Perception modelling by invariant representation of deep learning for automated structural diagnostic in aircraft maintenance
A study case using DeepSHM

Ewald, Vincentius; Sridaran Venkat, Ramanan; Asokkumar, Aadhik; Benedictus, Rinze; Boller, Christian; Groves, Roger M.

DOI
10.1016/j.ymssp.2021.108153

Publication date
2021

Document Version
Final published version

Published in
Mechanical Systems and Signal Processing

Mechanical Systems and Signal Processing xxx (xxxx) xxx

Contents lists available at ScienceDirect

Mechanical Systems and Signal Processing

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

Perception modelling by invariant representation of deep learning for automated structural diagnostic in aircraft maintenance: A study case using DeepSHM

Vincentius Ewald [a,*], Ramanan Sridaran Venkat [b], Aadhik Asokkumar [b,d], Rinze Benedictus [c], Christian Boller [b], Roger M Groves [a]

[a] *Aerospace Non-Destructive Testing Laboratory, Faculty of Aerospace Engineering, Delft University of Technology, Delft, the Netherlands*
[b] *Chair of Non-Destructive Testing and Quality Assurance, University of Saarland, Saarbrücken, Germany*
[c] *Chair of Structural Integrity & Composites, Faculty of Aerospace Engineering, Delft University of Technology, Delft, the Netherlands*
[d] *Kaunas University of Technology, Kaunas, Lithuania*

A R T I C L E I N F O

Keywords:
Ultrasonic lamb wave
Structural health monitoring
Deep learning
Convolutional neural network
Time-frequency-analysis
Perception
Feature learning
Invariant representation

A B S T R A C T

Predictive maintenance, as one of the core components of Industry 4.0, takes a proactive approach to maintain machines and systems in good order to keep downtime to a minimum and the airline maintenance industry is not an exception to this. To achieve this goal, practices in Structural Health Monitoring (SHM) complement the existing Non-Destructive-Testing (NDT) have been established in the last decades. Recently, the increasing computational capability such as utilization of a graphical processing unit (GPU) in combination with advanced machine learning techniques such as deep learning has been one of the main drivers in the advancement of predictive analytics in condition monitoring.

In our previous work, we proposed a novel approach using deep learning for guided wave based structural health called DeepSHM. As a study case, we treated an ultrasonic signal from guided Lamb wave SHM with a convolutional neural network (CNN). In that work, we only considered a single central frequency excitation. This led to a single governing wavelength which is normally good for the detection of a single damage size.

In classical signal processing, applying a broader excitation frequency poses an analysis and interpretation nightmare because it contains more complex information and thus is difficult to understand. This problem can be overcome with deep learning; however, it creates another problem: while deep learning typically results in a more accurate result prediction, it is specifically made for solving only certain types of tasks. While many papers have already introduced deep learning for diagnostics, many of these works are only proposing novel predictive techniques, however the mathematical formalization is lacking, and we are not informed about why we should treat acoustic signal with deep learning. So, the basis of 'explainable AI' for SHM and NDT is currently lacking.

For this reason, in this paper, we would like to extend our previous work into a more generalized. Rather than focusing on a novel technique, we propose a plausible theoretical perspective inspired from neuroscience for signal representation of deep learning framework to model machine perception in structural health monitoring (SHM), especially because SHM typically involves multiple sensory input from different sensing locations. To do this, we created a set of

* Corresponding author.
E-mail address: V.Ewald@tudelft.nl (V. Ewald).

https://doi.org/10.1016/j.ymssp.2021.108153
Received 10 September 2020; Received in revised form 19 May 2021; Accepted 15 June 2021

artificial data from a finite element model (FEM) and represented DeepSHM in two different ways: 1). Perpetual representation of observation and 2). Hierarchical structure of entities that is decomposable in a smaller sub-entity. Consequently, we assume two plausible models for DeepSHM: 1). Either it behaves as a single deciding actor since the observation is regarded as perpetual, and 2). Or it acts as a multiple actor with independent outputs since multiple sensors can form different output probabilities. These artificial data were split into several different input representations, classified into several damage scenarios and then trained with commonly used deep learning training parameters. We compare the performance metrics of each perception model to describe the training behavior of both representations.

1. Introduction

Airlines typically generate their revenue by using their aircrafts to transport passengers and cargo. Besides fuel and ground services, one of the most crucial aspects in airline operating costs in the maintenance. In 2017, it was reported that 70 billion USD was spent by airlines for the maintenance, repair, and overhaul (MRO) [74] and this figure was expected to grow to 88 billion USD in 2018 and would further increase to 115 billion USD in 2028 due to increasing number of aircraft deliveries [1,21]. However, due to the current global situation such as trade wars and global pandemic, we believe the prediction for 2028 might be overly optimistic, although we still think that the growth is still there and by 2028, the airline industry would have very much recovered.

One of the important objectives of the aircraft maintenance is to ensure that safety and reliability remain within their design limits to maintaining structural integrity within the damage tolerance design philosophy, so that the aircraft is able to sustain predictable damages until the next repair cycle [39,116]. While the damage tolerance design itself offers some sort of “passive protection” against catastrophic failure, there is no way to fully ensure reliability without active intervention due to the uncertain nature of aircraft operation and therefore an accurate structural diagnostics in aircraft maintenance must be performed regularly, commonly known as scheduled aircraft maintenance [35].

1.1. *Role of structural diagnostic in aircraft maintenance*

In world of aircraft maintenance, the Maintenance Steering Group – Task 3 (MSG-3) logic [29] has been a root methodology for modern aircraft maintenance and among the important tasks in the MSG-3 process is the Special Detail Inspection (SDI) [34] that has the purpose to detect hardly visible aircraft structural damages by a human inspector and requires non-destructive testing (NDT). Training an NDT inspector with various inspection methods until the proper certification is reached, however, is a relatively expensive in matter of time and costs. To induce more efficient inspection process, two mainstreams of automated SDI have been proposed: automated NDT [76] and Structural Health Monitoring (SHM) [45]. While automated NDT typically means robot-assisted non-destructive inspection (NDI), in SHM the NDI instruments become integrated into the structure itself [12].

Both SHM and NDI have their own pros and cons, but in both approaches, it is required to have a more complex processing for the data captured by the NDI instruments as the degree of complexity (not only the complexity of the aircraft structures on the geometrical level, but also on the material level) of the system increases. For this reason, we profoundly believe that more sophisticated digital signal processing (DSP) techniques are needed to be able to cope with these ever-increasing complexities.

1.2. *State of the art*

The main problem in classical signal processing is to extract signal features from one or multiple sensors and to process them to understand the behavior of a particular physical phenomenon of interest. Nowadays, in the era of increasing data volume and parallel computing, the major problem in signal processing has shifted into statistical signal modelling and the incorporating stochastic parameters into the statistical model. In the field of computer science, machine learning has gained in popularity in recent years. As machine learning is typically used to model statistical process, it is one of suitable techniques to model ‘real-world’ signals, which generally exhibit stochastic behavior. As an example, one popular machine learning technique that was introduced in 1990′s was Support Vector Machine (SVM) [24]. One example of using SVM for NDT can be found in the work of Virupakshappa and Oruklu [111] for ultrasonic flaw detection in steel block and in the work of Gong et al. [47] for defect classification by using thermal sensor.

More recently, people have been increasingly talking about ‘deep learning’ or a deep neural network (DNN) [97], which was introduced for the first time in the 1950′s as a perceptron and later emerged as a network and was called multilayer perceptron (MLP). Since MLP is an imitation of a biological neural network, it is also sometimes referred as artificial neural network (ANN). Deep learning as it is known today is a complex multilayered ANN, but technically a 2-layered MLP which was already known in 1970′s would also qualify as deep learning. ANN grew in popularity until the 1990′s [56], where at that time the computational ability was low and eventually the ANN popularity was overshadowed by SVM [68]. It was not until 2006 when [55] introduced the deep belief network (DBN), a class of DNN, where the neural network regained in popularity.

Since then, many approaches within the neural network realm have been proposed. The pure discriminative method in neural network includes the variants convolutional neural network (ConvNet or CNN) such as Inception Network [105], VGG-16 and VGG-19 [100], and CNN with skip connections, also known as Residual ConvNet or ResNet [52]. Beside ConvNet, there is also the recursive neural network that includes 1). linear chain of sequential operation such as recurrent neural network (RNN) and its derivative Enhanced Long-Short Term Memory (LSTM) [18] and Gated Recurrent Unit [20], 2). Reservoir based operation such as Deep Echo State Network (D-ESN) [40] and Deep Liquid State Machine (D-LSM) with neural plasticity [101].

For generative modelling, there are Deep Autoencoders (DAE) [6] and variational Autoencoders (VAE) [65], which is normally used for learning efficient data encoding and decodes the learned feature representation by reconstructing its input. A typical application for this kind of network is image and audio signal denoising. Besides these encoding techniques, there is semi-supervised approach that combines the discriminative and the generative approach such as Deep Convolutional Generative Adversarial Network (DCGAN) [92] and Energy-based Generative Adversarial Network [125].

In the area of model-free learning such as reinforcement learning, deep learning can be combined with a multi-agent system of reinforcement learning [38,67,83]. Since such a model-free environment is even more data-exhaustive than supervised learning, currently, the applications of these approaches are most limited to simulated reality such as computer game. The realization of an industrial commercialization such as fully robotics automation by multi-agent deep reinforcement learning still has a long way to go.

Focusing on diagnostic application, particularly in NDT, several applications of deep learning – which is largely based on CNN for crack visual detection have been proposed and these works generally focused on surface inspection of structures such as: [123,15,16,33,86,88] and many more. Beside for crack detection at surfaces, there are several other works involving CNN in NDT, such as for phase discriminating detection in shearography proposed by Sawaf and Groves [96], welding detection using X-Ray images by Hou et al. [57], and damaged steel and CFRP using infrared (IR) images by Zeng et al. [121], Yousefi et al. [119]. We are also in line with [117] who pointed out that advances in AI and machine learning will have a huge impact in several key areas of NDI.

In a similar way, deep learning has also brought some wave of excitement to diagnostic SHM, although there are less works exploiting deep learning for diagnostic SHM in comparison to NDT. In recent years, several works that incorporates deep learning in SHM have been proposed by Ebrahimkhanlou and Salamone [28], Ebrahimkhanlou et al. [27] who used deep autoencoder (deep AE) for acoustic emission (AE) source localization [22] and [25] who used CNN for processing electromechanical impedance (EMI), and [3], who used CNN for damage identification and localization of vibration sensor data in civil infrastructure. A review on the state of the art of data science for health monitoring in civil infrastructure was given in [7,8]. Some of the techniques they cover include anomaly detection for time-series data, various computer-vision technologies for crack identification, 3D-reconstruction, and fatigue life prediction. They also discussed AI-based disaster management. Lei et al. [69] used CNN for detection of structural damages imposed with seismic waves. They used wavelet-based transmissibility data of structural responses as an input into the deep CNN. Not only for damage classification, ConvNet has also been used by to localize damage in pressure vessel [58] and in plate-like structures [124].

Since CNN is a discriminative model that is specifically tailored to learn how to solve a certain task, i.e., once the model is trained, its parameters are fixed for solving that particular task only. Consequently, when the particular task is slightly changed (e.g., recognizing a car in autonomous driving instead of recognizing face in Facebook), a new deep learning model must be created. For this reason, transfer learning might be a temporary solution. Nevertheless, not only it would require a pretrained model based on a large dataset, but also a new model that still needs to be retrained based on the pretrained parameters. In many other domains such as NDT, such a large dataset is not publicly available thus making it more difficult to perform transfer learning. For Lamb wave based SHM there was a trial done by [72] although we doubt the efficacy of transferring image parameters for audio or acoustic wave signal processing. In line with other computer scientists [17], the reason for the non-transferability between image and audio is because such a pre-trained model was trained specifically for recognizing images that has a physical origin of photon particles captured by a RGB camera sensor which is a fundamentally different physical phenomenon from an acoustic wave. While in their work we believe at the end the network finally learns the features from the time–frequency spectrogram, we also think that there would be no advantage of using pre-trained image recognition models in comparison to the case where the network weights were just randomly initialized.

Another approach using online active learning has been proposed by Bull et al. [14]. They clearly recognized that the lack of descriptive labels made conventional supervised learning is hardly feasible and they proposed a novel adaptive learning process that updates the learning algorithm as soon as a new class of data is discovered. This approach bridges the gap of lack of labelled data temporarily, however it will fail at some point due to the incapability of the models to capture all possible cluster distributions since these tend to be infinite in nature. Conclusively, while we appreciate the numerous works that propose deep learning approach for SHM and NDT, a theoretical foundation that formalizes the utilization of deep learning for NDT and SHM is currently lacking. Therefore, some insight into understanding why deep learning might work for acoustic wave signal modelling for application in structural diagnostic is needed.

Further, there has been an increasing trend of using active acoustic based SHM approaches such as Lamb waves for continuously monitoring the ageing or repair of large aircraft with both metallic and composite structures. The concept of modelling and simulation can be applied to different stages while developing Lamb wave systems for monitoring large structures. During the design stage, it helps to choose optimal sensor positions and to examine the system response for various damage scenarios. During the operational phase, the simulation may be employed as a part of the system providing a virtual baseline result [93]. Some interesting contributions have been recently made by researchers especially in the development of theoretical Lamb wave models in SHM applications for both isotropic and anisotropic structures are [48,103,45,19,118,93,11].

Another major application of modelling of Lamb waves in SHM is to find an optimized sensor configuration for the given tolerable damages in a structure. In conventional ultrasonic NDT, the ultrasonic transducer head is free to move or scan over the entire structure to detect the damages, whereas in guided wave based SHM, the area to be monitored has to actively inspected by using a few transducers bonded on to the structures. For the latter case, the location of transducers plays a major role in the damage detection performance and hence, an optimized sensor configuration is one of the primary importance and it can be achieved through simulation. An example approach to determine the optimized sensor location for damage detection using numerical simulation was demonstrated by Boller et al. [11].

Numerical simulation of guided waves brings a thorough understanding of the governing mechanisms of the wave propagation when it interacts with structural features. Although extraction of desired modes using signal processing methods does exist Gopalakrishnan et al. [48], numerical analysis can typically aid the interpretation of those damages being observed through waves which have been mode converted as a consequence of the presence of damage and in such cases in order to obtain an appropriate understanding of what a physical principle such as mechanical and hence guided waves does when travelling through a structure, numerical simulations can be of invaluable help. Starting from such a simulation, the time domain signals to be monitored in practice can be generated for different damage configurations and will subsequently be processed in a sequence of steps ending up in a machine learning algorithm such that key damage features are derived that will then be used to realize an appropriate SHM system in practice. We previously proposed in such an approach to identify the damage features using simulation driven guided wave data for various damage configurations Asokkumar et al. [2] and it is needless important to mention that it would otherwise be impractical to perform this study using experimental data.

1.3. *Objective and outline*

Based on the state-of-the art in Section 1.2, we can assume the following premises to determine the research objective: 1). more complexity in geometry and material properties would require enhanced signal processing to capture signal features, 2). A pure physical model is normally more powerful, but typically requires a lot of effort and sometimes it also idealizes some assumptions that might not always correspond to the real-world situation, and 3). A pure statistical model can only find correlation, but not causation (also known as the 'black box property'), thus conclusions are difficult to understand. In the end, a compromise between a physical and a statistical model must be made in order to further progress the advancement of automated damage detection, be it in SHM or NDT.

In our previous work Ewald et al. [30], we demonstrated how to bias CNN with appropriate aerospace domain knowledge for both NDT and SHM applications. This was also in line with the approach proposed by Gardner et al. [41]. For the SHM application using active Lamb waves, we previously proposed a hybrid model that we called DeepSHM framework [32]. Specifically, it is a statistical signal modelling based on deep learning biased by a physical nature and we showed it easily worked for a complex signal classification. Like any other deep learning algorithm, the advantage of DeepSHM is its ***agnosticism***: it treats any input, and it gives any output given the input, so no matter how complex the signal is, the classification accuracy is tendentially very high. The biggest disadvantage of DeepSHM is also its ***agnosticism***: for any given bad input, the outcome would be a poor output, which is known in computer science as a ***Garbage in*** – ***Garbage out*** [64] process. While deep learning would work given any input sequence, we choose to align this research with our previous work and thus we limit the use of DeepSHM solely for active Lamb wave based SHM. One specific problem that we still encountered in [32] was that some of the algorithms could not make a distinction between the signals that come from a slightly similar distribution. The reason for this was the physical limitation that one particular wavelength is in general only suitable for detecting damage of a certain size. Our hypothesis to overcome this problem is:

1. Applying broadband frequency excitation since this will involve a broader wavelength distribution.
2. Varying the sensing locations to potentially obtain more information.

To do so, the ***business as usual*** of the machine learning is performed: the machine learning performance metrics in the confusion matrix will be compared with given captured signals from different sensing locations. As stated before, one of the associated problems with signal processing with multitude sensing locations is that it would result in different sensor responses which might give conflicting predictions (e.g., no damage based on the response of one sensor and there is damage based on the response of the other sensor) and this leads to following research questions:

1. How much do the varying sensing location and the different sensing representations of time–frequency Lamb wave signal influence the deep learning training behavior?
2. Given a posteriori knowledge from question (1), what consequence can be drawn for the engineering application in SHM and why should this approach work?

We would like to address these questions in a technical way to encompass the engineering knowledge. Nevertheless, to align with the main purpose of scientific research, we believe that just the demonstration of the technical approaches alone is insufficient to become a scientific contribution. To emphasize this, [91] stated: "*It is now generally accepted that the purpose, or at least the central purpose, of science is to explain, or perhaps to explain and predict*".

We are aware that many works involving deep learning – not only in SHM or NDT – propose advanced techniques that are superior to previous techniques without any explanation of why they worked. Thus, rather than proposing ***novel superior technique***, we took a step back in this paper to rather explain why deep learning should work for treating Lamb wave signal and where would it stop working as we are aiming to develop a ***methodology that makes reproducible result*** rather than just focusing on the result. In this paper we highlight the most important theorems and lemmas that bring the assertion of the reason why Lamb wave signal can be treated by deep learning. We believe that it is the SHM and NDT practitioners who aim to employ deep learning in their field work who will benefit from domain transfer from computer science into mechatronics system and engineering.

The way we organize the article is as follows: first we conceptualize the generic but fundamental idea about diagnostics, then we formalize DeepSHM by dividing it into several tuples of physical domains, and last we propose our thoughts about modelling SHM perception from a neuroscientific perspective in Sections 2.1 and 2.3. The formulation of CNN is briefly described in Section 2.4. To ease the data generation process, we chose to create synthetic data from numerical simulation and the brief theory behind this modelling is given in Section 2.5. The methodology regarding the sensor placement, numerical model, data pre-processing, and neural network training are described in Sections 3.1–3.4. The sample results and discussion are given in Section 4. Finally, the conclusion of our work is given in Section 5.

2. Concept and theoretical foundation

To explain the concept of the DeepSHM framework in more detail, we divide Section 2 into two parts: Section 2.1 describes the generalized diagnostic and the general SHM workflow, while in Section 2.2 we formalize the DeepSHM framework. Our abstraction regarding the modelling the perception of SHM from neuroscientific perspective is given in Section 2.3. The brief explanation of CNN is given by Section 2.4. The theory of Lamb wave propagation and its simulation is only briefly discussed in Section 2.5 as these are widely known.

2.1. *Diagnostics*

To be more generic and domain-agnostic with the explanation, let us consider two following real-world diagnostics examples given in Table 1. These are glucose monitoring (CGM) [54] for diabetic patient and magnetic particle inspection (MPI) in an aircraft hangar. We choose not to give any example within aircraft SHM as we are going to explain this in more detailed way later in this section. There are 5 aspects in each diagnostic scheme that can be considered: actor, transition, medium, phenomenon, and environment.

Categorizing these 5 aspects in the diagnostic realm, we abstract our idea as depicted in Fig. 1. Whether it is targeted for medical purposes such as CGM or electroencephalography (EEG), or for structural engineering purpose such as infrared-based NDT and guided-wave based SHM, these aspects can be summarized in following quintuplet $\boldsymbol{D}$: $\{\boldsymbol{\pi}, \boldsymbol{\psi}, \boldsymbol{\tau}, \boldsymbol{\lambda}, \boldsymbol{\omega}\}$. The meaning of all these symbols have been explained in Section 2.1 of [32].

Focusing our work on SHM, we follow the SHM workflow proposed by Ooijevaar [84] which is summarized in Fig. 2. Further, the function of each SHM levels have also been described in our previous works.

2.2. *Generalization of DeepSHM framework*

The most direct formulation to describe the relations between the domains in the diagnostic realm D can be written as:

$$\lambda = f(X(\pi, \psi, \tau, \omega)) \tag{1}$$

where $\boldsymbol{X}$ is an observable variable that describes the latent variable $\boldsymbol{\lambda}$ as a dependency on $\boldsymbol{\pi}$, $\boldsymbol{\psi}$, $\boldsymbol{\tau}$, $\boldsymbol{\omega}$, i.e., in reality $\boldsymbol{X}$ is the measured signal and can be mathematically expressed as either vector or matrix form and more generically as $\boldsymbol{X}$ is a finite-dimensioned observation tensor. However, as $\boldsymbol{\lambda}$ is generally hidden, what we typically observe as the measurement $\boldsymbol{X}(\boldsymbol{\pi}, \boldsymbol{\psi}, \boldsymbol{\tau}, \boldsymbol{\omega})$, so the relation can be rewritten as the inverse model:

$$X_{\pi,\psi,\tau,\omega} = f^{-1}(\lambda) = g(\lambda) \tag{2}$$

Since our universe tends to behave stochastically, every observation $\boldsymbol{X}$ perpetually changes for any given domain parameters. Thus, it would be naturally logical to describe the behavior of $\boldsymbol{X}$ as probabilistic variables rather than as deterministic ones. For brevity, let us assume the null hypothesis h_θ where the existence of $\boldsymbol{\lambda}$ is caused only by changing parameters in $\boldsymbol{\psi}$. Due to the first assumption of the stochastic nature of observation $\boldsymbol{X}$, the relation can be formulated via Bayes conditional probability P:

$$P(h_\theta(\lambda)|X_\psi) = \frac{P(X_\psi|h_\theta(\lambda)) \cdot P(h_\theta(\lambda))}{P(X_\psi)} \tag{3}$$

where in Eq. (3), $P(h_\theta(\lambda)|\boldsymbol{X}_\psi)$ is the posterior probability of the existence of λ given observations $\boldsymbol{X}_\psi$, $P(\boldsymbol{X}_\psi|h_\theta(\lambda))$ is the prior probability

Table 1
Sample comparison between medical and structural diagnostic.

Tuple	CGM [Heinemann and Stuhr [54]	MPI [Babu et al. [4]
Actor π	Amperometric enzyme (bio)-sensors and insulin pump, needle, control software	UV-Light, Human eye, Fluorescent magnetic particle powder, Magnetizer
Transition τ	Whole blood in suspension	Interface between the human eye and air, between air and suspension, and between suspension and magnetic powder
Medium ψ	Sera (blood plasma)	Air and suspension
Phenomenon λ	Enzymatic reaction called glucose oxidase (GOx) of the blood glucose level that causes fluctuation of peroxide (H_2O_2) concentration.	Interaction between photon (UV-Light) and fluorescent magnetic particle powder
Environment ω	Human body and its surrounding (e.g., high temperature) that might cause anomality during GOx	Aircraft hangar, external light that is not meant to be present during the observation of the phenomenon

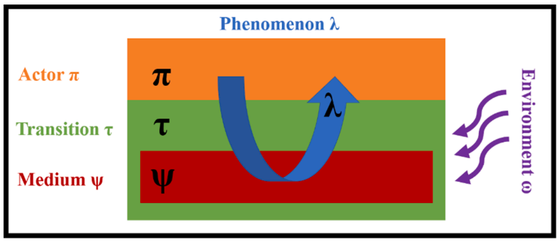

Fig. 1. Diagnostic realms *D*.

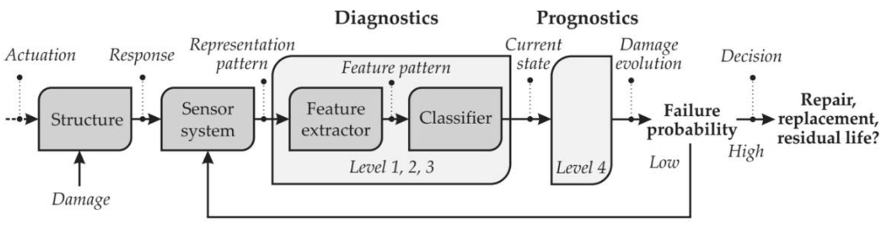

Fig. 2. SHM Workflow in General [84].

where $\boldsymbol{X}_\psi$ occurs given the hypothesis $h_\theta(\lambda)$. $P(\boldsymbol{X}_\psi)$ and $P(h_\theta(\lambda))$ are the marginal probabilities of observing $\boldsymbol{X}_\psi$ and $h_\theta(\lambda)$ independently, respectively. Furthermore, in Eq. (3), θ is the black-box parameters (which in machine and deep learning, these are normally called either synaptic parameters or simply neural network weights) that are to be optimized during the learning process. Given the fact that in the majority of the case, the observable $\boldsymbol{X}_\psi$ is multi-dimensional, it is logical to formulate the problem as a multivariate distribution rather than a uni- or bivariate distribution. To understand the concept of probabilistic formulation in diagnostic in more detail, we consider following:

Definition 1. Joint probability distribution as the product of marginal and conditional distribution [98].

Assume M is the product space of $(K \times L)$, a random variable p on (M, U) is the form (ξ, η) where ξ is a random variable on (K, S) and η on (L, T). Let $\{p_i\}$ be a i-th sequence of random variables on M and let let κ_i, μ_i, and υ_i denote the joint distribution of (ξ_i, η_i), the marginal distribution of ξ_i and the conditional distribution of η_i given $\xi_i = k$, respectively. Further, let us define the product (M, U) of two measure spaces (K, S) and (L, T) and a sequence of random variables (ξ_i, η_i) taking values in M. Assuming the existence of a conditional probability υ_i given $\xi_i = k$ which is a probability measure on $T\ \forall k \in K$ and a measurable function on $K\ \forall B \in T$, where A is a sub-space of S and B is a sub-space of T, it satisfies the equation:

$$\kappa_i(A \times B) = \int_A \upsilon_i(k, B) d\mu_i \forall A \in S \text{ and } B \in T$$

Furthermore, we consider the most suitable expression of the joint probability distribution. Naturally, there is an infinitely possible combination within $\boldsymbol{\psi}$ (e.g., for the length of a crack, the size of delamination, or corrosion depth) and very small variation within $\boldsymbol{\psi}$ normally only causes small variation within $\boldsymbol{X}_\psi$. Therefore, instead of discrete probability mass distribution, the probability density function is the more suitable formulation for a joint distribution in diagnostics since it expresses the density of a continuous random variable. Now, we have to consider how to associate the probability distribution with machine learning. Generally, when considering any machine learning algorithms, the following questions naturally arise: which learning problems can be solved efficiently and which are easier to solve than others? How many N training samples do we need, and which parameters θ must be tuned during the learning process? In computer science, the proper intuition would be the learnability of the function itself, commonly known as the probably approximately correct (PAC)-learning framework. Concretely speaking, the underlying assumption for the assumption stated in Eq. (1) and (2) is that we know h_θ: $f(\boldsymbol{X}_\psi) \rightarrow \lambda$ belongs to concept class C thanks to our physical understanding. Consider the PAC learning theory in Lemma 1.

Lemma 1. *PAC-learning* [109,110,49,79].

For any hypothesis $h_\theta \in H_C$, where H_C is the hypothesis space in the concept class C, the generalization error J is defined as $J_{h_\theta} = P(h_\theta(\lambda) \neq f(X_\psi))$. Such generalization error is impossible to calculate since the hypothesis space is infinitely large. Thus, we can come up with approximated measurable error over N samples, commonly referred as empirical error $J_{\widehat{h_\theta}} = \frac{1}{N}\sum_{n=1}^{N} P(h_\theta(\lambda) \neq f(X_\psi))$. Assume the existence of a learning algorithm A such that for any ε > 0 and δ > 0, for any distribution κ on input $\boldsymbol{X}_\psi$, the following holds for the training sample size $\left\{N = \frac{1}{\varepsilon} \cdot \ln\frac{|C|}{\delta}\right\} : \mathbf{P}\left(J_{h_\theta} \leq \varepsilon\right) \geq 1 - \delta$. Then, the concept class C is said to be PAC-learnable.

Consequently for Eq. (2), an algorithm A can be thought as a function that can inversely map $\boldsymbol{X}_\psi$ into $\boldsymbol{\lambda(\psi)}$. We denote the family of this function as generalizer L such that $A \in L$. As above stated, our assumption is that the universe tends to behave stochastically, thus L does not actually map the observation $\boldsymbol{X}_{\boldsymbol{\psi}}$ into $\boldsymbol{\lambda(\psi)}$ directly, but rather mapping $\boldsymbol{X}_{\boldsymbol{\psi}}$ into the joint probability density $P(h_{\theta}(\lambda)|\boldsymbol{X}_{\boldsymbol{\psi}})$. The definition of generalizer is as following:

Definition 2. Multidimensional Generalizer [114,115]

A k-dimensional generalizer is a countable infinite set of continuous functions from a subset of $(\mathbb{R}^k \times \mathbb{R} \times \mathbb{R}^k)$ to$\mathbb{R}$, from a subset of $(\mathbb{R}^k \times \mathbb{R} \times \mathbb{R}^k \times \mathbb{R} \times \mathbb{R}^k)$ to$\mathbb{R}$, etc. Per notation, a k-dimensional generalizer is a set of continuous functions $L_{\{i\}}$ along with domains of definitions, where$i \in \mathbb{N}$ and $L_{\{i\}}$ being from $\mathbb{R}^{i(k+1)+k}$ to $\mathbb{R}$. An $L_{\{i\}}$ that is invariant under any rotation, parity, translation, or non-zero scaling transformation, applied simultaneously to all the $\mathbb{R}^k$, including the question, or to all the $\mathbb{R}$, including the output, is a called heuristic binary engine (HERBIE). Further, a generalizer is said to upwardly compatible if $\forall L_{\{i\}}, i > k+1$, if the values of the entries of two datum spaces are identical, then the output of $L_{\{i-1\}}$ working on the rest of the learning set and on one of the two identical datum spaces.

Wolpert stated that an upwardly compatible HERBIE is a "*proper generalizer*". We would like to connect the upwardly compatible HERBIE with machine learning in [77], where he defined the algorithm of machine learning process as: "*A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E*". Specifically, this means that a trained algorithm, which is just the generalizer L in disguise from Def. 2, that has learned during machine learning process E and is said to be able to generalize on the class of tasks T from certain probability distribution as defined by Sethuraman [98]. In the diagnostic realm, task T is the diagnostic itself, that is retrieving the information from the observable variables $\boldsymbol{X}_{\boldsymbol{\psi}}$ regarding damage state, experience E is the iterative process for enhancing the algorithm to increase the accuracy performance of the trained algorithm that best generalizes the distribution over $\boldsymbol{X}_{\boldsymbol{\psi}}$. The probability for the i-th class of information given the $\boldsymbol{X}_{\boldsymbol{\psi}}$ in k-dimensional Hilbert space is typically written as a logit or sigmoid function [77], USDOD [75] and can be generalized in a softmax function which is defined by:

$$P(h_\theta(\lambda) = i|X_\psi) = \frac{\exp\left([X_\psi]^T \cdot \theta_i\right)}{\sum_{k=1}^{K} \exp\left([X_\psi]^T \cdot \theta_k\right)} \tag{4}$$

The posterior of $P(h_{\theta}(\lambda)|\boldsymbol{X}_{\boldsymbol{\psi}})$ is formally defined as [0,1], but is also sometimes informally written as between 0% and 100%. The posterior is maximized by minimizing the information difference, which can be referred as either as lost, cost or error between the predicted value $h_{\theta}(\lambda)$ and the true information contained $\boldsymbol{X}_{\boldsymbol{\psi}}$. The central task of machine learning is to minimize the loss function by iteratively adjusting θ. Depending on the formulation problem, different loss functions must be defined. For instance, in a regression problem, the mean squared error is typically chosen, while when looking further into classification problem, a cross-entropy loss is chosen since it maps a logistic output between 0 and 1 and thus is an appropriate measure to calculate a similarity between two distributions. The cross-entropy $H(p,q)$ between the true distribution $p(\boldsymbol{X}_{\boldsymbol{\psi}})$ and the estimated distribution $q(\boldsymbol{X}_{\boldsymbol{\psi}})$ is [94] defined as:

$$H(p,q) = -\sum_{X_\psi} p(X_\psi = \lambda) \cdot \log(q(X_\psi = h_\theta(\lambda))) \tag{5}$$

Within the context of machine learning formulation for a classification problem, the minimization of cross-entropy loss J_{θ} for N training samples can be rewritten from Eq. (5) as:

$$\operatorname{argmin} J_\theta = \min\left[-\frac{1}{N}\left(\sum_{i=1}^{N} p(X_\psi = \lambda) \cdot \log(q(X_\psi = h_\theta(\lambda)))\right)\right] \tag{6}$$

Depending on the machine learning algorithm complexity, θ can be infinitely dimensional, meaning that from Eq. (6), it can be easily assumed that learning in machine learning is a difficult optimization problem (also commonly called NP-hard) as it contains a high-dimensional combinatorial problem that is $\sim \boldsymbol{O}(\theta!)$. While it is almost impossible to reach $J_{\theta} = 0$, learning means we are striving for $J_{\theta} \to 0$, so there must be an upper limit where the probability measures converge – this theorem has been proposed by Sethuraman in Lemma 2, while the consequence can be summarized in Corollary 1.

Lemma 2. *Converging probability measure* [98]

Take the notations from Definition 1. Let $C \in U$ and $P_1, P_2, \ldots$, be the sequence of probability measures defined on a measurable space (M, V), then P_i converges strongly to $P(P_i \to P$ in symbols) if $P_i\,(C) \to P(C)\ \forall C \in V$.

Corollary 1. A proper probabilistic generalizer (upwardly compatible HERBIE) is said to have an upper converging generalization bound if the probability measures over the defined measure spaces strongly converges according to Lemma 2.

One statistical metric that is useful to indicate the upper convergence limit of generalization bound is the statistical classification accuracy A which is defined as:

$$A = \frac{\sum TP + TN}{\sum(TP + TN + FP + FN)} \tag{7}$$

where in Eq. (7), TP, TN, FP, FN is the true positive, true negative, false positive, and false negative rate, respectively. The definition of TP, TN, FP, FN can be found in [107]. It should be noted that often it is useful to influence the algorithm by domain knowledge, as we demonstrated before in our previous work [30]. Biasing the algorithm also includes determining the data distribution which is fed into the learning algorithm to have interpretable outcome and to avoid a ***Garbage in – Garbage out*** process. Therefore, it is very important to determine which task T the algorithm should perform at the beginning. For this reason, we expand the definition of an upwardly compatible generalizer in Def. 2 on deep learning by considering Def. 3:

Definition 3. Generalization gap and Rademacher Complexity [62,78,5]

Let N-samples exist in the sampling space of a training dataset $S_N \triangleq \{(x_1, y_1), \cdots, (x_N, y_N)\}$, where $x \in X_\psi$ and $y \in h_\theta(\lambda)$, and the generalizer $L(S) : X_\psi \to h_\theta(\lambda)$. The expected generalization risk is denoted as $R_{(L(S))}$ and the computable empirical risk is denoted as $R_{\widehat{L(S)}} = \frac{1}{N}\sum_{i=1}^{N} J_\theta(L(x_i), y_i)$, where J_θ is the error or loss function associated with L. The generalization gap G is defined as $G \triangleq R_{L(S)} - R_{\widehat{L(S)}}$. Further, if the codomain $J \in [0, 1]$, we have that $\forall\delta > 0$ with probability at of least $1 - \delta : \sup G \leq 2 \cdot R_N(L) + \sqrt{\frac{\ln\left(\frac{1}{\delta}\right)}{2N}}$, where $R_N(L)$ is the Rademacher complexity of L.

The Rademacher complexity R, together with the Vapnik – Chervonenkis (VC) dimension [46,23] are a measure of richness of a class of real-valued functions – that is, the capability of generalizer L is related to how complex it is. This can be explained more simply as: a linear function can be approximated with higher degree polynomial functions, but not the other way around. In deep learning, we usually split the training and validation set during the training and search an appropriate model in the hypothesis space by tuning the network parameters to minimize the error. In order to guarantee the upper bound of the generalizer, Kawaguchi et al. [62] proposed a theorem via validation error as given in Lemma 3.

Lemma 3. Generalization bounds of deep learning via validation [62].

Let the sampling space S_N be split according to the true distribution $P_{(X,Y)}$ in S_N^{train} and S_N^{val} that denote the training and validation datasets, respectively. Let $\aleph_{L,i} = R_L - J_\theta(L(x_i), y_i)$ for each pair $(x_i, y_i) \in S_N^{\text{val}}$. For any $\delta > 0$, with probability at least 1-δ, then the following holds $\forall L \in L^{\text{val}}$:

$$R_L \leq R_{L(S_N^{\text{val}})} + \frac{2 \cdot (\sup|\aleph_{L,i}|) \cdot \ln\left(\frac{|L^{\text{val}}|}{\delta}\right)}{3N_{\text{val}}} + \sqrt{\frac{2 \cdot (\sup E(\aleph_{L,i}^2)) \cdot \ln\left(\frac{|L^{\text{val}}|}{\delta}\right)}{N_{\text{val}}}}$$

where L^{val} is defined as a set of models L that is independent of a held-out validation dataset S_N^{val}, but can depend on the training dataset S_N.

Lemma 3 practically explains why deep learning can generalize well if the generalization bound is reached which is guaranteed by the converging validation error despite possible sharp local minima or non-robustness. Thanks to corollary 1, we can now summarize the conclusion in corollary 2.

Corollary 2. An upwardly compatible deep learning model reached its upper generalization bound when the validation loss converges.

In their proposition, they mentioned that in the worst case where $\sup|\aleph_{L,i}| = 1$ and $\sup E\left(\aleph_{L,i}^2\right) = 1$, given a large hyperparameter cardinality (e.g., 10^6) and 1000 training epochs in a larger dataset $N_{\text{val}} = 10000$, the second and third terms of the equations sums up only to 6.94%. In the non-worst case, this figure decreases to 0.49%. Extrapolating this for the case where the dataset amount is smaller

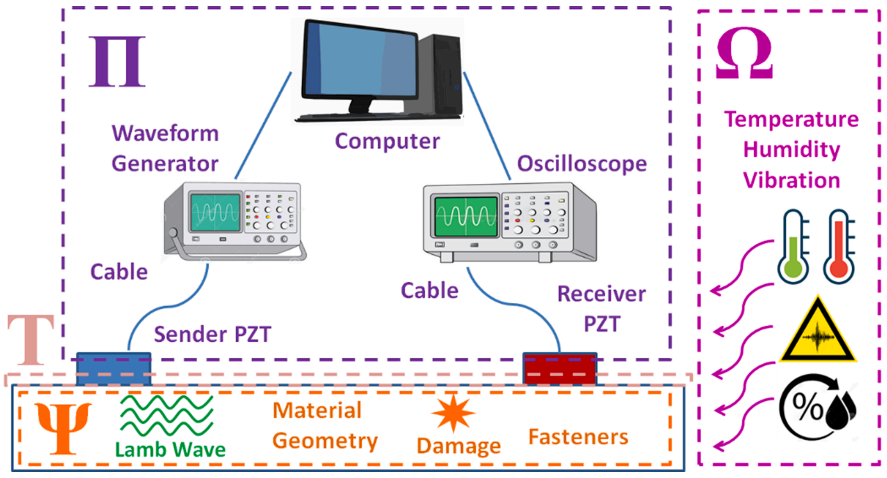

Fig. 3. Diagnostic SHM by using an active guided Lamb wave as physical phenomenon.

(e.g., $N_{val} = 100$) with a less complex deep neural network architecture with a smaller cardinality (e.g., 10^2) and 100 training epochs, we have a probability of 0.9 that $R_L \leq R_{L(S_N^{val})} + 55.66\%$. This means that eventually a smaller deep neural network would not generalize well over the hypothesis space as the generalization gap is large. Depending on the problem however, one could argue that for some problem settings, an inferior generalization capability is still better than anecdotal generalization. Further, the implication of an upwardly generalizable of a diagnostic algorithm, be it either for NDT or SHM – or even for a medical application (such as cancer detection), is to increase true positive (*TP*) and decrease false negative (*FN*, equivalent to statistical error of type II) detection rate, thus maximizing the probability of detection (*POD*), or sometimes called the sensitivity or recall rate, which is defined as:

$$POD = \frac{\sum TP}{\sum (TP + FN)} \tag{5}$$

The current standard practice for diagnostic NDT according to MIL-HDBK [75] is a $POD = 0.9$ (or 90%) within 95% statistical confidence σ, although this might not be suitable for diagnostic SHM [51]. To be more concrete with the concept of DeepSHM, we redraw the generic SHM formulation proposed by Ooijevaar in Fig. 2 for an active Lamb wave SHM system as shown in Fig. 3. The processing framework of the observable $\boldsymbol{X}_{\boldsymbol{\psi}}$ containing damage information captured by the actor π using a trained deep learning to predict the hypothesis $h_\theta(\boldsymbol{\lambda})$ can be seen in Fig. 4.

In line with any machine learning optimization process, the central task within the DeepSHM framework is to find generalizable parameters θ to fit the correlation between $\boldsymbol{X}_{\boldsymbol{\psi}}$ and $h_\theta(\boldsymbol{\lambda})$, where in guided Lamb wave SHM, $h_\theta(\boldsymbol{\lambda})$ is defined as the hypothesis of the damage information contained in medium domain $\boldsymbol{\psi}$ that is influenced by interaction between phenomenon Lamb wave ($\boldsymbol{\lambda}$) and the damage itself.

2.3. *Model abstraction of DeepSHM behavior*

We believe that DeepSHM according to Fig. 4 should be specified in the physical domain, and thus in this section we discuss the modelling behavior of the deep learning framework for Lamb wave SHM. Further, rather than just focusing on technical aspects on how to optimize the network parameters or improve the performance metrics, we consider it be more valuable to take into account an inspirational insight from other domain such as biology (especially auditory and neuroscience) and control theory. We believe that in the future AI will behave close to that of human intelligence. As SHM itself is inspired by biology, we believe that DeepSHM should not just be capable of actuating and sensing, but it should have its own perception. While the term ***machine perception*** is loosely defined, let consider the following definition from the example of autonomous driving [89]:

> “*Environment perception is a fundamental function to enable autonomous vehicles, which provides the vehicle with crucial information on the driving environment, including the free drivable areas and surrounding obstacles’ locations, velocities, and even predictions of their future states. Based on the sensors implemented, the environment perception task can be tackled by using LIDARs, cameras, or a fusion between these two kinds of devices.*”

Analogously, we can adapt the above-mentioned definition for SHM: ***Perception in diagnostic SHM is a fundamental function to enable the full functionality of an autonomous damage detection system, which provides the SHM system with crucial information on the changes in the sensory input (such as change in amplitude, frequency, or phase-shift)***. Before going deeper into the technicalities, consider a real-world setting that is inspired by the biological auditory cortex, that is a human-centered perception of hearing a song in a closed room as depicted in Fig. 5.

Assume the room can have any arbitrary length, width, or height. Let person 1 sing a song ***G*** in the room, while persons 2, 3, and 4 are listening. Their standing position in the room is arbitrary. The condition here would be, that while person 2 is listening, person 3 and 4 cannot listen to anything, and while person 3 is listening, person 2 and 4 cannot listen anything either, or vice versa. Given that persons 2, 3, and 4 do not have any hearing impairment, we might assume that they would probably tell us that person 1 is singing the song ***G***. However, when we replace person 2, 3, and 4 with just a headphone 2, 3, and 4, and record the electronic signal from the song, the signal set would entirely be different from each other.

Now, for Lamb wave SHM that practically uses the physical phenomenon of dispersive wave on the surface, we can think to have a machine perception built on deep learning framework that is biologically inspired particularly from human auditory perception.

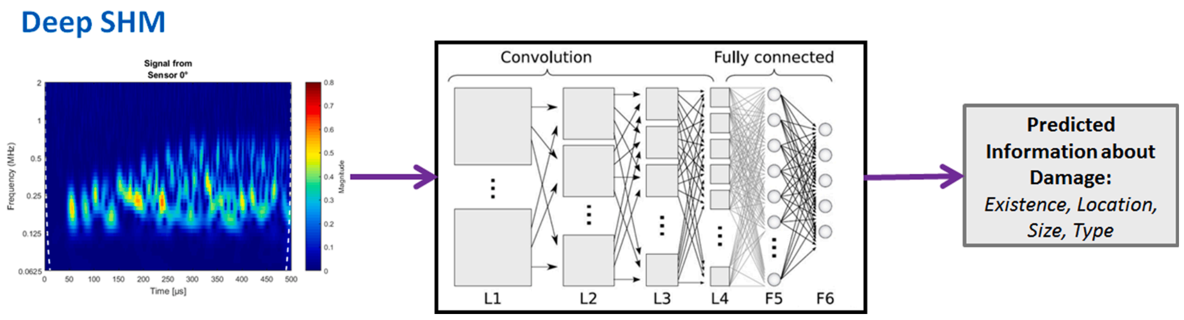

Fig. 4. DeepSHM Framework.

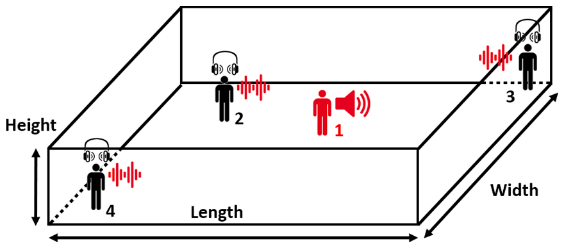

Fig. 5. Simplified setting on a person 1 singing in an enclosed space where the others are listening.

Consider the hearing process [104], hearing consists of two parts: 1). The conversion of mechanical (audio) waves that travels through concha, malleus and cochlea into electrical stimulation by stereocilia and 2). Information processing that is carried by neurotransmitter through auditory nerves [90] into the auditory cortex located in the temporal lobe of human brain.

Part 1 is the biological analogy of the conversion of an analog Lamb wave signal into an electrical signal by piezoelectric transducer (PZT) thanks to the piezoelectric effect. Part 2 is analogous to SHM signal processing to extract information from the recorded electrical signal to have a meaningful interpretation. The only difference here is, while normal person would have two ears (equivalent to 2 biological sensors), in Lamb wave SHM we only have a single sensor per position (and thus analogous to a half-deaf person). While there are many possibilities to perform the first part, as this paper is focused on DeepSHM – logically we limit the scope of our work to part 2 only.

To understand about this idea, we must first consider the Hebbian postulate [53]: "*Let us assume that the persistence or repetition of a reverberatory activity tends to induce lasting cellular changes that add to its stability…. When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased.*" In short, the Hebbian learning rule [112] states that the connectivity between two neurons is determined by how strongly they are linked to each other, i.e., the connectivity between two neurons increases if the they are simultaneously activated and decreases otherwise. Practically, if the auditory cortex is fed with the same song multiple times, at some point the brain would automatically recognize it the next time it is played thanks to the neuronal memory allocation induced by the neuroplasticity [66,108].

More interestingly, sometimes our brain still recognizes the same music even if it has been remixed – so the assumption here is that there must be shared invariant features between the original and remixed song. For cross-domain knowledge transfer between neuroscience into Lamb wave SHM application, consider that for any given set of programming rules (or learning algorithm) without human supervision, the signal sets can hardly be told to originate from song ***G***, as the recorded signals would be likely to have entirely different amplitudes, phase shift, and frequency from each other. We believe that human perception, on recognizing song ***G*** from different timeframe and locations in the room has a ***quantization vector*** that creates an invariant representation given any same object, which in this case is the song ***G***. To help understand what invariant representation means, consider Lemmata 4–6:

Lemma 4. The affine transformation of multivariable function composition O (trivial proof)

Let a set F^k, where $k \in \mathbb{N}$ be defined as $F^k \triangleq \{f | f : \mathbb{R}^k \to \mathbb{R}^k\}$ and $(X_1,\ldots,X_k)$ affine spaces. A composition O is defined as: $O : F^k \times F^k \to F^k$. Let (X, Y, Z) be the hyperspaces of the affine spaces $(X_1,\ldots,X_k)$, $(Y_1,\ldots,Y_k)$, and $(Z_1,\ldots,Z_k)$ that contain point sets $(x_1^{(k)}, \cdots, x_i^{(k)})$, $(y_1^{(k)}, \cdots, y_i^{(k)}$ and $(z_1^{(k)}, \cdots, z_i^{(k)})$ where $i \in \mathbb{N}$. Let (U,V) be a set of linear maps, and $(U : X \to Y)$, $(V : Y \to Z)$ be affine transformations. A map $U : X \to Y$ is said to be an affine map if $\exists V : Y \to Z$ such that $V(y - z) = f(y) - f(z) \forall y, z$ in (Y,Z). Then, the composition $O : V^\circ(U : X \to Y)$ is an affine transformation with tangent map $V^\circ U$.

Lemma 5. The cumulative distribution function (CDF) as the estimate of invariant for affine transformation [71].

Assume an observation $x \in X_\psi$,where X_ψ is a set of all possible observations. Further, we consider transformation $g_t \in G$, where G is a set of 2D affine groups that consists of all possible transformations T, such as translation, resizing, etc. Assume a set of smallest atomic representations of observation $\times$ as τ_i where $i \in \mathbb{N}$. For the set $\{g_t \tau_i | t = 1, \cdots, T\}$, the distribution of $\langle x, g_t \tau_i \rangle$ is invariant and unique to each observation. The empirical distribution κ of the inner products $\langle x, g_t \tau_i \rangle$ can be used as the signature: $\kappa_n^i(x) = \frac{1}{T} \sum_{t=1}^{T} \sigma(\langle x, g_t \tau_i \rangle + n\Delta)$,where σ is the sigmoid function and Δ is the resolution parameter for each observation. Since each observation $\times$ has its own characteristic empirical κ, it also shows that these signatures could be used to discriminate between them.

Lemma 6. Invariant latent [37].

Let us consider class label $y_{\lambda^{(n)}}$ that can be associated with the phenomenon λ in Eq. (1). The full set of class y-labeled data $\{x_n | h_\theta(x_n) = y_{\lambda^{(n)}}\}$ will be denoted as $D_{h_\theta}^y$ .The invariant latent $r_{y_{\lambda^{(n)}}}$ of label $y_{\lambda^{(n)}}$ is thus defined as: $r_{y_{\lambda^{(n)}}} = \mathrm{E}_{x\sim}($ where N is the number of sampled observations.

The importance of these above-mentioned Lemmata is as follows: 1). Any (non)-dispersive acoustic wave signal (including body and surface waves such as P-, SV-wave, Lamb-, Love-, Rayleigh, SH-wave, and/or any possible but finite combinations within the wave subset) can be regarded as a decomposable multivariable function of multiple polynomial functions and there exists an affine function that transforms the space of signal features to affine spaces of those signal features. 2). The transformation of each atomic representation of wave is invariant – these has been used e.g., in wavelet transformation, and 3). In a more high-level waveform composition such as longer time-series, the invariant latent can be explicitly calculated in order to provide the information needed to learn which wave packet within the time-series from all possible damage classes have in common – e.g., the first and second wave packet for similar damage sizes are all similar, i.e., having a cosine similarity close to 1. Logically, the reverse is also valid, and this is stated in Lemma 5: the invariant signatures can be used in order to discriminate [71] between waveform composition.

In March 2019, the innovator club at Merck proposed these research questions in the brain hack challenge that they called ***The Future of AI*** [73]. In one of the challenges, they were looking for a formalization of the cortical algorithm, a mechanism which might be able to imitate the pattern recognition process that is believed to be taking place in the grid neuron, which is located in mammal neocortex. To rephrase this is plain English: during this recognition process [61,50] (e.g., recognizing a face or a particular soundtrack), the human brain is known to have the ability to capture an invariant representation of the object. In those challenges, they defined the jargon and terms very loosely, but we understand what they are trying to communicate, and we believe they could be making a fundamental breakthrough in neuroscience. Should this be the case, it would have a tremendous impact on machine and deep learning community, and thus any subsequent domain that would benefit from the progress from machine learning, such as predictive maintenance by NDT and SHM. In one of their core problems statements, two assumptions are made:

1. "*Hierarchical structure: Entities are hierarchically structured, which means that an entity E is either fundamental* (i.e.*, it cannot be broken down any further) or it is composed of other entities* $E_1, E_2, \ldots, E_i$ *such that every perception p of E is associated with a set of perceptions* $p_1, p_2, \ldots, p_i$*.*"
2. "*Entity conservation over time: Subsequent perceptions in time are (usually) generated by the same set of entities.*"

As the result of the challenge has not yet been published at the time of the writing of this article, we currently can only assume that there are no other plausible assumptions other than those two mentioned. The reason we are considering this fringe idea from neuroscience is because it gives us a hint how we should treat the SHM signals. Concretely, given any arbitrary structure with k-sensors, we can adapt those two assumptions as follows:

1. ***Hierarchical Representation of signals***. The signals can be captured anywhere in the structure, but they are represented separately. As such, treating such images would require multiple learners in parallel as no sensor data fusion is required because each node acts as an independent actor. Thus, multiple output comes from k-sensory inputs forming k-possible perceptions. From there, we can summarize the total perceptions, by averaging the sum of k-perceptions. In this case, the invariant representation is the atomic decomposition of each observation.
2. ***Conserved Entity over Time***. In this case, DeepSHM is represented as a single entity – just like the song ***G*** in our previous example, assuming a bijective projection as given by Lemma 7, any arbitrary signal that comes from certain (un)-damaged structure can only be associated with that structure, no matter where the sensors are placed. As such, k-sensory inputs can be represented in a stack that consists of k-dimensional array. Consequently, treating such images would only require a single model and thus DeepSHM would only act as a single actor and in this case, each layer of the k-dimensional image becomes the invariant representant of the whole observations.

Lemma 7. The bijective relation maps an input space X into output sets Y (trivial proof)

Consider $f : X \to Y$, where $X, Y \in \mathbb{R}^k$, then f is surjective if for $f(x_1, \cdots, x_k) = [y_1, \cdots, y_k]$ there exists at least one solution $x \in X$ for each $y \in Y$, i.e., $\forall y \in Y : \exists x \in X : [y_1, \cdots, y_k] = f(x_1, \cdots, x_k)$. Further, f is said to be injective, if for $f(x_1, \cdots, x_k) = [y_1, \cdots, y_k]$ there exists at most one solution $x \in X$ for each $y \in Y$, i.e., $\forall x^{(1)}, x^{(2)} \in X : f(x^{(1)}) = f(x^{(2)}) \Rightarrow x^{(1)} = x^{(2)}$. Finally, f is said to be bijective if f is both surjective and injective.

For the conserved entity over time, ideally not only the full-length signal but also the different observation perspective must be taken into account. An analogy for this assumption would be looking at particular car which is moving on the street: there is only one car that we are looking at once, but it can be looked from different angles, e.g., from the front, left, back, etc. – see Fig. 7. But at the end, this car represents exactly only a single entity.

The corollary of lemma 7 for DeepSHM can be then rephrased as:

Fig. 7. Representation of entity conservation over time, example showing a car from different perspectives [Opel (online)].

Corollary 3. Changing any parameter in the diagnostic quintuplet **D** given by Table 1 would cause the phenomenon λ to change. In other word, each λ is only unique to each parameter combination.

For example, the change of the material, dimension, boundary conditions, sensor geometry, etc. and/or combination of these parameters, would cause λ to behave differently, so that the observation set $\boldsymbol{X}$ would also change. Under the assumption of corollary 3, we postulate that every observation $\boldsymbol{X}$ to the damage class $\boldsymbol{Y}$ belongs to a certain joint probability distribution that is distinct to each other but shares certain invariant features that can be estimated according to Lemma 7 and 3 and this can be summarized further in corollary 4.

Corollary 4. If a certain probability of damage class $P(\mathbf{Y_p})$ from every possible observation $\mathbf{X_p}$ shares the same invariant estimate with probability of damage class $P(\mathbf{Y_q})$ of every possible observation $\mathbf{X_q}$, then they belong to the same joint probability distribution.

The implication of corollary 4 is: if we put two sampled observations from distribution p and q into two distinct classes, the value of the joint probability of each class would be the same because they share the same invariant features. Clearly, we would like to know how these two perception models behave. We are interested on their capability to relate the perception with the actual phenomenon λ described in Section 2.1. The metric used can be for instance the classification accuracy, as given by Eq. (7).

2.4. Convolutional neural network (CNN)

To process the observed signal $\boldsymbol{X}_\psi$, there are several deep learning architectures available, depending on how we would like to treat $\boldsymbol{X}_\psi$. For brevity, we focus on the CNN as this has been used in our previous work. In general, there are two kinds of layers that are normally used in the core architecture of CNN: the convolutional layer and the pooling layer, as depicted in Fig. 8 [113].

The core CNN is normally attached to the fully connected (FC)-layer, which is also called the dense layer. Should this dense layer consist of multiple layers, then these are simply called multilayer perceptrons (MLP). The convolutional layer is made of a filter (or kernel) which performs the Hadamard transformation to the tensor that it receives from the previous layer. This process can be thought as layer-wise feature extraction to generate feature maps by sliding over the input data every s-pixels (called stride) as depicted in Fig. 9.

2.5. Model-assisted SHM by active Lamb wave propagation

Although the efforts that need to be taken to obtain Lamb wave data from a single experiment is less time consuming, the process of fabricating a test component with various damage configurations is expensive and time consuming, and in most cases a large parametric study would be impossible to perform through experimental methods. So, it is needless to mention that a Lamb wave analytical model is an essential requirement for such studies. However, building a such analytical model is complicated as most of the engineering applications are driven by complex systems and are governed by multivariate Partial Differential Equations (PDEs). Subsequently, full-analytical solutions for these systems are not possible in most situations and numerical solutions are often sought in those cases.

There are different mathematical ways one can obtain numerical solutions of Lamb waves such as Finite element method (FEM), Finite difference method (FDM), Elastic Finite integration technique (EFIT) and spectral FEM (SFEM) [85]. In the current study, a 3D-FEM model has been developed by means of COMSOL-Multiphysics to generate and receive GW in the given plates [43,44]. This is achieved by coupling the piezoelectric devices and the structural mechanics modules in the COMSOL software. The material behavior due to applied force is studied within the structural mechanics module and the piezoelectric effect is studied within the electrostatics domain. This is shown in Fig. 10 where a piezoelectric material is being applied as a common interface coupling the two physical models.

In an early study, a 2D-FEM model was analyzed by Nieuwenhuis et al. [82] regarding the generation and detection of guided waves using PZT. Zennaro et al. [122] has developed a numerical simulation to study and model the transducers for Lamb wave generation. The governing equations for the piezoelectric effect, i.e., the generated electric field for the applied stress, and the inverse piezoelectric effect, i.e., the total strain generated for the given electric field, can be written with Voigt notation index i, j, k, l as follows Giurgiutiu [45]:

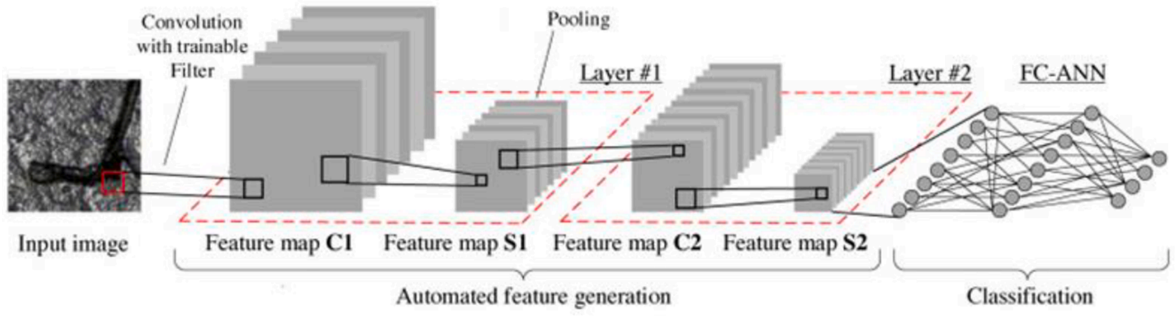

Fig. 8. Typical example architecture of a convolutional neural network (ConvNet) [Weimer et al. [113].

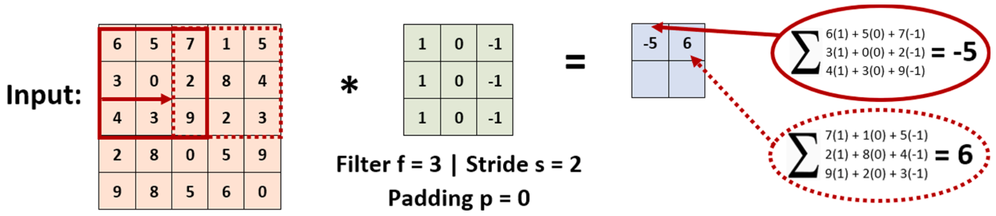

Fig. 9. Convolution operation in CNN using stride $s = 2$ and zero-padding [Ewald et al. [30]].

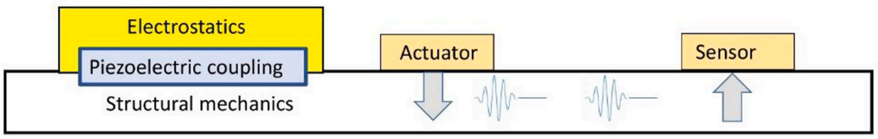

Fig. 10. Finite Element multi-physics model using COMSOL-Multiphysics.

$$S_{ij} = S^{E}_{ijkl}\sigma_{kl} + d_{kij}E_k + \delta_{ij}\alpha^{E}_{i}T \tag{6}$$

$$D_i = d_{ikl}\sigma_{kl} + \xi^{\sigma}_{ik}E_k + \widetilde{D}_i T \tag{7}$$

where S^{E}_{ijkl} is the mechanical compliance tensor measured at zero electric field $E = 0$, S_{ij} is the mechanical strain tensor, σ_{kl} is the mechanical stress tensor, and d_{kij} represents piezoelectric coupling effects. E_k is the electric field, D_i represents electrical displacement and ξ^{σ}_{ik} denotes dielectric permittivity at zero mechanical stress ($\sigma = 0$). T is a temperature term and α_i is the coefficient of thermal expansion. $\widetilde{D}_i$ represents the electric displacement temperature coefficient and δ_{ij} is the Kronecker delta. Eqs. (6) and (7) are called the piezoelectric constitutive equations and they are used to predict how much strain and electrical displacement will be created at given stress, electric field, and temperature. Eq. (7) can be represented in the following form to predict the electric field for a given applied stress:

$$E_i = g_{ikl}\sigma_{kl} + \beta^{\sigma}_{ik}D_k + \widetilde{E}_i T \tag{8}$$

where E_i is the electric field and g_{ikl} is the piezoelectric voltage coefficient that denotes how much electric field is induced by the applied stress. The coefficient $\widetilde{E}_i$ is the pyroelectric voltage coefficient that represents how much electrical field is induced per unit temperature change and β^{σ}_{ik} represents the electric permittivity matrix. Eq. (8) is called the sensing effect equation. Piezoelectric materials can be used as sensors to measure deformations within the structure through their direct piezoelectric effect and as actuators to send acoustic waves into the structure through their converse piezoelectric effect.

When a voltage is applied to this piezoelectric material whilst attached onto the structural surface an in-plane motion is generated causing Lamb waves to propagate. Several different materials with piezoelectric properties are available. Although piezoelectric properties occur naturally in some crystalline materials (e.g., quartz crystals and Rochelle salt), the piezoelectric effect can be induced by electrical poling of certain polycrystalline materials such as piezoceramics. Due to their high piezoelectric coefficients, Lead Zirconate Titanate is one example of a piezoceramics material that is being widely used for SHM. In this case the respective transducer geometry is given a time dependent boundary load. The application of such models can be referred to in [45].

3. Methodology

As a study case for the preliminary concept DeepSHM, the active Lamb wave SHM system that has been previously employed in our previous works was used. Section 3 of our paper is organized as follows: the sensor placement of the DeepSHM framework for Lamb wave is described in Section 3.1, while method to generate and multiply the data in a Finite Element (FE) environment is described in Section 3.2. The explanation on the data pre-processing method to convert from time-domain into time–frequency domain and the CNN training method are described in Sections 3.3 and 3.4, respectively.

3.1. Lamb wave propagation setup and sensor placement method for SHM

In this study, an aluminum 2024 (Al-2024) plate is used which is a commonly used material in aerospace fuselage structures. Since the energy of the Lamb waves are mostly confined within the plate, they can be efficiently used to monitor a relatively large area and depending of the wavelength, it can be between less than a meter up to several hundreds of meter. The Lamb waves are dispersive in nature and analytically the dispersion phenomenon can be expressed in terms of wavenumber vs frequency relation [93] from which relationship such as phase velocity vs frequency and group velocity vs frequency such as shown in Fig. 11 (left and right, respectively) can be derived.

To save the computational resource, the size of aluminum plate is limited to 600 × 400 × 2 mm. Additionally, a crack with the full-length $2a$ is assumed to grow from the rivet hole which is located at 400|200 mm as depicted in Fig. 12.

The half-crack length including the notch is denoted as a, which measured from the center of the rivet hole. Due space constraints, we only demonstrate 3 crack growth scenarios. They vary from 0 mm (pristine plate) up to 200 mm length which is assumed to be the critical crack length a_{crit}. The total crack length $2a$ in percentage as a proportion of 200 mm is presented for multiple classification scenarios in Table 2.

To determine the hotspot sensing location, we used the previously developed blob detection algorithm [31]. The algorithm to obtain the fused image of the wave propagation as depicted can be downloaded in our [Github repository]. Two sensing locations of the centroid with maximal blob area were used: at 1). 360|200 mm and 2). 510|200 mm. To compare the metrics, we can assign another randomly additional sensing location (numbered as location 3 – 8, see Fig. 12).

3.2. Simulated PZT sensor response using FEM

In each simulation scenario, there are two sub-simulations running: 1). The Lamb wave propagation in the mechanical regime, and 2). The piezoelectrical conversion from the mechanical regime into the electrical regime. Each additional piezoelectrical response simulation means additional simulation time. The total simulation time is the time needed for sub-simulation 1 plus the time needed for sub-simulation 2. The sub-simulation 1 runs only once because there was only one plate, however, the sub-simulation 2 will take a longer time as the number of sensor position increases. Depending on the number of assigned PZT locations, the total simulation time varies between 6 and 12 h. Since there are 6 crack length classes (including the baseline) in scenario 1, 6 simulations must be performed, which will take at least 36 h to complete. Accordingly, the complete scenario would take 54 h to complete.

To reduce the simulation time and to ensure that there was enough space in our hard drive, in the simulation of scenario 1 we only simulated for sensors located at positions 1 – 4 as the classification scenarios in the crack lengths are more distant from each other (i.e., the simulation request in sensor location 5 – 8 was disabled). In scenario 2, however, as the classifications become more difficult because sensor signals from 10% crack length would be more likely to be similar to a 20% crack length, we simulated the response from all 8 sensor positions, making each simulation twice longer due to the FE calculation of additional electromechanical conversion in sensor position 5 – 8. The combined sensor data from scenarios 1 and 2 are made into scenario 3, although due to the lack of the data of the other 4 sensors (i.e., sensor locations no. 5 – 8), we could only merge the 4 sensors data with the same location from scenario 1 and 2 (i.e., sensor locations no. 1 – 4). To generate the Lamb waves using the piezoelectric effect in the numerical model, COMSOL Multiphysics was used. The guiding medium is an Al-2024 plate with dimensions 600 × 400 × 2 mm and PZT transducer disks of diameter and thickness 9.52 and 1 mm, respectively, were considered. The material properties are Young's modulus E = 73.1 GPa, Poisson's ratio μ = 0.33 and density ρ = 2780 kg/m^3 Bauccio [9].

In COMSOL, the predefined PZT properties (orthotropic, Type 5A) were defined as follows (values in GPa): $C_{11} = C_{22} = 120.35$; $C_{33} = 110.88$; $C_{44} = C_{55} = 21.05$; $C_{66} = 22.58$; $C_{12} = C_{13} = C_{23} = C_{21} = C_{31} = C_{32} = 75.1$; the coupling matrix values are as follows (values in C.m^{-2}): $e_{15} = e_{24} = 12.3$; $e_{31} = e_{32} = -5.35$; $e_{33} = 15.78$; and the relative permittivity matrix elements are as follows: $\varepsilon_{11} = \varepsilon_{22} = 919.1$ and $\varepsilon_{33} = 826.6$ with density ρ = 7750 kg/m^3. The excitation pulse is a chirped Gaussian pulse with central frequency of 310 kHz and bandwidth of 100 to 500 kHz. To fulfil the Courant-Friedrichs-Lewy condition [26], a sampling frequency of 10 MS/s 20 times above the Nyquist frequency, was used to measure the signal to ensure there are enough sampling points. Such a kind of a numerical model has been already validated by the time of arrival method and a previous experiment [106].

The simulation was modeled without considering external environmental influences, and human factors, which results in identical

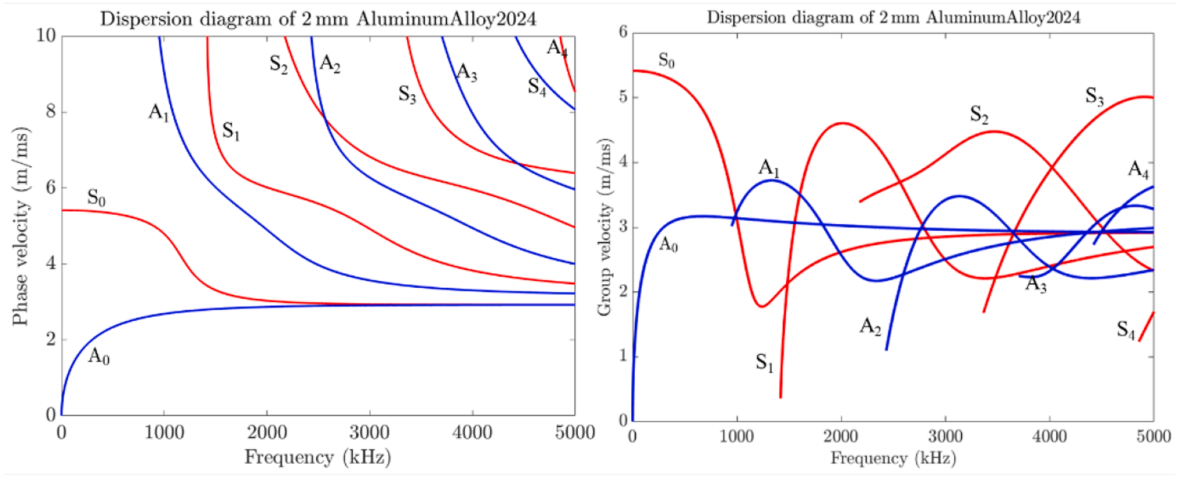

Fig. 11. Dispersion diagram for Aluminum 2024 with a thickness of 2 mm. Red: Symmetric modes Blue: Anti-symmetric modes. The dispersion curve is generated using German Aerospace Center (DLR) dispersion calculator [[60]].

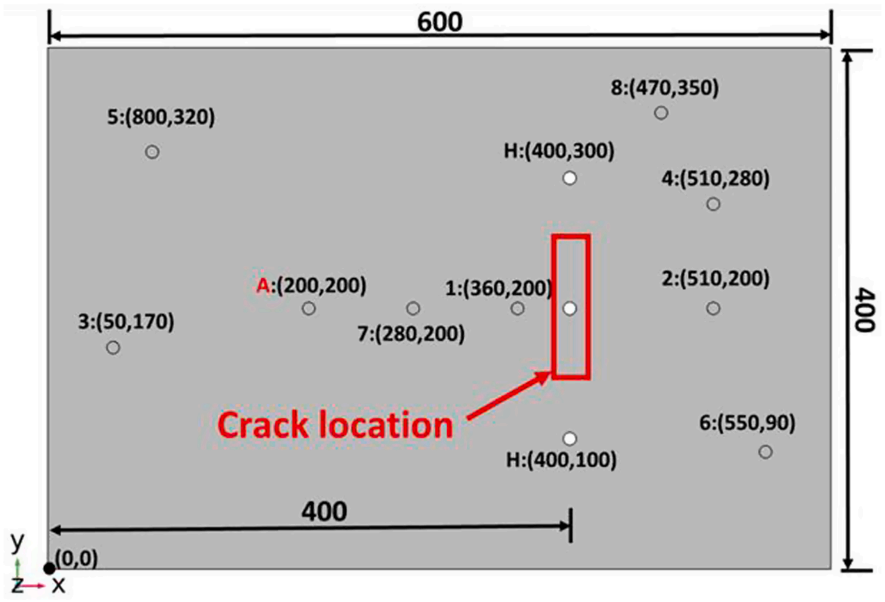

Fig. 12. Sensor positions are numbered from 1 to 8. The coordinates of each sensor position are written in the bracket. A is the actuator; and H are the rivet holes. All dimensions are expressed in mm. The sensor locations are numbered from 1 to 8.

Table 2
Classification scenarios with different percentage of crack length with reference to $a_{crit} = 200$ mm.

Scenario 1	6 classes: Baseline – 20% – 40% – 60% – 80% – 100% of a_{crit}
Scenario 2	9 classes: 10% – 20% – 30% of a_{crit} with varying angle (0°, 15°, and 45°)
Scenario 3	15 classes: Combination of scenario 1 and 2

output signal every time the simulation is repeated. Hence, as a data augmentation technique, a random white Gaussian noise was added to the sensor signals to make them look like a signal obtained from a sensor data in an experiment. An example of this method is demonstrated with a sensor signal obtained from the simulation in the Fig. 13 and from that we considered signal an SNR of 15 to be bad and an SNR of 25 to resembles a typical non-averaged signal measured from an experiment.

3.3. Data pre-processing

The data from the sensors can be transformed from a 1D representation (time domain signal) into a 2D Time-Frequency representation (TFR) and since Lamb waves are dispersive waves, signals from each crack condition that will result in distinct images which is then fed into a ConvNet. There are quite a few methods that can be used to perform TFR for Lamb waves. The reassignment method is a technique used in TFRs to sharpen and to localize the frequencies nearer to their true regions along time of the signal [81]. Therefore,

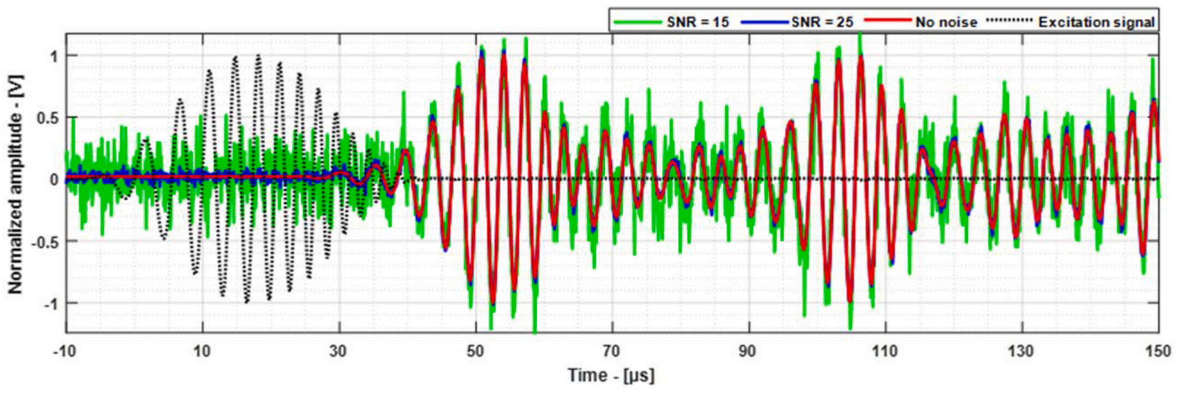

Fig. 13. Example of simulated signal without noise, with SNR = 15 (green) and SNR = 25 (blue).

we used the reassignment method implemented on Short Time Fourier Transform (STFT) [36]. An example of such transformation is shown in Fig. 14.

3.4. Entity representation

In Section 2.3 we talked about entity representation and there are two assumptions that can be made: 1). ***The entity is hierarchically represented over smaller sub-entities***, and 2). ***The entity is conserved over time***. In this sub-section, we describe how to represent the entity in both ways.

3.4.1. Hierarchical representation in multiple sensor

The analogy of hierarchical representation of human face recognition is that a face consists of eyes, mouth, noses, ears, chin, etc. The eye consists of eyebrow, pupil, iris, sclera etc. In-line with this analogy where a larger entity can be broken down into smaller sub-entities, we can randomly sample the signal over certain time window $\boldsymbol{W}$ and then pre-process the randomly sampled sequence by the method described in Section 3.3.

Examples of randomly sampled TFR from a sensor that is located at 51|28 cm in undamaged aluminum plate are depicted in Fig. 15a – e, where in these figures, $\boldsymbol{W}$ is varied between 20 and 320 μs and thus corresponds to a varying image size between 200 and 3200 pixels in length and between 7 and 253 pixels in width. The width represents the STFT coefficient of the frequency component. Applying a smaller window length would mean imply the probability that certain features occur in other classes increases, leading to higher invariant estimate. As for data augmentation, the invariant transformation would be shifting the features to the left or right to represent small sensor displacements.

3.4.2. Conserved entity over time

Translating the analogy of the different car perspective in Fig. 7 for DeepSHM, this would mean that an ideal representation of the signals would be a stack(s) of pre-processed data from all possible observation points, that is all signals are ideally captured by a grid neuron. However, this is very difficult to realize with SHM, particularly because in SHM are would typically only have small number of sampling observations (e.g., 10 sensors for a monitoring area of 1 m^2).

For ultrasonic based SHM, it might be possible in the future to combine an actuator PZT and a moving phased array (PA)-probe as a sensor that acts as a grid neuron, where the smallest discretization unit is the size of PZT actuator. Nevertheless, we can only partially reconstruct the responses by simplifying the conserved representation, e.g., for 3 sensors, the pre-processed data from Section 3.3 can be visualized by stacking them together in RGB array, depicted in Fig. 16. For >4 sensors, the data cannot be visualized without reducing information anymore – but it can be stacked in *k*-dimensional array.

Applying a larger window length means that the probability that certain features occurs in other classes decreases, thus equivalent to a lower invariant estimate. As for data augmentation, the invariant transformation would be shifting the features to the left or right to represent a small sensor displacement but also swapping the channel (e.g., red, green, and blue channel first represents sensors 1,2,3 respectively and then swapped to sensors 2,3,1 respectively). The reason for this channel equivalence is because the feature representations of each channel will be summed together before being passed through to the next layer. To simply rephrase this: what would be the perception difference in hearing a music when the left and right speaker of a stereo headset are turned around?

3.5. Training deep neural network

3.5.1. Hardware

As for hardware, a standard specification PC was used: Dell Precision T5810 running with Intel Xeon(R) E5-1620 3.5 GHz and 32 GB DDR-RAM running on Windows 7 with an additional NVidia GPU GeForce GTX1080Ti to increase the computational performance. At the time of purchase (June 2018), the graphic card had the highest performance on the end-user market. Alternatively, there are more powerful GPUs for large-scale exploitation such as the NVidia DGX-2 or the hourly-priced machine learning services such Microsoft Azure or Amazon AWS. Another alternative for training a deep neural network would be a Field Programmable Gate Array (FPGA) as it promises even faster processing – however, at the time our work was conducted in mid-2019 there was no mass-available commercial FPGA framework.

3.5.2. Software and libraries

Software-wise, we used the [TensorFlow (online)] library which is developed by Google Brain and Keras API [Keras (online)] because currently these are the richest library available on market. Both libraries are available in Python. Note that, recently, there is a

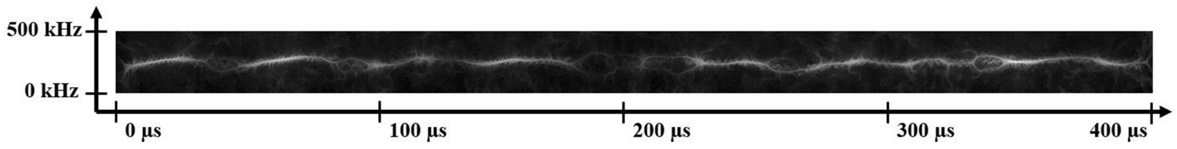

Fig. 14. Example of normalized reassigned STFT. Black pixel signifies an STFT coefficient about 0 and it contains mostly noise and no meaningful information, where the white pixel represents an STFT coefficient close to 1, which contains the waveform information.

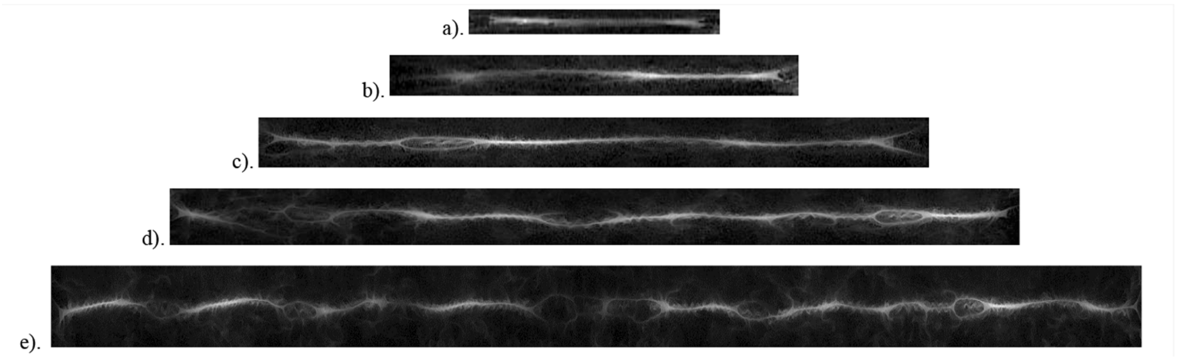

Fig. 15. Spectrogram from randomly sampled signals from sensor that is located at 51|28 cm over a convolution window length W of a). 20 μs, b). 40 μs, c). 80 μs, d). 160 μs, and e). 320 μs. 1 μs input length corresponds to 10 input samples. Sampling frequency = 10 MHz. The meaning of greyscale color scaling is analogous to Fig. 14.

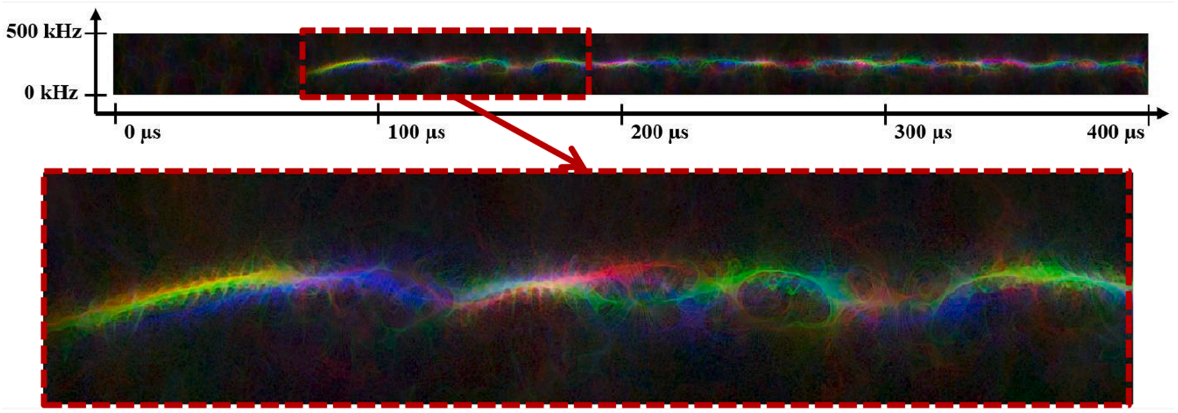

Fig. 16. Feature representation of merged input signal from 3 sensors channeled in RGB array.

3rd option which is called [PyTorch (online)] which is developed by the Facebook AI Team. It is slightly less rich than TensorFlow, but recently gaining its popularity among deep learning academic community. More recently MATLAB has also launched its own GUI-based deep learning toolbox, making it is even more practical to use, although it has very limited support from community. To enable the GPU acceleration, the API NVidia CUDA (currently we have the latest version 10.1) must be installed so that the API can interact with the graphic card.

3.5.3. Optimization

Since training a neural network is NP-hard and a brute force tactic to solve it would take non-polynomial effort, one would rather select converging iterative methods over search metaheuristics given the high dimensionality of parameters θ. When talking about converging iterative methods, the natural choice for high-dimensionality of θ would be the first-order methods, which is also known as the gradient-based method which is a single-dimensional Jacobian. Involving any second-order methods (i.e., Hessian matrix) would typically solve the problem faster than gradient-methods, however as there is no such thing as a free lunch, the Hessian-based method would also require much larger amount of computational space (i.e., RAM/memory).

Among the first-order methods, some of well-known techniques are (L)-BFGS [80], Gauß-Newton [99], (steepest) gradient descent [95], and Levenberg-Marquardt [120]. Like other libraries, currently only the gradient descent methods are already being implemented in Keras. The native optimizer is called stochastic gradient descent (SGD) with variable batch training size which also supports neural momentum and Nesterov acceleration [13] as described in Eq. (9) to escape local minima.

$$\begin{aligned}\delta_{t+1} &= \gamma \cdot \delta_t + \eta \cdot [\nabla_\theta J(\theta - \gamma \cdot \delta_t)] \\ \theta_{t+1} &= \theta_t - \delta_{t+1}\end{aligned} \tag{9}$$

where in Eq. (9), δ_{t+1} is the update vector for parameter θ at iteration $t+1$, η is the learning rate, J is the assigned cost function and γ is the momentum which is typically set to 0.9 [102]. There are more optimizer available in Keras such as RMSProp, adaptive moment estimate (Adam) and its variants (Adagrad, Adadelta, Adamax, Nesterov-Adam). The more details explanations of these techniques can be read in [42].

3.5.4. Neural network architectures and optimizers

The convolutional filter of ConvNet is designed to capture spectrotemporal features of the TFR and after that, the learned filters are fed into a fully connected layer (MLP). Without capturing the spectrotemporal features, the information coherence within a certain time–frequency window will be lost. In order to demonstrate that spectrotemporal features are learned during the training, we must compare the training results from ConvNet + MLP and pure MLP architectures without convolutional kernel for each of the 3 different classification scenarios.

Like neural network parameters optimization, finding a suitable network architecture has an NP-hard property since the architecture combinations are infinitely extendable and unfortunately, there is no strict rule to design the number of layers. Thus, it would be logical to start the choice of architecture with the less complex series, as per *lex parsimonae*, and therefore, this time decided to use the architectures that we have been using in the previous work [32], as described in Table 3.

We are aware that there are more sophisticated CNN architectures such as VGG-16 [100], inception layers [105], or ResNet [52] that can be employed in the future work, but for the sake of a simple demonstration in this paper we limit the network to a maximum of 8 hidden layers. The VC dimension and the Rademacher complexity from Def. 3 can be later quantified to determine the necessary adjustment of the complexity of the network. The example code can be downloaded in our repository [Github (online)].

For the training purpose, the data should be normalized between 0 and 1 so that the many useful training parameters proposed by the computer science community can be directly used. For deep neural network, it is recommended to always activate the dropout regularization and according to Srivastava et al. [102], 0.5 is the best rate found, which means 50% of the neurons at that particular layer are deactivated. The default parameter for the optimizers are presented in Table 4.

4. Results and discussion

The training results for the given parameters in Section 3 are presented in this section and organized as following: in Section 4.1, the results and discussion for modelling DeepSHM as a hierarchical entity as discussed in Section 3.4.1 are presented, where the discussion includes the training results trained under both the SGD and Adam optimizers for different sensing locations in each given damage case scenario for variable window length. In Section 4.2, the results for modelling DeepSHM as a conserved entity, as discussed in Section 3.4.2, that gets the input from fused sensor data are presented. In this case, we will describe the training results for different sensing locations as well, however as a consequence of the assumption of the conserved entity over time, only the full window length can be used.

While our computer was relatively up-to date for training the network, it rarely meant that we always had enough space to store all the models created, thus we limit our training samples only for 100 samples per class to see the network behavior. During the trial-and-error phase, we actually tried with 1000 samples first, but since it took too long for preprocessing (about 2 days per dataset). While the number of samples we tried is nothing compared to what is typically processed within the computer vision community, the features that our network has to learn are much higher. Unlike in the machine learning community, the models they are training are using a CIFAR-10 or MNIST dataset that has an input size of 32×32 pixel, while in our case, the input size varied between 251×7 pixels up to 4101×247 pixels, depending on the dataset generated.

4.1. On modelling DeepSHM as multiple actors with independent decision making

4.1.1. Influence of network architecture and sensor locations

In order to demonstrate the superiority of CNN in comparison to multilayer perceptron (MLP) to handle spectrotemporal data, we first present the training result of MLP handling the training set. We trained the MLP for all damage scenarios given in Table 2 with various reserved memory from 800 to 3200 input samples. In order to save space, we only show the training results for the training and validation datasets from the presumably best sensor location, which the sensor at location 2, depicted in Fig. 17a – f. For simplification, we limit the training process up to 50 epochs only. As a reminder, the sensor locations are given in Fig. 12.

We compared both of the optimizer SGD and Adam in this case. Adam uses the adaptive learning rate from the moments of the gradients estimates thus it is generally faster in finding the gradient path in the beginning epochs and keeps an exponentially decay of prior gradients average. On the other side, SGD is slightly slower in finding an optimal gradient in the beginning epochs as it only computes the gradient without the estimate approximation. For a general training purpose, Adam is therefore recommended as it can find the gradient path faster.

From Fig. 17a, b, for a window length of 80 μs, it can be seen that the MLP model quickly overfits regardless of the optimizer used. While the training accuracy reaches almost 1.0 (or 100%) in Fig. 17a, b, it can be seen that the validation accuracy stays below 0.6 (or 60%). Note that we will discuss the effect of window length in Section 4.1.2 in detail. This behavior is confirmed again in Fig. 17c, d, where the window length was extended to 160 μs and in both cases, the difference between training and validation accuracy is>35%.

Table 3
Neural network architectures used. **C**(*i*): *i*-filter convolutional kernel; **MP**: MaxPooling layer; **DO**(*j*): dropout regularization with rate of *j*; **D**(*k*): dense (fully connected) layer with *k*-neurons, **CL**: Classification layer.

MLP	**D**(128)-**DO**(0.5)-**D**(16)-**CL**
CNN	**C**(8)-**MP**-**DO**(0.5)-**C**(16)-**MP**-**DO**(0.5)-**C**(32)-**MP**-**DO**(0.5)-**D**(16)-**DO**(0.5)- **CL**

Table 4
Default optimizers parameters in Tensorflow library. η: Learning rate, γ: neural momentum, η_{decay}: learning rate decay, NESTEROV: Nesterov momentum parameter, β_1, β_2: exponential decay rate for 1st and 2nd moment estimates.

SGD	$\eta = 0.01$, $\gamma = 0.0$, $\eta_{decay} = 0.0$, NESTEROV = FALSE
Adam	$\eta = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\varepsilon = 10^{-8}$, $\eta_{decay} = 0.0$, AMSGrad = FALSE

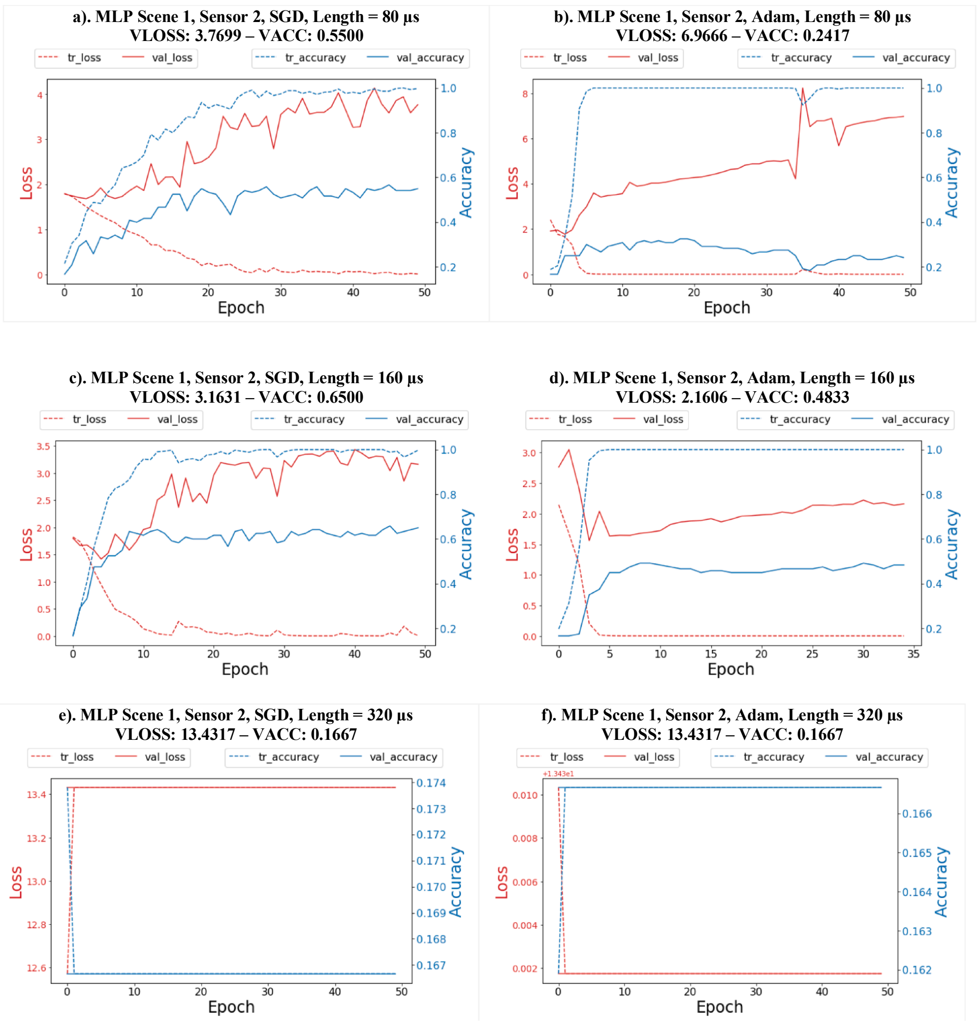

Fig. 17. a–f: Training results of MLP for signal from sensor 2 with various window length trained under Adam and SGD optimizer up to 50 epochs. 1 µs input length corresponds to 10 input samples. VACC = Final validation accuracy, VLOSS = Final Validation Loss. Note that the scaling is not uniform due to the default library settings.

When the window length was extended once more to 320 µs (see Fig. 17e, f), only a constant flat line over the whole training epoch was reached, meaning that the memory of our graphic card was overloaded. This problem can only be solved by adding more physical memory which would mean we have to purchase a newer GPU – however this only solves the problem temporarily until it meets a larger input matrix dimension. With such an unacceptable training behavior, we choose not to further proceed with exploiting the MLP network and assume a simple MLP would very highly likely to fail to capture a spectrotemporal feature such as depicted in Fig. 15.

For the CNN architecture, we kept our CNN very simple as it only contains 3 hidden convolutional layers that are attached to the same MLP we used previously. Each convolutional layer was followed by pooling and dropout layers as described in Section 2.4. In general, the network may learn the pattern representation after long training, however in reality, we always have a time–cost limit and a training budget threshold must be determined depending on application and budget resources. To demonstrate a simplified training budget, we limit the training epochs to 50. The training results from CNN trained both under Adam and SGD optimizer for damage scenario 1 (Table 1) are depicted in Fig. 18a–h.

In this case the CNN is trained to distinguish 6 different damage classes including one from the baseline: The highest validation accuracy reached was 0.9917 (or 99.17%) from data captured by sensors 1, 2, and 4 when optimized by SGD, while the lowest validation accuracy reached was 51.67% from data captured by sensor 3. However, when using Adam, the highest validation accuracy reached was 98.50% from data captured by sensor 2 (which is presumably the best sensor position) followed by 96.67%, 94.17%, and 92.50% for sensor 1, 4, and 3. Because the Adam optimizer yields a better result than the SGD, from this point on we only show the result of the trained network under Adam optimizer.

For the classification of damage scenario 2, we also simulated the PZT response from sensor location 5 – 8, because in this scenario, the CNN is trained to classify less distinguishable signals from each class as per Table 2 and we believe that more sensor responses

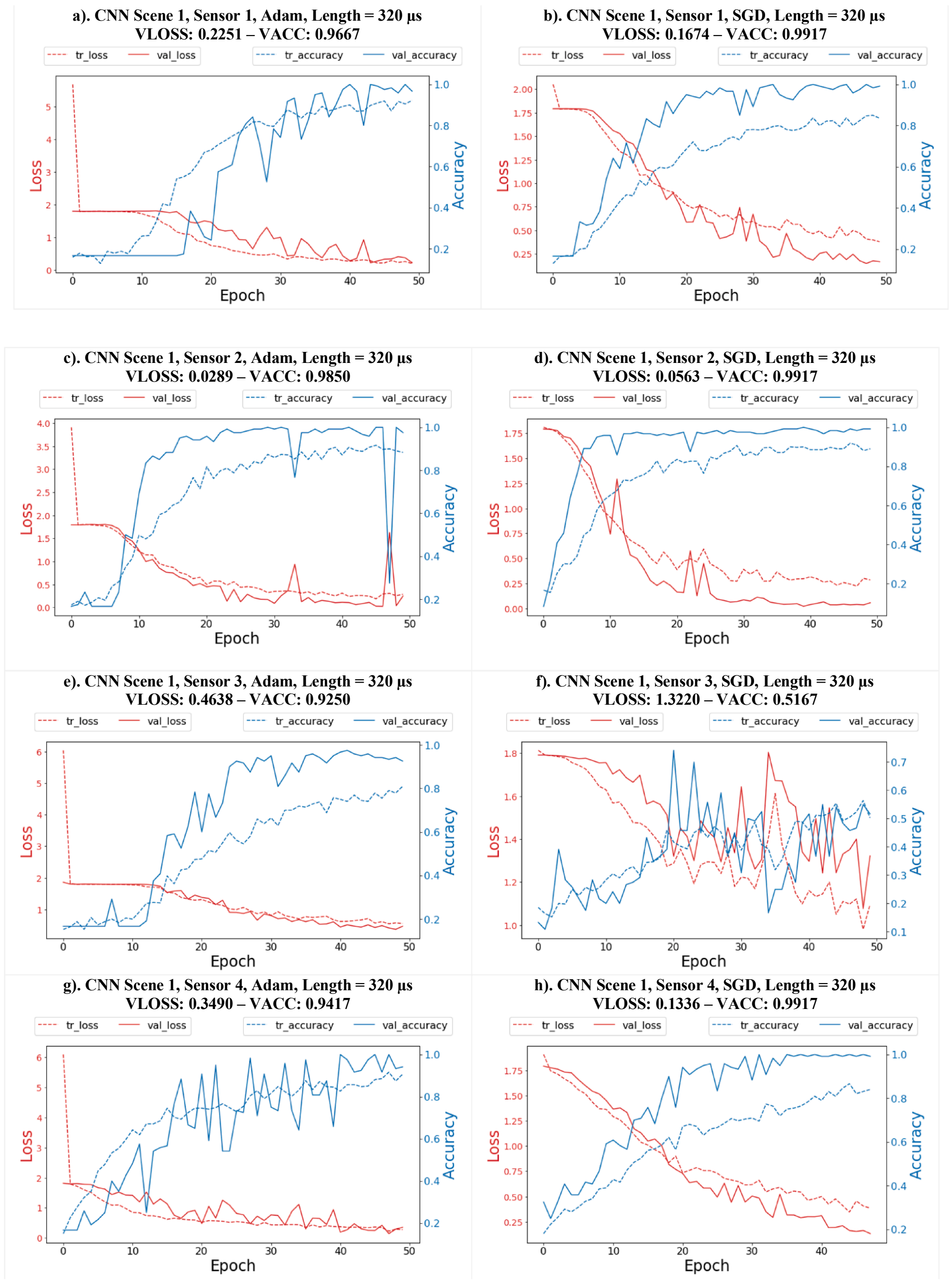

Fig. 18. a–h: Training results of CNN for signal from sensor 1–4 in scene 1 with window length of 320 µs trained under Adam and SGD optimizer up to 50 epochs. Sensor locations are given in Fig. 12. 1 µs input length corresponds to 10 input samples. VACC = Final validation accuracy, VLOSS = Final Validation Loss. Note that the scaling is not uniform due to the default library settings.

would give an advantage because there is more data available. Nevertheless, for brevity we only show the training results using Adam optimizer from sensor location 1 – 4 in Fig. 19a–d, which can be directly compared to Fig. 18 a, c, e, and g, respectively. The results depicted in Fig. 19a–d met our expectation because validation accuracy is on average lower than in scenario 1, ranging from 32.22% in sensor 3 (where it also overfits – s. Fig. 19c and sensor 3 is one of the worst PZT sensing location) to 93.33% in sensor 1, depicted in Fig. 19a which is one of the best PZT sensing locations. This is to be expected, because if we look closely into scenario 2, we can easily assume that the TFR signal from baseline plate and the damaged plate with a smaller crack length (e.g., only 10% a_{crit}) and smaller deviated angle (i.e., 15°) are likely to be similar with each other, especially because the recording length was short (320 μs). Thus, the neural network would not be able to distinguish these TFR. This explanation is clearly supported by corollary 4 in Section 2.3.

Nevertheless, unlike classical signal processing, the TFR from better sensing locations such as sensor position 1, 2, and 4 were able to be trained to reach better validation accuracy (Fig. 19a, b, and d). As a side note, if we double or even quadruple the window length from 320 μs to 640 μs or 1280 μs, we believe the neural network will be very likely to be able to learn the distinguishing pattern between the TFR and thus increase the validation accuracy. This is because in the later time-series response, the wave scatter from the crack would eventually travel through sensor location 3, too. While this might be an advantage of deep learning over classical signal processing, we would like to remind the readers that doubling or quadrupling the window length would require double or quadruple amount of data storage. If one TFR image takes up 2 MB, doubling this would mean at least 4 MB. For small-scale research, this is surely not a problem but scaling this up on industrial level would mean double or quadruple required investment for data storage. So, to be fair we would like to mention that in classical signal processing, multiplying the data storage might not be necessary.

As previously mentioned in Table 2, damage scenario 3 is just the combination of scenario 1 (very distinctive classification ranging from 0% to 100% length of critical crack in each 20% step) and scenario 2 (less distinguishable signal between each class with varying angled crack), the training results are slightly better than scenario 2, but worse than scene 1. Note that in this case, only the data from the 4 sensors in scenario 2 which are located in the same location as in scenario 1 can be added to the training set. Also, the importance of sensor positioning is also now clearly highlighted. The training results are depicted in Fig. 20a–d.

We already mentioned in Section 3.1, that sensors 1 and 2 are at one of the best sensing locations and thus, the validation training accuracy reached was>0.9 or 90% (Fig. 20a, b), while in sensor 3, which is located very far away from the crack location, only reached a validation accuracy of 43.67% (Fig. 20c). Again, this is far from surprising, because we expect that at the sensing location occupied by the sensor 3, the accumulated energy from the scattered wave from the crack to be far less than that captured by sensors 1 and 2. The data from sensor 4, which is located close to the crack location but not directly placed along the wave scatter propagation path, reaches a final validation accuracy of 69.67%, as depicted in Fig. 20d. Analogous to the explanation for the training results in scenario 2, here we also believe the neural network will be very likely to be able to learn the distinguishing pattern between the TFR if the window length is at least doubled.

From all these experiments, we concluded that one cannot just rely solely on deep or any other advanced machine learning algorithm to tackle physical limitation and thus there must be a physical intervention (such as adding more PZT sensors until

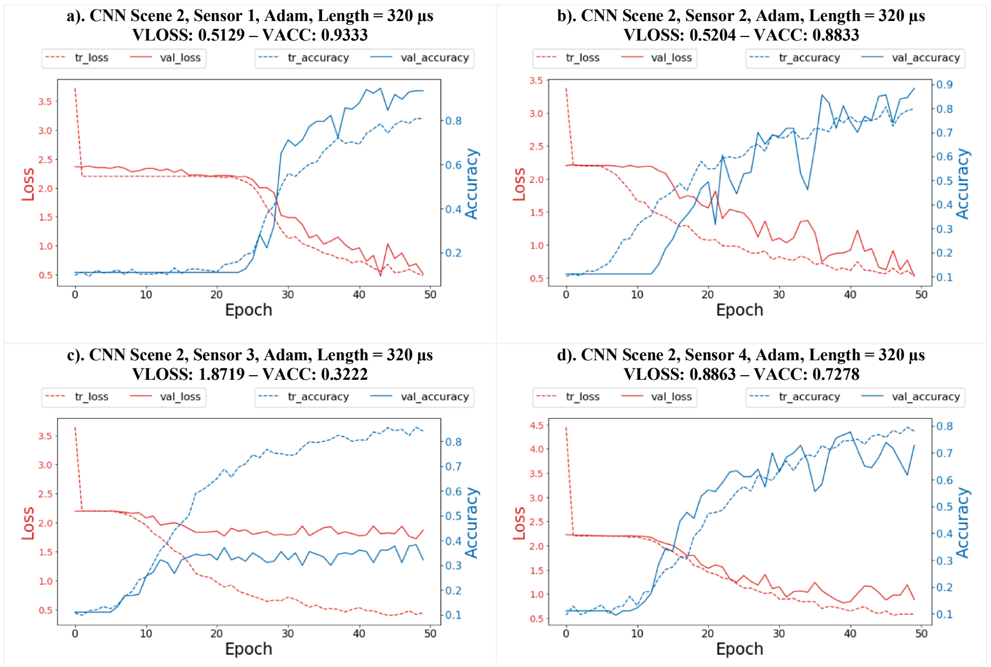

Fig. 19. a–d: Training results of CNN for signal from sensor 1–4 in scene 2 with window length of 320 μs trained under Adam optimizer up to 50 epochs. Sensor locations are given in Fig. 12. 1 μs input length corresponds to 10 input samples. VACC = Final validation accuracy, VLOSS = Final Validation Loss. Note that the scaling is not uniform due to the default library settings.

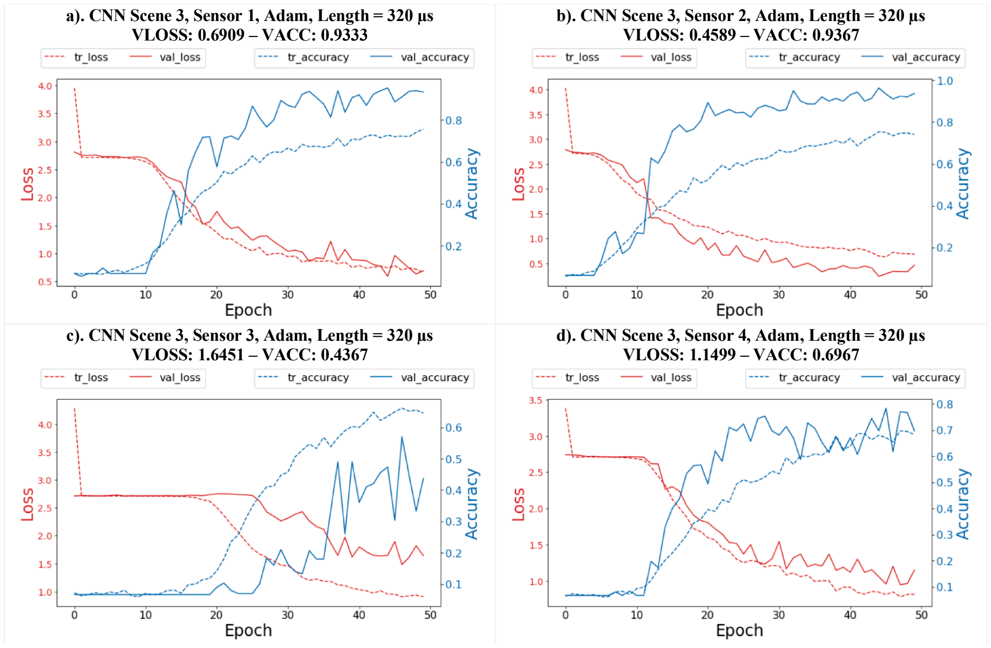

Fig. 20. a–d: Training results of CNN for signal from sensor 1 – 4 in scene 3 with window length of 320 µs trained under Adam optimizer up to 50 epochs. 1 µs input length corresponds to 10 input samples. VACC = Final validation accuracy, VLOSS = Final Validation Loss. Note that the scaling is not uniform due to the default library settings.

detectability convergence) to improve the result. The good news is, however, our previous work regarding sensor placement strategy for hotspot sensor placement using blob detection Ewald et al. [31] can be used because a sensor placement strategy is still needed even when applying a sophisticated machine learning technique such as deep learning. The bad news on the other side is that a hotspot sensor placement strategy is still needed – thus the design effort for an SHM sensor placement strategy is needed and there would be a cost for this both in time and money.

4.1.2. *Effect of the length of convolution window*

In this scenario, we describe the influence of the reserved convolutional window length of the signal on the required time per training step and on the final validation accuracy (VACC), where in this case, we only use the data from the best sensing locations, particularly sensor 2. The convolutional window length was varied between 10 µs to 320 µs sampled from the full signal as previously explained in Sections 3.3 and 3.4. All the scenarios are trained with Adam optimizer and to save time, we limit the training only up to 50 epochs.

The results for all scenes are summarized in Fig. 21 and Table 5, where it can be seen that at least a window length of 200 µs is necessary to surpass a 90% training accuracy threshold, while for the more difficult scenarios 2 and 3, a windows length of 320 µs is needed. The computation time per time step varies between around 16 ms for TFR with up to 60 µs window length and increases quadratically as the window length is increased as depicted in Fig. 21b.

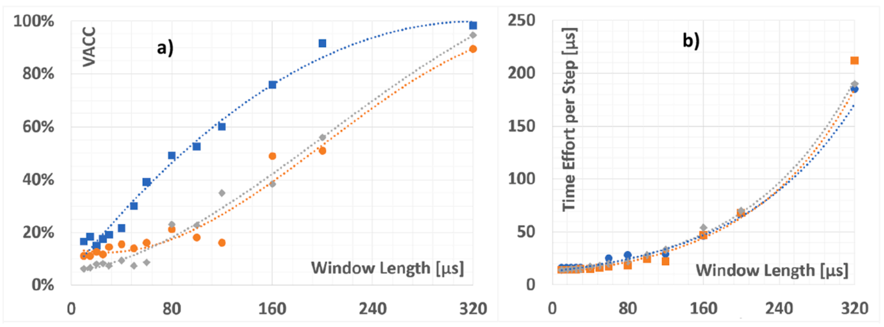

Fig. 21. Left: The relation between final validation accuracy (VACC) and reserved window length in [µs]; right: The relation between time effort needed per training step and reserved window length in [µs].

Table 5
Effect of Window Length on Training Time per Epoch and Final VACC* at 50 Epochs. *VACC = Validation Accuracy.

Window Length [μs]	Scene 1: 6 classes		Scene 2: 9 classes		Scene 3: 15 classes	
	Time/Step [ms]	Final VACC	Time/Step [ms]	Final VACC	Time/Step [ms]	Final VACC
10	16	0.1667	14	0.1111	14	0.0633
15	16	0.1833	14	0.1111	14	0.0667
20	16	0.1500	14	0.1278	15	0.0800
25	16	0.1750	14	0.1167	15	0.0833
30	16	0.1917	15	0.1444	15	0.0733
40	16	0.2167	15	0.1556	17	0.0933
50	16	0.3000	16	0.1389	18	0.0733
60	25	0.3917	17	0.1611	20	0.0867
80	28	0.4917	18	0.2111	22	0.2300
100	24	0.5250	24	0.1811	28	0.2267
120	29	0.6000	22	0.1611	33	0.3500
160	46	0.7593	47	0.4889	54	0.3833
200	67	0.9167	68	0.5086	70	0.5600
320	185	0.9833	212	0.9044	190	0.9467

4.2. *On modelling DeepSHM as conserved entity over time*

As has been discussed in Section 3.4.2, we are now interested in seeing the training behavior when the perception is modelled as a single entity conserved over time. The consequence for this assumption is that: the training set consists of a k-dimensional data cube composed of multi-layer full-length TFRs, e.g., see Fig. 16 for an example of this representation. Ideally, the data cube would comprise all possible layers which represents all possible responses from each sensor. However, for brevity and simplification, we only represent the data cube in 3-layers (meaning that only 3 sensor responses are used) so that it can easily be converted into an image and trained with the libraries we used (Keras and Tensorflow). While Tensorflow can read bitmap (BMP) files, we do not recommend converting the STFT coefficient matrix into BMP since it took too much space. Thus, a portable network graphic (PNG) format, which is lossless compression, is chosen in order to compromise the information richness contained in BMP and the sparsity of JPEG format.

Most of the ready-to-use deep learning libraries only support a 3-channel color image because most of the deep learning community are focused with recognizing 3-color channel RGB images. By customizing the library, it is possible in the future to train a data cube with>3-dimensions, which typically can be saved as data frame in Python. For this, there are already research works related to classification of hyperspectral images by using CNN [70,87]. Unfortunately, those works are in general very niche and many of the codes are not made publicly accessible.

As in our case, we use a combination of several sensor responses to create the data. As there are only 4 sensors in scenario 1, we were able to show all possible combination of the training behavior of the sensor response (1;2;3, 1;3;4, 1;2;4, and 2;3;4) and these are depicted in Fig. 22a–d. However, as there are too many possible combinations in scenario 2, we only show the result of several combinations as given in Fig. 22e–h. The rest of the results are available in the dataset, or alternatively the readers are can also download the code and the dataset as these have been made available online. Finally, as scenario 3 is the combination of the result of scenario 1 and 2, only 4 sensor responses can be combined, and the corresponding training results are given in Fig. 22i–l.

From all sub-figures in Fig. 22, it can be clearly concluded that modelling the SHM perception as a single conserved entity has a quickly converging training rather than modelling the perception as hierarchically ordered but separate entities (cf. to Figs. 19–21). For the first case, it is obvious that the sensing locations now become the less important issue. There are clearly several different behaviors before the training accuracy reaches its 1.0 plateau as depicted in Fig. 22a–l, from all these figures, we postulate that the CNN was able to capture the correlated 3D-spectrotemporal features within certain region of interest in the data cube.

We assumed the existence of the correlated 3D-spectrotemporal features to be evidence for the theory of invariant latent and its estimate proposed in Lemma 5 and 6. It is very difficult, or if not impossible to deductively proof the applicability of this lemma as deep learning model parameters can only propose correlation but rarely have explanatory power – but this is widely *de facto* accepted in mathematics and computer science community. The possible physical explanation to our belief has already been stated previously in Section 2.3 regarding music recognition in human brain.

While a Lamb wave signal is not a music or a song, it is not merely random noise either – there is an interconnection between each particle wave movement during guided wave propagation because an acoustic wave in a continuum can be regarded as harmonic motion and thus do not fall into a sudden singularity, so we assume there must be a spectrotemporal features that are locally connected between each layer in Fig. 16. We are aware that this assumption is of course not valid for randomly occurring singularities within a continuum – but we never expect sudden singularities within acoustic wave signals either (except when the sensor is broken, but that is a separate topic).

It is very difficult to understand what is happening inside the deep neural network, but we postulate that these local spectrotemporal features, while being very deeply embedded in the CNN model, are very distinct to each other for every different damage class. This was due to our previous assumption that every different damage classes would produce a varying, non-homogenous TFR at each different sensing location and therefore making the neural network easily captures these (cf. Rademacher complexity in Def. 3) and learns from them after several training epochs. Each TFR is then a special distinctive signature of each probability classes as supported by the Lemma 5.

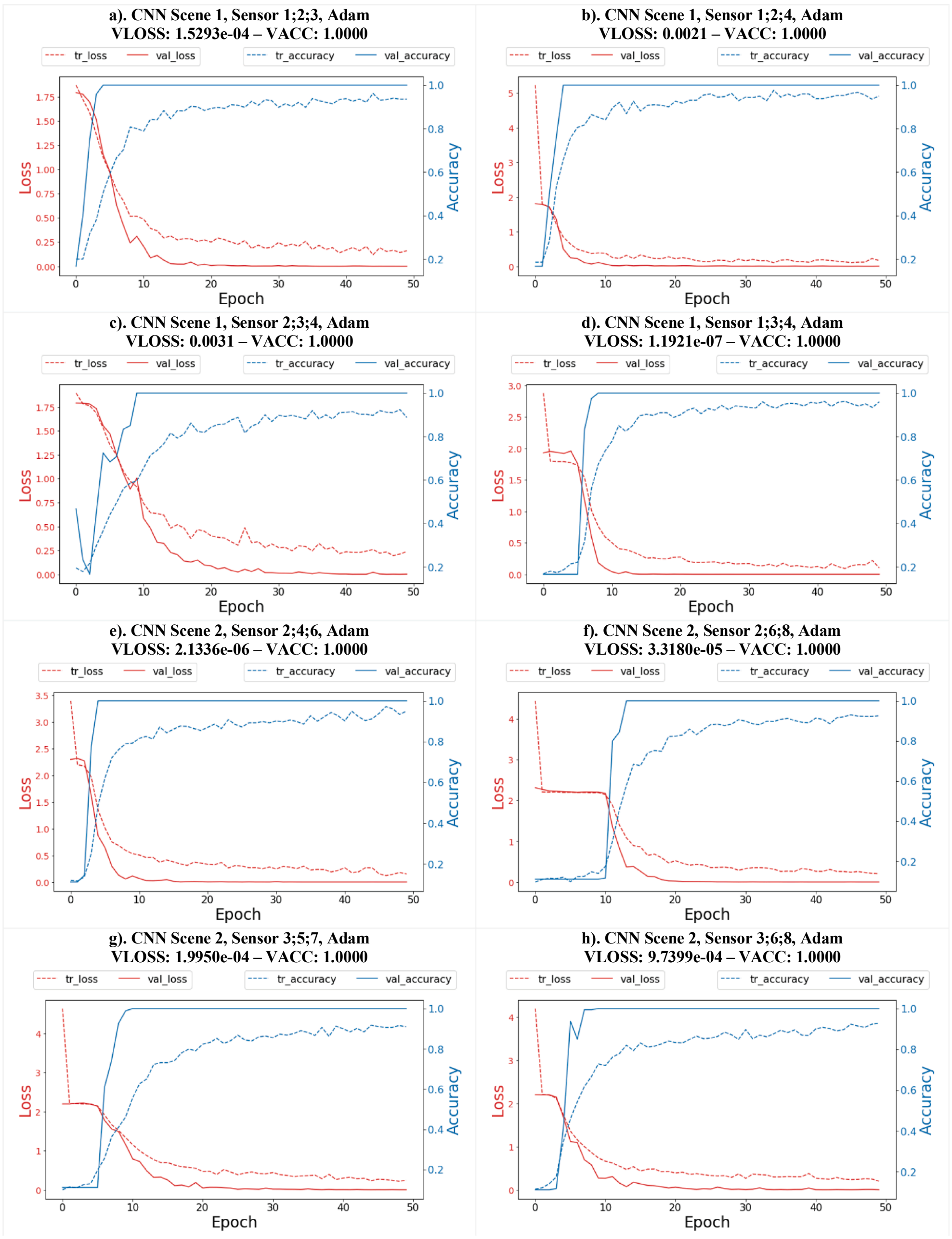

Fig. 22. a–l: Training results of CNN for combined signal from various sensor combination in all damage scenarios trained under Adam optimizer up to 50 epochs. VACC = Final validation accuracy, VLOSS = Final Validation Loss. Note that the scaling is not uniform due to the default library settings.

In general, it is common to use a single-frequency centered Hann window as an excitation wave because it would be easier to analyze the sensor response based on single-frequency signal. As this has always been most of the case, our training results gives evidence that it might be an advantage to try out the less exploited and more “adventurous” chirplet-like excitation signal for Lamb wave SHM. Our training results suggest that these distinctive spectrotemporal features within the representation that are more heavily augmented due to the quasi-chirplet excitation in comparison to when it is excited by simple Hann window signal, which in general produces more complex material response and thus more complex information embedded in the signal representations that helps the CNN to learn from this distinctive spectrotemporal information – as per Lemma 5. As this is our speculation, we think that there would be a future opportunity to quantitatively investigate this matter in a more rigorous research.

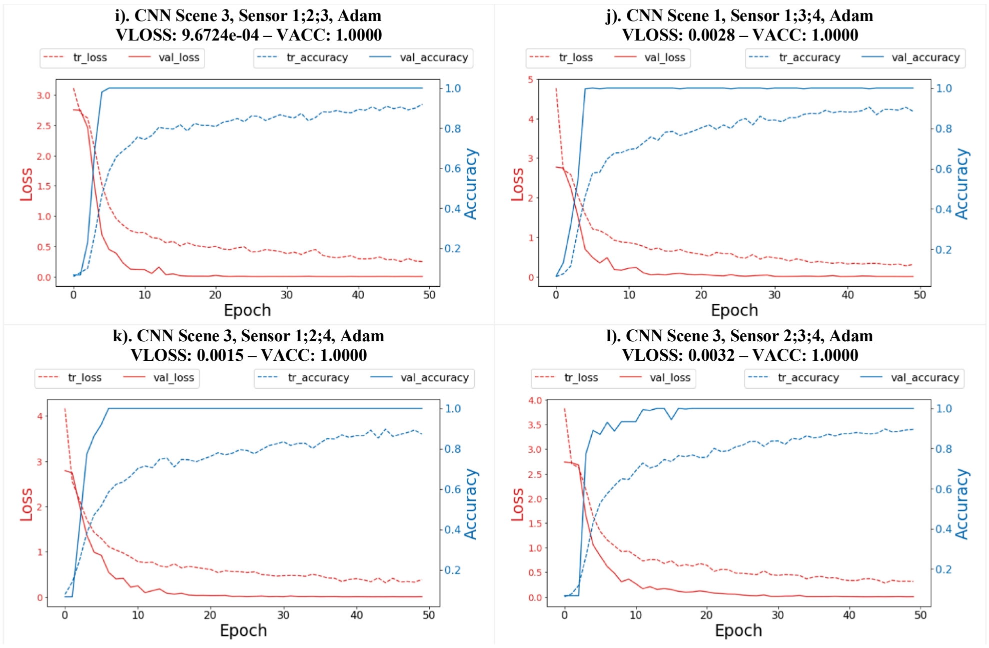

Fig. 22. (*continued*).

4.3. Concept validation

4.3.1. Result comparison with random noise training

To deliver further evidence regarding the invariant latent, we set up a mock-up test to compare the training behavior between the simulated signal with added white Gaussian noise with varying SNR between 5 and 15 from previous and pure Gaussian white noise. The reason to do random noise training comes from the classical A/B testing in statistics. Without trial like this, we will never know whether the network actually learns the pattern, or it merely remembers the noise behind the pattern. An example of TFR of Gaussian white noise is depicted in Fig. 23. The noise was trained under exactly the same training parameters as before and split into 6 different folders like in scenario 1.

We repeated the white noise training several times and give two sample results in Fig. 24a, b which depicts the training behavior for a white noise signal with window length of 320 μs. It can be seen from Fig. 24a, b, the networks failed to reach a validation accuracy of >60% even after 50 training epochs, while the training accuracy reached a value much higher than the validation accuracy.

The distance between the training and validation accuracy is relatively far (between 20% and 40%), which is a clear sign of heavy network overfitting, but this still is not the most important fact. It is more important to know that they failed to deliver a consistent result, i.e., the network did not learn anything from the TFR pattern because there is no locally connected spectrotemporal features in the TFR of a noise – thus indicating that the invariant latent within the noise group does not hold. In fact, the network did learn the noise. More figures are available upon demand or the reader are more than welcome to download the source code [Ewald et al. [32], Github] and data to self-experiment with it.

4.3.2. Model testing

4.3.2.1. Hierarchical representation in multiple sensors. To test the validity of the models as to whether they can classify the signal or not, a confusion matrix must be calculated for each model. As a sample result, we show the test results from scene 1 and 2 for sensor 2 for convolutional window lengths of 80 μs, 160 μs, and 320 μs which are depicted in Fig. 25a–f, respectively. Further, the labels “0″ to “5” in Fig. 25a–c correspond to the damage condition of scene 1 described in Table 4 with “0” as the baseline and “5” as the critical

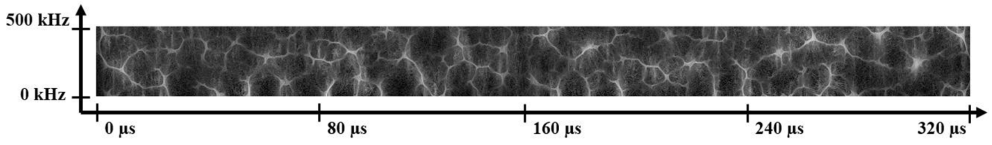

Fig. 23. TFR of a white Gaussian noise with a length of 320 μs.

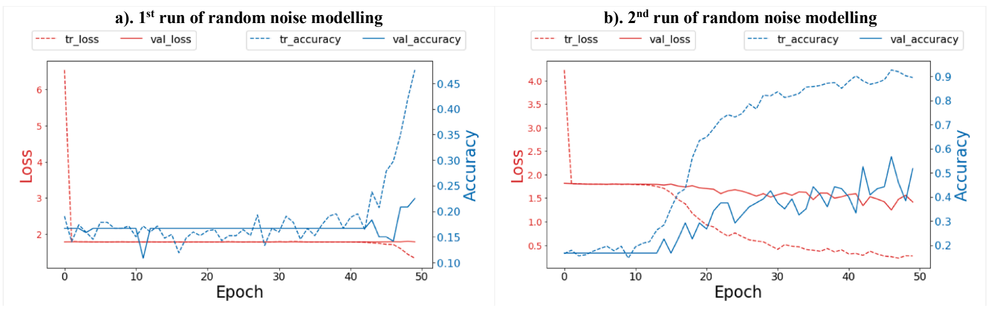

Fig. 24. a, b: Sample training results of random noise modelling. Note that the scaling is not uniform due to the default library settings.

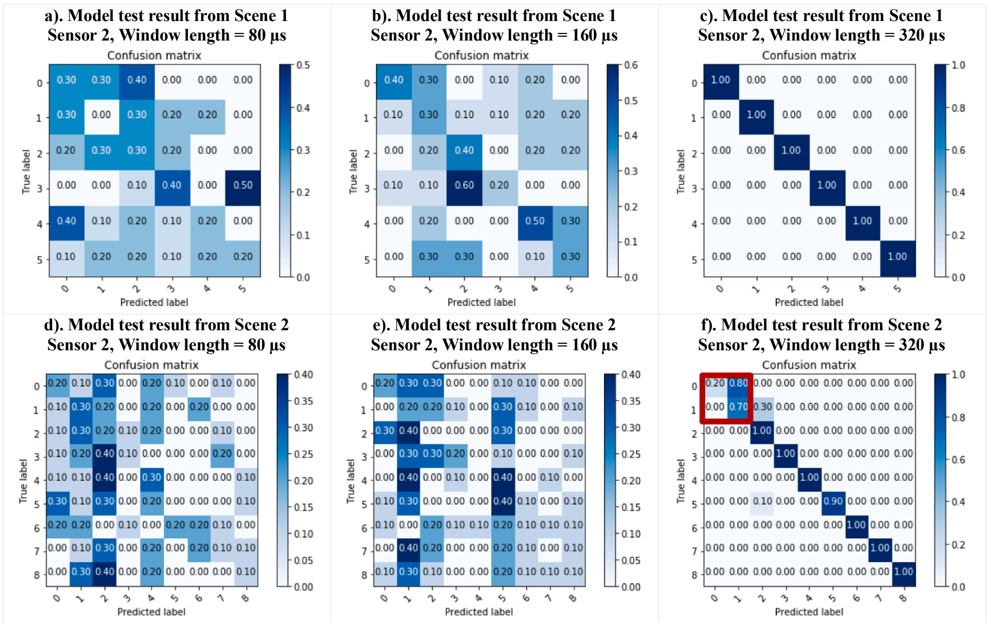

Fig. 25. a–f: Model test result from scene 1 and 2 for sensor 2 with varying window length between 80 and 320 µs.

crack length a_{crit}. The same logic applies for Fig. 25d–f from scene 2 described in Table 4, where in this case, “0” is the 0°-oriented crack with length of 10% a_{crit} and “8” is the 45°-oriented crack with length of 30% a_{crit}.

The labelling itself is currently not important as this can be changed in the code. A diagonal with an average of 1.00 such as Fig. 25c corresponds to 100% POD. In Fig. 25f (marked in red rectangle), it is obvious that the CNN cannot correctly classify the smaller cracks, which in this case “0″ is an 0°-oriented crack of 10% a_{crit}, whereas “1” is a 15°-oriented crack of 10% a_{crit}. This is to be expected because for two small cracks with slightly angled orientation, as previously explained in Section 4.1, we assume the signal from both cracks is less likely to be very different. For a detailed proof, the relative entropy and the cosine similarities between two time-series can be calculated, however this is not the focus of our paper. Nevertheless, we provide our data in the dataset.

4.3.2.2. Representation as conserved entity over time. When modelling the feature representation in a single conserved entity, the network does not seem to have a problem at all classifying the test data in accurate way with 100% POD as depicted in Fig. 26a–c. The meaning of the labels “0” to “8” is analogous to our explanation in Section 4.3.2.1. While more test results are available, for brevity we only show three example results in Fig. 26a–c that depict result from scene 2 for conserved entity representation that fused responses from sensor 1;2;3, 1;5;7, and 2;4;6, respectively. All other tests yield into a 100% POD. Also, here we hypothesize that it learned the locally spectrotemporal features within the entity in a more subtle and concise way as there is more information embedded in this ‘data cube’.

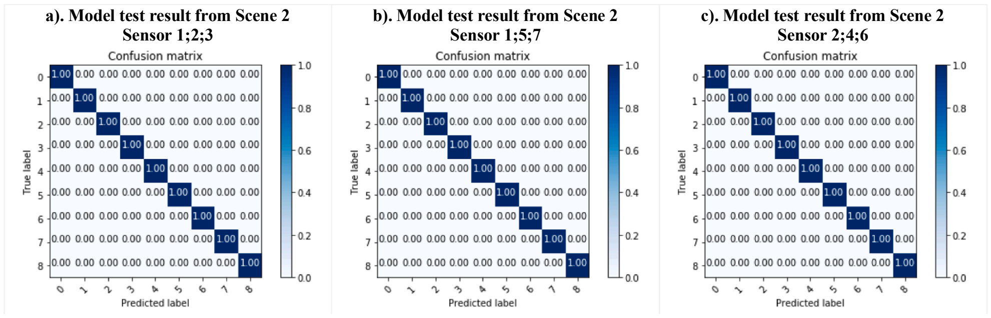

Fig. 26. a–c: Model test results from scene 2 for conserved entity representation that fused different sensor responses.

5. Concluding remarks

5.1. *Summary*

This paper is an extension to our previous work [32], in which we conceptualize the generalized idea about diagnostics and formalize our DeepSHM framework. In Section 2, we include the abstract thought about SHM perception from a neuroscientific perspective, while the methodology used such as the numerical model, data pre-processing, and neural network training are described in Sections 3.1–3.4. The results and discussion are given by Section 4.

5.2. *Conclusion and recommendation*

Before making any recommendation, recall the hypothesis stated in Section 2:

> … that some of the algorithms could not make a distinction between the signals that come from slightly similar a distribution. We hypothesize that this problem might be overcome by:

1. Applying broadband frequency excitation since broadband excitation frequency will normally invoke more variable wavenumber, and broader wavelengths.
2. Varying sensing location to obtain more potential information

Based on our results in general, we could state that the CNN captures more information due to a varying sensing location and varying wavelength thanks to the chirplet-like excitation, nevertheless this approach encounters its limit as well as demonstrated in Fig. 25f. It is possible to enlarge the excitation band frequency to include more information, however this would have two drawbacks: 1). a more expensive wideband PZT must be used, and 2). Assigning a human data analyst to analyze such complex signal with traditional signal processing would require more time and thus would be a disadvantage if we would like to understand and relate meaningful signal features to the physical domain of the structure. Our recommendation is to use the quasi chirp-alike excitation is only when it is coupled with advanced machine learning, which in any case is already a black-box. Thus, we would like to re-emphasize the principle of ***Garbage-in*** – ***Garbage-out*** as in the middle of the black box it is hardly to extract any meaningful causation. Employing a machine learning algorithm such as CNN requires a quality control on both input and output. Further, we stated following questions at the beginning:

1. How much do the varying sensing locations and the different sensing representations of time–frequency Lamb wave signal influence the deep learning training behavior?
2. Given “a posteriori knowledge” from (1), what consequence can be drawn for the engineering application in SHM and why should this approach work?

From the results, we are firm there is not a single ultimate answer, but we can state that the highly optimally located set of sensors give the most desired training behavior due to better response capture. On the other side, when the information is fused first from different sensing locations, that is in the case of conserved entity representation, it can clearly be seen that the sensing position does not matter anymore as has already been explained in Section 4.3.2.2.

However, as there is no free lunch, we must state the training cost in terms of time also increases as the network must learn more parameters than the first case, thus at this moment limiting the amount of model parameters by decreasing the neural network size is the only feasible way to make deep learning for structural diagnostic is scalable for industrial production, with the caveat that limiting the amount of model parameters would risk that the training would be longer and in worst case where the size of data is limited, this would lead into less generalizable model. In future work, we are going to investigate the scalability level of Deep SHM for given data size, model parameters, and restriction on physical memory (in this case RAM from the GPU).

Also, another consideration that we should discuss is the actuator position. If the actuator position is changed, the signal representation is also changing. While it is possible to determine the damage even if the actuator position is changed, but the model has to be changed as well. This is the limitation of deep learning where it learns the statistical distribution from given samples, and this principle applies to all type of supervised networks such as MLP and its derivates, CNN and its derivatives (LeNet, VGG, ResNet, etc.), recurrent and its derivates (LSTM, GRU, RNN), etc. To ensure that the damages types from different actuator location can be detected, there are two possibilities:

1. A new model based on that actuator-sensor pair must be created, or
2. The training data from all possible actuator-sensor pair must be included during the training

It is possible to train data coming with 2 sensors only and we expect that the performance lies between single sensor response (Section 4.1 in Figs. 18–20) and 3 sensor response (Section 4.2 in Fig. 22). As mentioned, we represent the entity via two different perceptions:

1. The behavior of Lamb wave propagation is sampled in single sensor and different window lengths are applied to generate multiple perception, resulting into different chopped time-series which were converted via reassigned STFT to generate greyscale array (Fig. 15)
2. The behavior of Lamb wave propagation is sampled in n-sensor locations (e.g., $n = 3$ in our case) with a full window length (full length is adjustable, but we have 400 μs). These time-series were again converted via STFT and joined together to form “polar-light-alike” image in Fig. 16. It is possible to join more sensors data ($n > 3$), but of course these cannot be depicted as RGB images anymore.

Conclusively, the second perception modelling can be regarded as the extension of the first modelling and as academia, we might be tricked into thinking “higher performance = better” so we will be tempted to use more and more sensors data ($n > 20$) to improve performance, until we meet the big O in the time–space complexity. In far future however, all these computational limitation problems might be tackled with quantum computing [10,59] to accelerate the computation time. For instance, it has been tried in a smaller scale hobby project [63] to combine the quantum library PennyLane and deep learning library PyTorch.

A further limitation in this current study is that we only employ damage classification techniques. In order to localize the damage, the label in the datasets can be expanded with the location of the corresponding damages such as proposed by [58] and [124] and retraining the models. In this case, the model will be assigned to classification and regression tasks at the same time. Computationally, this is not a problem per se. However, the availability of the data in such case is very scarce so that our proposed DeepSHM may only work well in medium or long-term run until a big enough database is available. For short-term period however, some augmentation techniques and generative modelling should probably be incorporated to bypass the data scarcity. For conservative industry like aerospace, this is actually a perfect pace because the industry and its people are moving slowly – this will provide enough time to the major OEMs to place strategy on their investment and make right prioritization.

Finally, we would like to emphasize that our paper is not about proposing novel methodology, but rather to give theoretical justification why a supervised deep learning can work for classification of Lamb wave data to ensure its reproducibility. As a final conclusion, by providing our results, we hope to have contribute a piece further how to employ advanced machine learning technique, in particular CNN for designing appropriate network architecture for application in diagnostic SHM.

6. Funding and acknowledgement

This project was funded by Topconsortium voor Kennis en Innovatie (TKI) from of the Dutch Enterprise Agency (Rijksdienst voor Ondernemend RVO). The authors would like to thank Mr. Xavier Goby from TU Delft for helping with coding the confusion matrix.

7. Source codes and online Documentation

Fedotenkova M. Available online https://github.com/mfedoten/reasspectro (Last online: FEB-2020).
Github repository. Available online https://github.com/vewald/DeepSHM (Last online: SEP-2020).
Keras Optimizers Documentation. Available online https://www.keras.io/optimizers (Last online: JUL-2020).
Opel Website. Available online https://www.opel.de (Last online: JUN-2020).
PyTorch Documentation. Available online www.pytorch.org (Last online: AUG-2020).
Tensorflow Documentation. Available online https://www.tensorflow.org (Last online: AUG-2020).

Credit authorship contribution statement

Vincentius Ewald: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Writing - original draft. **Ramanan Sridaran Venkat:** Methodology, Software, Validation, Writing - original draft. **Aadhik Asokkumar:** Methodology, Software, Validation, Writing - original draft. **Rinze Benedictus:** Resources, Supervision. **Christian Boller:** Resources, Writing - review & editing, Supervision. **Roger M Groves:** Writing - review & editing, Supervision, Project administration, Funding acquisition.

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.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.ymssp.2021.108153.

References

[1] Ann Shay L. Commercial Spending Will Lead MRO Field in 2018. Aviation Week & Space Technology (2018). Available online http://aviationweek.com/commercial-aviation/commercial-spending-will-lead-mro-field-2018 (Last online: FEB-2020).
[2] Asokkumar A, Boller C, Venkat RS. An Approach on How to Determine Key Performance Indicators for Guided Wave Based SHM Systems Based on Numerical Simulation. 12th International Workshop on SHM (IWSHM), Stanford (2019).
[3] M. Azimi, G. Pekcan, Structural health monitoring using extremely compressed data through deep learning, J. Comput. Aided Civ. Infrast. Eng. 12517 (2019) 1–18.
[4] S.K. Babu, W.T. Chan, A. Chan, Productivity & Reliability Study of Magnetic Particle Testing & Eddy current Testing for Inspection of Construction Welds, in: Proc. 15th Asia Pacific Conference for Non-Destructive Testing (APCNDT), 2017, pp. 1–13.
[5] M.F. Balcan, Rademacher Complexity. Lecture series CS 8803: Machine Learning Theory, Carnegie Mellon University, 2011.
[6] Baldi P. Autoencoders, Unsupervised Learning, and Deep Architectures. Proc. Intl Conf on Unsupervised and Transfer Learning Workshop, Washington, Vol. 27: 37-50 (2011).
[7] Y.Q. Bao, Z. Chen, S. Wie, Y. Xu, Z. Tang, H. Li, The state of the art of data science and engineering in structural health monitoring, J. Eng. 5 (2) (2019) 234–242.
[8] Y. Bao, H. Li, Machine learning paradigm for structural health monitoring, J. Struct. Health Monitor. (2020), https://doi.org/10.1177/1475921720972416.
[9] M.L. Bauccio, ASM Metals Reference Book, 3rd Ed, ASM International, Materials Park, 1993.
[10] Beer K, Bondarenko D, Farrelly T, Osborne TJ, Salzmann R, Wolf R. Efficient Learning for Deep Quantum Neural Networks. Available online: https://arxiv.org/abs/1902.10445 (Last online: FEB-2020).
[11] C. Boller, R. Mahapatra, R.S. Venkat, N.B. Ravi, N. Chakraborty, K. Simon, Integration of non-destructive evaluation based ultrasonic simulation: a means for simulation in structural health monitoring, Int. J. Struct. Health Monitor. 16 (5) (2017) 611–629.
[12] C. Boller, M.R. Mofakhami, From Structural Mechanics to Inspection Processes: Getting Structural Health Monitoring into Application for Riveted Metallic Structures, Proc. IUTAM Symp on Multi-Functional Material Structures and Systems, Bangalore, 2008.
[13] A. Botev, G. Lever, D. Barber, Nesterov's accelerated gradient and momentum as approximations to regularised update descent, Int. Joint Conf. Neural Netw. (IJCNN) 1–5 (2017).
[14] A. Bull, T.J. Rogers, C. Wickramarchchi, E.J. Cross, K. Worden, N. Dervilis, Probabilistic active learning: an online framework for structural health monitoring, J. Mech. Syst. Signal Process. 134 (2019), 106294.
[15] Y.J. Cha, W. Choi, O. Büyüköztürk, Deep learning based crack damage detection using convolutional neural networks, J. Comput. Aided Civ. Infrast. Eng. 32 (5) (2017) 361–378.
[16] K. Chaiyasarn, M. Sharma, L. Ali, W. Khan, N. Poovarodom, Crack detection in historical structures based on convolutional neural networks, Int. J. Geomate 15 (51) (2018) 240–251.
[17] Chen L, Shen LL, Tang M. Accent Classification and Neural Accent Transfer of English Speech (2018). Available online http://cs230.stanford.edu/files_winter_2018/projects/6939642.pdf.
[18] Chen Q, Zhu X, Ling Z, Wei S, Jiang H, Inkpen D. Enhanced LSTM for Natural Language Inference. Proc. 55th Annual Meeting of the Association for Computational Linguistics, Vancouver (2017).
[19] Chinta PK, Mayer P, Langenberg K. Three-Dimensional Elastic Wave Modeling in Austenitic Steel Welds using Elastodynamic Finite Integration Technique (EFIT). Proc. 18th World Conf on Non-Destructive Testing (WCNDT), Durban (2012).
[20] Cho K, van Merrienboer B, Gulcehre C, Bahdanau D, Bougare F, Schwenk H, Bengio Y. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. Proc. Conf. on Empirical Methods in Natural Language Processing (EMNLP), Doha (2014).
[21] Chong A. Global MRO spend to reach $115 billion by 2028 – Wyman. Flightglobal (2018). Available online https://www.flightglobal.com/news/articles/global-mro-spend-to-reach-115-billion-by-2028-oli-445243/ (Last online: FEB-2020).
[22] Choy AW. Structural Health Monitoring with Deep Learning. Proc. 2018 IAENG International Conf on Control and Automation, Hong Kong (2018).
[23] Clayton S. Topic 10: Rademacher Complexity. In Lecture series EECS 598: Lecture on Statistical Learning Theory. University of Michigan (2014).
[24] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines and Other Kernel-based Learning Methods, Cambridge University Press, Cambridge, 2000.
[25] M.A. De Oliveira, M.A. Monteiro, F.J. Vieira, A new structural health monitoring strategy based on PZT sensors and convolutional neural network, J. Sens. 18 (2018) 1–21.
[26] S. Duczek, M. Joulaian, A. Düster, U. Gabbert, Numerical analysis of lamb waves using the finite and spectral cell methods, Int. J Numer. Methods Eng. 99 (2014) 26–53.
[27] A. Ebrahimkhanlou, B. Dubuc, S. Salomone, A generalizable deep learning framework for localizing and characterizing acoustic emission sources in riveted metallic panels, J. Mech. Syst. Signal Process. 130 (2019) 248–272.
[28] A. Ebrahimkhanlou, S. Salamone, Single-Sensor Acoustic Emission Source Localization in Plate-Like Structures Using Deep Learning, Health Monitoring of Structural and Biological Systems XII, Proc. SPIE Smart Structures And NDE, 2018.
[29] European Aviation Safety Agency (EASA). ATA Maintenance Steering Group (MSG) Task Force 3. Rev. 1 (2009).
[30] Ewald V, Goby X, Jansen H, Groves RM, Benedictus R. Incorporating Inductive Bias into Deep Learning: A Perspective from Automated Visual Inspection in Aircraft Maintenance. Proc. 10th Intl Symposium on NDT in Aerospace, Dresden, 1-9 (2018a).
[31] V. Ewald, R.M. Groves, R. Benedictus, Transducer placement option of lamb wave SHM system for hotspot damage monitoring, MDPI J Aerospace 5 (2) (2018) 39.
[32] V. Ewald, R.M. Groves, R. Benedictus, DeepSHM: A Deep Learning Approach for Structural Health Monitoring Based on Guided Lamb Wave Techniques, Proc. SPIE Smart Structures and NDE, Denver, 2019, pp. 1–16.
[33] Fan Z, Wu Y, Lu J, Li W. Automatic Pavement Crack Detection Based on Structured Prediction with the Convolutional Neural Network (2018). Available online https://arxiv.org/abs/1802.02208 (Last online: FEB-2020).
[34] Federal Aviation Administration (FAA), US Department of Transportation (DOT). Advisory circular AC 43-204: Visual Inspection for Aircraft. (1997).

[35] Federal Aviation Administration (FAA), US Department of Transportation (DOT). Advisory circular AC 121-22C: Maintenance Review Boards, Maintenance Type Boards, and OEM/TCH Recommended Maintenance Procedures. (2012).
[36] M. Fedotenkova, P.B. Graben, J. Sleigh, A. Hutt Time-Frequency Representations as Phase SpaceReconstruction in Recurrence Symbolic Analysis. Intl Work Conf on Time Series Analysis (ITISE), Granada (2016).
[37] Feige I. Invariant-Equivariant Representation Learning for Multi-Class Data. 36th Conference on International Conference on Machine Learning (ICML), Long Beach, 1-5 (2019).
[38] Foerster JN, Assael YM, Freitas N, Whiteson N. Learning to Communicate with Deep Multi-Agent Reinforcement Learning. Proc. Conf on Neural Information Processing System (NIPS), Barcelona (2016).
[39] J. Gallagher, Damage Tolerant Design Handbook – A Compilation of Fracture and Crack-Growth Data for High-Strength Alloys, Metals and Ceramics Information Center, Columbus, 1983.
[40] C. Gallicchio, A. Micheli, Echo state property of deep reservoir computing networks, J. Cogn. Comput. 9 (3) (2017) 337–350.
[41] P. Gardner, K. Worden, X. Liu, On the application of domain adaptation in structural health monitoring, J. Mech. Syst. Signal Process. 138 (2020), 106550.
[42] A. Géron, Hands-On Machine Learning with Scikit-Learn and TensorFlow, O'Reilly Media, Sebastopol CA, 2017.
[43] Ghose B, Balasubramaniam K, Krishnamurthy CV, Rao AS. Two-Dimensional FEM Simulation of Ultrasonic Wave Propagation in Isotropic Solid Media using COMSOL. Proc. COMSOL Conference, Bangalore (2010).
[44] B. Ghose, K. Balasubramaniam, C.V. Krishnamurthy, A.S. Rao, COMSOL-Based 2D FEM Model for Ultrasonic Guided Wave Propagation in Symmetrically Delaminated Unidirectional Multilayered Composite Structures, Proc. Natl. Seminar and Exhibition on Nondestructive Evaluation, Chennai, 2011.
[45] V. Giurgiutiu, Structural Health Monitoring with Piezoelectric Wafer Active Sensors, 2nd Ed., Elsevier, Oxford & Waltham, 2014.
[46] P.W. Goldberg, M.R. Jerrum, Bounding the vapnik-chervonenkis dimension of concept classes parameterized by real numbers, J Mach. Learn. 18 (2–3) (1995) 131–148.
[47] R. Gong, C. Wu, M. Chu, Steel surface defect classification using multiple hyper-spheres support vector machine with additional information, J. Chemometr. Intell. Labor. Syst. 172 (2018) 109–117.
[48] S. Gopalakrishnan, M. Ruzzene, S. Hanagud, Computational Techniques for Structural Health Monitoring, Springer, London/Dordrecht/Heidelberg/New York, 2011.
[49] M. Gormley, Lecture 28: PAC Learning. Lecture series 10–601: Introduction to Machine Learning, Carnegie Mellon University, 2016.
[50] Han Y, Roig G, Poggio T. Is the Human Visual System Invariant to Translation and Scale? Association for the Advancement of Artificial Intelligence (AAAI) Symposium Series, 1-5 (2017).
[51] T. Hayo, B. Frankenstein, C. Boller, C. Bockenheimer, Approach to the Technical Qualification of a SHM System in Terms of Damage Detection in Aerospace Industry, in: Proc. Intl Workshop Smart Materials, Structures & NDT in Aerospace, 2011, pp. 1–9.
[52] He K, Zhang X, Ren S, Sun J. Deep Residual Learning for Image Recognition. Proc. Conf on Computer Vision and Pattern Recognition (CVPR), Las Vegas (2016). Available online: http://arxiv.org/abs/1512.03385 (Last online: JUL-2021).
[53] Hebb DO. The Organization of Behavior: A Neuropsychological Theory. John Wiley & Sons Inc, New York (1949).
[54] Heinemann L, Stuhr A. Self-measurement of Blood Glucose and Continuous Glucose Monitoring – Is There Only One Future? Proc. Satellite Symposium 11th Intl Conf on Advanced Technologies & Treatments for Diabetes (ATTD 2018), Vienna, 24-29 (2018).
[55] G.E. Hinton, S. Osindero, Y.W. Teh, A fast learning algorithm for deep belief nets, J. Neural Comput. 18 (7) (2006) 1527–1554.
[56] J.J. Hopfield, Neural Networks and physical systems with emergent collective computational abilities, Proc. Natl. Acad. Sci. U.S.A. 79 (8) (1982) 2554–2558.
[57] W. Hou, Y. Wei, J. Guo, Y. Jin, C. Zhu, Automatic Detection of welding defects using deep neural network, J. Phys. 933 (2018) (2018), 012006.
[58] C. Hu, B. Yang, J. Yan, Y. Xiang, S. Zhou, F.Z. Xuan, Damage localization in pressure vessel by guided waves based on convolution neural network approach, J. Pressure Vessel Technol. 142 (6) (2020), 061601.
[59] W. Hu, J. Hu, Training a quantum neural network to solve the contextual multi-armed bandit problem, J. Natural Sci. 11 (01) (2019) 17–27.
[60] Huber A. Dispersion Calculator Software. Available online https://www.dlr.de/zlp/en/desktopdefault.aspx/tabid-14332/24874_read-61142 (Last online: JUN-2020).
[61] L. Isik, A. Taccheti, P.T.A. Fast, Invariant representation for human action in the visual system, J. Neurophysiol. 119 (2) (2016) 631–640.
[62] Kawaguchi K, Kaelbling LP, Bengio Y. Generalization in Deep Learning. Mathematics of Deep Learning, Cambridge University Press, to appear. Preprint available as: MIT-CSAIL-TR-2018-014.
[63] Killoran N, Izaac J. Training Quantum Neural Networks with PennyLane, PyTorch, and TensorFlow. Available online https://github.com/XanaduAI/pennylane (Last online: FEB-2020).
[64] Kim Y, Huang J, Emery S. Garbage In, Garbage Out: Data Collection, Quality Assessment and Reporting Standards for Social Media Data Use in Health Research, Infodemiology and Digital Disease Detection. J Med Internet Res., Vol. 18: e41 (2016).
[65] Kingma DP, Welling M. Auto-Encoding Variational Bayes. Proc. Intl Conf on Representation Learning (ICLR), Banff (2014).
[66] J.J. Langille, R.E. Brown, The synaptic theory of memory: a historical survey and reconciliation of recent opposition, J. Front. Syst. Neurosci. 12 (2018) 52.
[67] V.F. Lavet, P. Henderson, R. Islam, M.G. Bellemare, J. Pineau, An Introduction to Deep Reinforcement Learning, Now Publisher Inc, Boston & Delft, 2018.
[68] M.C. Lee, C. To, Comparison of support vector machine and back propagation neural network in evaluating the enterprise financial distress, Int. J. Artif. Intell. Appl. 1 (3) (2010) 31–43.
[69] Y. Lei, Y. Zhang, J. Mi, W. Liu, L. Liu, Detecting structural damage under unknown seismic excitation by deep convolutional neural network with wavelet-based transmissibility data, Intl. J. Struc. Health Monitor. (IJSHM) 20 (4) (2020) 1583–1596.
[70] S. Li, W. Song, L.Y. Fang, Y. Chen, P. Ghamisi, J.A. Benediktss, Deep learning for hyperspectral image classification: an overview, IEEE Trans. Geosci. Remote Sens. 57 (9) (2019) 6690–6709.
[71] Q. Liao, J.Z. Leibo, T. Poggio, Learning invariant representations and applications to face verification, in: 27th Conference on Neural Information Processing System (NIPS), Lake Tahoe, 2013, pp. 1–9.
[72] H. Liu, Y. Zhang, Deep learning based crack damage detection technique for thin plate structures using guided lamb wave signals, J. Smart Mater. Struct. 29 (2019), 015032.
[73] Merck: Future of AI Challenge. Available online https://app.ekipa.de/challenges/future-of-ai/brief (Last online: FEB-2020).
[74] Michaels K. Opinion: OEMs Focus on Mature Aircraft for Aftermarket Growth. Aviation Week & Space Technology (2018). Available online http://aviationweek.com/commercial-aviation/opinion-oems-focus-mature-aircraft-aftermarket-growth (Last online: FEB-2020).
[75] US Department of Defense (USDOD), Wright-Patterson (2009). MIL-HDBK-1823A. Non-Destructive Evaluation System Reliability Assessment.
[76] C. Mineo, S.G. Pierce, P.I. Nicholson, I. Cooper, Robotic Path planning for non-destructive testing – A custom MATLAB toolbox approach, J. Robot. Comput. Integrat. Manuf. 37 (2006) 1–12.
[77] T. Mitchell, Machine Learning, McGraw-Hill, Redmond & Ithaca, 1997.
[78] Mohri M, Rostamizadeh A, Talwakar. Foundations of Machine Learning. 2nd Ed., MIT Press, Cambridge & London (2012).
[79] S. Moran, A. Yehudayoff, Sample compression schemes for VC classes, J. Commun. Assoc. Comput. Mach. (ACM) 63 (3) (2016) 1–21.
[80] J.L. Morales, A numerical study of limited memory BFGS methods, J. Appl. Mathemat. Letter. 15 (4) (2002) 481–487.
[81] M. Niethammer, L.J. Jacobs, J. Qu, J. Jarzynski, Time-frequency representations of lamb waves, J. Acoust. Soc. Am. 109 (5) (2001) 1841–1847.
[82] J.H. Nieuwenhuis, J. Neumann, D.W. Greve, I.J. Oppenheim, Generation and detection of guided waves using PZT wafer transducers, IEEE Trans. Ultrason. Ferroelectr. Freq. Control 52 (11) (2005) 2103–2111.
[83] N.D. Nguyen, T. Nguyen, S. Nahavandi, Multi-agent behavioral control system using deep reinforcement learning, J. Neurocomput. 359 (2019) 58–68.
[84] Ooijevaar T. Vibration-based Structural Health Monitoring of Composite Skin-stiffener Structures. PhD Diss, University of Twente (2014).
[85] W. Ostachowicz, P. Kudela, M. Krawczuk, A. Zak, Guided Waves in Structures for SHM: The Time-Domain Spectral Element Method, John Wiley & Sons Ltd., West Sussex, 2012.

[86] Panella F, Boehm J, Loo Y, Kaushik A, Gonzalez D. Deep Learning and Image Processing for Automated Crack Detection and Defect Measurement in Underground Structures. Proc. Conf ISPRS TC II Mid-term Symposium “Towards Photogrammetry 2020”, Riva del Garda (2018).
[87] M.E. Paoletti, J.M. Haut, J. Plaza, A. Plaza, A new deep convolutional neural network for fast hyperspectral image classification, J. Photogramm. Remote Sens. (ISPRS) 145(A) (2018) 120–147.
[88] Pauly P, Peel H, Luo S, Hogg D, Fuentes R. Deeper Networks for Pavement Crack Detection. Proc. Intl Symp on Automation and Robotics in Construction, Taipei (2017).
[89] S.D. Pendleton, H. Andersen, X. Du, X. Shen, M. Meghjani, Y.H. Eng, D. Rus, M.H. Ang, Perception, planning, control, and coordination for autonomous vehicles, MDPI J. Mach. 5 (1) (2017) 1–54.
[90] Petralia RS, Wenthold RJ. Neurotransmitters in the Auditory System. Encyclopedia of Neuroscience (2009).
[91] R. Purtill, The purpose of science, J. Philos. Sci. 37 (2) (1970) 301–306.
[92] Radford A, Metz L, Chintala S. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Network. Proc. Intl Conf on Learning Representations (ICLR), San Juan (2016).
[93] J.L. Rose, Ultrasonic Guided Waves in Solid Media, Cambridge University Press, 2014.
[94] R.Y. Rubinstein, D. Kroese, The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte Carlo Simulation and Machine Learning. Springer-Verlag, New York, 2004.
[95] Ruder S. An Overview of Gradient Descent Optimization Algorithms (2016). Available online https://arxiv.org/abs/1609.04747 (Last online: FEB-2020).
[96] F. Sawaf, R.M. Groves, Phase discontinuity predictions using a machine-learning trained kernel, Appl. Opt. 53 (24) (2014) 5439–5447.
[97] J. Schmidhuber, Deep learning in neural networks: an overview, J. Neural Netw. 61 (2015) 85–117.
[98] J. Sethuraman, Some limit theorems for joint distributions, Indian J. Statist. Ser. A 23 (4) (1961) 379–386.
[99] S. Shalev-Shwartz, S. Ben-David, Understanding Machine Learning: From Theory to Algorithms, Cambridge University Press, Cambridge, 2014.
[100] Simonyan K., Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition. Proc. Intl Conf on Learning Representations (ICLR), San Diego (2014).
[101] N. Soures, D. Kudithipudi, Deep liquid state machines with neural plasticity for video activity recognition, J. Front. Neurosci. 13 (686) (2019) 1–12.
[102] Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. J Machine Learning Research, Vol. 15: 1929-1958 (2014).
[103] T. Stepinski, T. Uhl, W. Staszewski, Advanced Structural Damage Detection: From Theory to Engineering Applications, John Wiley & Sons Ltd., West Sussex, 2013.
[104] Stöver T, Diensthuber M. Molecular Biology of Hearing. J GMS Current Topics in Otorhinolaryngology - Head and Neck Surgery. Vol. 10: 1-15 (2011).
[105] Szegedy G, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A. Going Deeper with Convolutions. Proc. IEEE Conf on Computer Vision and Pattern Recognition, Boston (2015).
[106] A. Taltavull, Structural Health Monitoring Solutions for Patched Metallic Aircraft Repairs Based on Guided Ultrasonic Waves, MSc Thesis, University of Saarland, 2017.
[107] Ting KM. Confusion Matrix. In Encyclopedia of Machine Learning and Data Mining. Springer, Boston (2017).
[108] P.C. Trettenbein, The demise of the synapse as the locus of memory: a looming paradigm shift? J Front. Syst. Neurosci. 10 (2016) 88.
[109] L.G. Valiant, A theory of the learnable, J. Commun. Assoc. Comput. Mach. (ACM) 27 (11) (1984) 1134–1142.
[110] L.G. Valiant, Probably Approximately Correct: Nature’s Algorithms for Learning and Prospering in a Complex World, Basic Books Inc., New York, 2013.
[111] Virupakshappa K, Oruklu E. Ultrasonic Flaw Detection Using Support Vector Machine Classification. Proc. IEEE Intl Ultrasonics Symp (IUS), Taipei (2015).
[112] P. Wallisch, M.E. Lusignan, M.D. Benayoun, T.I. Baker, A.S. Dickey, N.G. Hatsopoulus, Chapter 36 - Neural Networks Part I: Unsupervised Learning. MATLAB for Neuroscientists: An Introduction to Scientific Computing in MATLAB, 2nd Ed.,, Elsevier, London/Waltham/San Diego, 2014.
[113] D. Weimer, B. Scholz-Reiter, M. Shpitalni, Design of deep convolutional neural network architectures for automated feature extraction in industrial inspection, J. CIRP Ann. Manuf. Technol. 65 (1) (2016) 417–420.
[114] D.H. Wolpert, A mathematical theory of generalization: Part I, J. Complex Syst. 4 (1990) 151–200.
[115] D.H. Wolpert, Stacked generalization, J. Neural Netw. 5 (2) (1992) 241–259.
[116] H.A. Wood, R.M. Engle Jr, USAF Damage Tolerant Design Handbook: Guidelines for the Analysis and Design of Damage Tolerant Aircraft, Air Force Flight Dynamics Laboratory, Wright-Patterson AF Base, 1979.
[117] Wunderlich C, Tschöpe C, Duckhorn F. Advanced Methods in NDE using Machine Learning Approaches. Proc. 44th Annual Review of Progress in Quantitative Nondestructive Evaluation, Provo (2017).
[118] H. Xu, C. Xu, X. Li, L. Wang, Study on single mode lamb wave interaction with defect of plate by finite element model, Procedia Eng. 15 (2011) 5067–5072.
[119] Yousefi B, Kalhor D, Usamentiaga R, Lei L, Castanedo CI, Maldague X. Application of Deep Learning in Infrared Non-Destructive Testing. Proc. 14th Quantitative InfraRed Thermography Conf, Berlin (2018).
[120] Zayani R, Bouallegue R, Roviras D. Levenberg-Marquardt Learning Neural Network for Adaptive Predistortion for Time-Varying HPA with Memory in OFDM Systems. 16th European Signal Processing Conf (EUSIPCO), Lausanne (2008).
[121] Z. Zeng, J. Zhou, N. Tao, L. Feng, C. Zhang, X. Han, Support vector machines based defect recognition in SonicIR using 2D heat diffusion features, J. NDT&E Int. 47 (2012) 116–123.
[122] Zennaro M, Haig A, O’Boy D, Walsh S. Experimental and Numerical Analysis of a Transducer for the Generation of Guided Waves. In: 9th NDT in Progress, Prague (2017).
[123] Zhang L, Yang F, Zhang YD, Zhu YJ. Road Crack Detection Using Deep Convolutional Neural Network. Proc. IEEE Intl Conf on Image Processing, Phoenix (2016).
[124] S. Zhang, C.M. Li, W. Ye, Damage localization in plate-like structures using time-varying feature and one-dimensional convolutional neural network, J. Mech. Syst. Signal Process. 147 (2021), 107107.
[125] Zhao J, Mathieu M, LeCun Y. Energy-based Generative Adversarial Network. Proc. Intl Conf on Learning Representations (ICLR), Toulon (2017).