Perception Modelling by Invariant and Covariant Representation of Deep Learning for Automated Structural Diagnostic in Aircraft Maintenance: A Study Case using DeepSHM

Vincentius Ewald[1], Ramanan Sridaran Venkat[2], Adhi Asokkumar[2], Roger M Groves[1], Christian Boller[2], Rinze Benedictus[3]

1). Aerospace Non-Destructive Testing Laboratory, Faculty of Aerospace Engineering, Delft University of Technology, Delft, the Netherlands
2). Chair of Non-Destructive Testing and Quality Assurance, University of Saarland, Saarbrücken, Germany
3). Chair of Structural Integrity & Composites, Faculty of Aerospace Engineering, Delft University of Technology, Delft, the Netherlands

> High level introduction needed: State of the art of AI and what would be the benefits of using deep learning for structural aircraft maintenance.

In our previous work, we had 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 convolutional neural network (CNN). In that work, we only had a single central frequency excitation that was used at that time. This led to a single governing wavelength which is normally good for detection of single particular damage size. To enable a smoother transition between multiple damage size classifications, a broader frequency band (i.e. multiple wavelengths) are normally needed, thus giving a more expectable indication of damage growth process instead of binary output ("damaged vs pristine" only).

In classical signal processing, a broader excitation frequency poses an interpretation nightmare because it is more contains more complex information and thus difficult to understand. This problem can be overcome with deep learning; however, it creates another problem: while deep learning typically results in more accurate result prediction, it is specifically made for solving certain task only. While many papers have already introduced deep learning for diagnostics, many of these works are only proposing novel techniques, however the mathematical formalization lacks, and we are still left without any clue why we should treat acoustic signal with deep learning. So, the basis of 'explainable AI' for SHM and NDT currently lacks.

For this reason, in this paper, we would like to extend our work into more generalized concept of our previous work. Specifically, rather than focusing on novel technique, we are proposing a plausible theoretical perspective inspired from neuroscience on 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 so, we created a set of artificial data from finite element model (FEM) and represented DeepSHM in two different way: 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 probability.

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.

Keyword: *Ultrasonic Lamb Wave, Structural Health Monitoring, Deep Learning, Convolutional Neural Network, Time-Frequency-Analysis, Perception, Feature Learning, Invariant Representation*

1. Introduction

Airlines typically generate their revenue by using their aircrafts to transport passengers and cargo load. Beside fuel and ground services, one of the most crucial aspect in airline operating cost in the maintenance. In 2017, it was reported that 70 billion USD spent by airlines for the maintenance, repair, and overhaul (MRO) [Michaels (2018)] 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 [Ann Shay (2018), Cong (2018)].

One of the important aspects in the maintenance of the aircraft to ensure the safety and reliability remain within its design limit to maintain structural integrity, also commonly known as damage tolerance design, that is the aircraft is able to sustain predictable damages until the next repair cycle. While the damage tolerance design itself offers some sort of "passive protection" against catastrophic failure, there is no other way to ensure reliability without involving active intervention due to the nature of aircraft operation that faces safety uncertainty and this is the reason why structural diagnostic in aircraft maintenance must be performed regularly.

1.1. Role of Structural Diagnostic in Aircraft Maintenance

In world of aircraft maintenance, the Maintenance Steering Group – Task 3 (MSG-3) logic [EASA (2009)] has been a root methodology for modern aircraft maintenance and among the important tasks in MSG-3 process is the Special Detail Inspection (SDI) [FAA (1997)] that has a purpose to detect hardly visible aircraft structural damages by human inspector and requires non-destructive testing (NDT). One of the classical solutions that has been established for decades is to train an NDT inspector with various inspection methods, however it is relatively expensive in matter of time and costs to train human resources until the certification is reached. As an example, a level-3 certification provided by DGZfP for ultrasonic testing (UT) would cost between 20.000 to 25.000 Euro [DGZfP]. To induce more efficient inspection process, two mainstreams of automated SDI has been proposed: the automated NDT [Mineo et al. (2006)] and Structural Health Monitoring (SHM) [Giurgiutiu (2014)]. While automated NDT typically means robot-assisted non-destructive inspection (NDI), in SHM the NDI instruments become an integral part of the structure itself [Boller and Mofakhami (2008)].

Both approaches have its own advantages and disadvantages, 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 aircraft structures on the geometrical level, but also on the material level) of the system increase. Thus, we are profound that a more sophisticated data processing technique particularly in digital signal processing (DSP) is 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 analyze the captured signals by extracting the signal features from one or multiple sensors to understand the behavior of particular physical phenomenon of interest. Nowadays, in the era of increasing data volume and parallel computing, the main problem of signal processing has shifted into statistical signal modelling by incorporating stochastic parameters into the statistical model. In the meanwhile, in the field of computer science, machine learning has gained popularity in recent years. As machine learning is typically used to model statistical process, it is one of suitable techniques to model the 'real-world' signal, which in general exhibits stochastic behavior. As an example, one popular machine learning technique that was introduced in 1990's was Support Vector Machine (SVM) [Cristianini and Shawe-Taylor (2000)]. One example of using SVM for NDT can be found in the work of Virupakshappa et al. (2015) for ultrasonic flaw detection in steel block and in the work of Gong et al. (2017) for defect classification by using thermal sensor.

More recently, people were increasingly talking about 'deep learning' or deep neural network (DNN) [Schmidhuber (2015)], which was introduced for the first time in 1950's as a perceptron and later emerged as a network and called multilayer perceptron (MLP). Since MLP is an imitation of biological neural network found in human body, it is also sometimes referred as artificial neural network (ANN). As deep learning itself is just overhyped marketing buzzword for complex multilayered ANN, technically 2-layered MLP which is already known in 1970's would also qualify to be called deep learning. ANN had its popularity grew until 1990's [Hopfield (1982)], where at that time the computational ability

was low and eventually the ANN popularity was overshadowed by SVM [Lee (2010)]. It was not until 2006 when Hinton introduced deep belief network (DBN), a class of DNN, where neural network regained its popularity.

Since then, many approaches within neural network realm have been proposed. Pure discriminative method in neural network includes the variants convolutional neural network (ConvNet or CNN) such as Inception Network [Szegedy et al. (2014)], VGG-16 and VGG-19 [Simonyan and Zisserman (2014)], and CNN with skip connections, also known as Residual ConvNet or ResNet [He et al. (2015)]. Beside ConvNet, there is also 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) [Chen et al. (2017)] and Gated Recurrent Unit [Cho et al. (2014)], 2). Reservoir based operation such as Deep Echo State Network (D-ESN) [Gallicchio and Micheli (2017)] and Deep Liquid State Machine (D-LSM) with neural plasticity [Soures and Kudithipudi (2019)].

For generative modelling, Deep Autoencoders (DAE) [Baldi et al. (2012)] and variational Autoencoders (VAE) [Kingma and Welling (2019)]. Both are 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. Beside encoding techniques, there is semi-supervised approach that combines discriminative and generative approach such as Deep Convolutional Generative Adversarial Network (DCGAN) [Radford et al. (2016)] and Energy-based Generative Adversarial Network [Zhao et al. (2017)].

In the area of model-free learning such as reinforcement learning, deep learning can be combined with multi-agent system of reinforcement learning [Foerster et al. (2016), Lavet et al. (2018), and Nguyen et al. (2018)]. Since such model-free environment is even more data-exhaustive than supervised learning, current applications of such approach is most limited to simulated reality such as computer game. 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 are generally focused on surface inspection of structures have been proposed by: Zhang et al. (2016), Cha et. al (2017), Chaiyasarn et al. (2018), Fan et al. (2018), Panella et al. (2018), Pauly et al. and many more. Beside for crack detection for surface, there are several other works involving CNN in NDT, such as for phase detection technique in shearography proposed by Sawaf and Groves (2014), welding detection using X-Ray images by Hou et al. (2018), and damaged steel and CFRP using infrared (IR) images by Yousefi et al. (2018). As pointed out by Wunderlich et al. (2017), we believe that advances in machine learning will have a huge impact in several key areas of NDI.

In 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 work that incorporates deep learning in SHM has been proposed by of Ebrahimkhanlou and Salomone (2019) who used deep autoencoder (deep AE) for acoustic emission (AE) source localization, Choy (2018) and Oliveira et al. (2018) who used CNN for processing electromechanical impedance (EMI), and Azimi and Pekcan (2019), who used CNN for damage identification and localization of vibration sensor data in civil infrastructure.

However, as CNN is a discriminative model that is specifically tailored to learn how to solve certain task, 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 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, however not only it requires a pretrained model based on a large dataset but also a new model still needs to be retrained based on the pretrained parameters. In many other domains such as NDT, such large dataset is not publicly available thus making it more difficult to perform transfer learning. It has been tried for recognizing crack image as has it been proposed, and in Lamb wave based SHM there was trial done [Liu and Zhang (2019)] although we doubt the efficacy of transferring image parameters for audio or acoustic wave signal processing. In line with other computer scientist, the reason for non-transferability between image into audio is because such pre-trained model was trained specifically for recognizing images that has a physical origin of photon particles captured by RGB camera sensor which is fundamentally of different physical phenomenon from 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 was just randomly initialized and thus no need of transfer learning.

Another approach using online active learning has been proposed by Bull et. al (2019). They clearly recognized that the lack of descriptive labels made a conventional supervised learning is not feasible and they proposed a novel adaptive learning process that updates the learning algorithm as soon as a new class of data is discovered by calculating the entropy. 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 currently lacks. 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 to use active acoustic based SHM approaches such as Lamb waves for continuously monitoring the ageing or repair of large aircraft of both metallic and composite structures. 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 or examine the system's response for various damage configurations and during the operation phase, the simulation may be employed as a part of the system providing a virtual baseline result. 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 [Gopalakrishnan (2011), Stepinski et al. (2013), Giurgiutiu (2014), Chinta et al. (2012), Xu et al. (2011), Rose (1999), Venkat and Boller (2017)].

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, an ultrasonic transducer head is free to move or scan over the entire structure for detecting the damages, whereas in guided wave based SHM, the area to be monitored is actively inspected by using a few transducers bonded on to the structures. In the later, the location of transducers plays a major role in damage detection and hence, an optimized sensor configuration is of primary importance and it can be only achieved through simulation. An approach to determine the optimized sensor location for damage detection using numerical simulation is demonstrated various works [Boller et al. (2017)].

Numerical simulation of guided waves brings thorough understanding of governing mechanisms of the wave propagation when it interacts with structural features. Although extraction of desired modes using signal processing methods does exist [Gopalakrishnan (2011)], 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. Asokkumar et al. (2019) proposed such approach to identify the damage features using simulation driven guided wave data for various damage configurations and it is needless to mention here that it would otherwise impossible to perform this study using experimental data.

1.3. Objective and Outline

Based on the state-of-the art in section 1.2, we must assume the following premises to determine the research objective: 1). more complexity in geometry and material properties would require enhanced signal processing that can 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 real world situation, and 3). A pure statistical model can only find correlation, but not causation (also known as '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 SHM or NDT.

In our previous work [Ewald at al. (2018)], 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 and Worden

(2020). For the SHM application using active Lamb wave, we had proposed a hybrid model that we called DeepSHM framework. Specifically, it is a statistical signal modelling based on deep learning that is biased of physical model and we showed it worked like a charm for complicated signal classification. Remind that the advantage of DeepSHM, just like any other deep learning algorithm, 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 and sometimes surpasses human intelligence. The biggest disadvantage of DeepSHM is also its ***agnosticism***: for any given rubbish input, the outcome would be a rubbish output, or we called this process in computer science ***Garbage in – Garbage out***.

While DeepSHM would work given any input signal, we choose to align this research with our previous work and therefore we limit the use of DeepSHM solely for active Lamb wave based SHM. One specific problem that we still encountered in [Ewald at al. (2019)] 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. We hypothesize that this problem might be overcome by:

1. ***Applying broadband frequency excitation since this will invoke 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 confusion matrix will be compared, with given captured signals from different sensing locations. One of the associated problems with signal processing with multitude sensing locations is that it would result in different sensor response which might give obsolete predictions and the leads to following research questions:

1. ***How far 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 (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 technical way to encompass the engineering knowledge. Nevertheless, as with aligning with main purpose of scientific research, we believe that just the demonstration of the technical approaches of DeepSHM alone is not sufficient to become a scientific contribution. To emphasize this, we cite the purpose of science by Purtill: "*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 why it worked. Thus rather than proposing ***novel superior techniques*** which in the meanwhile becomes the ***devil's circle of academia*** due to insufficient research funding within increasing academic competition or whatever the reason is, in this paper we took a step back to rather explain why deep learning should work for treating Lamb wave signal and where would it stop working. Therefore, as we are aiming for ***methodology that makes reproducible result*** rather than just focusing on result. In this paper we highlight the most important theorems and lemmas that brings the assertion of the reason why Lamb wave signal can be treated by deep learning. The reason we chose to publish in this journal is because our target readers are SHM and NDT practitioner that aim to employ deep learning in their field work and certainly it is mostly engineers who are going to benefit from domain transfer from computer science community into mechatronics system and engineering – not the other way around.

The way we organize the article is as follow: 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 an abstract thought about modelling an SHM perception from neuroscientific perspective in section 2.1 – 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 it is given in section 2.5. The methodology regarding the sensor placement, numerical model, data pre-processing, and neural network training are described in section 3.1 – 3.4. The sample results and discussion are given by section 4, while the full results are attached in the attachment. Finally, the conclusion of our work is given in section 5.

2. Concept and Theoretical Foundation

To be more precise on what we mentioned in section 1.3 regarding the DeepSHM framework, we divide section 2 in 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. For brevity, we limited the theory section of Lamb wave propagation and its simulation in section 2.5 as these are widely known.

2.1. Diagnostics

To be more generic with the expression of our idea, let consider two following real-world examples given in Table 1 that we call diagnostics. 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. Thus, we took 2 typical and very well-known examples which is not SHM-related: continuous glucose monitoring (CGM) for diabetic patient and magnetic particle NDT inspection in an aircraft hangar. There are 5 aspects in each diagnostic scheme that can be considered: actor, transition, medium, phenomenon, and environment.

Tuple	CGM	Wet Magnetic Particle Inspection
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 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 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

Table 1: Example comparison between medical and structural diagnostic

Categorizing these 5 aspects in the diagnostic realm, we abstract our idea as depicted in Fig. 1. Whether it is targeted for medical purpose such as CGM and 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 $\boldsymbol{D}$: {$\boldsymbol{\pi}$, $\boldsymbol{\psi}$, $\boldsymbol{\tau}$, $\boldsymbol{\lambda}$, $\boldsymbol{\omega}$}. For SHM, we can describe each of these aspects as following:

$\boldsymbol{\pi}$ is the actor domain. This tuple contains the parameters that are needed to generate and measure the physical phenomenon of interest $\boldsymbol{\lambda}$. An example of the influencing parameters in this domain includes: the geometry of the actuator or sensor, the instrumentation quality including calibration accuracy of the oscilloscope, length of measuring cable and soldering quality, the excitation parameters such as frequency, waveform type and burst period, and the source location of the phenomenon of interest.

$\boldsymbol{\psi}$ is the medium domain. We define this tuple as the subject space where phenomenon of interest $\boldsymbol{\lambda}$ reigns after being excited in the actor domain as per fundamental physics law ***action = reaction*** and the philosophical expression ***ex nihilo nihil fit***. When the parameters of the medium domain changes, the behavior of $\boldsymbol{\lambda}$ changes. An example of the influencing parameters in this domain include: the geometrical and electro-thermomechanical material properties such as anisotropic elastic moduli, heat transfer coefficient, density, electrical conductivity, but also the presence of any possible inhomogeneity of any size inside the medium ranging from nanoscopic, microscopic, and macroscopic defect including but not limited to crystal defect, grain boundaries, dislocations, particle inclusion, composite delamination, micro- and macrocrack, corrosion, fiber break, and also the presence of mechanical fasteners, etc.

$\boldsymbol{\tau}$ is the transitional domain, which separates the actor from the medium. Concretely, as in many real-world situations where the medium has a finite property, the transitional domain can be understood as the boundaries of the medium. Examples of

this are the boundary condition of the medium and the boundary condition between the actor and medium. A more concrete examples of transitional domain in NDT and SHM are water, ultrasonic gel couplant or sensor glue.

λ is the domain of the phenomenon of interest, that is the physical phenomenon which is observed in actor domain **π**. The phenomenon **λ** typically interacts with at least one parameter in all other domains. For example, in a general diagnostic, one would use ultrasonic wave, electromagnetic effect such as eddy current, or (non)-visible light (ranging from X-ray to UV) to draw a conclusion whether inhomogeneities within the medium domain are present or not. Due to practicality, not all physical phenomenon (e.g. X-ray) are always suitable for diagnostic SHM. The cruciality in the diagnostic is that the phenomenon must be fluctuating so that the actor π can recognize the contrast between the family of phenomenon **λ**.

ω is the environmental domain which covers actor, transition, and medium domain **π**, **ψ**, and **τ**, respectively but are separated from them. This tuple contains parameters that could potentially changes the observation in actor domain **π** or even changes the behavior of **λ** completely but is not included in the other domains. This domain can be modelled as an external stochastic parameter and examples of the influencing parameters in this domain include measurement temperature, humidity, external vibration, electromagnetic radiation, etc.

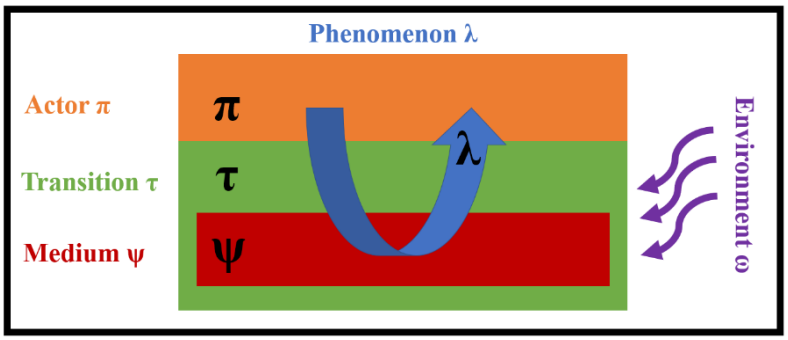


Fig. 1: Diagnostic realms ***D***

Focusing our work on SHM, we follow the SHM workflow proposed by Ooijevaar (2014) which is summarized in Fig. 2. A Diagnostic SHM consists of 3 functional levels: level 1 is concerned with load monitoring which we would not further discuss in this article due to resource and time constraint, while level 2 is concerned with damage detection within the structure to answer the question: ***does a damage exist within the structure?***. Level 2 SHM can easily be understood as a decision problem and the answer can easily be determined as either YES or NO and for that, one can just compare the signal from pristine structure – also called baseline signal and the signal of damaged structure and it is rather a question of thresholding to determine the corresponding a YES or NO response.

The functions in SHM level 3 can only proceed if the question in level 2 is answered with YES and concerned with the extracted information that can be retrieved from the signal regarding the damage, such as location, type, and size of the damage to answer the question: ***where, what type, and how severe is the concerned damage?*** The majority of diagnostic SHM works are focused in this area since it requires more complex answer rather than just binary outcome in level 2. The function of SHM level 4 would go beyond these questions as it approximates the residual lifetime (sometimes called remaining useful life) given the information retrieved in level 2 and 3. In this work, we are limiting our focus on diagnostic SHM only, and thus as level 4 is a part of prognostic SHM, we would not discuss it further in this paper.

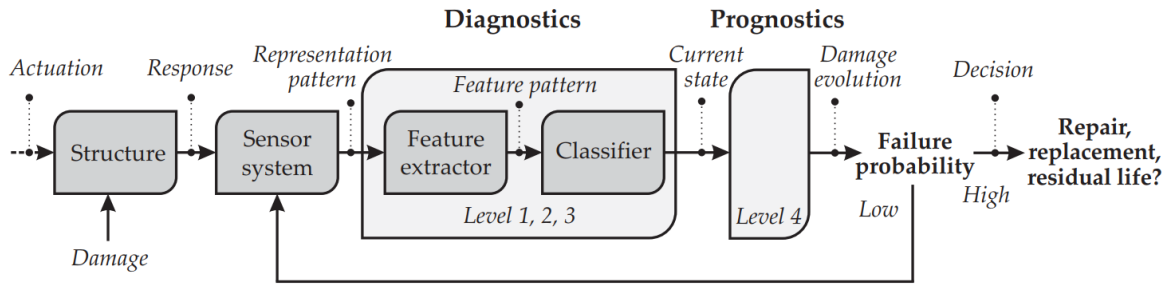


Fig. 2: SHM Workflow in General [Ooijevaar (2014)]

2.2. Generalization of DeepSHM Framework

The most direct formulation to describe the relations between the domains in the diagnostic realm *D* would be:

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

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, $\boldsymbol{X}$ is a finite-dimensional observation tensor. However, as $\boldsymbol{\lambda}$ is hidden and what we typically observe from outside is $\boldsymbol{X}(\boldsymbol{\pi}, \boldsymbol{\psi}, \boldsymbol{\tau}, \boldsymbol{\omega})$, the relation can be rewritten as the inverse model:

$$\boldsymbol{X}_{\boldsymbol{\pi},\boldsymbol{\psi},\boldsymbol{\tau},\boldsymbol{\omega}} = f^{-1}(\boldsymbol{\lambda}) = g(\boldsymbol{\lambda}) \quad (2)$$

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 $\boldsymbol{X}$ rather as probabilistic variables rather than deterministic one. For brevity, let assume the null hypothesis h_θ where the existence of $\boldsymbol{\lambda}$ is caused only by changing parameter in $\boldsymbol{\psi}$ and due to the first assumption of the stochastic nature of observation $\boldsymbol{X}$, the relation can be formulated via Bayes conditional probability *P*:

$$P(h_\theta(\boldsymbol{\lambda}) \mid \boldsymbol{X}_{\boldsymbol{\psi}}) = \frac{P(\boldsymbol{X}_{\boldsymbol{\psi}} \mid h_\theta(\boldsymbol{\lambda})) \cdot P(h_\theta(\boldsymbol{\lambda}))}{P(\boldsymbol{X}_{\boldsymbol{\psi}})} \quad (3)$$

Where in Eq. (3), $P(h_\theta(\boldsymbol{\lambda})|\boldsymbol{X}_{\boldsymbol{\psi}})$ is the posterior probability of the existence of $\boldsymbol{\lambda}$ given observations $\boldsymbol{X}_{\boldsymbol{\psi}}$, $P(\boldsymbol{X}_{\boldsymbol{\psi}}|h_\theta(\boldsymbol{\lambda}))$ is the prior probability where $\boldsymbol{X}_{\boldsymbol{\psi}}$ occurs given the hypothesis $h_\theta(\boldsymbol{\lambda})$, $P(\boldsymbol{X}_{\boldsymbol{\psi}})$ and $P(h_\theta(\boldsymbol{\lambda}))$ are the marginal probability of observing $\boldsymbol{X}_{\boldsymbol{\psi}}$ and $h_\theta(\boldsymbol{\lambda})$ 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 $\boldsymbol{X}_{\boldsymbol{\psi}}$ is multi-dimensional, it is logical to formulate the problem as multivariate distribution rather than 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 [Sethuraman (1961)]

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

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

Furthermore, we consider the most suitable expression of the joint probability distribution. Naturally, there is infinitely possible combination within $\boldsymbol{\psi}$ (e.g. a crack length in metallic structure can vary between 1 µm to 100 cm – or put any arbitrary number you want, a composite delamination can vary between 0.1 mm² to 1000 cm², a corrosion depth can also vary between 1 µm to 1 cm, etc.) and very small variation within $\boldsymbol{\psi}$ normally only causes small variation within $\boldsymbol{X}_{\boldsymbol{\psi}}$. Therefore, instead of discrete probability mass distribution, the probability density function is the more suitable formulation for joint distribution in diagnostic since it expresses the density of continuous random variable.

Now, we have to consider how to associate the probability distribution with machine learning, and in general, when considering any machine learning algorithms, following questions naturally arise: which learning problems can be solved efficiently and is it 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 domain, the proper intuition would be the learnability of the function itself, commonly known as probably approximately correct (PAC)-learning framework. More concretely, the underlying assumption for given assumption stated in Eq. (1) and (2) is we know that h_θ: $f(\boldsymbol{X}_\psi) \rightarrow \boldsymbol{\lambda}$ belongs to concept class $\mathfrak{C}$ thanks to the physical understanding. Let consider lemma 1.

Lemma 1: PAC-learning [Valiant (1984, 2017), Gormley (2016), Moran S et al. (2015)]

For any hypothesis $h_\theta \in H_{\mathfrak{C}}$, where $H_{\mathfrak{C}}$ is the hypothesis space in the concept class $\mathfrak{C}$, the generalization error J is defined as $J_{h_\theta} = P(h_\theta(\boldsymbol{\lambda}) \neq f(\boldsymbol{X}_\psi))$. 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 $\widehat{J_{h_\theta}} = \frac{1}{N}\sum_{n=1}^{N} P(h_\theta(\boldsymbol{\lambda}) \neq f(\boldsymbol{X}_\psi))$. Assume the existence of a learning algorithm $\mathfrak{A}$ such that for any $\varepsilon > 0$ and $\delta > 0$, $\forall$ distribution κ on input $\boldsymbol{X}_\psi$, the following holds for training sample size $\left\{ N = 1/\varepsilon \cdot \ln |\mathfrak{C}|/\delta \right\} : P(J_{h_\theta} \leq \varepsilon) \geq 1 - \delta$. Then, the concept class $\mathfrak{C}$ is said to be PAC-learnable.

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

Definition 2: Multidimensional Generalizer [Wolpert (1990, 1992)]

An 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. Notationally, a k-dimensional generalizer is a set of continuous functions $L_{\{i\}}$ along with domains of defininions, 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 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*". The connection between an upwardly compatible HERBIE and machine learning algorithm can be found in Mitchell [36], where he defined the algorithm of machine learning 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 is said to be able to generalize on the class of tasks T of certain probability distribution.

In the diagnostic realm, task T is the diagnostic itself, that is retrieving the information from the observable variables $\boldsymbol{X}_\psi$ 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 $\boldsymbol{X}_\psi$. The probability for the i-th class of information given the $\boldsymbol{X}_\psi$ in k-dimensional Hilbert space is typically written as logit or sigmoid function [Mitchell, USDOD] and can be generalized in softmax function which is defined by:

$$P(h_\theta(\boldsymbol{\lambda}) = i \mid \boldsymbol{X}_\psi) = \frac{\exp([\boldsymbol{X}_\psi]^T \cdot \theta_i)}{\sum_{k=1}^{K} \exp([\boldsymbol{X}_\psi]^T \cdot \theta_k)} \quad (4)$$

The posterior of $P(h_\theta(\boldsymbol{\lambda})|\boldsymbol{X}_\psi)$ is formally defined as [0, 1], but also sometimes informally written between 0% to 100% and as convention in this paper, both writings are interchangeable. The posterior is maximized by minimizing the information difference, which is sometimes referred as lost, cost or error between the predicted value $h_\theta(\boldsymbol{\lambda})$ and the true information contained $\boldsymbol{X}_\psi$. The central task of machine learning is to minimize the loss function by iteratively adjusting θ. Depending on the formulation problem, different loss function must be defined. For instance, in regression problem, 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 $p(\boldsymbol{X}_\psi)$, and the estimated distribution $q(\boldsymbol{X}_\psi)$ is defined as:

$$H(p,q) = -\sum_{\boldsymbol{X}_\psi} p(\boldsymbol{X}_\psi = \boldsymbol{\lambda}) \cdot \log(q(\boldsymbol{X}_\psi = h_\theta(\boldsymbol{\lambda}))) \quad (5)$$

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

$$\arg\min J_\theta = \min\left[-\frac{1}{N} \left(\sum_{i=1}^{N} p(\boldsymbol{X}_\psi = \boldsymbol{\lambda}) \cdot \log(q(\boldsymbol{X}_\psi = h_\theta(\boldsymbol{\lambda}))) \right) \right] \quad (6)$$

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 as NP-hard) as it contains highly-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 $J_\theta \rightarrow 0$, 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 [Sethuraman (1961)]

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

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

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:

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

Where in Eq. (7), TN is the true negative rate and FP is the false positive rate (equivalent to statistical error of type I). It is to be noted that often it is useful to influence the algorithm by domain knowledge, as we demonstrated before in our previous work [Ewald et al. (2018)]. 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 upwardly compatible generalizer in Def. 2 on deep learning by considering Def. 3:

Definition 3. Generalization gap and Rademacher Complexity [Kawaguchi et al. (2017), Mohri et al. (2012), Balcan (2011)]

Let N-samples in sampling space of training dataset $S_N \triangleq \{(x_1, y_1), ..., (x_N, y_N)\}$, where $x \in \boldsymbol{X}_{\psi}$ and $y \in h_{\boldsymbol{\theta}}(\boldsymbol{\lambda})$, and the generalizer $L(S): \boldsymbol{X}_{\psi} \to h_{\boldsymbol{\theta}}(\boldsymbol{\lambda})$. The expected generalization risk is denoted as $R_{L(S)}$ and the computable empirical risk is denoted as $\widehat{R}_{L(S)} = \frac{1}{N}\sum_{i=1}^{N} J_{\theta}(L(x_i), y_i)$. where J_{θ} is the error or loss function associated with L.

The generalization gap $\mathfrak{G}$ is defined as $\mathfrak{G} \triangleq R_{L(S)} - \widehat{R}_{L(S)}$. Further, if the codomain $J \in [0,1]$, we have that $\forall \delta > 0$ with probability at least $1-\delta$: $\sup\ G \leq 2 \cdot \mathfrak{R}_N(L) + \sqrt{\frac{\ln(1/\delta)}{2N}}$, where $\mathfrak{R}_N(L)$ is the Rademacher complexity of L.

The Rademacher complexity $\mathfrak{R}$, together with Vapnik – Chervonenkis (VC) dimension [Goldberg and Jerrum (1995), Clayton (2014)] 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. The easier language for dummies (especially in non-mathematics or computer science world) would be: approximating linear function can be done 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 (2017) proposed a theorem via validation error as given in Lemma 3.

Lemma 3. Generalization bounds of deep learning via validation [Kawaguchi et al. (2017)]

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

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

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.***

We are aware that in their proposition, even in worst case where $\sup\left|\aleph_{L,i}\right| = 1$ and $\sup \mathbf{E}(\aleph_{L,i}^2) = 1$, and by having large hyperparameter cardinality (e.g. 10^6) and 1000 training epochs in larger dataset N_{val} = 10000, the second and third term of the equations sums up only to 6.94% where in non-worst case, this figure decreases to 0.49%. For the case where the dataset is smaller (e.g. N_{val} = 100) and less complex deep neural network architecture with 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 smaller deep neural network would not generalize well over the hypothesis space as the generalization gap is large, but depending on the problem, one could argue that for some problem settings, an inferior generalization capability is still better than anecdotal generalization. Further, the implication of upwardly generalizable of a diagnostic algorithm, be it either for NDT or SHM – or even medical application (e.g. for early stage 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 sensitivity or recall rate, which is defined as:

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

The current standard of practice for diagnostic NDT according to MIL-HDBK1823 [USDOD] is $POD = 0.9$ (or 90%) within 95% statistical confidence σ, although this might not be suitable for diagnostic SHM [Hayo et al. (2011)]. To be more concrete with the concept of DeepSHM, we redraw the generic SHM formulation proposed by Ooijevaar in Fig. 2 in an active Lamb wave SHM system in Fig. 3. The processing framework of the observable $\boldsymbol{X}_{\psi}$ that contains 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.

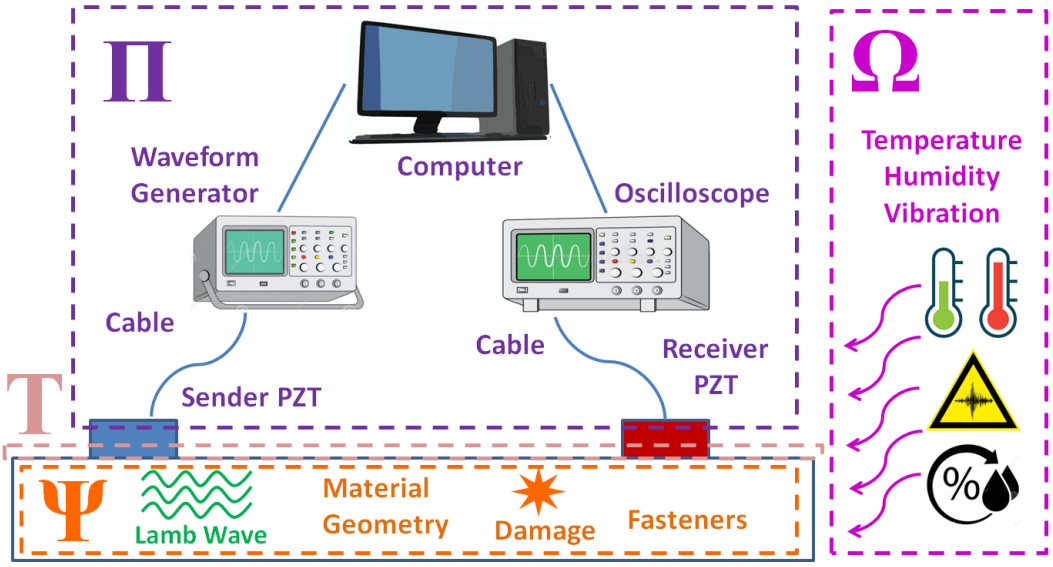


Fig. 3: Diagnostic SHM by using active guided Lamb wave as physical phenomenon

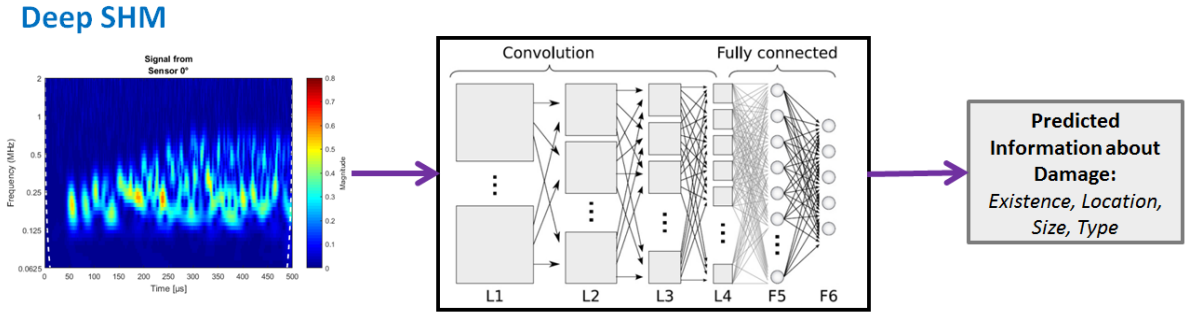


Fig. 4: DeepSHM Framework

In line with any machine learning optimization process, the central task of DeepSHM framework is to find generalizable parameters θ to fit the correlation between $\boldsymbol{X}_{\psi}$ 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 our idea on DeepSHM according Fig. 4 should be concretized in the physical domain, and thus in this section we are discussing on 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 to improve the performance metrics, we consider it be more valuable to have an inspirational insight from other domain such as biology (especially auditory and neuroscience) or control theory. We are profound that the future AI should behave close to that of human intelligence and within this approach we emphasize that scientific contribution should not just about the ***bog standard novel superior AI technique*** due to fierce academic competition especially on research funding. 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 an example of autonomous driving [Pendleton et al. (2017)]:

> "*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.*"

In analogous way, let adapt the above-mentioned definition for our case: ***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.

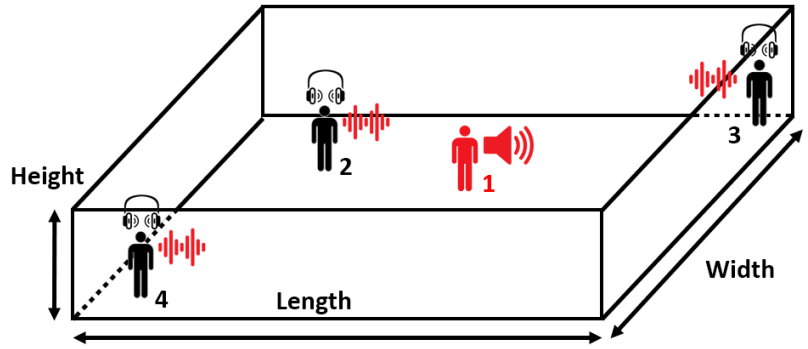


Fig. 5: Simplified setting on a person 1 singing in an enclosed space where the others are listening

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 hear 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. Consider the hearing process [Stöver and Diensthuber (2011)] in Fig. 6: Hearing consists of two parts: 1). 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 [Petralia and Wenthold (2009)] into auditory cortex located in the temporal lobe of human brain.

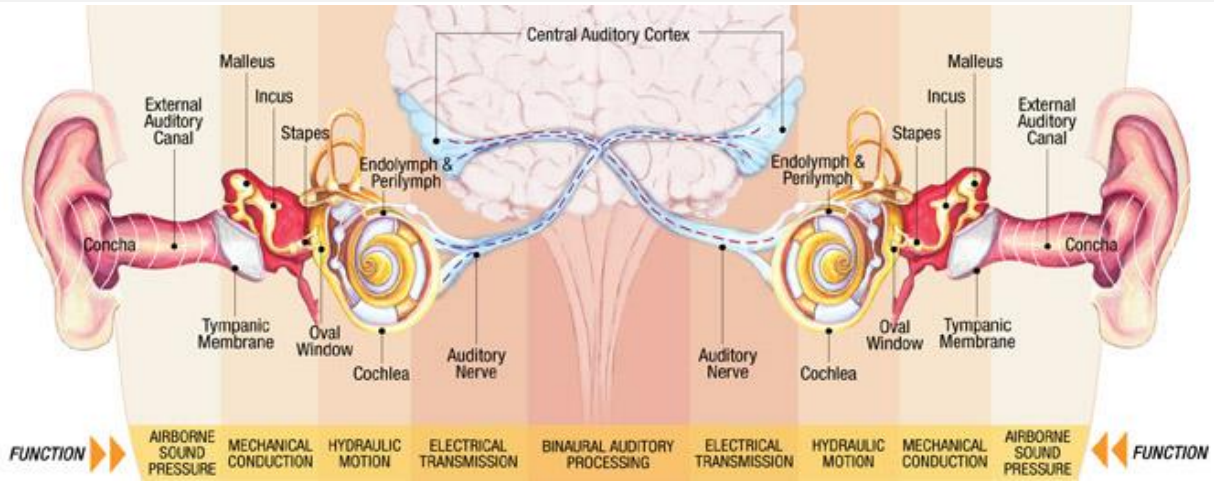


Fig. 6: Binaural hearing process [Global Care Company]

Part 1 is analogous to conversion of analog Lamb wave signal into 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 and thus analogically 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 our work on part 2 only. In line with the objective stated in section 1.3, it would be interesting to see whether a group of half-deaf people can classify certain music track correctly or not – or should this environment be regarded as one single entity representing a song heard by multiple ears (thus should be called ***multiaural hearing***) that is connected into single brain, especially if this song is played again and again?

To further understand about this idea, we took the Hebbian postulate (1949): "*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 [Wallisch et al. (2014)] states that the connectivity between two neurons is determined by how strong they are linked to each other, i.e. the connectivity between two neurons increases if the they are simultaneously activated and decreases if otherwise. Practically, if the auditory cortex is fed with the same song multiple times, at some point the brain would automatically recognize the next time it is played thanks to the neuronal memory allocation induced by the neuroplasticity [Langille and Brown (2018), Trettenbein (2016)]. More interestingly, sometimes our brain still recognizes the same music even if it has been remixed – so we assume 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 the fact 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 $\boldsymbol{G}$, as the recorded signals would likely to have entirely different amplitudes, phase shift, and frequency from each other. We believe that the human perception, on recognizing song $\boldsymbol{G}$ 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 $\boldsymbol{G}$. To help understand what invariant representation means, consider Lemma 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 $F^k \triangleq \left\{ f \mid f : \mathbb{R}^k \to \mathbb{R}^k \right\}$ and $(X_1,...,X_k)$ affine spaces. A composition $\boldsymbol{O}$ is defined as: $\boldsymbol{O} : F^k \times F^k \to F^k$. Let $(\boldsymbol{X},\boldsymbol{Y},\boldsymbol{Z})$ be the hyperspaces of the affine spaces $(X_1,...,X_k)$, $(Y_1,...,Y_k)$, and $(Z_1,...,Z_k)$ that contain point sets $(x_1^{(k)},...,x_i^{(k)}),(y_1^{(k)},...,y_i^{(k)})$, and $(z_1^{(k)},...,z_i^{(k)})$ where $i \in \mathbb{N}$. Let $(\boldsymbol{U},\boldsymbol{V})$ a set of linear map, and $(\boldsymbol{U} : \boldsymbol{X} \to \boldsymbol{Y})$, $(\boldsymbol{V} : \boldsymbol{Y} \to \boldsymbol{Z})$ be affine transformations. A map $\boldsymbol{U} : \boldsymbol{X} \to \boldsymbol{Y}$ is said to be an affine map if $\exists \boldsymbol{V} : \boldsymbol{Y} \to \boldsymbol{Z}$ such that $\boldsymbol{V}(y-z) = f(y) - f(z)\ \forall y, z$ in $(\boldsymbol{Y},\boldsymbol{Z})$. Then, the composition $\boldsymbol{O} : \boldsymbol{V} \circ \left(\boldsymbol{U} : \boldsymbol{X} \to \boldsymbol{Y}\right)$ is an affine transformation with tangent map $\boldsymbol{V} \circ \boldsymbol{U}$.

Lemma 5. The cumulative distribution function (CDF) as the estimate of invariant for affine transformation. [Liao et al. (2013)]

Assume an observation $x \in \boldsymbol{X}_{\boldsymbol{\psi}}$, where $\boldsymbol{X}_{\boldsymbol{\psi}}$ is a set of all possible observations. Furthermore, we consider transformation $g_t \in \boldsymbol{G}$, where $\boldsymbol{G}$ is 2D affine groups that consists of all possible tranformations $\boldsymbol{T}$, such as translation, resizing, etc. Assume a set of smallest atomic representations of observation x as τ_i where $i \in \mathbb{N}$. For the set $\left\{ g_t \tau_i \mid t = 1,..., \boldsymbol{T} \right\}$, the distribution of $\left\langle x, g_t \tau_i \right\rangle$ is invariant and unique to each observation. The empirical distribution κ of the inner products $\left\langle x, g_t \tau_i \right\rangle$ is used as the signature:

$\kappa_n^i(x) = \frac{1}{\boldsymbol{T}} \sum_{t=1}^{\boldsymbol{T}} \sigma\left(\left\langle x, g_t \tau_i \right\rangle + n\Delta \right)$, where σ is the typical sigmoid function and Δ is the resolution parameter for each observation. Since each observation x has its own characteristic empirical κ, it also shows that these signatures could be used to discriminate between them.

Lemma 6. Invariant latent. [Feige (2019)]

Let 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 $\left\{ x_n \mid h_\theta(x_n) = y_{\lambda^{(n)}} \right\}$ will be denoted as $\boldsymbol{D}_{h_\theta}^{y}$. The invariant latent $r_{y_{\lambda^{(n)}}}$ of label $y_{\lambda^{(n)}}$ is thus defined as: $r_{y_{\lambda^{(n)}}} = \mathbf{E}_{x \sim (\boldsymbol{D}_{h_\theta}^{y} \setminus \{x_n\})} \left[f_\theta(x) \right] \approx \frac{1}{N} \sum_{i=1}^{N} f_\theta(x^i)$ where N is the number of sampled observations.

The importance of these above-mentioned lemmas is following: 1). Any (non)-dispersive acoustic wave signal (including body waves such as P- and SV-wave and/or surface wave such as Lamb-, Love-, Rayleigh wave, SH-wave, and/or any possible but finite combinations within the wave subset) can be regarded as a decomposable multivariable function of multiple polynomial function 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 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*** [Ekipa (2019)]. In one of the challenges, they were looking for a formalization of 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 [Isik, Han (2016)] (e.g. recognizing a face or a particular soundtrack), the human brain is known to have the ability to capture the invariant representation of the object. For instance, if we would go sunbathing and getting tanner or taking selfie from different smartphone angle – it is not that our neighbor suddenly recognizes our face as completely different person. The same way when we are listening to particular soundtrack such as "*Justin Bieber – Love Yourself*" and then we listen to its remix, it is not that our brain suddenly recognizes the remix as "*Mozart – Symphony 40*".

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, ..., E_i such that every perception p of E is associated with a set of perceptions p_1, p_2, ..., 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 in following way:

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.

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 particular car which is driving 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 to have an easy way to imagine this. But at the end, this particular car represents exactly only a single entity.

Fig. 7: Representation of entity conservation over time

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

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

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

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 ***X*** would also change. Under the assumption of corollary 3, we postulate that every observation ***X*** to the damage class ***Y*** belongs to 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. *Should a certain probability of damage class $P(\boldsymbol{Y_p})$ from every possible observation $\boldsymbol{X_p}$ shares the same invariant estimate with probability of damage class $P(\boldsymbol{Y_q})$ of every possible observation $\boldsymbol{X_q}$, 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 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 the $\boldsymbol{X_\psi}$. For brevity, we focus on the CNN as this has been used in our previous work. In general, there are two kind of layers that are normally used in the core architecture of CNN: the convolutional layer and the pooling layer, as depicted

in Fig. 8 [Weimer et al. (2016)]. The core CNN is normally attached to the fully connected (FC)-layer, which is also called dense layer. Should this dense layer consist of multiple layers, then these are simply called multilayer perceptron (MLP).

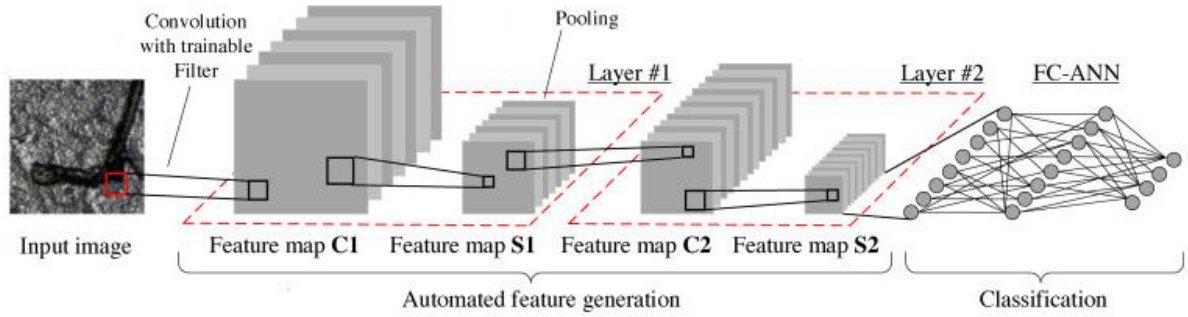


Fig. 8: Typical example architecture of convolutional neural network (ConvNet) [Weimer (2016)]

The convolutional layer is made of filter a (or kernel) which performs Hadamard transformation to the tensor that it receives from 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.

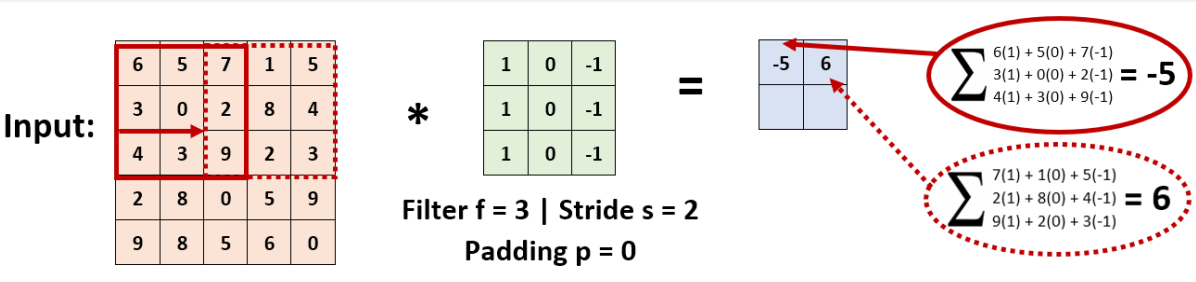


Fig. 9: Convolution operation in CNN using stride $s = 2$ and zero-padding [Ewald et al. (2018)]

2.5. Model-Assisted SHM by Active Lamb Wave Propagation

Although the efforts need to be taken to obtain a GW data from an experiment is less time consuming, a process of fabricating a test component with various damage configurations is highly expensive and time consuming, in most of the cases a parametric study would be impossible to perform through experimental methods. So, it is needless to mention that a GW model is an essential requirement for such studies. However, building a numerical model of GW is complicated as most of the engineering applications are driven by complex systems and are governed by complicated multivariate Partial Differential Equations (PDEs). The 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) [Ostachowitz et al. (2016)]. 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 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.

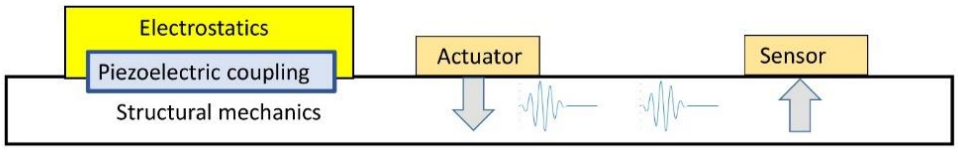


Fig. 10: Finite Element multi-physics model using COMSOL-Multiphysics

In an early study, a 2D-FEM model was analyzed by Nieuwenhuis et al. (2005) regarding the generation and detection of guided waves using PZT. Similarly, Zennaro et al. (2017) 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 as follows [Giurgutiu (2014)]:

$$S_{ij} = S_{ijkl}^{E}\sigma_{kl} + d_{kij}E_k + \delta_{ij}\alpha_i^{E}T \quad (6)$$

$$D_i = d_{ikl}\sigma_{kl} + \xi_{ik}^{\sigma}E_k + \tilde{D}_i T \quad (7)$$

where S_{ijkl}^{E} 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_{ik}^{σ} denotes dielectric permittivity at zero mechanical stress ($\sigma = 0$). T is a temperature term and α_i is the coefficient of thermal expansion. $\tilde{D}_i$ represents the electric displacement temperature coefficient and δ_{ij} is the Kronecker delta. Eq. (6) and (7) are called the piezoelectric constitutive equations that are used to predict how much strain and electrical displacement will be created at given stress, electric field, and temperature. Eq. (6) is the actuation equation, but Eq. (7) can be represented in the following form to predict the electric field for a given applied stress:

$$E_i = g_{ikl}\sigma_{kl} + \beta_{ik}^{\sigma}D_k + \tilde{E}_i T \quad (8)$$

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 $\tilde{E}_i$ is the pyroelectric voltage coefficient that represents how much electrical field is induced per unit temperature change and β_{ik}^{σ} 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 the voltage is applied to this piezoelectric material attached onto the structure's 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 piezoceramics materials that is being widely used as piezoelectric materials 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 [Giurgutiu (2014)]. In this study, Aluminum 2024 (Al-2024) is used which is a commonly used material in aerospace structure for example in fuselage area. Since the energy of the guided lamb waves are focused inside the plate, it can be used to monitor greater distance. The Lamb waves are dispersive in nature and analytically the dispersion phenomenon can be expressed in terms of wavenumber vs frequency relation [Rose (1999)] from which relationship such as Phase velocity vs frequency, group velocity vs frequency etc., as shown in Fig. 11 can be derived.

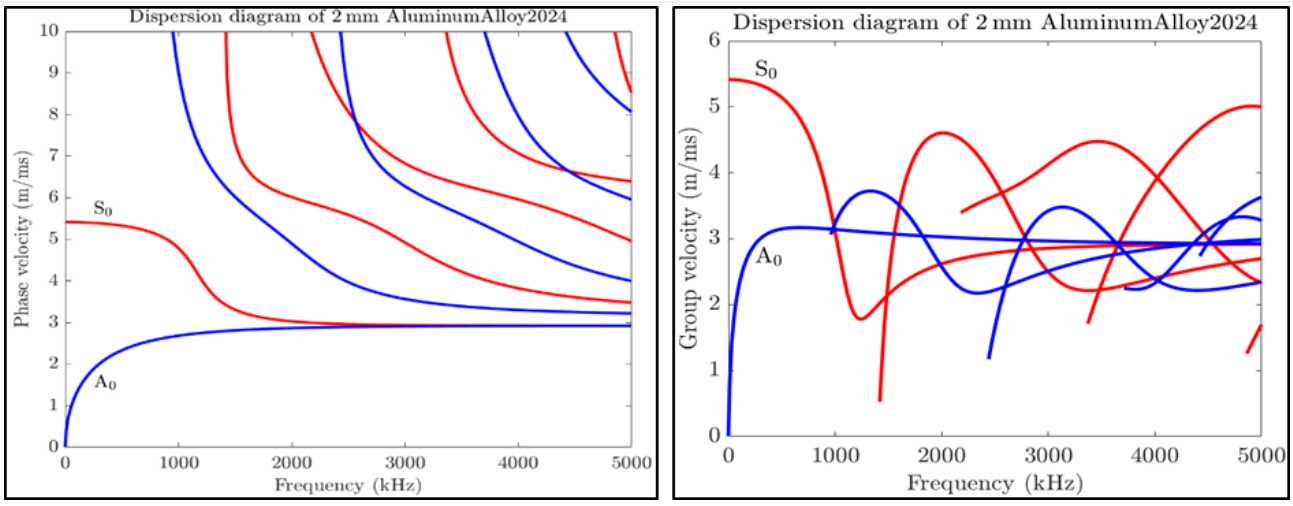


Fig. 11: Dispersion diagram for Aluminum 2024. Red: Symmetric modes Blue: Anti-symmetric modes

The Lamb waves in a plate can be generated in the plate by using PZT element which is a piezoelectric wafer and works by piezoelectricity phenomenon. The advantage of using PZT element is that it is small and it convenient to be integrated as a part of the structure.

3. Methodology

As a study case for the preliminary concept DeepSHM, active guided Lamb wave SHM that has been previously employed in our previous works are used. Section 3 of our paper is organized as follow: the sensor placement method of the DeepSHM framework for guided Lamb wave is described in section 3.1, while method to generate and multiply the data in 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 is described in section 3.3 and 3.4, respectively.

3.1. Sensor Placement Method for SHM

To determine the hotspot location, we have developed a blob detection algorithm in our previous work [Ewald (2018)] and this result was reproduced. 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). 36|20 cm and 2). 51|20 cm. To compare the DeepSHM metrics, we can assign another randomly selected location, e.g. 5|17 cm; 51|28 cm, etc.

3.2. Data Generation using FEM

In order to generate the Lamb waves using piezoelectric effect in the numerical model, physics such as solid mechanics and electrostatics must be coupled to produce piezoelectricity and it was achieved using finite element software called COMSOL Multiphysics. The guided medium is the Al-2024 plate with dimensions 600 x 400 x 2 mm and the 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 $\mu = 0.33$ and density $\rho = 2780$ kg/m^3. The material for piezo transducer was PZT-5A and due to the orthotropic behavior, its material properties are defined in matrices.

The elastic matrix values as follows 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 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 $\varepsilon_{33} = 826.6$.. The excitation pulse is a chirped Gaussian pulse with central frequency of 310 kHz and bandwidth of 100 to 500 kHz. A sampling frequency of 10 MS/s was used to measure the signal. Such kind of a numerical model has been already validated by time of arrival method and previous experiment [Taltavull (2017)].

The simulation was modeled without considering external environmental influences, and human factors, which results in identical output signal every time the simulation is repeated. Hence, as a data augmentation technique, a random white Gaussian noise is added to the sensor signals to make it look like a signal obtained from a sensor in an experiment. An example of this method is demonstrated with a sensor signal obtained from the simulation in the Fig. 12 and from that we considered signal SNR of 15 to be very bad but SNR 25 and resembles close to reality to a non-averaged signal measured from an experiment.

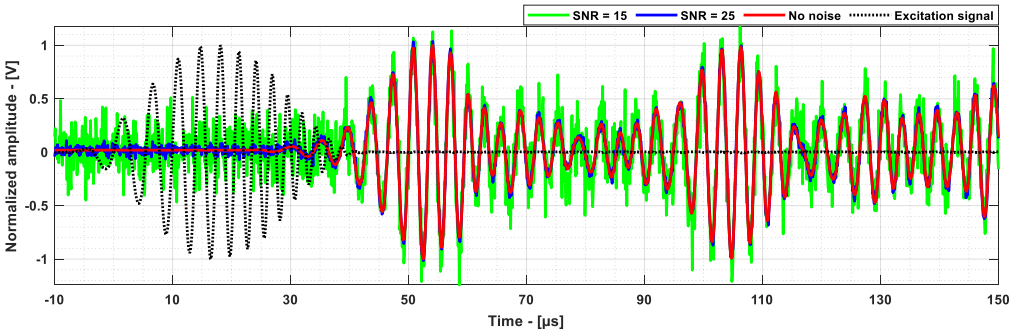


Fig. 12: Example of simulated signal without noise, with SNR = 15 (green) and SNR = 25 (blue)

3.3. Data Pre-Processing

The data from the sensors can be transformed from 1D representation (time domain signal) into 2D Time-Frequency representation (TFR) and since Lamb waves are dispersive waves, signals from each crack condition will result in distinct images which then can be fed into CNN. There are quite a few methods that can be used to perform TFR for Lamb waves. 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. For this study, the reassignment method implemented on Short Time Fourier Transform (STFT) [Fedotenkova (2016)] was used. An example of the transformation is shown in the Fig. 13.

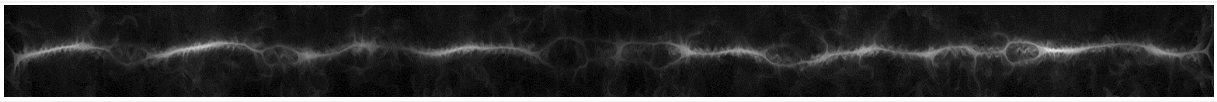


Fig. 13: Example of reassigned STFT

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 is can be broken down into smaller sub-entities, we randomly sampled the signal over certain time window ***W*** and then pre-processed the randomly sampled sequence by the method described in section 3.3. An example of randomly sampled pre-processed signals from sensor that is located at 51|28 cm in baseline aluminum plate are depicted in Fig. 14a – e, where in these figures, ***W*** is varied between 20 and 320 µs. Applying smaller window length would mean imply the probability that certain features occurs in other classes increases, thus leads 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 displacement.

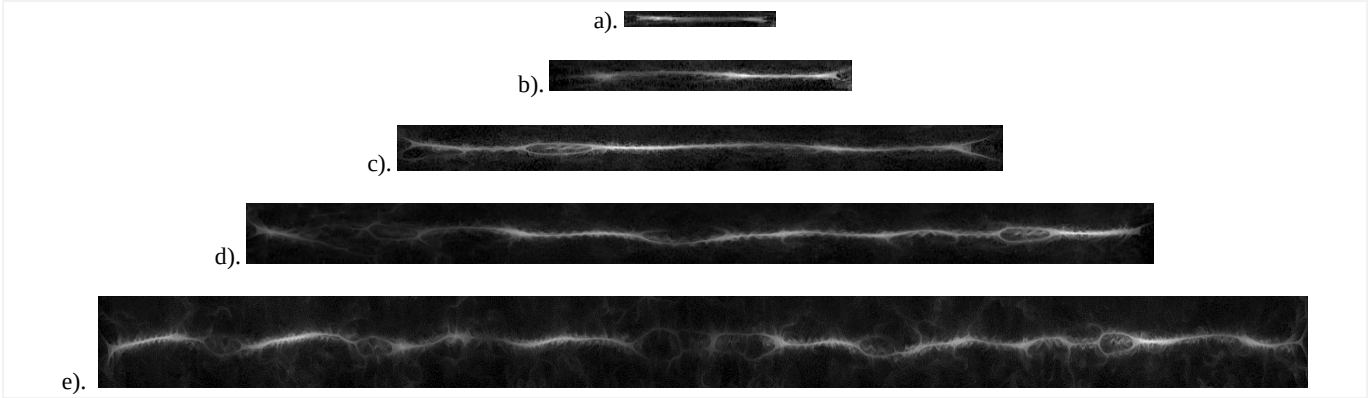


Fig. 14: Spectrogram from randomly sampled signals from sensor that is located at 51|28 cm over convolution window length *W* of a). 20 µs, b). 40 µs, c). 80 µs, d). 160 µs, and e). 320 µs.

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 section 3.3 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 SHM would typically only have small number of observation sampling (e.g. 10 sensors for a monitoring area of 1 m²).

For ultrasonic based SHM, it might be possible in the future to combine an actuator PZT and moving phased array (PA)-Probe as 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. 15.

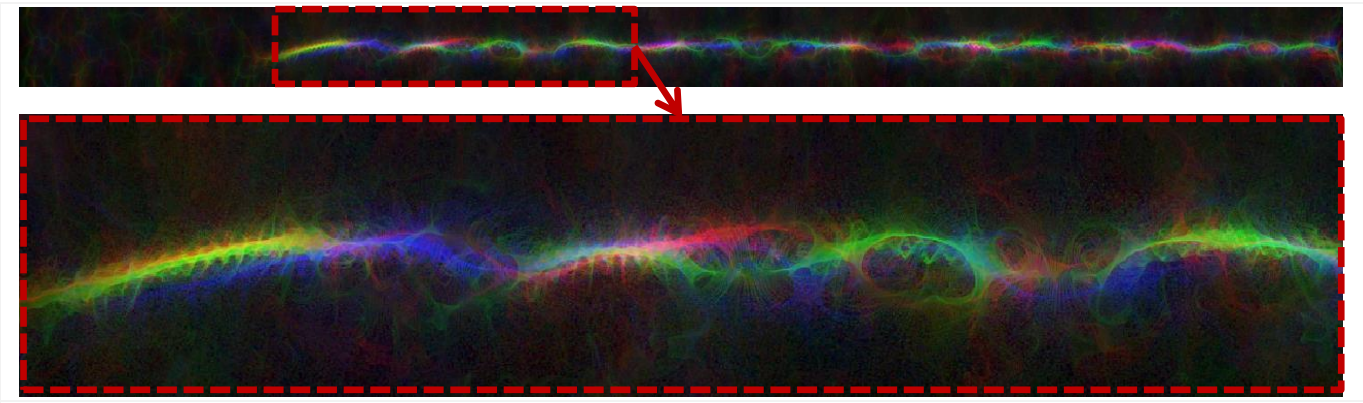


Fig. 15: Feature representation of merged input signal from 3 sensors channeled in RGB array

For more than 4 sensors, the data cannot be visualized without reducing information anymore – but it can be stacked in *k*-dimensional image array. Applying larger window length means that the probability that certain features occurs in other classes decreases, thus equivalent to lower invariant estimate. As for data augmentation, the invariant transformation would be shifting the features to the left or right to represent small sensor displacement but also changing the channel (e.g. here red channel = sensor 1, green channel = sensor 2, blue channel = sensor 3, etc. – this is equivalent to red channel = sensor 2, green channel = sensor 1, blue channel = sensor 3). The reason for this channel equivalence is because the feature representations of each channel will be summed together before being passed through the next layer.

3.4. Training Deep Neural Network

3.4.1. Hardware

As for hardware, the standard PC specification at TU Delft is: Dell Precision T5810 running on Intel Xeon(R) E5-1620 3.5 GHz and 32 GB DDR-RAM running on Windows 7. The only exception we had is that we had our PC performance boosted with NVidia GPU GeForce GTX1080Ti which at the time we purchased (June 2018) had highest performance on end-user market. Note that, currently there are more powerful GPU for large-scale exploitation such as NVidia Tesla or even cloud processing center such Microsoft Azure or Amazon AWS which is available for manifold of price that we paid for the GTX1080Ti. The other alternative on training deep neural network would be on Field Programmable Gate Array (FPGA) and it promises even more faster processing – however, at the time our work was conducted in mid-2019 there was no mass-available commercial FPGA framework.

3.4.2. Software and Libraries

Software-wise, we used TensorFlow [TensorFlow website] (developed by Google Brain) and Keras API [Keras website] (both are available on Python) as currently these are the richest library available on market. Note that, recently, there is 3rd option which is called PyTorch [PyTorch website] (developed by 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 10.0) must be installed so that the API can interact with the graphic card.

3.4.3. Optimization

Since training a neural network is NP-complete and generally the 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 gradient-based method. Involving second-order methods (i.e. Hessians

matrix) would typically solve the problem faster than gradient-method, however as there is no such thing as free lunch, 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 [Morales (2002)], Gauß-Newton [Shalev-Shwartz and Ben-David (2014)], (steepest) gradient descent [Ruder (2016)], and Levenberg-Marquardt [Zayani et al. (2008)].Like any other libraries, currently there are only the gradient descent methods 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 as described in Eq. (9).

$$\delta_{t+1} = \gamma \cdot \delta_t + \eta \cdot \left[\nabla_\theta J(\theta - \gamma \cdot \delta_t)\right]$$
$$\theta_{t+1} = \theta_t - \delta_{t+1} \qquad (9)$$

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 [Srivastava et al. (2014)]. There are more optimizers options 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 [Keras documentation, Géron (2017)].

3.5. CNN Architectures & Test Plan

Due to space and time constraints in this paper, we restrict the result presentation to only CNN and MLP architectures for each 3 different classification scenarios. Thus, for this paper we present the result as follows: in section 3.5.1 we briefly explain several different CNN architectures we had, and the training methods employed. Section 3.5.2 we presented 3 different classification scenarios under which each CNN architectures were trained.

3.5.1. Network Architectures and Training Methods

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 [Ewald (2018)], as described in Table 2. We are aware that there are more sophisticated CNN architectures such as VGG-16 [Simonyan (2014)], inception layers [Szegedy (2014)], or ResNet [He (2015)] that can be employed in the future work, but for the sake of simple demonstration purpose in this paper we do not go further beyond 8 hidden layers. This number can be adjusted according to the complexity of input features, as the main purpose of this paper is not to optimize the CNN architecture being used. 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].

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 2. 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

For the training purpose, the data should be normalized between 0 and 1 so that 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 Hinton (2014), 0.5 is the best rate found, which means 50% of the neurons at that particular layer being deactivated. The default parameter for the optimizers as described in Table 3.

SGD	η = 0.01, γ = 0.0, η_{decay} = 0.0, NESTEROV = FALSE
Adam	η = 0.001, β_1 = 0.9, β_2 = 0.999, $\varepsilon = 10^{-8}$, η_{decay} = 0.0, AMSGrad = FALSE

Table 3. 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.

3.5.2. Classification scenarios

Due space constraint, we will only demonstrate 3 scenarios under which the CNN architectures were trained. Assume that we have dataset from following half-crack length (including initial notch) measured from the center of the rivet as depicted in Fig. 16: 0 mm (pristine plate) up to 200 mm length that is assumed to be the critical crack length a_{crit}. The crack length 2a in percentage as proportion from 200 mm. We presented the classification scenarios in Table 4.

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

Table 4: Classification scenarios with different percentage of crack length with reference to a_{crit}

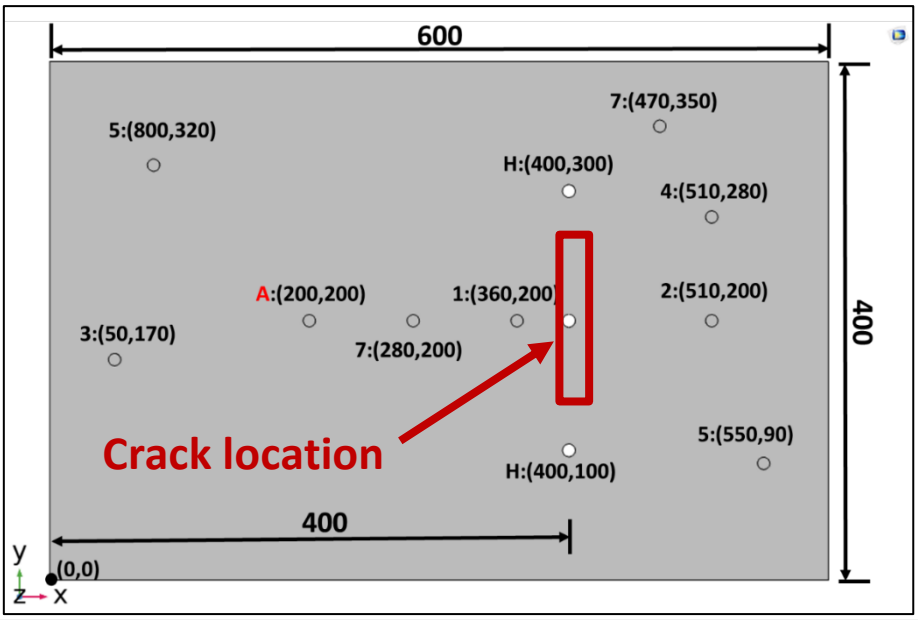


Fig. 16: Sensor positions on the plate (dimension stated in mm)

In scenario 1, 4 sensors data are available, while in scenario 2, 8 sensors data are available. The depiction of the sensor location from scenario 1 and 2 is given in Fig. 16. The combined sensor data from scenario 1 and 2 are made into scenario 3, although due to the lack of the data of the other 4 sensors, we could only merge the 4 sensors data with the same location from scenario 1 and 2.

4. Results and Discussion

The training results for given parameter in section 3 is presented in this section. We organized this section as following: in section 4.1, the results for modelling DeepSHM as a multiple acting decision maker are presented, where in this section we will describe the training results trained under both SGD and Adam optimizer for different sensing locations in each given damage case scenario for variable window length according to section 3.5.2. In section 4.2, the results for modelling DeepSHM as single acting decision maker 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 the 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 means that we always have enough space to store all the model created, thus we limit our training samples only for 100 samples to see the network behavior. During the trial and error phase, we actually tried with 1000 samples first, but as it took too long for preprocessing (about 2 days per dataset) such large amount of data – therefore we cut this amount down to 100. 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 larger. Normally, in top computer vision conferences such as CVPR, the models they are training are using CIFAR-10 or MNIST dataset that has an input size of 32 x 32 pixel, while in our case, the this input size vary between 251 x 7 pixel up to 4101 x 247 pixel, 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 spatiotemporal data, we first presence the training failure of MLP handling the training set. To do so, we trained the MLP from for all damage scenarios given by section 3.5.2. In order to save space, we only show the training results from the presumably best sensor location, namely sensor 2, depicted in Fig. 17a – f.

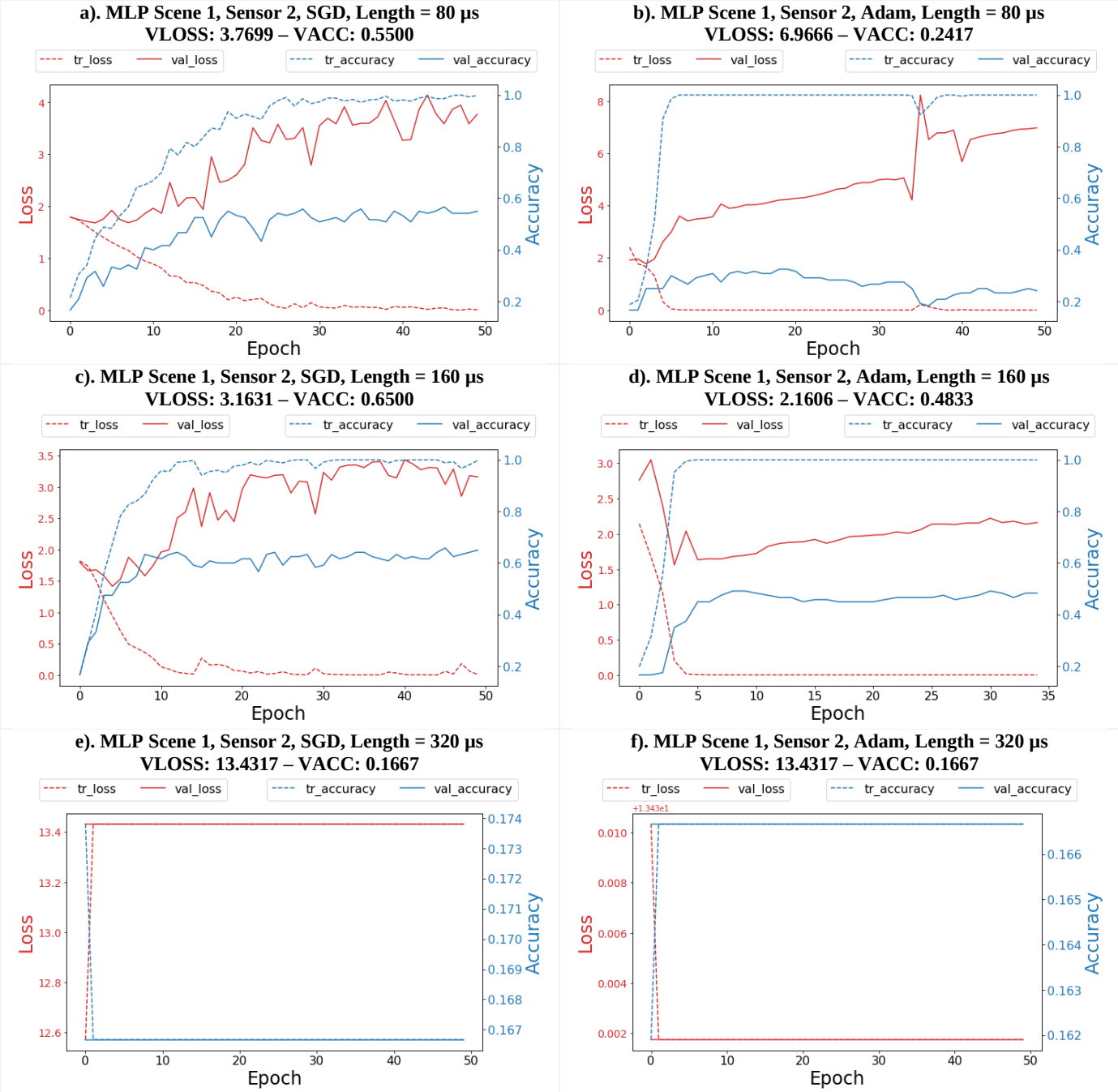
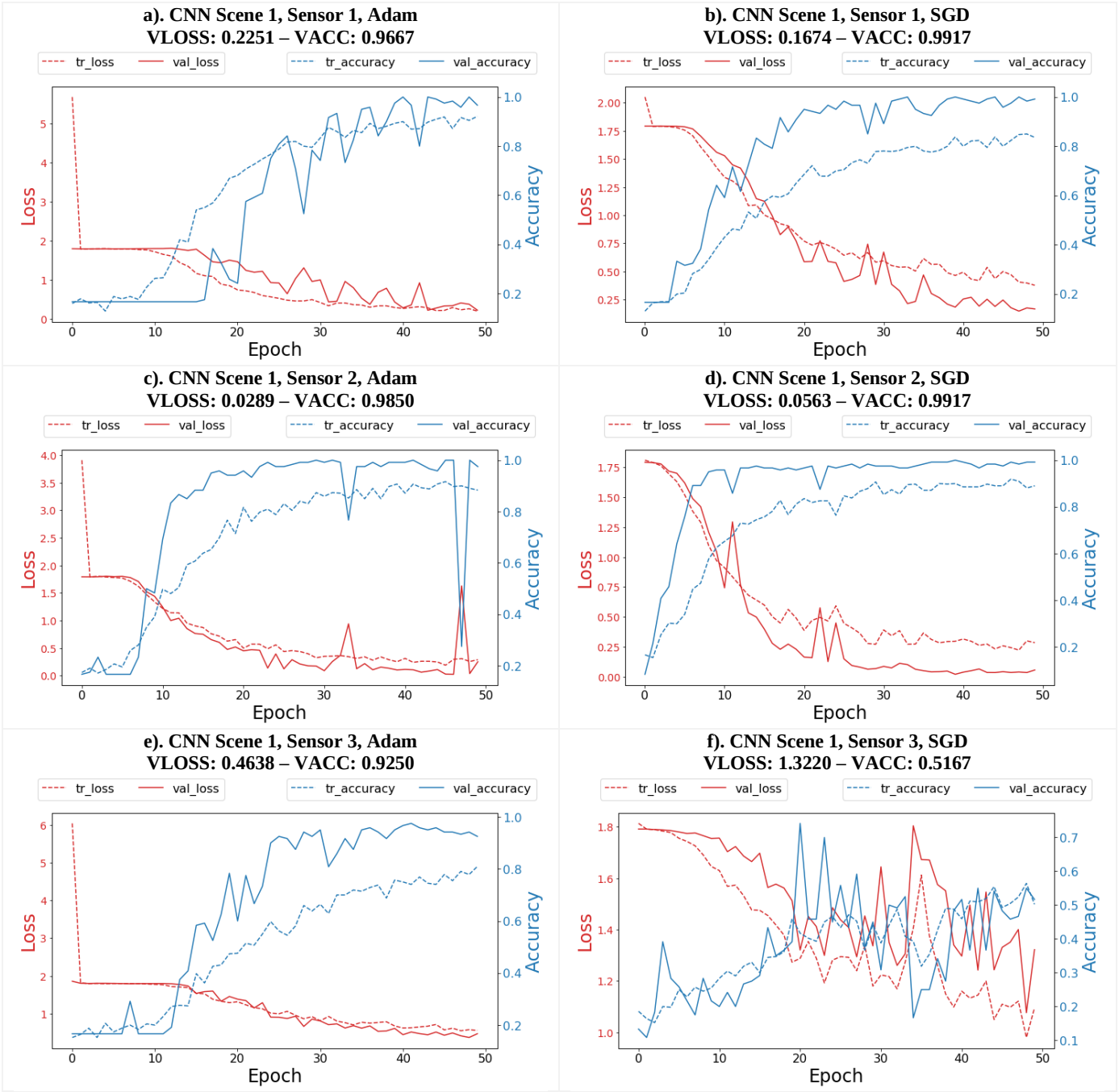


Fig. 17a – f: Training results of MLP for signal from sensor 2 with various window length trained under Adam and SGD optimizer up to 50 epochs. VACC = Final validation accuracy, VLOSS = Final Validation Loss

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 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. From Fig. 17a – b, for a window length of 80 µs, the MLP model quickly overfits regardless of the optimizer used. While the training accuracy reaches almost 1.0 (or 100%) in Fig. 17a – b, 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 once 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 more than 35%. 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 spatiotemporal feature such as depicted in Fig. 15.

Moving to CNN architecture, rather than trying a complex CNN such as VGG, AlexNet, or InceptionNet, we kept our CNN very simple as it only contains 3 hidden convolutional layers that are attached to the same MLP we used previously. The training results from CNN trained both under Adam and SGD optimizer for damage scenario 1 are depicted in Fig. 18a – h, where 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 sensor 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 optimized 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.



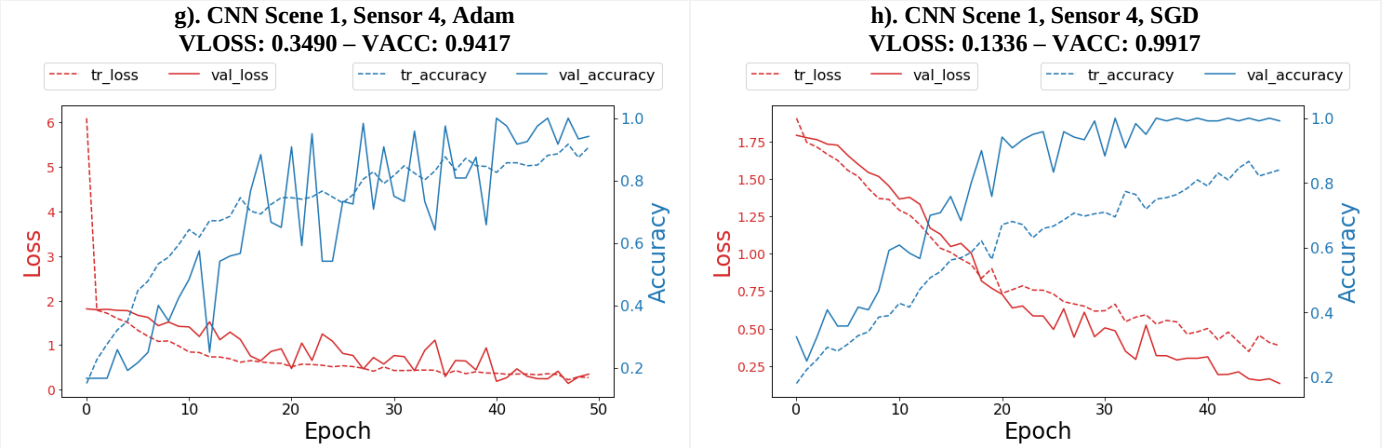


Fig. 18a – h: Training results of CNN for signal from sensor 1 – 4 in scene 1 with window length of 320 µs trained under Adam and SGD optimizer up to 50 epochs. VACC = Final validation accuracy, VLOSS = Final Validation Loss

As 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. Further training results are available on demand – also, we decided to make our source code publicly available so that it can be downloaded and further improved for future research. We would like to remind that in scenario 2, there were 8 sensors available, and the sensor location are given in Fig. 16. In this scenario, the CNN is trained to classify less distinguishable signals from each class as per Table 4. Thus, the results (depicted in Fig. 19a – d) met our expectation: the validation accuracy is 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 location.

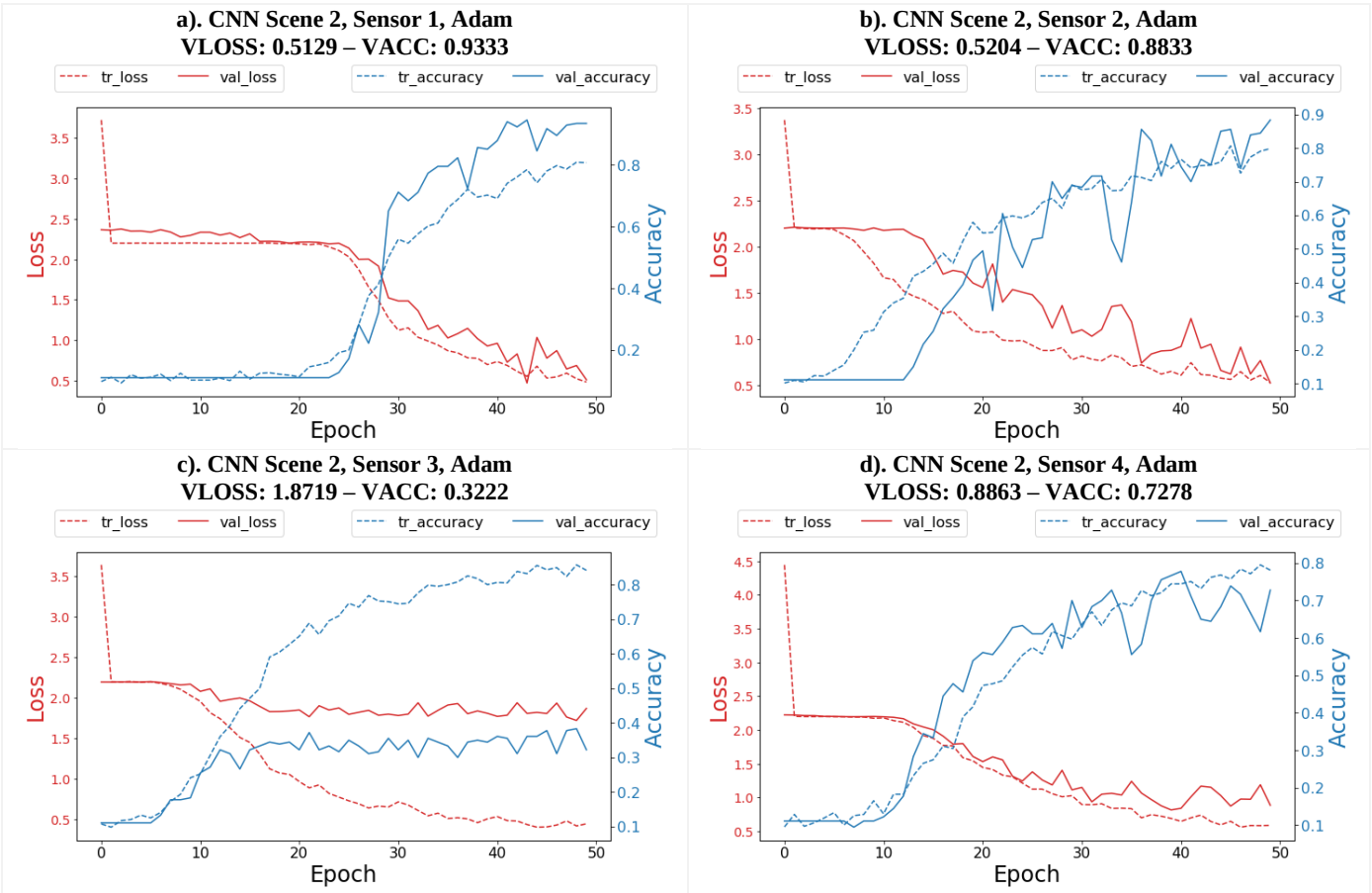


Fig. 19a – d: Training results of CNN for signal from sensor 1 – 4 in scene 2 with window length of 320 µs trained under Adam optimizer up to 50 epochs. VACC = Final validation accuracy, VLOSS = Final Validation Loss

Moving forward to scenario 3, which is just the combination of scenario 1 and 2. Note that in this case, only the data from 4 sensors in scenario 2 which are located in the same location as in scenario 1 can be added to the training set. As scenario 3 is the combination of scenario 1 (very distinctive classification ranging from 0% – 100% length of critical crack in each 20% step) and scenario 2 (less distinguishable signal between each classes), the training results are slightly better than

scenario 2, but worse than scene 1. Also, the importance of sensor positioning is also now clearly highlighted. The training results are depicted in Fig. 20a – d.

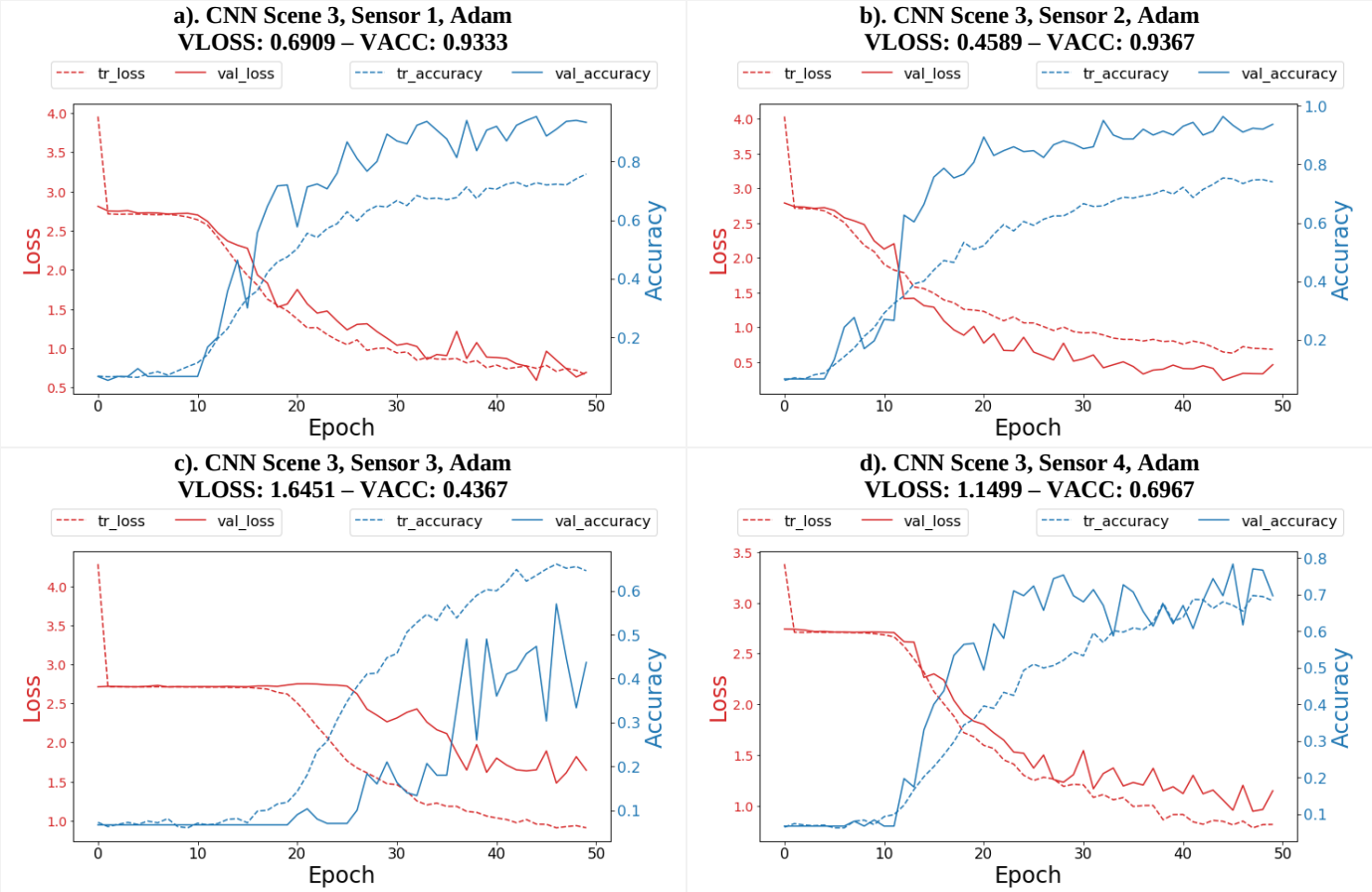


Fig. 20a – d: Training results of CNN for signal from sensor 1 – 4 in scene 3 with window length of 320 µs trained under Adam optimizer up to 50 epochs. VACC = Final validation accuracy, VLOSS = Final Validation Loss

As previously described in section 3.1, sensor 1 and 2 are the best sensing locations and thus, the validation training accuracy reached was more than 0.9 or 90% (Fig. 20a – b), while sensor 3, which is located very far away from the crack location could only reached a validation accuracy of 43.67% (Fig. 20c). This is far than surprising, because we expect that at the sensing location occupied by the sensor 3, the accumulated energy from the scattered wave from the crack is far less than those 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. From all these experiments, it can be concluded easily 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 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 (2018)] can be used because a sensor placement strategy is still needed even for sophisticated machine learning techniques 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 both in time and money associated with that.

4.1.2. Effect of the Length of Convolution Window

In this scenario, we describe the influence of the 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 from the signal as previously explained in section 3.4.1. All the scenarios are trained with Adam optimizer and we limit the training only up to 50 epochs and the results for all scenes are summarized in Table 5 and Fig. 21.

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	0.8944	190	0.9467

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

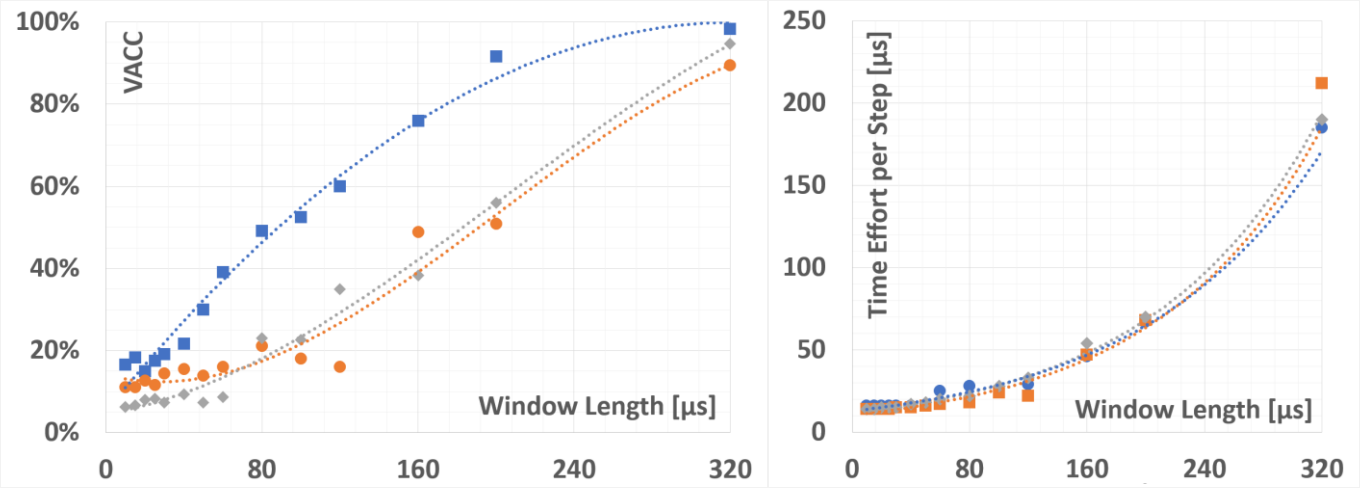


Fig. 21: left: The relation between final validation accuracy (VACC) and window length in [µs]; right: The relation between time effort needed per training step and window length in [µs]

4.2. On Modelling DeepSHM as Conserved Entity over Time

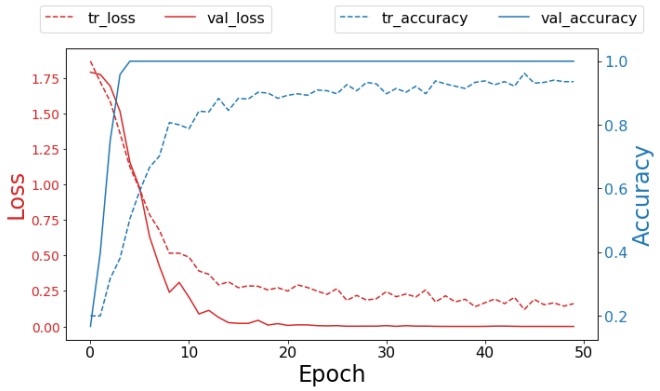
As has been discussed in section 3.4, we are now interested to see the training behavior when the perception is modelled as a single entity conserved over time. The consequence for this assumption: the training set consists of a *k*-dimensional data cube composed by multi-layer full-length time-frequency signal representation, e.g. see Fig. 15 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-layer as it can easily be trained as an image and the library we used (Keras and Tensorflow), supports data flow from folder for image training. Most of the ready-to-use deep learning libraries only support 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 more than 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 [Paoletti, Audebert]. Unfortunately, those works are in general very niche and rarely exploited in machine and deep learning community and many of the codes are not made publicly accessible and thus if we are to start to customize the library, it would take years until the code is ready and in academia, this is not the favorable situation for faculty dean.

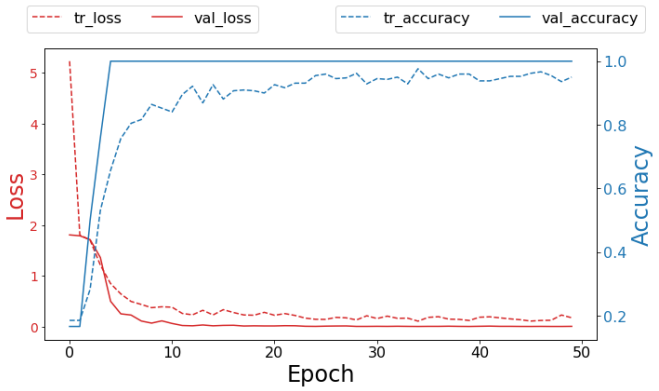
As in our case, we use a combination of several sensor responses to create the data. As there are only 4 sensors in scenario 1, we were able to show all possible combination of the training behavior of the sensor response (1;2;3, 1;3;4, 1;2;4, and 2;3;4) and these are depicted in Fig. 22a – d. However, as there are more 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 upon request, 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.

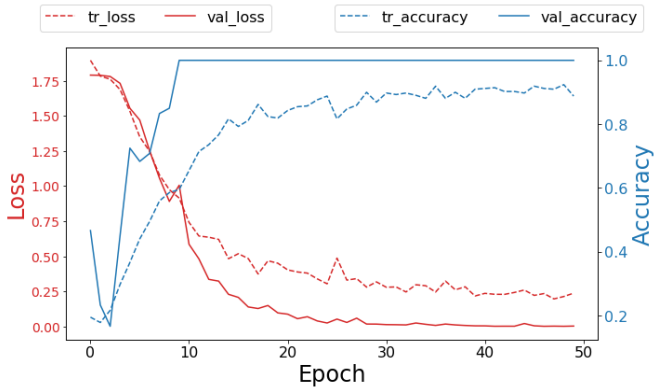
a). CNN Scene 1, Sensor 1;2;3, SGD
VLOSS: 1.5293e-04 – VACC: 1.0000



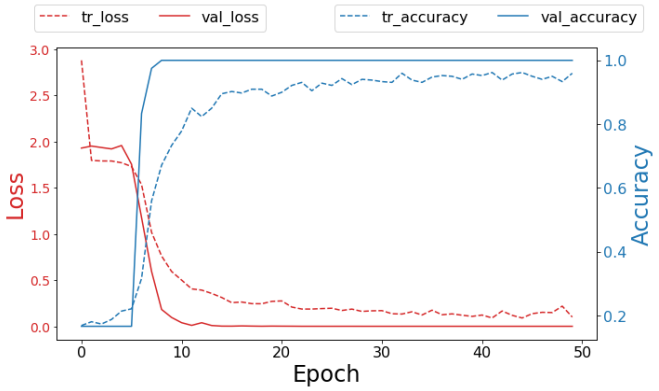
b). CNN Scene 1, Sensor 1;2;3, Adam
VLOSS: 0.0021 – VACC: 1.0000



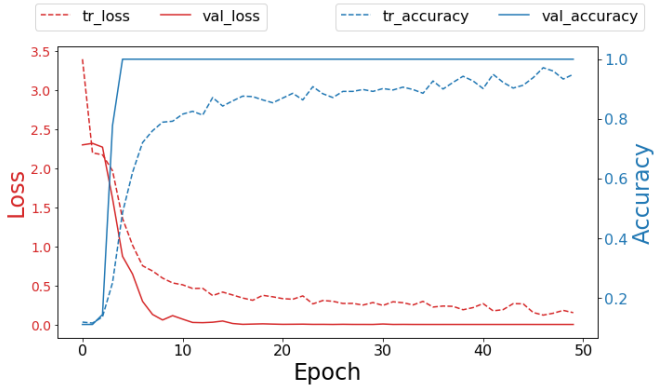
c). CNN Scene 1, Sensor 1;2;4, SGD
VLOSS: 0.0031 – VACC: 1.0000



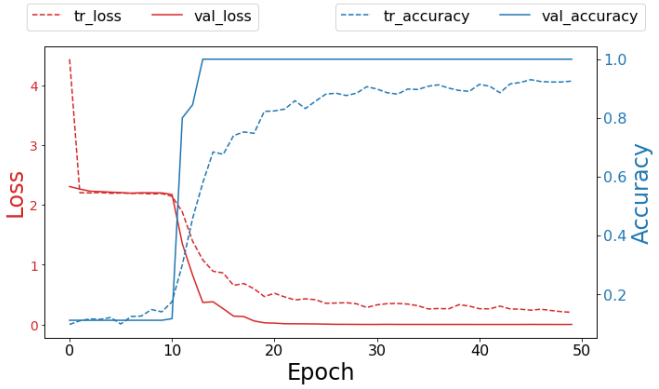
d). CNN Scene 1, Sensor 1;2;4, Adam
VLOSS: 1.1921e-07 – VACC: 1.0000



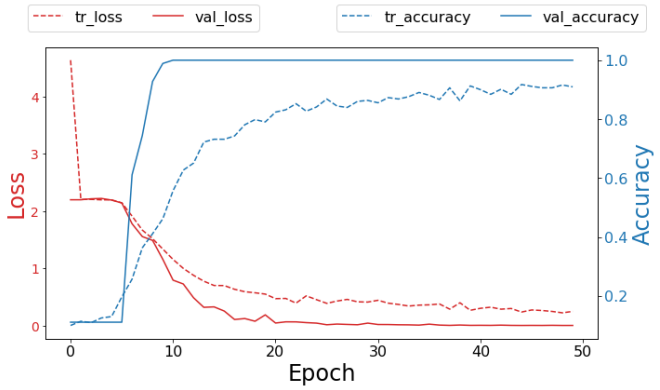
e). CNN Scene 2, Sensor 2;4;6, Adam
VLOSS: 2.1336e-06 – VACC: 1.0000



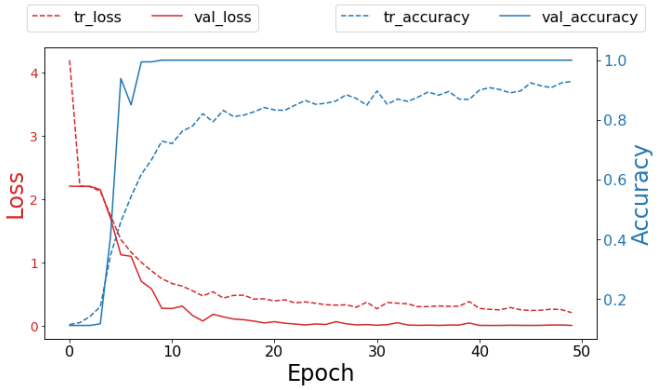
f). CNN Scene 2, Sensor 2;6;8, Adam
VLOSS: 3.3180e-05 – VACC: 1.0000



g). CNN Scene 2, Sensor 3;5;7, Adam
VLOSS: 1.9950e-04 – VACC: 1.0000

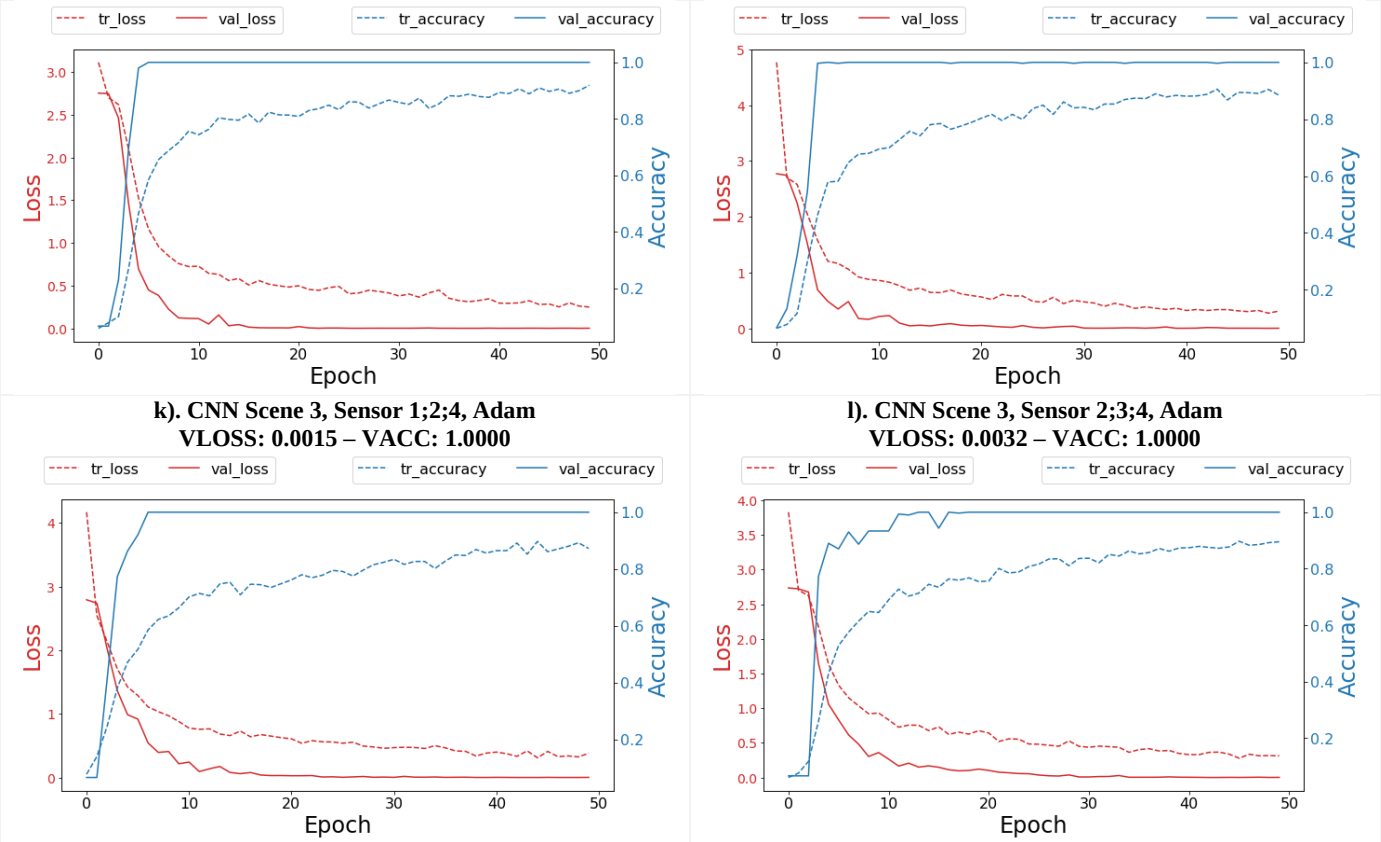


h). CNN Scene 2, Sensor 3;6;8, Adam
VLOSS: 9.7399e-04 – VACC: 1.0000



i). CNN Scene 3, Sensor 1;2;3, Adam
VLOSS: 9.6724e-04 – VACC: 1.0000

j). CNN Scene 1, Sensor 1;3;4, Adam
VLOSS: 0.0028 – VACC: 1.0000



k). CNN Scene 3, Sensor 1;2;4, Adam
VLOSS: 0.0015 – VACC: 1.0000

l). CNN Scene 3, Sensor 2;3;4, Adam
VLOSS: 0.0032 – VACC: 1.0000

Fig. 22a – 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

From all sub-figures in Fig. 22, it can be clearly concluded when modelling the SHM perception as single conserved entity (we referred this as a ***single person with multiple ears*** previously) has a quick converging training rather than modelling the perception as hierarchically ordered but separate entities (cf. Fig. 18 – 20, we referred this as ***multiple half-deaf person in a room*** in previous section). For the first case, the sensing location 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-spatiotemporal features within certain region of interest in the data cube. We assumed the existence of the correlated 3D-spatiotemporal features to be the evidence for the theory of invariant latent proposed in Lemma 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 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 single continuum can be regarded as harmonic motion and thus do not fall into a sudden singularity, so we assume there must be spatiotemporal features that are locally connected between each layer in Fig. 15. We are aware that this assumption is of course not valid for randomly occurring singularities within continuum – but we do never expect sudden singularities within acoustic wave signals either. We are aware that it is very difficult to understand what is happening inside the deep neural network, we postulate that these local spatiotemporal features, while very deeply embedded in the CNN model, they are very distinct to each other for every different damage classes, since every different damage classes would produce different responses (signal) at each different sensing location and therefore making the complex neural network easily capture these thanks to Rademacher complexity in Def. 3 and learned it completely after several training epochs, making it as a special distinctive signatures of each probability classes as supported by the Lemma 5.

In general, it is common to use 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 an evidence to try out the less exploited and more “adventurous” chirplet-alike excitation signal for Lamb wave SHM. We believe our training results suggests that these distinctive spatiotemporal features within the representation

that are heavily more 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 spatiotemporal information – as per Lemma 5. As this is our speculation, we think that there would be an opportunity to quantitatively investigate this matter in more rigorous research.

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 – 15 from previous and pure Gaussian white noise. An example of time-frequency domain representation of pure Gaussian white noise is depicted in Fig. 23. The noise was trained under exactly the same training parameters as before and the results are given below. We repeated the white noise training several times and give two sample results in Fig. 24a – b which depicts the training behavior for white noise signal with convolutional window length of 320 µs.

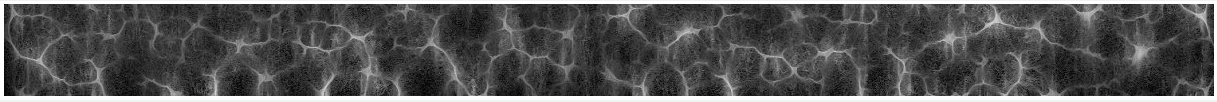


Fig. 23: White Gaussian noise with length of 320 µs

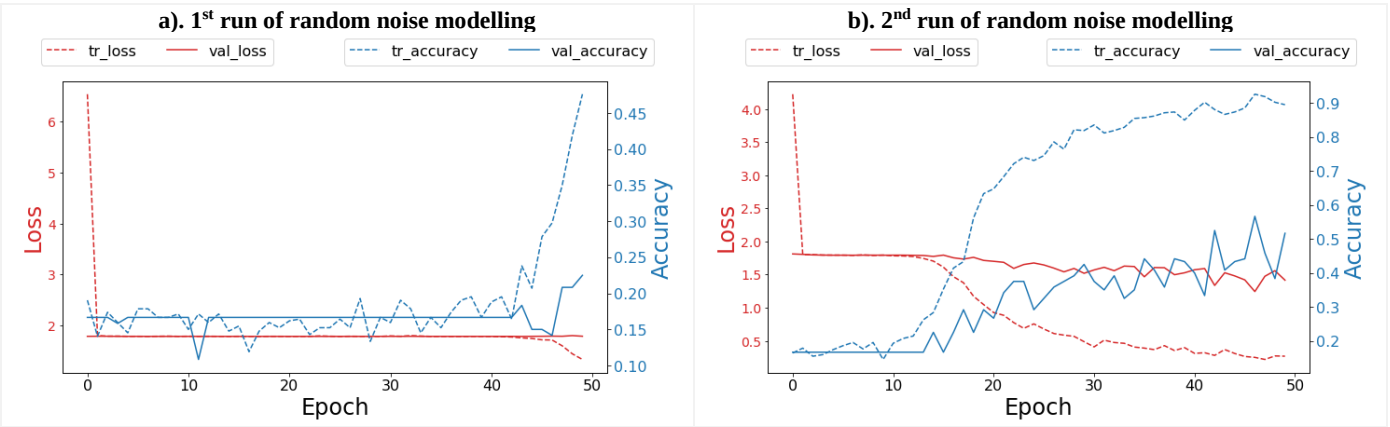


Fig. 24a – b: Sample training results of random noise modelling

As it can be seen from all figures in Fig. 24a – b, the networks failed to reach validation accuracy of more than 80%. Furthermore, the training and validation accuracy is relatively far (between 20% and 40%), which is clearly signification of heavy network overfitting, but this still is not the most important fact. It is more important to know is that they failed to deliver a consistent result. This clearly signifies that the network did not learn anything from the noise because there is no locally connected spatiotemporal features in the noise – thus indicating that the invariant latent within the noise group does not hold. More figures are available upon demand or the reader are more than welcome to download the source code [Ewald (2019), 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 it can classify the signal or not, a confusion matrix must be calculated from each model. As an example of result, we show the test result from scene 1 and 2 for sensor 2 for convolutional window length 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 corresponds to the damage condition of scene 1 described in Table 4 with "0" as the baseline and "5" as the critical crack length. 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}.

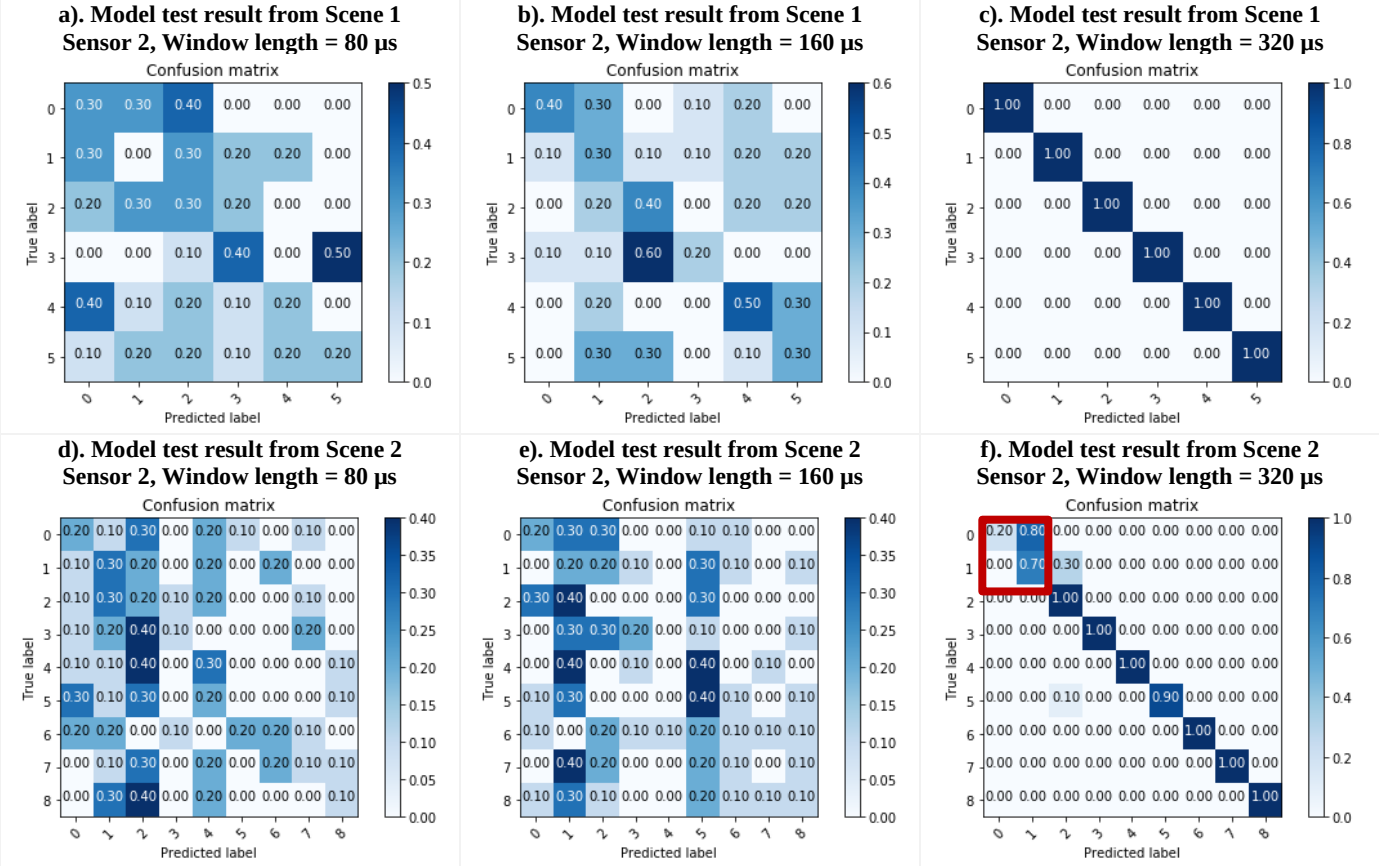


Fig. 25a – f: Model test result from scene 1 and 2 for sensor 2 with varying window length between 80 and 320 µs.

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 0°-oriented crack of 10% a_{crit}, where "1" is 15°-oriented crack of 10% a_{crit}. This is to be expected because for two small cracks with slightly angled orientation, 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.

4.3.2.2. Representation as Conserved Entity over Time

When modelling the feature representation in a single conserved entity (which we previously referred as ***a person with multiple ears*** in previous section), 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 analogue to our explanation in section 4.3.2.1.

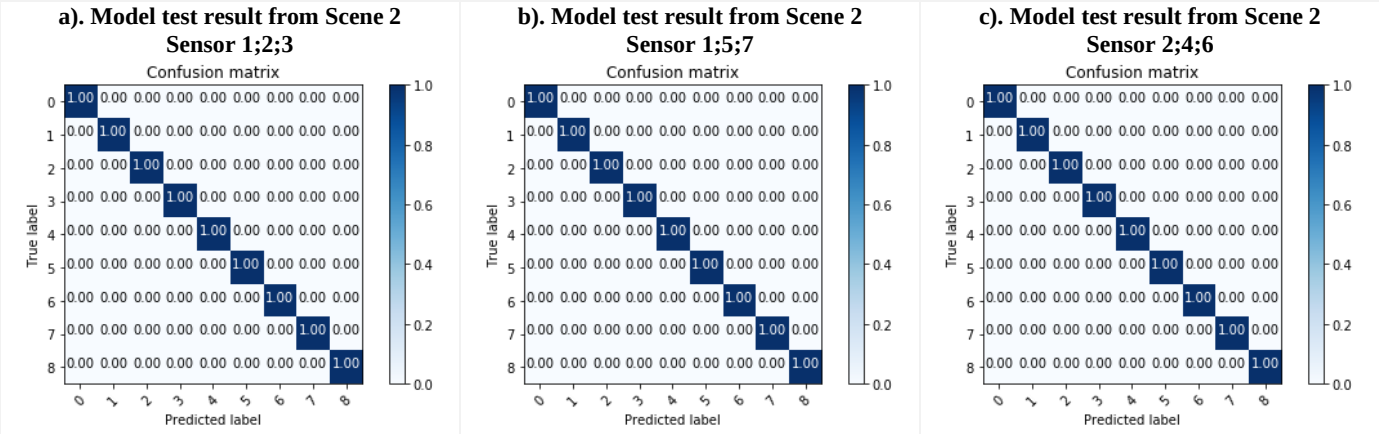


Fig. 26a – c: Model test results from scene 2 for conserved entity representation that fused different sensor responses.

While more test results are available upon request, 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. We hypothesize that it learned the locally spatiotemporal features within the entity in a more subtle and concise way as there are more information embedded in this 'data cube'.

5. Concluding Remarks

5.1. Summary

This paper is an extension to our previous work [Ewald (2019)], 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 neuroscientific perspective, while the methodology used such as the numerical model, data pre-processing, and neural network training are described in section 3.1 – 3.4. The results and discussion are given by section 4.

5.2. Recommendation

Before making any recommendation, let us review 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 from 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-alike excitation, nevertheless this approach encounters its limit as well as demonstrated in Fig. 22f. 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 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 to use the quasi chirp-alike excitation is only when it is coupled with advanced machine learning, which is in any case already a black-box. Thus, we would like to re-emphasize our principle on ***Garbage-in – Garbage-out*** as in the middle of the black box it is hardly to extract any meaningful causation. Employing 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 far 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 (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 answer, but we can state that the best sensor (or better said: highly optimally located set of sensors) give the most desired training behavior due to better response capture. However, when the information is fused first from different sensing locations, that is in the case of conserved entity representation which we previously referred as "a single person with multiple ears", 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).

In far future however, the neural network training process could be combined with quantum computing [Beer et al. (2019), Hu et al. (2019)] to accelerate the computation time. Recently, it has been tried on smaller scale hobby project [Killoran and Izaac (2018)] to combine quantum library PennyLane and PyTorch. 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.

Declaration of No Conflict Interests

The authors hereby declare no conflict of interest.

Acknowledgement

The authors would like to thank Mr. Xavier Goby for helping with coding the confusion matrix.

Literatures

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)

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)

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)

ATA Maintenance Steering Group (MSG) Task Force 3. European Aviation Safety Agency. Rev. 1 (2009).

Advisory circular AC 43-204: *Visual Inspection for Aircraft*. USDOT Federal Aviation Administration (FAA) (1997).

Mineo C, Pierce SG, Nicholson PI, Cooper I. *Robotic Path Planning for Non-Destructive Testing – A Custom MATLAB Toolbox Approach*. J Robotics and Computer-Integrated Manufacturing, Vol. 37: 1-12 (2006).

Purtill R. *The Purpose of Science*. J of Philosophy of Science, Vol. 37(2): 301-306 (1970).

Zeng Z, Zhou J, Tao N, Feng L, Zhang C, Han X. Zeng Z. *Support Vector Machines Based Defect recognition in SonicIR Using 2D Heat Diffusion Features*. J of NDT&E International, Vol.47: 116-123 (2012).

Giurgiutiu V. *Structural Health Monitoring with Piezoelectric Wafer Active Sensors*. 2nd Ed., Elsevier, Oxford & Waltham (2014).

Boller C, Mofakhami MR. *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).

Cristianini N, Shawe-Taylor J. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, Cambridge (2000).

Virupakshappa K, Oruklu E. *Ultrasonic Flaw Detection Using Support Vector Machine Classification*. Proc. IEEE Intl Ultrasonics Symp (IUS), Taipei (2015).

Schmidhuber J. *Deep Learning in Neural Networks: An Overview*. J Neural Networks, Vol. 61: 85-117 (2015).

Hopfield JJ. *Neural Networks and Physical Systems with Emergent Collective Computational Abilities*. Proc. Natl Academy of Science USA, Vol. 79(8): 2554-2558 (1982).

Lee MC, To C. *Comparison of Support Vector Machine and Back Propagation Neural Network in Evaluating the Enterprise Financial Distress*. Intl J of Artificial Intelligence & Applications, Vol. 1(3): 31-43 (2010).

Hinton GE, Osindero S, Teh YW. *A Fast Learning Algorithm for Deep Belief Nets*. J Neural Computation, Vol. 18(7): 1527-1554 (2006).

Baldi P. *Autoencoders, Unsupervised Learning, and Deep Architectures.* Proc. Intl Conf on Unsupervised and Transfer Learning Workshop, Washington, Vol. 27: 37-50 (2011).

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).

Azimi M, Pekcan G. *Structural Health Monitoring using Extremely Compressed Data through Deep Learning*. J Computer Aided Civil and Infrastructure Engineering, Vol. 12517: 1-18 (2019).

Nguyan TT, Nguyden ND, Nahavandi S. *Multi-agent Behavioral Control System Using Deep Reinforcement Learning*. J Neurocomputing. Vol. 359: 58-68 (2019).

Lavet VF, Henderson P, Islam R, Bellemare MG, Pineau J. *An Introduction to Deep Reinforcement Learning*. Now Publisher Inc, Boston & Delft (2018).

Kawaguchi K, Kaelbling LP, Bengio Y. *Generalization in Deep Learning*. Mathematics of Deep Learning, Cambridge University Press, to appear. Preprint avaliable as: MIT-CSAIL-TR-2018-014

Mohri M, Rostamizadeh A, Talwakar. *Foundations of Machine Learning*. 2nd Ed., MIT Press, Cambridge & London (2012).

Balcan MF. *Rademacher Complexity*. In Lecture series CS 8803: Machine Learning Theory, Carnegie Mellon University (2011).

Goldberg PW, Jerrum MR. *Bounding the Vapnik-Chervonenkis Dimension of Concept Classes Parameterized by Real Numbers*. J Machine Learning, Vol. 18(2-3): 131-148 (1995).

Clayton S. *Topic 10: Rademacher Complexity*. In Lecture series EECS 598: Lecture on Statistical Learning Theory. University of Michigan (2014).

Pendleton SD, Andersen H, Du X, Shen X, Meghjani M, Eng YH, Rus D, Ang MH. *Perception, Planning, Control, and Coordination for Autonomous Vehicles*. MDPI J Machines, Vol. 5(1): 1-54 (2017).

Gallicchio C, Micheli A. *Echo State Property of Deep Reservoir Computing Networks.* J Cognitive Computing, Vol. 9(3): 337-350 (2017).

Soures N, Kudithipudi D. *Deep Liquid State Machines with Neural Plasticity for Video Activity Recognition*. J Frontiers in Neuroscience. Vol. 13 (686): 1-12 (2019).

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).

Kingma DP, Welling M. *Auto-Encoding Variational Bayes*. Proc. Intl Conf on Representation Learning (ICLR), Banff (2014).

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)

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).

Simonyan K., Zisserman A. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. Proc. Intl Conf on Learning Representations (ICLR), San Diego (2014)

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).

Zhao J, Mathieu M, LeCun Y. *Energy-based Generative Adversarial Network*. Proc. Intl Conf on Learning Representations (ICLR), Toulon (2017).

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).

Cha YJ, Choi W, Büyüköztürk O. *Deep Learning Based Crack Damage Detection Using Convolutional Neural Networks*. J Computer-Aided Civil and Infrastructure Engineering, Vol. 32(5): 361-378 (2017).

Chaiyasarn K, Sharma M, Ali L, Khan W, Poovarodom N. *Crack Detection in Historical Structures Based on Convolutional Neural Networks*. Intl J of Geomate, Vol. 15(51): 240-251 (2018).

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)

Sawaf F, Groves RM. *Phase Discontinuity Predictions Using a Machine-Learning Trained Kernel*. Applied Optics, Vol. 53(24): 5439-5447 (2014).

Hou W, Wei Y, Guo J, Jin Y, Zhu C. *Automatic Detection of Welding Defects using Deep Neural Network*. J Physics, Vol. 933 (2018): 012006 (2018).

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).

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).

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).

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).

Gong R, Wu C, Chu M. *Steel Surface Defect Classification using Multiple Hyper-Spheres Support Vector Machine with Additional Information*. J Chemometrics and Intelligent Laboratory System. Vol. 172: 109-117 (2018).

Ebrahimkhanlou A, Dubuc B, Salomone S. *A Generalizable Deep Learning Framework for Localizing and Characterizing Acoustic Emission Sources in Riveted Metallic Panels*. J Mechanical System and Signal Processing. Vol. 130: 248-272 (2019)

Ebrahimkhanlou A, Salamone S. *Single-Sensor Acoustic Emission Source Localization in Plate-Like Structures Using Deep Learning*. Proc. SPIE Smart Structures And NDE: Health Monitoring of Structural and Biological Systems XII (2018).

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).

Liu H, Zhang Y. *Deep Learning Based Crack Damage Detection Technique for Thin Plate Structures using Guided Lamb Wave Signals*. J Smart Materials and Structure. Vol. 29: 015032 (2019).

Bull A, Rogers TJ, Wickramarchchi C, Cross EJ, Worden K, Dervilis N. *Probabilistic Active Learning: An Online Framework for Structural Health Monitoring*. J Mechanical System and Signal Processing. Vol. 134: 106294 (2019).

Gardner P, Worden K, Liu X. *On the Application of Domain Adaptation in Structural Health Monitoring*. J Mechanical Systems and Signal Processing. Vol. 138: 106550 (2020).

Choy AW. *Structural Health Monitoring with Deep Learning*. Proc. 2018 IAENG International Conf on Control and Automation, Hong Kong (2018).

S. Gopalakrishnan, Ruzzene M, Hanagud S. *Computational Techniques for Structural Health Monitoring*. Springer, London / Dordrecht / Heidelberg / New York (2011).

Stepinski T, Uhl T, Staszewski W. *Advanced Structural Damage Detection: From Theory to Engineering Applications*. John Wiley & Sons Ltd., West Sussex (2013).

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).

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).

Xu H, Xu C, Li X, Wang L. *Study on Single Mode Lamb Wave Interaction with Defect of Plate by Finite Element Model*. Procedia Engineering, Vol. 15: 5067-5072 (2011).

Taltavull A. *Structural Health Monitoring Solutions for Patched Metallic Aircraft Repairs Based on Guided Ultrasonic Waves*. MSc Thesis, University of Saarland (2017).

Fedotenkova M. Available online https://github.com/mfedoten/reasspectro (Last online: FEB-2020)

Sethuraman J. *Some Limit Theorems for Joint Distributions*. The Indian Journal of Statistics: Series A. Vol. 23(4): 379-386 (1961).

Valiant LG. *A Theory of the Learnable*. J Communications of the Association of Computing Machinery (ACM). Vol. 27(11): 1134-1142 (1984).

Valiant LG. *Probably Approximately Correct: Nature's Algorithms for Learning and Prospering in a Complex World*. Basic Books Inc., New York (2013).

Gormley M. *Lecture 28: PAC Learning*. In Lecture series 10-601: Introduction to Machine Learning. Carnegie Mellon University (2016).

Moran S, Yehudayoff A. *Sample Compression Schemes for VC Classes*. J Communications of the Association of Computing Machinery (ACM). Vol. 63(3): 1-21 (2016).

Wolpert DH. *A Mathematical Theory of Generalization: Part I*. J Complex Systems. Vol. 4: 151-200 (1990).

Wolpert DH. *Stacked Generalization*. J Neural Networks. Vol. 5(2): 241-259 (1992).

De Oliveira MA, Monteiro MA, Vieira Filho J. *A New Structural Health Monitoring Strategy Based on PZT Sensors and Convolutional Neural Network*. J Sensors, Vol. 18: 1-21 (2018).

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 (2018).

Ooijevaar T. *Vibration-based Structural Health Monitoring of Composite Skin-stiffener Structures*. PhD Diss, University of Twente (2014).

Mitchell T. *Machine Learning*. McGraw-Hill, Redmond & Ithaca (1997).

MIL-HDBK-1823A. *Non-Destructive Evaluation System Reliability Assessment*. US Department of Defense, Wright-Patterson (2009).

Hayo T, Frankenstein B, Boller C, Bockenheimer C. *Approach to the Technical Qualification of a SHM System in Terms of Damage Detection in Aerospace Industry*. Proc. Intl Workshop Smart Materials, Structures & NDT in Aerospace, Montreal, 1-9 (2011).

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).

Ewald V, Ochoa P, Groves RM, Benedictus RM. *Design of a Structural Health Monitoring System for a Damage Tolerance Fuselage Component*. Proc. 7th Intl Symposium on NDT in Aerospace, Bremen, 1-9 (2015).

Giurgiutiu V, Yu L. *Comparison of Short-time Fourier Transform and Wavelet Transform of Transient and Tone Burst Wave Propagation Signals for Structural Health Monitoring*. Proc. 4th Intl Workshop for Structural Health Monitoring (IWSHM), Stanford, 1-9 (2003).

Morales JL. *A Numerical Study of Limited Memory BFGS Methods*. J Applied Mathematics Letter. Vol. 15(4): 481-487 (2002).

Shalev-Shwartz S, Ben-David S. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, Cambridge (2014).

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).

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).

Stöver T, Diensthuber M. *Molecular Biology of Hearing*. J GMS Current Topics in Otorhinolaryngology - Head and Neck Surgery. Vol. 10: 1-15 (2011).

Petralia RS, Wenthold RJ. *Neurotransmitters in the Auditory System*. Encyclopedia of Neuroscience (2009).

Ruder S. *An Overview of Gradient Descent Optimization Algorithms* (2016). Available online https://arxiv.org/abs/1609.04747 (Last online: FEB-2020)

Tensorflow Documentation. Available online https://www.tensorflow.org (Last online: FEB-2020)

Keras Optimizers Documentation. Available online https://www.keras.io/optimizers (Last online: FEB-2020)

PyTorch Documentation. Available online www.pytorch.org

Géron A. *Hands-On Machine Learning with Scikit-Learn and TensorFlow*. O'Reilly Media, Sebastopol CA (2017).

Github repository. Available online https://github.com/vewald/ (Last online: FEB-2020)

Ewald V, Groves RM, Benedictus R. *Transducer Placement Option of Lamb Wave SHM System for Hotspot Damage Monitoring*. MDPI J Aerospace. Vol. 5(2): 39 (2018).

Ewald V, Groves RM, Benedictus R. *DeepSHM: A Deep Learning Approach for Structural Health Monitoring Based on Guided Lamb Wave Techniques*. Proc. SPIE Smart Structures and NDE, Denver, 1-16 (2019).

Liao Q, Leibo JZ, Poggio T. *Learning Invariant Representations and Applications to Face Verification*. 27th Conference on Neural Information Processing System (NIPS), Lake Tahoe, 1-9 (2013).

Feige I. *Invariant-Equivariant Representation Learning for Multi-Class Data*. 36th Conference on International Conference on Machine Learning (ICML), Long Beach, 1-5 (2019).

Zhao H, de Combes RT, Zhang K, Gordon GJ. *On Learning Invariant Representation for Domain Adaptation*. 36th Conference on International Conference on Machine Learning (ICML), Long Beach, 1-12 (2019).

Isik L, Taccheti A, Poggio T. *A Fast, Invariant Representation for Human Action in the Visual System*. Journal of Neurophysiology, Vol.: 119(2): 631-640 (2016).

HanY, 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).

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)

Hu W, Hu J. *Training a Quantum Neural Network to Solve the Contextual Multi-Armed Bandit Problem*. J Natural Science, Vol. 11(01): 17-27 (2019).

Killoran N, Izaac J. *Training Quantum Neural Networks with PennyLane, PyTorch, and TensorFlow*. Available online https://github.com/XanaduAI/pennylane (Last online: FEB-2020)

Paoletti ME, Haut JM, Plaza J, Plaza A. *A New Deep Convolutional Neural Network for Fast Hyperspectral Image Classification*. J of Photogrammetry and Remote Sensing (ISPRS), Vol. 145(A): 120-147 (2018).

Li S, Song W, Fang LY, Chen Y, Ghamisi P, Benediktss JA. *Deep Learning for Hyperspectral Image Classification: An Overview*. IEEE Transactions on Geoscience and Remote Sensing. Vol. 57(9): 6690-6709 (2019).

Botev A, Lever G, Barber D. *Nesterov's Accelerated Gradient and Momentum as Approximations to Regularised Update Descent*. International Joint Conference on Neural Networks (IJCNN), 1-5 (2017).

Global Care Company. Available online http://www.globalcarecompany.com/benifits-of-binaural-hearing.html (Last online: FEB-2020)

Weimer D, Scholz-Reiter B, Shpitalni M. *Design of Deep Convolutional Neural Network Architectures for Automated Feature Extraction in Industrial Inspection*. J CIRP Annals Manufacturing Technology. Vol. 65(1): 417-420 (2016).

Hebb DO. *The Organization of Behavior: A Neuropsychological Theory*. John Wiley & Sons Inc, New York (1949).

Wallisch P, Lusignan ME, Benayoun MD, Baker TI, Dickey AS, Hatsopoulus NG. *Chapter 36 - Neural Networks Part I: Unsupervised Learning*. In MATLAB for Neuroscientists: An Introduction to Scientific Computing in MATLAB. 2nd Ed., Elsevier, London / Waltham / San Diego (2014).

Langille JJ, Brown RE. *The Synaptic Theory of Memory: A Historical Survey and Reconciliation of Recent Opposition*. J Frontiers in Systems Neuroscience. Vol. 12: 52 (2018).

Trettenbein PC. *The Demise of the Synapse as the Locus of Memory: A Looming Paradigm Shift?* J Frontiers in Systems Neuroscience. Vol. 10: 88 (2016).

Merck: Future of AI Challenge. Available online https://app.ekipa.de/challenges/future-of-ai/brief (Last online: FEB-2020).

Ostachowicz W, Kudela P, Krawczuk M, Zak A. Guided Waves in Structures for SHM: The Time-Domain Spectral Element Method. John Wiley & Sons Ltd., West Sussex (2012).