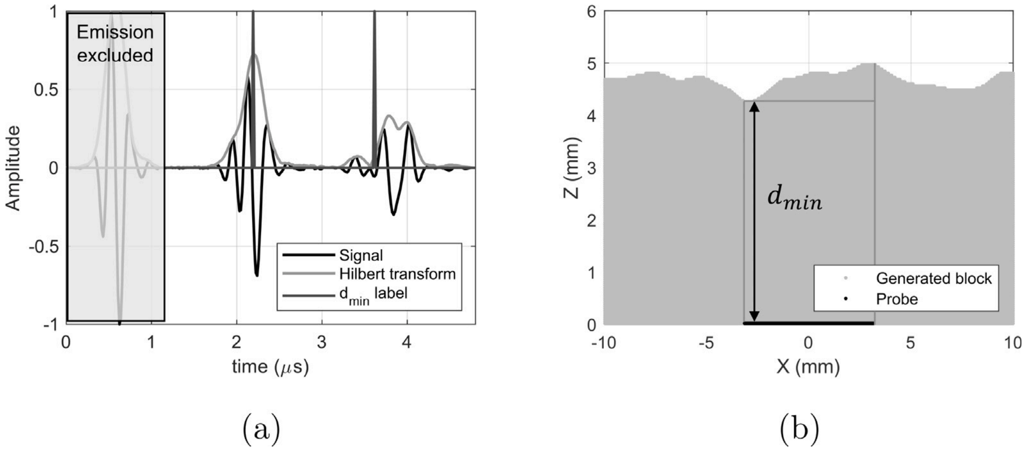


Fig. 1. Example of the labeling methodology for minimal distance under the probe with: (a) the result coming from probe on the generated block (b).

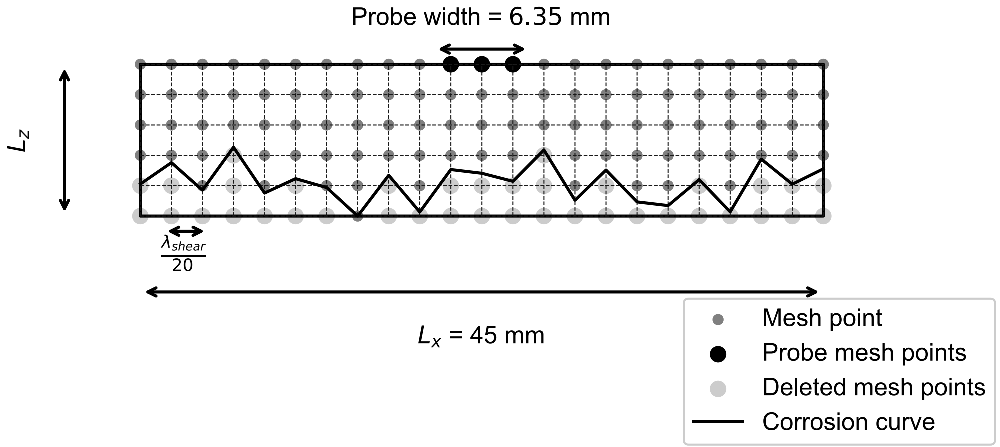


Fig. 2. Schematic of the block generated for simulation with its mesh.

of a probe was performed by selecting the nodes over a width of 6.35 mm. The same input signal was applied to all nodes, as illustrated in Fig. 2.

To accurately simulate real-world applications, emitted waveforms must be representative of typical transducer emission patterns. Li et al. [38] showed that the Nakagami distribution model had the best proximity to echo patterns. While signal generation in the time domain is the standard approach, signal generation in the frequency domain was not discussed. The frequency spectrums of real transducer emissions were acquired on a steel sample, and to match real echo patterns, the spectrums were approximated with Laplace's distributions in the frequency domain. Then, an inverse Fourier Transform was performed to obtain the waveform. The advantages of this method are its flexibility and the ability to easily obtain variables from experimental data, such as μ, the central frequency desired during the emission and β, the bandwidth:

$$W(f_n) = \frac{\exp\left(-\frac{|f_n-\mu|}{\beta}\right)}{2\beta} \tag{2}$$

Finally, a phase-shifted Hann's window was applied to add an asymmetry to the signal:

$$x(n) = \frac{1}{Len} \cdot \cos^2\left(\frac{\pi(n-d)}{Len}\right) \cdot IFFT(W(f_n)) \tag{3}$$

where $Len = \lfloor \frac{(n_{cycles} \cdot F_{sampling})}{\mu} \rfloor$ is the signal's length – with $\lfloor . \rfloor$ being the integer part operator – determined by the sample rate $F_{sampling}$ = 62.5 MHz, number of cycles n_{cycles} and frequency μ, and d the phase-shift.

To conclude, four variables dictate the generated signal's variation and randomness: (1) the central frequency $\mu \in [2;2.5] \cup [4.75;5.25]$ MHz; (2) the bandwidth $\beta \in 1.7 \cdot [0.63;1.37]$; (3) the number of cycles $n_{cycles} \in \{3,5\}$ and (4) the phase-shift $d \in [|-\frac{Len}{3};\frac{Len}{3}|]$. The selected variables were adapted according to the Verasonics Vantage 64 LE and Evident single and multi-element transducer properties, which were used experimentally: V125-RM and 5L64-32X10-A32-P-2.5-OM.

Using this algorithm to approximate the signal emitted by the probes on flat surfaces, an average proximity of 94.86% was achieved using the Normalized Cross-Correlation (NCC) indicator, as described in [38].

To simulate the corrosion on Pogo FEA [37], corrosion profiles were randomly generated with an algorithm adapted from [24] and subtracted to the backwall of the generated 2D block, as illustrated Fig. 2. Firstly, the corrosion profile of the corroded step block used experimentally and described in the next subsection was acquired from its laser scan as shown in Fig. 3a. The properties of that profile, including the amplitudes σ_i at each length frequency λ_i, were calculated and used to generate similar profiles following the algorithm schematized in Fig. 3c. In this paper, the maximum corrosion amplitude σ_{tot}, as specified Fig. 3c, was taken within the set $\{0, 0.2, 0.4, 0.6, 0.8, 1\}$ mm, which is in accordance with the corrosion profiles obtained on the experimental sample (Fig. 4).

The FE simulated results were filtered with a second-order bandpass between 2 and 10 MHz. The data were then normalized by the maximum amplitude to rescale the values within the interval $[-1,1]$:

$$y'(n) = \frac{y(n)}{\max(|y(n)|)} \tag{4}$$

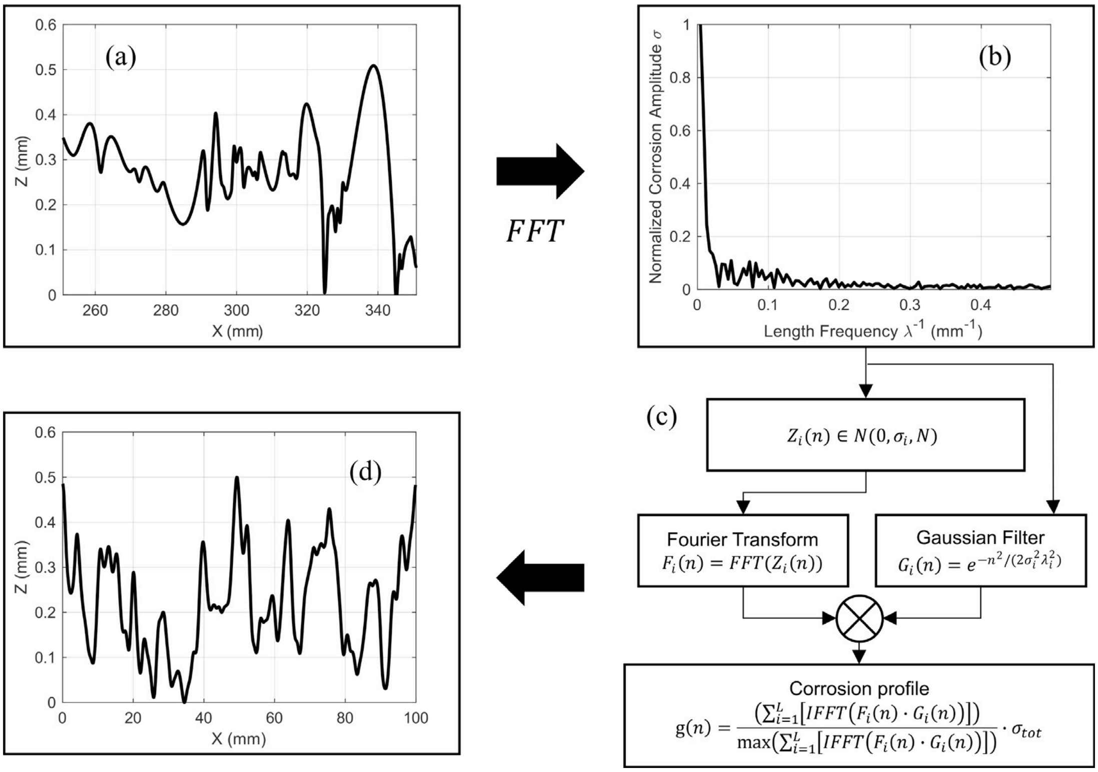

(a)
Z (mm)
X (mm)
0
0.1
0.2
0.3
0.4
0.5
0.6
260
280
300
320
340
FFT
(b)
Normalized Corrosion Amplitude σ
Length Frequency λ^-1 (mm^-1)
0
0.2
0.4
0.6
0.8
1
0
0.1
0.2
0.3
0.4
(c)
$Z_i(n) \in N(0, \sigma_i, N)$
Fourier Transform
$F_i(n) = FFT(Z_i(n))$
Gaussian Filter
$G_i(n) = e^{-n^2/(2\sigma_i^2\lambda_i^2)}$
Corrosion profile
$g(n) = \frac{(\sum_{i=1}^{L}[IFFT(F_i(n) \cdot G_i(n))])}{\max(\sum_{i=1}^{L}[IFFT(F_i(n) \cdot G_i(n))])} \cdot \sigma_{tot}$
(d)
Z (mm)
X (mm)
0
0.1
0.2
0.3
0.4
0.5
0.6
0
20
40
60
80
100

Fig. 3. Schematic of the corrosion profile generation with: (a) a corrosion profile of the corroded block, (b) the FFT of (a), (c) the algorithm used to randomly generate a corrosion profile with similar properties to the experimental profile, and (d) a typical simulated profile.

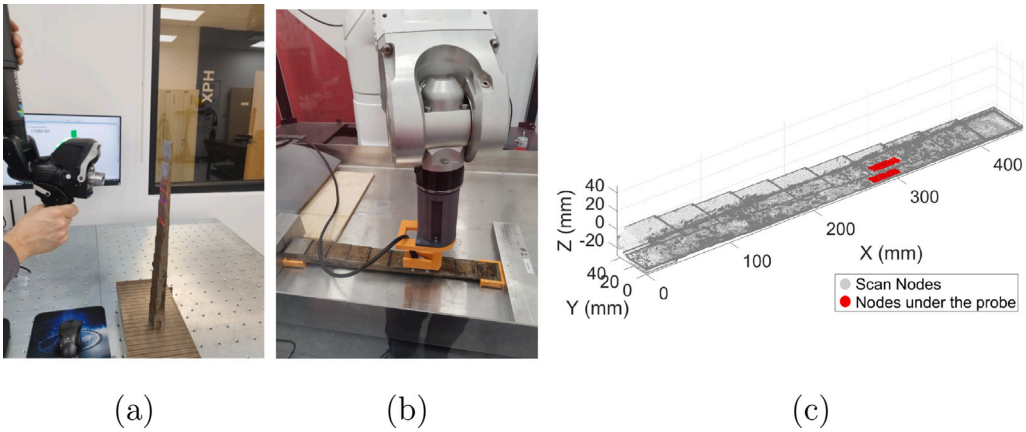


Fig. 4. Pictures of the experimental setup: (a) laser scan of the corroded staircase specimen, (b) example of an acquisition with the multi-element probe mounted on the robot, and (c) the probe's absolute position on the specimen to determine the minimal distances beneath it.

Then, a similarly filtered white Gaussian noise was added, with a SNR varying from 0 to 40 dB.

Finally, a time window and a tapered cosine window, calculated based on the echo size and the maximum thickness beneath the probe, were applied to the simulated data to isolate the first two echoes (Fig. 1). This approach excludes the emitted signal, which, in our case, was not experimentally accessible.

Labeling was performed using the minimum distance from the simulated corrosion profile directly beneath the probe (Fig. 1). Due to experimental constraints, where the emitted signal was inaccessible, the ultrasonic waveform $x(n)$ used as a reference was taken from the first echo. To ensure a consistent position of the first peak across the datasets used for ML training and to provide a clear visual output, the first peak was identified as the maximum of the Hilbert transform of the first echo. The second peak, corresponding to $x(n)$ crossing $2 \cdot d_{min}$, was then calculated based on the first peak using the longitudinal wave velocity V_l in the material and the sampling frequency $F_{sampling}$ (Eq. (6)).

From these calculations, the labeled deconvolution signal, representing h' as described in Section 2.1, was obtained, with an offset introduced by the position of the first reference point.

$$h'(n) = \begin{cases} 1 & \text{if } n = peak_1 \text{ or } n = peak_2 \\ 0 & \text{else} \end{cases} \tag{5}$$

with:

$$\begin{cases} peak_1 = index\{\max[|Hilbert\{y(n)\}|]\} \\ peak_2 = peak_1 + \lfloor \frac{2d_{min} \cdot F_{sampling}}{V_l} \rfloor \end{cases} \tag{6}$$

Table 1
Tabular summary of the datasets distribution.

Datasets	2.25 MHz A-scans	5 MHz A-scans	Total A-scans
Exp. total	162	1728	1890
Exp. train	103	1097	1200
Exp. validation	59	631	690
Sim. total	1377	1623	3000
Sim. train	931	1069	2000
Sim. testing	446	554	1000

Thus, the proposed TNN and CNN models in this paper will have as input the processed signal $y(n)$, and will compare their prediction to the labeled deconvolution $h'(n)$ for training.

2.2.2. Experimental database

In order to test the proposed architecture with experimental data and to verify the need to use experimental data in the training, an experimental database was acquired. An AISI 1018 carbon steel stair block was machined, and its longitudinal speed of sound was measured at $V_l = 5932.1$ m/s. To induce corrosion, the flat face of the block was immersed in saline water to accelerate the corrosion process with electrolysis. A laser scan was subsequently performed, capturing the surface profile with a resolution of 40 μm (Fig. 4a), providing the corrosion parameters required for the simulations (Fig. 3a).

A and B-scans were acquired throughout the block using a 2.25 MHz single element longitudinal probe (Olympus V125-RM) and a 5 MHz 64-element longitudinal probe (Olympus 5L64-32X10-A32-P-2.5-OM). Data acquisition was done using a high-frequency Verasonics Vantage 64 LE system with a sampling rate of 62.5 MHz.

A robotic arm was used to position the mono and multi-element probes on various predefined targets on the staircase sample, with a positioning accuracy of approximately 1 mm thanks to RoboDK software (Fig. 4b). As a result, 162 A-scans and 27 B-scans composed of 64 A-scans each were acquired, uniformly distributed across 9 steps with depths ranging from 5 mm to 25 mm, and a step size increment of 2.5 mm, as illustrated in Fig. 4.

Owning the absolute position of each A and B-scan, alongside the scan of the staircase sample, minimum distances were extracted for labeling purposes (Fig. 4c). labeling adhered to the same criteria as the simulation (Eqs. (5) and (6)). Altogether, a total of 1890 experimental labeled A-scans were acquired (162 from the single element probe and 1728 from the 64-element probe).

2.2.3. Training datasets

To train the architecture in Sections 3.1.1 to 3.1.3, two databases were used. The first consisted of 2000 simulated A-scans, with depths randomly distributed between 2 mm and 20 mm thanks to the corrosion algorithm (Fig. 3). This range allows for varying thicknesses to address cases where traditional methods struggle to detect the minimal thickness. The second one was made out of 1200 A-scans randomly picked from the pool of 1890 experimental tagged A-scans. The 690 unused experimental A-scans were used as validation while testing was done separately using the methodology outlined in Section 2.6 with 1000 simulations. The datasets distribution for training, validation and testing are described in Table 1.

Finally, the dataset composition is analyzed in Section 3.1.4 and the optimal configuration is employed in Section 3.2.

2.3. Deep neural network architecture

The proposed network architecture $f^*(y) : \mathbb{R}^T \to \mathbb{R}^T$, illustrated in Fig. 5, consists of three primary components: a pre-processing module, a transformer module, and a post-processing module. This network takes an A-scan $y(t)$ of length T as input and produces a deconvoluted signal $h'(t)$ of equal duration T. In this part, details of each of these modules are given.

2.3.1. Pre-processing module

Inspired by the architectures of Chapon and Shpigler [22,23], the pre-processing module follows a similar approach, initially employing a convolutional layer with 32 filters of kernel size $k = 157$. This configuration covers the Region of Interest (ROI) given by the product of the number of cycles n_{cycles} and the sampling frequency $F_{sampling}$, all divided by the signal frequency F_{signal}. Keeping in mind the variability in signal frequency and the number of cycles, the maximal ROI was set to be 156.25:

$$ROI = \frac{n_{cycle} \times F_{sampling}}{F_{signal}} \tag{7}$$

This kernel size is fixed for Sections 3.1.2 to 3.1.4. However, the significance of the ROI in determining the first kernel size will be examined in Section 3.1.1.

Multiple filters gather different types of information and interpret various echo signatures, effectively increasing the model's dimensionality. In contrast to the original architectures, the model was enhanced by introducing batch normalization after the convolutional layer to accelerate training and improve generalization [39,40].

To introduce non-linearity to the model, the batch normalization is followed by a Rectified Linear Unit (ReLU) activation function [41]. In addition, to avoid overfitting, a dropout layer is applied [42]. Through testing, an optimal drop rate of $p = 0.3$ was found and used for optimization and comparative analysis. Since the drop rate is a subject that has already been covered in other articles [43], it will not be studied in depth in the present paper.

2.3.2. Transformer encoding module

The transformer module receives processed local features from the pre-processing module and passes them through the Multi-Head Self-Attention (MHSA) block. Shpigler et al. [23] proposed to use 3 convolutional layers of kernel size 5, stride 1 and dilatation 2 to confront each filter's data on the proposed detected echoes. While this method may work locally, the limitation of the kernel size prevents it from effectively contrasting the final deconvolution peaks, making it unable to reliably select the two correct ones [44]. Thus, this paper proposes a transformer architecture similar to the one introduced by Vaswani et al. in 2017 [26]. This setup encourages relating long-range dependencies without resorting to Recurrent Neural Networks (RNN) or serial convolutional layers, conserving computational effort and enhancing global understanding. Since a convolutional layer containing the positional information precedes the transformer, position encoding is not required [33,45].

Inspired by the architectures of [25,26,31], the input goes through the MHSA block, comprised of multiple self-attention units operating together. Each self-attention block projects queries Q_i, keys K_i and values V_i using learned matrix weights $W_i^Q, W_i^K, W_i^V \in \mathbb{R}^{1200\times d_k}$. In self-attention, the matrices Q, K and V correspond to the output of the previous layer:

$$\begin{aligned} Head_i &= SelfAttention\left(Q_i, K_i, V_i\right) \\ &= softmax\left(\frac{Q_i K_i^T}{\sqrt{d_k}}\right) V_i \end{aligned} \tag{8}$$

where $Q_i, K_i, V_i \in \mathbb{R}^{d_{model}\times d_k}$ and $d_k = \frac{d_{model}}{n_{head}}$ using $d_{model} = 32$ from the reference Fig. 5.

$Head_i$ represents the self-attention computed outputs, which are ultimately concatenated and passed through another projection matrix $W^O \in \mathbb{R}^{d_{model}\times 1200}$ to yield the final MHSA output:

$$MHSA(Q, K, V) = concat\left(Head_1, \dots, Head_{n_{head}}\right) W^O \tag{9}$$

Using multiple heads allows to focus on details across various subspace and positions. To capture distinct information within the model, using multiple heads may be useful [26]. For instance, having two heads could help in distinguishing longitudinal waves from transversal ones within an A-scan. Since deconvolution simplicity does not warrant

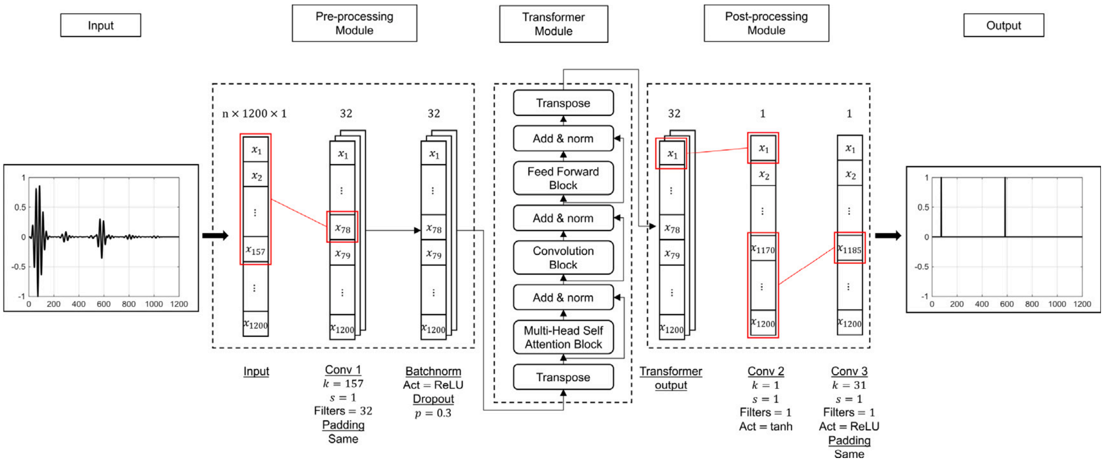


Fig. 5. Proposed CNN-TNN architecture with reference sizes.

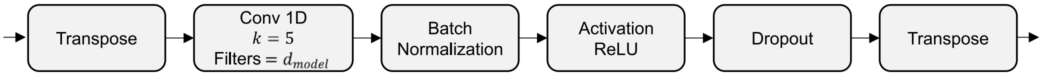


Fig. 6. Convolution Block details.

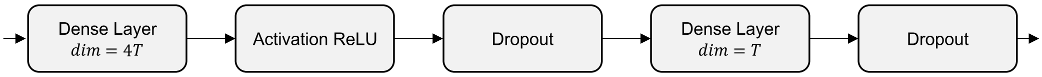


Fig. 7. Feed Forward Block details.

multiple heads, and only longitudinal waves are addressed, a single head $n_{head} = 1$ is deployed to minimize the complexity of the architecture. Finally, a dropout layer with a drop rate of $p = 0.3$ accompanies the MHSA. The same drop rate is used over the transformer module.

Inspired by [25], a convolution-augmented transformer consolidates global information locally, connecting the global and local information processing stages. Wrapping up the transformer module, a Feed Forward Network (FFN) lets parameters within each A-scan interact both locally and globally. The Convolution Block and Feed Forward Block are detailed in Figs. 6 and 7, respectively.

2.3.3. Post-processing module

To conclude, the final module consists of two convolutional layers aimed at deconvolving the signal, condensing all earlier information into a vector. The first layer ends with a hyperbolic tangent (tanh) activation used to capture both positive and negative data, avoiding sigmoids and thereby hastening convergence [46]. Since only positive information is to be seized, the last layer captures the final data to convolve it into a positive vector with a ReLU activation.

Unlike the architectures of Chapon [22] and Shpigler [23], which conclude with a convolutional layer of kernel size $k = 1$ and a ReLU activation, this model introduces an innovative approach. The output is first encoded using the tanh activation, transforming all values into the interval $[-1, 1]$. Subsequently, a final convolutional layer with a kernel size $k = 31$ and ReLU activation refines the information. This larger kernel size enhances the model's capacity to consolidate features over a broader context and accurately construct the final deconvolution peaks.

Finally, the $L_2\!-\!loss$ function compares the output with the expected result, emphasizing errors between estimated and real deconvolution, quickening convergence and lifting the model accuracy [47]. To prevent the occurrence of vanishing gradients and dead neurons linked to the terminal ReLU activation, labeled data got amplified by 1000, yielding two deconvolution peaks of magnitude 1000.

2.4. Training

2.4.1. Training and batching

Using TensorFlow and Keras [48], the architectures were generated and trained for 80 epochs with a training batch size of 32 and a validation batch size of 690, utilizing the datasets from Table 1. Checkpoint callbacks were employed to save each trained model at the point of its highest validation loss, based on the metric defined in Eqs. (13) and (14), throughout the 80 epochs. This approach ensures that, for each trained architecture, the best-performing model is selected for comparison.

2.4.2. Optimizer

The models were trained using the Adam optimizer [49], with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-7}$. The learning rate varied from epoch to epoch according to the formula:

$$\begin{cases} l_{start} + l_{step} \cdot epoch & \text{if } epoch < warmup \\ \frac{l_{start}+l_{step}\cdot warmup}{1+0.1\cdot(epoch-warmup)} & \text{if } epoch \geq warmup \end{cases} \tag{10}$$

The learning rate rose linearly to hit a ceiling before descending inversely proportional to the epoch number. For this article, the values used were $l_{start} = 0.0001$, $l_{step} = 0.0001$ and $warmup = 10$.

2.5. Comparison to other architectures and methods

The proposed architecture was compared with those outlined in NDE-focused publications by Chapon, Shpigler and Cantero-Chinchilla [22–24], ensuring a fair comparison by employing identical databases and conditions. Each architecture was evaluated using the parameters and optimized setups outlined in their respective studies.

To further evaluate the capabilities of the proposed TNN architecture against traditional CNN models, popular architectures such as ResNet-50 [50] and EfficientNet-B0 and B4 [51] were also included in the comparison. Input signals were resized to 224 × 224 images using bilinear interpolation, then converted from grayscale to RGB to match the standard input dimensions of 224 × 224 × 3. The models were implemented as described in their respective papers, with 1200 output classes and a linear activation function to perform the deconvolution, effectively identifying two classes among the 1200 possible ones. Training was conducted using the L2-loss function over 100 epochs with an initial learning rate of 0.001, which was halved if no improvement in loss was observed after 10 epochs.

Additionally, the industry-standard threshold algorithm was used for comparisons with AI techniques, as done in [24]. Using the Hilbert transform as the envelop of the signal, the two rising points crossing the threshold were taken to calculate the ToF [52]. Adaptive thresholds were used according to the signal's frequency for better results :

1. 19% for 2.25 MHz and 38% for 5 MHz for experimental testing
2. 58% for 2.25 MHz and 87% for 5 MHz for simulation testing

2.6. Evaluation methodology

The trained CNN and TNN models, and the threshold algorithm were tested on both experimental and simulation datasets to ensure their adaptability and performance in broad case scenarios. The datasets were separated into 690 unused labeled A-scans from the experimental dataset and 1000 unseen simulated A-scans.

The distance was calculated with the time-of-flight (ToF) metric, provided by the top two AI-derived deconvolution peaks separated by a minimum of 3 mm, $peak_1$ and $peak_2$. Knowing the longitudinal speed V_l and sampling frequency of the machine $F_{sampling}$, we got:

$$d_{predicted} = \frac{\left|peak_1 - peak_2\right|}{2 \cdot F_{sampling}} \cdot V_l \tag{11}$$

Given the true distance d_{exp} for each A-scan, $d_{predicted}$ was compared to d_{exp}. A success was recorded if the difference was lower than a predetermined precision p. Success percentages were acquired based on the cumulative success count across the experimental or simulated datasets.

$$\left|d_{predicted} - d_{exp}\right| \le p \tag{12}$$

The success criterion allows to calculate the success percentage (SP) of each trained model (AI_i) as follows:

$$\mathrm{SP}(AI_i) = \frac{\sum_{k=1}^{N} \begin{cases} 1 & \text{if } \left|d_{predicted_k} - d_{exp_k}\right| \le 0.5\lambda \\ 0 & \text{else} \end{cases}}{N} \tag{13}$$

To comprehensively evaluate every solution, a precision criterion of $p = 0.5\lambda$ was imposed (1.32 mm for 2.25 MHz and 0.59 mm for 5 MHz), as per the minimal axial resolution described in [53], serving as a benchmark for assessing the performance in both simulations and real-world scenarios. For each configuration or architecture tested, five AI models were trained under the same conditions. The success rate for each configuration was then computed as the average of the success percentages of the five trained models:

$$\text{Success Rate} = \frac{\sum_{i=1}^{5}(\mathrm{SP}(AI_i))}{5} \tag{14}$$

To provide a more robust representation of the results, error bars were added to each success rate, representing the Margin of Error of the success percentages across the five evaluated models, with a confidence level of 95%:

$$\text{Margin of Error} = 1.96 \cdot \frac{\sigma\left(\{\mathrm{SP}(AI_i)\}_{i\in[|1,5|]}\right)}{\sqrt{5}} \tag{15}$$

3. Results

3.1. Transformer study

An efficient utilization and maximum exploitation of the Transformer architecture require a thorough investigation of its key parameters such as the kernel size, the overall size, and layer implementations, on unseen simulation and real-world datasets. This strategic approach seeks to underline the correlations between the model's inner workings and external performances. Parallel to this study, a comprehensive survey of the database size and composition will be presented. This critical appraisal provides essential insights into the data requirements for training a proficient deconvolution CNN-TNN model, ensuring an efficient training with available resources.

3.1.1. Convolution kernel size

To investigate the significance of the pre-processing module in capturing the local context through echo signatures, the kernel size of the conv 1 layer from Fig. 5 was studied, as shown in Fig. 8a. Its influence on the overall structure is examined by varying it from 1 to 301.

Similarly, the conv 3 layer in the post-processing module being the core of the deconvolution process, its understanding is necessary for a complete survey of the architecture. By keeping the architecture shown in Fig. 5, a kernel size scan from 15 to 157 was performed, with results presented in Fig. 8b.

3.1.2. Architecture size

For computational efficiency, an optimized model size is desirable. To fully understand the connection between the architecture scale and task performance, the impact of the parameter count on the overall AI results was studied and illustrated in Fig. 9. Accordingly, the number of filters in the pre-processing module, corresponding to d_{model}, was adjusted to values ranging from 8 to 128. This trial was run twice: once with a solitary Transformer, and again with two Transformers in series.

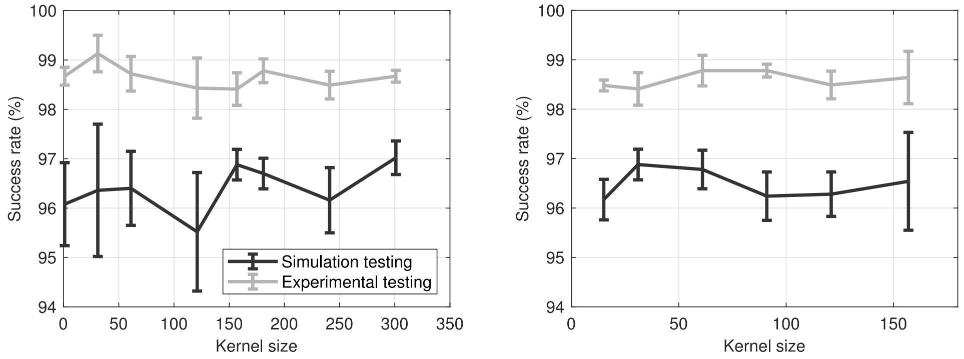
3.1.3. Ablation studies

To study the internal dynamics of the proposed architecture and isolate the effects of each module – namely the Transformer, Pre-processing, and Post-processing modules – selected components were strategically disabled, and the results are presented in Fig. 10. This approach allows for a focused evaluation of each module's contribution, helping to gauge the significance and impact of each constituent element on the overall performance of the architecture.

The entire Transformer module was first disabled to highlight its pivotal role within the overall structure. Next, the Transformer module was modified by omitting the local interaction achieved through the convolution block, granting insights on its effects. The impact of the echo detection and projection modules were then independently evaluated by analyzing the individual performance of the Pre-processing and Post-processing modules in conjunction with the Transformer module. Finally, the Transformer was assessed in isolation, without the Pre/Post-processing modules or the convolution block, to evaluate its standalone performance.

3.1.4. Database composition

Since running simulations is more convenient and budget-friendly than acquiring experimental data, training an AI model exclusively using simulation data with satisfactory real-world performance is desirable. Thus, the impact of the composition of the training dataset was investigated, as shown in Fig. 11. Additionally, a TNN model trained solely on experimental data was tested to: (1) evaluate the realism of the simulations and (2) establish a benchmark comparison for the simulation-supplemented database.



(a) (b)

Fig. 8. Success rate according to the kernel size of: (a) the conv 1 and (b) the conv 3 layer on the simulation and the experimental test datasets.

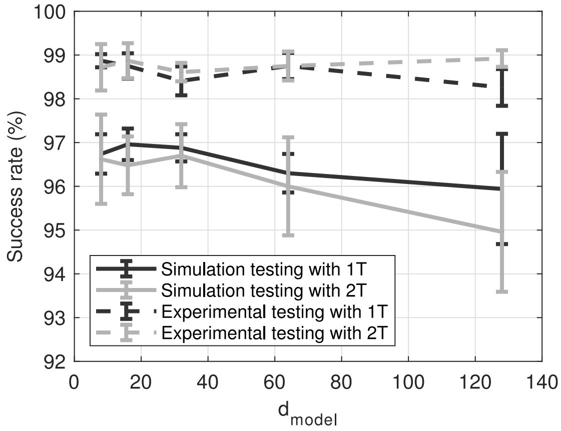


Fig. 9. Success rate with respect to the dimension of the model d_{model} with one or two transformers in series on the simulation and the experimental test datasets.

3.2. Comparative results

The fine-tuned Transformer architecture was compared with those presented in [22–24,50,51] and with the thresholding method, with the results summarized in Fig. 12 and Table 2. The refined TNN employs 16 filters with a conv 1 layer kernel size of 31 and a conv 3 layer kernel size of 31. All dropout rates were standardized to 0.3, while the single Transformer remained untouched. Lastly, to ensure a balanced training, 10,000 simulations were augmented with the experimental dataset comprising 1200 measurements to compose the training dataset.

To further analyze and compare models with similar success rates, another perspective using the MAE is presented in Fig. 12b. Furthermore, using keras-flops to measure the number of parameters and calculations (FLOPS) performed by each architecture, additional visualizations of the models' performances are shown in Fig. 13.

4. Discussion

4.1. Convolution kernel size

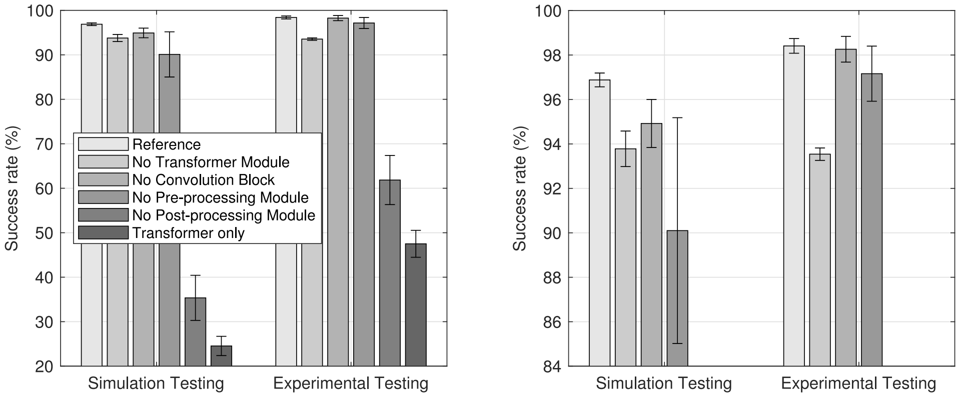
The initial assumption regarding the role of the Conv 1 layer in capturing potential echoes is challenged by the results presented in Fig. 8a. Surprisingly, using a kernel size of 1 does not negatively impact the architecture, yielding satisfactory results. Moreover, the best performance is achieved when the kernel size of the Conv 1 layer matches that of the Conv 3 layer. While the Conv 1 layer is involved in echo detection, as demonstrated in Shpigler's work [23] and in Fig. 8a (with the average line), its primary function is to encode the A-scans for better interpretation by the Transformer [33]. The Conv 3 layer, responsible for decoding the encoded signal to perform deconvolution, is more effective when its kernel size matches that of the encoding layer, as illustrated in Fig. 8a with $k = 31$ and Fig. 8b with $k = 157$. Integrating a TNN between a CNN encoder and decoder with matching kernel sizes has been explored previously for speech enhancement [32]. In this study, the best feature extraction was achieved with a kernel size of 31. Further investigation would be required to fine-tune the ideal kernel sizes for both convolution layers, but this lies beyond the scope of the present work.

4.2. Size

Increasing d_{model} causes the model to grow exponentially. However, this does not lead to a proportional improvement in performance. As seen in Fig. 9, the proposed architecture's success rate reaches an upper bound in both simulation and experimental testing, and even declines when $d_{model} \geq 64$ during simulation testing. This trend may be attributed to the growing complexity of the Transformer Module as d_{model} increases, combined with the low-size training dataset, as demonstrated in [54]. Indeed, as the model size grows, so does the demand for a larger training dataset to fully exploit the model's capacity. Without sufficient data, the model struggles to generalize, leading to diminished performance despite its increased complexity. Therefore, increasing the size of the architecture beyond a certain point offers diminishing returns, resulting in a decelerated processing speed and increased resource consumption without noticeable improvements. Adapting the architecture's dimensions to match the task's intricacy and training dataset size thus ensures a balance between the success rate and computational efficiency.

4.3. Ablation

Considering the best obtainable success rate with a 95% confidence level, removing the Transformer Module results in a 4.92% reduction in success rate on experimental data and a 2.61% reduction on simulation data, as shown in Fig. 10. This highlights the importance of the global context provided by the Transformer in improving the overall understanding of the A-scan for its final deconvolution. Interestingly, removing the Convolution Block within the Transformer Module slightly increases the success rate by 0.10% on experimental data but decreases it by 1.19% on simulation data. However, the local interactions provided by the convolution block contribute to increased model stability, reducing the margin of error between each trained model. This can be attributed to the better generalization capabilities of CNNs when trained with smaller dataset sizes [34,54].



(a) (b)

Fig. 10. Success rate depending on the removed blocks or modules on the simulation and the experimental test datasets with: (a) unzoomed and (b) zoomed view.

Table 2
Number of parameters, number of FLOPS, Success Rate (SR) and MAE between actual and predicted distances shown in Fig. 12 of the different methods on the simulation and the experimental test datasets.

Architecture	#Parameters	#FLOPS	SR exp (%)	SR sim (%)	MAE exp (mm)	MAE sim (mm)
TNN (this article)	11,613,233	373.8M	98.92 ± 0.25	98.70 ± 0.29	0.147 ± 0.024	0.171 ± 0.010
ResNet-50 [50]	26,046,512	7756M	96.32 ± 1.04	96.84 ± 0.40	0.235 ± 0.028	0.284 ± 0.004
EfficientNet-B0 [51]	5,586,771	804M	94.26 ± 0.67	97.12 ± 0.41	0.289 ± 0.012	0.273 ± 0.014
EfficientNet-B4 [51]	19,825,423	3089M	94.09 ± 0.77	96.54 ± 0.30	0.321 ± 0.012	0.270 ± 0.009
CNN Chapon et al. [22]	30,913	74.1M	91.68 ± 0.17	90.36 ± 1.10	0.361 ± 0.045	0.536 ± 0.048
CNN Shpigler et al. [23]	15,451	36.9M	91.54 ± 0.40	93.20 ± 0.59	0.388 ± 0.019	0.451 ± 0.032
CNN Cantero-Chinchilla et al. [24]	650,497	1203M	93.22 ± 0.50	92.70 ± 0.94	0.265 ± 0.013	0.311 ± 0.026
Threshold	//	//	81.58	78.20	1.14	1.76

MAE = Mean Absolute Error, SR = Success Rate.

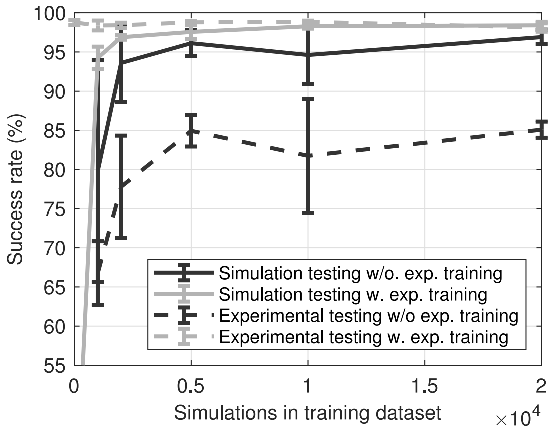


Fig. 11. Success rate following the quantity of simulations within the training dataset on the simulation and the experimental test datasets.

Similarly, while removing the pre-processing module has little effect on experimental data, it leads to a significant drop in success rate on simulation data and is associated with high variability across the five trained models (±5.08%). These results suggest that the pre-processing module plays a key role in position encoding [33] and facilitates learning, particularly with small datasets, enhancing the model's comprehension, learnability, and stability. However, its primary role in echo detection is questioned, given the relatively high success rates achieved without it.

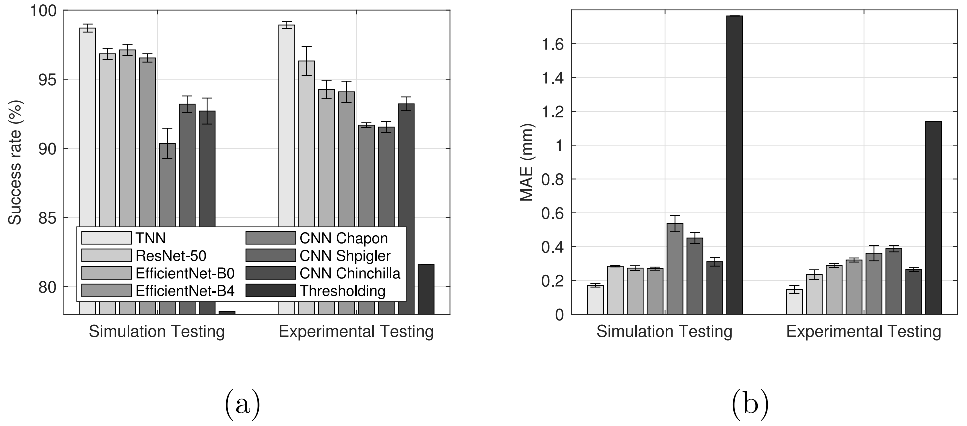
In contrast, removing the post-processing module drastically lowers the success rate, dropping it below 70% on experimental data and 30% on simulation data. The Transformer Module alone does not provide adequate signal processing capabilities, leading to poor signal regression and deconvolution performances. This is further emphasized by the low success rates observed when the Transformer Module is used without other components (50.54% on experimental data and 26.71% on simulation data). This decline can be attributed to the absence of key elements such as adequate number of training samples, position encoding, and the ability to capture local features, as described in [33].

Thus, while the Transformer is less effective for regression tasks, it excels at capturing the global context of the A-scans. This contextual information supplements the CNN modules, providing key insights that are essential for the model's overall performance, especially in tasks like signal deconvolution and echo detection. By combining the strengths of both the Transformer and CNN, the architecture achieves improved precision and stability.

4.4. Database composition

The Transformer's poor generalization capabilities in the context of small datasets have been well-documented in the AI community [33,54,55], and this limitation is evident in Fig. 11 for dataset sizes under 10k. While Vision Transformers (ViT) and other Transformer architectures require datasets in the millions to outperform traditional CNN architectures, such as ResNet [50] or EfficientNet [51], in tasks like image classification, the Transformer architecture proposed in this paper achieves high performance with much smaller datasets as shown in Fig. 11. Specifically, the performance of the CNN-TNN model converges after 10k training samples, with minimal improvement when the dataset size is doubled. This phenomenon can be attributed to two key factors.

Firstly, the task addressed in this study is simpler compared to those tackled by larger models. Here, the model deconvolve a 1D signal of size 1200 × 1, whereas image classification tasks typically



(a) (b)

Fig. 12. Comparative bar chart of (a) the success rates and (b) the MAE of different AI architectures and industrial algorithm on the simulation and the experimental test datasets.

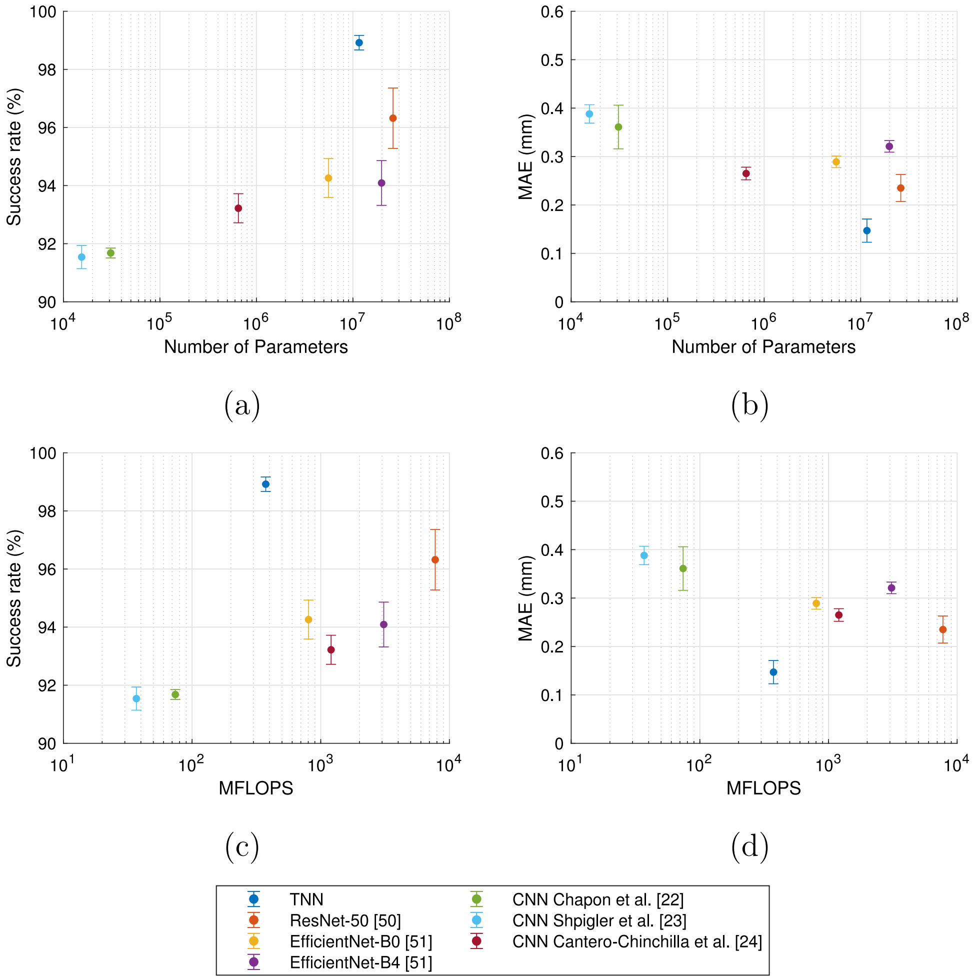


Fig. 13. Comparative analysis of various AI architectures on the experimental test dataset: (a) success rates versus the number of parameters, (b) MAE versus the number of parameters, (c) success rates versus the number of FLOPS, and (d) MAE versus the number of FLOPS.

involve processing input of size 224 × 224 × 3 across 1k, 18k or 21k classes [54]. Supporting this observation, Pu et al. successfully trained a Transformer model to denoise EEG signals of size 512 × 1 with a training dataset size of 13,512 (approximately 45k after augmentation), surpassing CNN performance in their application [31]. While their model predicted 512 values, the model in this study needs to identify 2 values within the 1200 possible classes, significantly reducing the task complexity and the required dataset size.

Secondly, the proposed architecture incorporates convolutional layers for encoding and decoding. These layers effectively introduce local information, facilitating learning during backpropagation [25,34]. This hybrid design leverages the strengths of both CNNs and Transformers, enabling efficient training even on smaller datasets. Additionally, the design could be further enhanced by incorporating the convolutional block in parallel with the MHSA layer, as demonstrated in [56].

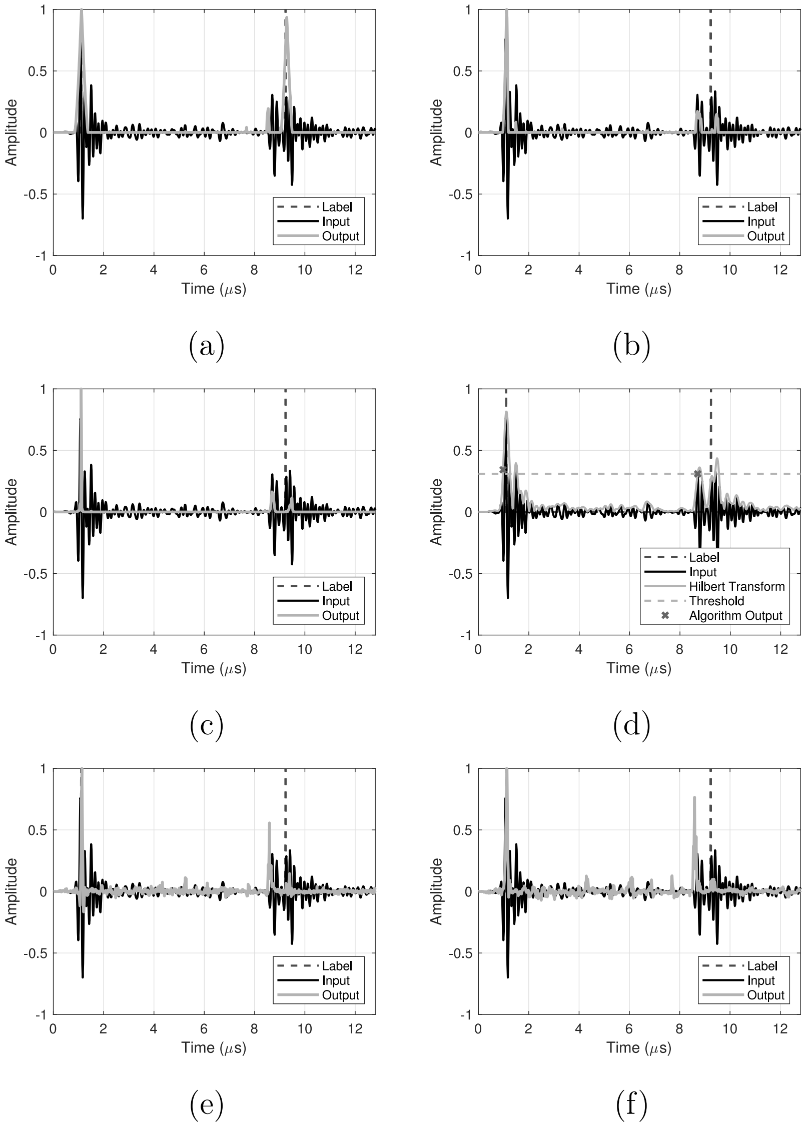


Fig. 14. Distance and deconvolution prediction of the experimental sample 106 with: (a) TNN, (b) Chapon's CNN, (c) Shpigler's CNN, (d) Thresholding technique, (e) ResNet-50, and (f) EfficientNet-B0.

With limited access to experimental data but abundant simulation samples, finding the optimal balance between these resources is critical for effective learning. At a 95% confidence level, when no experimental data are included, increasing the size of the simulation-based training dataset from 1000 to 10,000 data improves the success rate on both experimental and simulation testing datasets by 18.19% and 4.36% respectively. However, the Transformer struggles to generalize to experimental data when trained exclusively on simulation samples. Adding 1200 experimental samples to the 10,000 simulation samples boosts the success rate on experimental data from 89.02% to 99%. This result underscores the model's limited generalization ability and highlights the necessity of including experimental data in the training process.

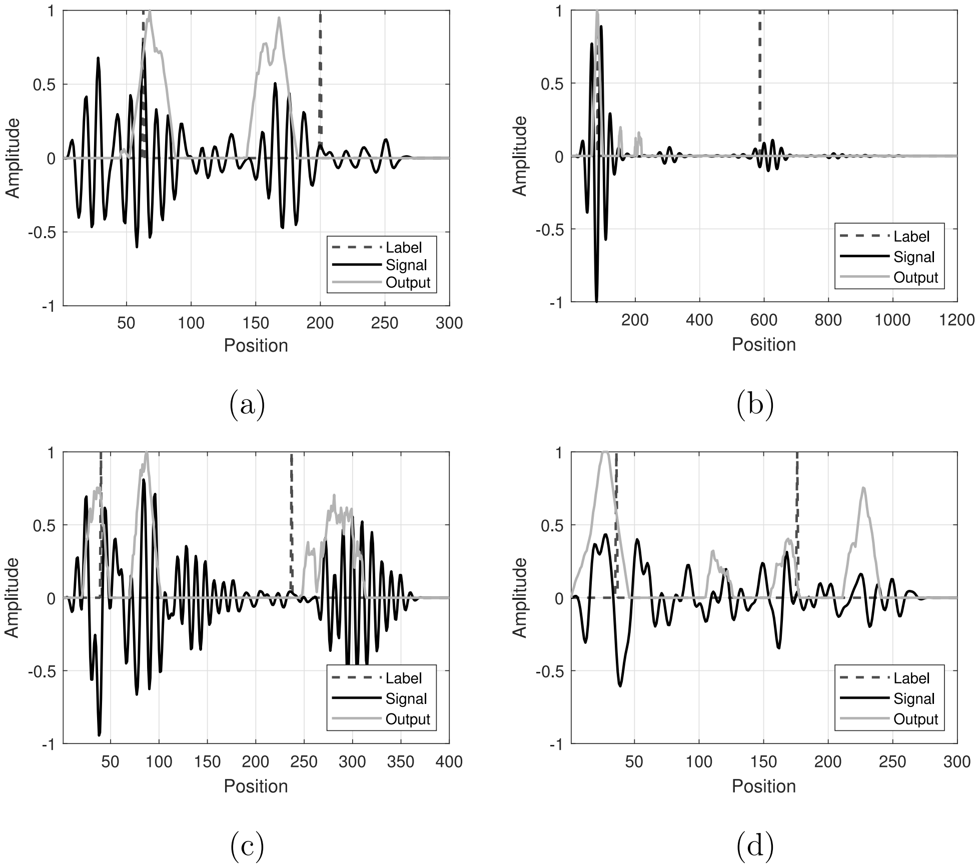
Incorporating experimental data also enhances the success rate on the simulation testing set and improves the model's learning stability. This improvement is likely due to the increased overall dataset size, as adding experimental data augments the total number of training samples.

However, training the model exclusively on experimental data leads to overfitting, as demonstrated by the success rate of under 55% on the simulation testing when no simulation data is included (Fig. 11). Therefore, adding simulation data with high variability helps the model generalize and prevents overfitting.

For this application, the optimal ratio of training data was found to be approximately 90% simulation and 10% experimental data (10,000 simulations for 1200 experimental samples), offering the most effective balance between training efficiency and generalizability across different datasets. This ratio ensures that the model can leverage the abundance of simulation data while still incorporating enough experimental data to generalize well in real-world scenarios.

4.5. Comparative results

At a 95% confidence level, following the training of each architecture as described in Section 2.5, the top-performing TNN demonstrated a 5.20% higher success rate compared to the best CNN result in simulation testing (CNN Shpigler) and a 5.45% improvement in experimental testing (CNN Cantero-Chinchilla). These CNN architectures, described in NDE-focused studies [22–24], were specifically designed for tasks in this domain. For deeper and more general models like ResNet and EfficientNet, the TNN exhibited a 1.46% improvement in simulation testing and a 1.81% improvement in experimental testing, as shown in Fig. 12a and summarized in Table 2.



(a)

(b)

(c)

(d)

Fig. 15. Failure cases of the TNN's deconvolution predictions on experimental sample: (a) 622, (b) 6, (c) 489, and (d) 600.

Additionally, according to Fig. 12b and Table 2, the Transformer architecture emerged as the most accurate option for predicting distances based on 2.25 MHz and 5 MHz experimental data. Specifically, the TNN was 1.7 times more precise in terms of total MAE compared to the best CNN alternative (ResNet-50). Similarly, the best CNN model was 5.5 times more precise than traditional thresholding methods, while the TNN variant achieved an even greater improvement, reducing MAE by a factor of 9.3. This highlights the superiority of AI-based methods over conventional techniques.

Finally, despite its relatively high parameter count, the optimized TNN model remains competitive in terms of FLOPS. It is 3.2 times less computationally demanding than the best CNN architecture described in NDE papers (CNN Cantero-Chinchilla), and approximately 21 times less demanding than the top overall CNN model (ResNet-50). Coupled with this efficiency, the TNN consistently outperforms all CNN alternatives for deconvolution tasks in experimental testing, achieving notable improvements in both MAE and success rate, as shown in Fig. 13.

To better illustrate individual model behavior on complex experimental samples, Fig. 14 presents a typical indistinct A-scan for CNN models. Limited by their regional focus, CNNs decomposed each local echo, thereby misinterpreting the dominant one. A secondary echo originating from lateral reflections in the corroded sample was identified by all CNN architectures, introducing interpretive complications. Thanks to the addition of long-range dependencies with the MHSA module, the TNN model successfully identified the correct echo to deconvolve within the main echo packet (Fig. 14a). Finally, although the thresholding method struggles to distinguish the minimum separation within echo packets, it grasps the average distance reasonably well if the threshold is appropriately selected (Fig. 14d), as outlined by Cantero-Chinchilla et al. [24]. In this specific example, however, it first selects the wrong echo, resulting in failure.

4.6. Error analysis

Using the 5 optimal TNNs referenced in Table 2, 12 unique experimental A-scans were identified as failure cases, with four examples shown in Fig. 15 for analysis.

Firstly, while the maximum of the Hilbert transform generally provides a reliable reference, it can mislabel data and mislead the ML during training, as observed in Fig. 15a and c. In these cases, the deconvolutions failed because the first peak corresponded to an edge reflection rather than the intended reference echo.

Secondly, destructive interference and high surface roughness caused low-amplitude echoes to be reflected back to the probe, as seen in Fig. 15b. The lack of sufficient 2.25 MHz data with low echo amplitudes in the training set caused the model to misinterpret these signals as artifacts rather than valid echo packets.

Finally, mode conversion effects were observed in several A-scans due to angled reflections, producing both longitudinal and transverse waves. These waves, originating from the backwall or edges of the block, can lead to misleading peak amplitudes. As seen in Fig. 15a and c., the largest peak did not always correspond to the backwall echo. Additionally, Fig. 15d highlights how an unwanted reflection can mimic the first echo waveform, causing the model to incorrectly identify it as the second peak.

5. Conclusion

In this paper, a novel CNN-TNN architecture was developed to partially deconvolve A-scans of severely corroded profiles on the minimal thickness and was compared against multiple CNN architectures sourced from the literature. Utilizing a simulation-based dataset – developed with [24] as a foundation – alongside experimental data acquired from a corroded carbon steel step specimen, the study demonstrated TNN's superior deconvolution capabilities. The CNN-TNN architecture achieved a notable 1.81% success rate improvement over the best-performing CNN and a 17.5% advantage over established thresholding methods in detecting minimal depths on experimental data, with a resolution under 0.5 λ (1.32 mm for 2.25 MHz and 0.59 mm for 5 MHz).

Furthermore, the TNN demonstrated exceptional precision and accuracy, achieving an overall MAE that was 1.7 times lower that the widely recognized ResNet-50. Its ability to enhance signal decomposition alongside CNN models further underscores its superior performance in complex signal processing tasks.

Although TNN emerged successful in this study, it is important to bear in mind that demonstrated learning capabilities shown in this paper are confined to specified emission frequencies (around 2.25 MHz and 5 MHz) and sampling rates (62.5 MHz). Moreover, the proposed method focuses on two echoes to identify minimal distances within fixed-length A-scans. This approach requires a pre-processing step to isolate echo pairs and padding to standardize the input length.

Additionally, the limitations of the simulation were highlighted and could be improved through enhanced signal generation or the use of GAN-based synthetic data generation.

Finally, the deconvolved output allows the model to be adapted for various applications. Beyond distance measurement, it can precisely estimate echo arrival times for multimodal TFM, unlocking new possibilities in advanced imaging and diagnostics. Given its focus on signal processing, the model could also be extended for signal denoising or reconstructing FMC amplitudes from binary FMC data.

CRediT authorship contribution statement

T. Sendra: Writing – original draft, Validation, Software, Methodology, Investigation, Conceptualization. **P. Belanger:** Writing – review & editing, Validation, Supervision, Resources, Project administration, Funding acquisition.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work the authors used ChatGPT in order to improve the readability and language of the manuscript. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the content of the published article.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment and fundings

This research was funded by Natural Sciences and Engineering Research Council of Canada (NSERC) grant number ALLRP 569922-21, Nucleom and Eddyfi Technologies.

Data availability

Data will be made available on request.

References

[1] F. Cegla, J. Allin, Ultrasonic monitoring of pipeline wall thickness with autonomous, wireless sensor networks, in: Oil and Gas Pipelines, John Wiley & Sons, Ltd, 2015, pp. 571–578, http://dx.doi.org/10.1002/9781119019213.ch39, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119019213.ch39. Section: 39 _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781119019213.ch39.

[2] L.J.B. T. Eason, Ultrasonic Thickness Structural Health Monitoring of Steel Pipe for Internal Corrosion (Ph.D. thesis), Iowa State University, Digital Repository, Ames, 2017, 11055505, http://dx.doi.org/10.31274/etd-180810-5130, URL https://lib.dr.iastate.edu/etd/15513/.

[3] Ø. Baltzersen, T.I. Waag, R. Johnsen, C.H. Ahlen, E. Tveit, Wall thickness monitoring of new and existing subsea pipelines using ultrasound, vol. all days of nace corrosion, 2007, arXiv:https://onepetro.org/NACECORR/proceedings-pdf/CORR07/All-CORR07/NACE-07333/1839313/nace-07333.pdf.

[4] W.E. Norris, D.J. Naus, H.L. Graves, Inspection of nuclear power plant containment structures, Nucl. Eng. Des. 192 (2) (1999) 303–329, http://dx.doi.org/10.1016/S0029-5493(99)00125-9, URL https://www.sciencedirect.com/science/article/pii/S0029549399001259.

[5] J. Krautkrämer, H. Krautkrämer, Pulse-echo method, in: J. Krautkrämer, H. Krautkrämer, W. Grabendörfer, L. Niklas, R. Frielinghaus, W. Rath, H. Schlemm, U. Schlengermann (Eds.), Ultrasonic Testing of Materials, Springer, Berlin, Heidelberg, 1977, pp. 193–264, http://dx.doi.org/10.1007/978-3-662-02296-2_11.

[6] H. Liu, L. Zhang, H.F. Liu, S. Chen, S. Wang, Z.Z. Wong, K. Yao, High-frequency ultrasonic methods for determining corrosion layer thickness of hollow metallic components, Ultrasonics 89 (2018) 166–172, http://dx.doi.org/10.1016/j.ultras.2018.05.006, URL https://www.sciencedirect.com/science/article/pii/S0041624X18300301.

[7] J.D.N. Cheeke, Fundamentals and applications of ultrasonic waves, CRC series in pure and applied physics, CRC Press, Boca Raton, 2002, URL http://catdir.loc.gov/catdir/toc/fy032/2002018807.html, Section : 462 pages : illustrations ; 24 cm.

[8] P.J. Shull (Ed.), Nondestructive evaluation: theory, techniques, and applications, in: 142 in Mechanical engineering, Dekker, New York Basel, 2002.

[9] D. Marioli, C. Narduzzi, C. Offelli, D. Petri, E. Sardini, A. Taroni, Digital time-of-flight measurement for ultrasonic sensors, IEEE Trans. Instrum. Meas. 41 (1) (1992) 93–97, http://dx.doi.org/10.1109/19.126639, URL https://ieeexplore.ieee.org/abstract/document/126639, Conference Name: IEEE Transactions on Instrumentation and Measurement.

[10] M. Parrilla, J. Anaya, C. Fritsch, Digital signal processing techniques for high accuracy ultrasonic range measurements, IEEE Trans. Instrum. Meas. 40 (4) (1991) 759–763, http://dx.doi.org/10.1109/19.85348, URL https://ieeexplore.ieee.org/abstract/document/85348, Conference Name: IEEE Transactions on Instrumentation and Measurement.

[11] Y. Chang, Y. Zi, J. Zhao, Z. Yang, W. He, H. Sun, An adaptive sparse deconvolution method for distinguishing the overlapping echoes of ultrasonic guided waves for pipeline crack inspection, Meas. Sci. Technol. 28 (3) (2017) 035002, http://dx.doi.org/10.1088/1361-6501/aa52ae, Publisher: IOP Publishing.

[12] H. chen, L. bing, G. fei, An overlapping echo separation method using blind deconvolution in ultrasonic testing, J. Xi'an Jiaotong Univ. (China), J. Xi' an Jiaotong Univ. (2021) 129–137, http://dx.doi.org/10.7652/xjtuxb202112015, Place: China Publisher: Editorial Board of Journal of Xi'an Jiaotong University.

[13] S. Mallat, Z. Zhang, Matching pursuits with time-frequency dictionaries, IEEE Trans. Signal Process. 41 (12) (1993) 3397–3415, http://dx.doi.org/10.1109/78.258082.

[14] J. Tropp, A. Gilbert, Signal recovery from random measurements via orthogonal matching pursuit, IEEE Trans. Inform. Theory 53 (12) (2007) 4655–4666, http://dx.doi.org/10.1109/TIT.2007.909108.

[15] E. Mor, A. Azoulay, M. Aladjem, A matching pursuit method for approximating overlapping ultrasonic echoes, IEEE Trans. Ultrason. Ferroelectr. Freq. Control 57 (9) (2010) 1996–2004, http://dx.doi.org/10.1109/TUFFC.2010.1647.

[16] W. Rawat, Z. Wang, Deep convolutional neural networks for image classification: A comprehensive review, Neural Comput. 29 (9) (2017) 2352–2449, http://dx.doi.org/10.1162/neco_a_00990.

[17] E. Jiang, A review of the comparative studies on traditional and intelligent face recognition methods, in: 2020 International Conference on Computer Vision, Image and Deep Learning (CVIDL), 2020, pp. 11–15, http://dx.doi.org/10.1109/CVIDL51233.2020.00010, URL https://ieeexplore.ieee.org/abstract/document/9270454?casa_token=J8Ah-aHz54YAAAAA:7dwvLIYxioOj8kdOrQmqRBoTKI35bL8YKIXTf3Tzs_9HFmnyL0WpakbFZupEUY8hGN_vG9ScWhs.

[18] X. He, L. Deng, Deep learning for image-to-text generation: A technical overview, IEEE Signal Process. Mag. 34 (6) (2017) 109–116, http://dx.doi.org/10.1109/MSP.2017.2741510, URL https://ieeexplore.ieee.org/abstract/document/8103169?casa_token=GRujYghEDvwAAAAA:5tKFYQOIrXybMdJHRGaf5nYTSx1W_IyKrO7PvtQvi3tpaIWA-UDm88VP_iHDAOBrATHyF8Pej68, Conference Name: IEEE Signal Processing Magazine.

[19] L. Posilović, D. Medak, M. Subašić, T. Petković, M. Budimir, S. Lončarić, Flaw detection from ultrasonic images using YOLO and SSD, in: 2019 11th International Symposium on Image and Signal Processing and Analysis (ISPA), (ISSN: 1849-2266) 2019, pp. 163–168, http://dx.doi.org/10.1109/ISPA.2019.8868929, URL https://ieeexplore.ieee.org/abstract/document/8868929?casa_token=ekXwITVqTt0AAAAA:0gw3VOeCv6RDCPuhCU12vhgCWSOEnxKTsy0EnnInrwk-ia7bDVVwbC1HzqA3M8lskt_WclRQkoc.

[20] R.J. Pyle, R.L.T. Bevan, R.R. Hughes, R.K. Rachev, A.A.S. Ali, P.D. Wilcox, Deep learning for ultrasonic crack characterization in NDE, IEEE Trans. Ultrason. Ferroelectr. Freq. Control 68 (5) (2021) 1854–1865, http://dx.doi.org/10.1109/TUFFC.2020.3045847, URL https://ieeexplore.ieee.org/abstract/document/9298790, Conference Name: IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control.

[21] S. Uhlig, I. Alkhasli, F. Schubert, C. Tschöpe, M. Wolff, A review of synthetic and augmented training data for machine learning in ultrasonic non-destructive evaluation, Ultrasonics 134 (2023) 107041, http://dx.doi.org/10.1016/j.ultras.2023.107041, URL https://www.sciencedirect.com/science/article/pii/S0041624X23001178.

[22] A. Chapon, D. Pereira, M. Toews, P. Belanger, Deconvolution of ultrasonic signals using a convolutional neural network, Ultrasonics 111 (2021) 106312, http://dx.doi.org/10.1016/j.ultras.2020.106312, URL https://linkinghub.elsevier.com/retrieve/pii/S0041624X2030247X.

[23] A. Shpigler, E. Mor, A. Bar-Hillel, Detection of overlapping ultrasonic echoes with deep neural networks, Ultrasonics 119 (2022) 106598, http://dx.doi.org/10.1016/j.ultras.2021.106598, URL https://www.sciencedirect.com/science/article/pii/S0041624X21002183.

[24] S. Cantero-Chinchilla, C.A. Simpson, A. Ballisat, A.J. Croxford, P.D. Wilcox, Convolutional neural networks for ultrasound corrosion profile time series regression, NDT E Int. 133 (2023) 102756, http://dx.doi.org/10.1016/j.ndteint.2022.102756, URL https://www.sciencedirect.com/science/article/pii/S0963869522001554.

[25] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, R. Pang, Conformer: Convolution-augmented transformer for speech recognition, 2020, arXiv:2005.08100 URL https://arxiv.org/abs/2005.08100.

[26] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems, 30, Curran Associates, Inc., 2017, URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[27] S. Karita, N. Chen, T. Hayashi, T. Hori, H. Inaguma, Z. Jiang, M. Someki, N.E.Y. Soplin, R. Yamamoto, X. Wang, S. Watanabe, T. Yoshimura, W. Zhang, A comparative study on transformer vs RNN in speech applications, in: 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), 2019, pp. 449–456, http://dx.doi.org/10.1109/ASRU46091.2019.9003750, URL https://ieeexplore.ieee.org/abstract/document/9003750.

[28] Q. Zhang, H. Lu, H. Sak, A. Tripathi, E. McDermott, S. Koo, S. Kumar, Transformer transducer: A streamable speech recognition model with transformer encoders and RNN-t loss, in: ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), (ISSN: 2379-190X) 2020, pp. 7829–7833, http://dx.doi.org/10.1109/ICASSP40776.2020.9053896, URL https://ieeexplore.ieee.org/abstract/document/9053896.

[29] C. Che, P. Zhang, M. Zhu, Y. Qu, B. Jin, Constrained transformer network for ECG signal processing and arrhythmia classification, BMC Med. Inform. Decis. Mak. 21 (1) (2021) 184, http://dx.doi.org/10.1186/s12911-021-01546-2.

[30] J. Guan, W. Wang, P. Feng, X. Wang, W. Wang, Low-dimensional denoising embedding transformer for ECG classification, in: ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), (ISSN: 2379-190X) 2021, pp. 1285–1289, http://dx.doi.org/10.1109/ICASSP39728.2021.9413766, URL https://ieeexplore.ieee.org/abstract/document/9413766.

[31] X. Pu, P. Yi, K. Chen, Z. Ma, D. Zhao, Y. Ren, Eegdnet: Fusing non-local and local self-similarity for EEG signal denoising with transformer, Comput. Biol. Med. 151 (2022) 106248, http://dx.doi.org/10.1016/j.compbiomed.2022.106248, URL https://www.sciencedirect.com/science/article/pii/S0010482522009568.

[32] K. Wang, B. He, W.-P. Zhu, TSTNN: Two-stage transformer based neural network for speech enhancement in the time domain, in: ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), (ISSN: 2379-190X) 2021, pp. 7098–7102, http://dx.doi.org/10.1109/ICASSP39728.2021.9413740, URL https://ieeexplore.ieee.org/abstract/document/9413740.

[33] H. Wu, B. Xiao, N. Codella, M. Liu, X. Dai, L. Yuan, L. Zhang, Cvt: Introducing convolutions to vision transformers, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, ICCV, 2021, pp. 22–31.

[34] K. Li, R. Yu, Z. Wang, L. Yuan, G. Song, J. Chen, Locality guidance for improving vision transformers on tiny datasets, in: S. Avidan, G. Brostow, M. Cissé, G.M. Farinella, T. Hassner (Eds.), Computer Vision – ECCV 2022, Springer Nature Switzerland, Cham, 2022, pp. 110–127, http://dx.doi.org/10.1007/978-3-031-20053-3_7.

[35] A. Nandi, D. Mampel, B. Roscher, Blind deconvolution of ultrasonic signals in nondestructive testing applications, IEEE Trans. Signal Process. 45 (5) (1997) 1382–1390, http://dx.doi.org/10.1109/78.575716, URL https://ieeexplore.ieee.org/abstract/document/575716, Conference Name: IEEE Transactions on Signal Processing.

[36] K. Kaaresen, Deconvolution of sparse spike trains by iterated window maximization, IEEE Trans. Signal Process. 45 (5) (1997) 1173–1183, http://dx.doi.org/10.1109/78.575692, URL https://ieeexplore.ieee.org/abstract/document/575692, Conference Name: IEEE Transactions on Signal Processing.

[37] P. Huthwaite, Accelerated finite element elastodynamic simulations using the GPU, J. Comput. Phys. 257 (2014) 687–707, http://dx.doi.org/10.1016/j.jcp.2013.10.017, URL https://www.sciencedirect.com/science/article/pii/S0021999113006931.

[38] D. Li, Z. Chen, Y. Zhang, S. Zhao, W. Liu, Separation of multi-echo overlapping ultrasonic signals for increasing the axial resolution using a neural network, Meas. Sci. Technol. 34 (12) (2023) http://dx.doi.org/10.1088/1361-6501/acefee.

[39] S. Ioffe, C. Szegedy, Batch normalization: Acceleratingdeep network training by reducing internal covariate shift, in: Proceedings of the 32nd International Conference on Machine Learning, PMLR, 2015, pp. 448–456, URL https://proceedings.mlr.press/v37/ioffe15.html, issn:1938-7228.

[40] N. Bjorck, C.P. Gomes, B. Selman, K.Q. Weinberger, Understanding batch normalization, in: S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, R. Garnett (Eds.), Advances in Neural Information Processing Systems, 31, Curran Associates, Inc., 2018, URL https://proceedings.neurips.cc/paper_files/paper/2018/file/36072923bfc3cf47745d704feb489480-Paper.pdf.

[41] A.F. Agarap, Deep learning using rectified linear units (relu), 2018, http://dx.doi.org/10.48550/arXiv.1803.08375, URL http://arxiv.org/abs/1803.08375, arXiv:1803.08375 [cs, stat] version: 1.

[42] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (2014) 1929–1958.

[43] C. Garbin, X. Zhu, O. Marques, Dropout vs. batch normalization: an empirical study of their impact to deep learning, Multimedia Tools Appl. 79 (19) (2020) 12777–12815, http://dx.doi.org/10.1007/s11042-019-08453-9.

[44] W. Han, Z. Zhang, Y. Zhang, J. Yu, C.-C. Chiu, J. Qin, A. Gulati, R. Pang, Y. Wu, ContextNet: Improving convolutional neural networks for automatic speech recognition with global context, 2020, http://dx.doi.org/10.48550/arXiv.2005.03191, URL http://arxiv.org/abs/2005.03191, arXiv:2005.03191 [cs, eess].

[45] S. Yuan, P. Li, B. Wu, Towards single-component and dual-component radar emitter signal intra-pulse modulation classification based on convolutional neural network and transformer, Remote. Sens. 14 (15) (2022) 3690, http://dx.doi.org/10.3390/rs14153690, URL https://www.mdpi.com/2072-4292/14/15/3690, Number: 15 Publisher: Multidisciplinary Digital Publishing Institute.

[46] M.M. Lau, K. Hann Lim, Review of adaptive activation function in deep neural network, in: 2018 IEEE-EMBS Conference on Biomedical Engineering and Sciences, IECBES, 2018, pp. 686–690, http://dx.doi.org/10.1109/IECBES.2018.8626714.

[47] Q. Wang, Y. Ma, K. Zhao, Y. Tian, A comprehensive survey of loss functions in machine learning, Ann. Data Sci. 9 (2) (2022) 187–212, http://dx.doi.org/10.1007/s40745-020-00253-5.

[48] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D.G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, X. Zheng, TensorFlow: A system for large-scale machine learning, in: 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), USENIX Association, Savannah, GA, 2016, pp. 265–283, URL https://www.usenix.org/conference/osdi16/technical-sessions/presentation/abadi.

[49] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2017, http://dx.doi.org/10.48550/arXiv.1412.6980, URL http://arxiv.org/abs/1412.6980, arXiv:1412.6980 [cs].

[50] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.

[51] M. Tan, Q. Le, EfficientNet: Rethinkingmodel scaling for convolutional neural networks, in: Proceedings of the 36th International Conference on Machine Learning, PMLR, (ISSN: 2640-3498) 2019, pp. 6105–6114, URL https://proceedings.mlr.press/v97/tan19a.html.

[52] B. Barshan, Fast processing techniques for accurate ultrasonic range measurements, Meas. Sci. Technol. 11 (1) (2000) 45, http://dx.doi.org/10.1088/0957-0233/11/1/307.

[53] S.K. Shastri, S. Rudresh, R. Anand, S. Nagesh, C.S. Seelamantula, A.K. Thittai, Axial super-resolution in ultrasound imaging with application to non-destructive evaluation, Ultrasonics 108 (2020) 106183, http://dx.doi.org/10.1016/j.ultras.2020.106183, URL https://www.sciencedirect.com/science/article/pii/S0041624X20301220.

[54] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, N. Houlsby, An image is worth 16x16 words: Transformers for image recognition at scale, 2021, http://dx.doi.org/10.48550/arXiv.2010.11929, URL http://arxiv.org/abs/2010.11929, arXiv:2010.11929 [cs].

[55] C. Li, C. Zhang, Toward a deeper understanding: RetNet viewed through convolution, Pattern Recognit. 155 (2024) 110625, http://dx.doi.org/10.1016/j.patcog.2024.110625, URL https://www.sciencedirect.com/science/article/pii/S0031320324003765.

[56] Z. Wu, Z. Liu, J. Lin, Y. Lin, S. Han, Lite transformer with long-short range attention, 2020, http://dx.doi.org/10.48550/arXiv.2004.11886, URL http://arxiv.org/abs/2004.11886, arXiv:2004.11886 [cs].