ENIQ position on

Qualification of an Artificial Intelligence / Machine Learning Non-Destructive Testing System

ENIQ Publication no. 64

POSITION PAPER

NUGENIA is an international non-profit association under Belgian law established in 2011. Dedicated to the research and development of nuclear fission technologies, with a focus on Generation II & III nuclear plants, it provides scientific and technical basis to the community by initiating and supporting international R&D projects and programmes. The Association gathers member organisations from industry, research, safety organisations and academia.

The activities of NUGENIA cover plant safety & risk assessment, severe accidents, reactor operation, integrity assessment and ageing of systems, structures & components, development of fuel, waste & spent fuel management & reactor decommissioning, innovative light water reactor design & technologies, harmonisation and in-service inspection & their qualification.

NUGENIA Association
c/o EDF, avenue des Arts, 53
B - 1000 Bruxelles
Belgium

For more information please contact the NUGENIA Secretariat:
secretariat@nugenia.org

FOREWORD

Machine learning has developed rapidly in recent years. The state-of-the-art models are now powerful enough to be successfully applied to complex data, including various non-destructive testing (NDT) data sets. Very recently, machine learning models have demonstrated human-level performance in detecting flaws in complex ultrasonic data.

Before these models can be used in nuclear inspections, their performance and reliability must be verified. Procedures using machine learning must be qualified, similar to conventional procedures. However, using machine learning deviates significantly from traditional inspection scenarios and thus may need special attention in the technical justification and practical trials during qualification.

This document provides a high-level review of the current position on qualifying NDT systems for use in the nuclear industry which make use of machine learning. It reviews various potential approaches to utilize machine learning and discusses areas of special needs in qualification. The document is meant to provide the basis for a recommended practice to qualify NDT systems containing machine learning.

The authors gratefully acknowledge the comments and contribution received from Michael Klotzbücher (Framatome GmbH), Liudas Mazeika (Kaunas University), Vykintas Samaitis (Kaunas University) and Phil Ashwin (EPRI).

TABLE OF CONTENT

1. Introduction

Artificial intelligence (AI) and machine learning (ML) have developed significantly in recent years and have solved many tasks that for long were considered impossible to automate. Such tasks include handwriting recognition, image classification, detecting anomalies and more recently, generation of human-like text and speech.

While automated systems have long been used in non-destructive testing (NDT), their main application has been in simple highly automated inspection setups relying on simple decision algorithms like an amplitude threshold. For most inspections and, in particular, for the more demanding applications, such simplistic automated systems are inadequate. The difficult, tedious and error prone task of detecting flaws with low signal-to-noise ratio from complex signals, such as ultrasonic data has evaded automation and requires the use of human expert analysts. In addition to flaw detection, flaw classification (e.g., distinguishing between service-induced flaws or manufacturing flaws) and flaw sizing remain tasks for the human experts. These are, flaw detection in particular, time consuming tasks that require high level of expertise. In addition, the tasks require maintaining high level of concentration for long periods of time. Thus, the work is taxing and may be susceptible to human errors.

The recent advancements in ML applications in complex classification tasks share many features with NDT signal classification and flaw detection. It is generally expected, that ML technologies will also find applications in the NDT field. However, the performance requirements in the NDT field are quite high, and thus the use of ML systems requires thorough validation and qualification.

The ENIQ framework is highly flexible and a methodical framework for qualifying different NDT systems. It has been successfully used in many different countries, for different inspection applications and inspection methods. The flexibility is highly valuable in taking advantage of new technologies like ML. Despite this flexibility, ML systems represent in some ways a fundamentally different decision process than the traditional systems. Hence, they may need special attention in justifying and validating the expected performance, that are not included in the current ENIQ recommended practices (RPs) which assume human judgement-based systems. ML systems also offer the possibility for continuously improved / trained systems. Controlling procedure stability and changes forms an important part of the ENIQ framework and continuously changing systems are largely incompatible with the current methodologies for controlling change.

This document provides the current ENIQ position for including ML systems in ENIQ qualified NDT systems. Special requirements presented by the use of ML systems are discussed and identified. The document describes in particular the guidelines for qualification of a NDT system that utilises AI or ML systems.

This document does not aim to give guidance in developing or using any particular ML software or system. The discussion is at an abstract level and the examples given are for illustrating generic considerations and not to be considered as a recommendation of using a specific model or software for a certain application area.

2. Artificial Intelligence / Machine Learning Fundamentals

"Artificial intelligence (AI) is the field, that aims to develop systems, that mimic human intelligence or perform tasks that have been thus far thought to require human intelligence" [1]. This is a vast field with long tradition and includes, for example, various expert systems that seek answers from databases based on set questions or adaptive sequence of questions. Recently, attention within AI has focused mainly to ML and deep learning systems (see Figure 1).

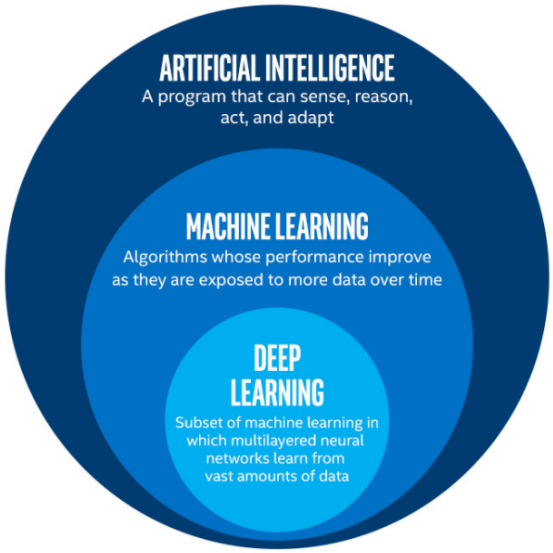

Figure 1: A schematic overview of AI sub-items

Various algorithms have been developed to implement ML behaviour. Each of these algorithms provide a wide area of potential application areas and for any given problem, several algorithms may provide a viable solution. Depending on what and how the ML algorithms "learn" they are divided in three broad categories: supervised learning, unsupervised learning and reinforcement learning (see Figure 2).

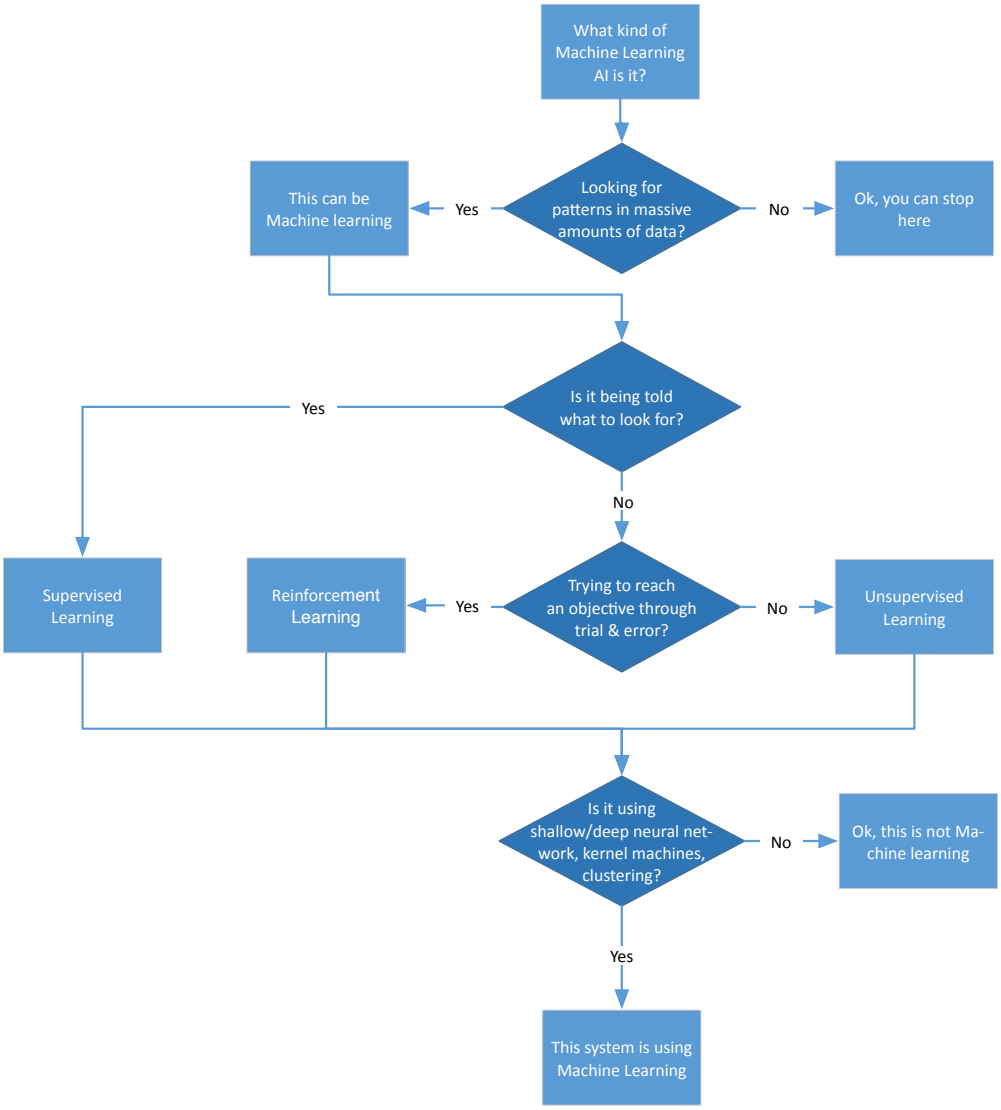

Figure 2: A schematic overview on artificial intelligence sub-items [2]

In **supervised learning**, the ML algorithm is provided with a (large) set of known data that corresponds to the input in the problem domain and the desired output. This is akin to using open samples to train humans. The data is called labelled because it contains the desired output (label) for included input (inspection signal). The learning then proceeds to optimize the network so as to produce the desired output when given any relevant input. The key benefit is, that the desired outcome is clearly and explicitly defined. A typical problem is, that it is necessary to have a large set of labelled data, which is often costly to produce or unavailable. For NDT the labelled data takes the form of NDT data from known flawed and unflawed samples.

In **unsupervised learning**, the ML algorithm tries to optimize and create internal representation of the data, that follows the algorithm design. Such models can be used to find, e.g., anomalous input or items closely matching a given input. The key benefit is, that pre-labelled input data is not needed. Such models could be taught to flag suspiciously “anomalous” signals even when known data from open samples are not available.

In **reinforcement learning**, the ML algorithm takes active action during the training and learns to optimise its actions to maximise some “reward”, i.e. desired outcome. Such algorithms can be used, e.g., for learning to play an interactive game, where the problem domain needs to be “explored” to create the learning data set.

While all of these ML systems offer potential benefits in the field of NDT, it is recognized that **supervised learning** models are most easily to integrate in the current ENIQ framework. The applicable models can be trained with a controlled, verifiable training set and versioned (i.e. the learning stopped) so that an unchangeable trained model can be qualified and then used with predictable results. Annex 1 provides a more detailed discussion on such supervised learning models and, to give a concrete example, provides some hints on shallow and deep architectures [3] and the associated methods that are widely employed by the scientific community.

Various schemes of continuous and incremental learning modes can also be envisioned, e.g., if it is planned that the ML system uses reinforced learning after the qualification, this could potentially be justified by locking the qualified ML system (M1) and in parallel let a second ML system (M2) evolve using new data collected during inspections. M2 evaluation results cannot be used during the inspections but can be utilised in the next qualified revision of the software. Such continuous learning schemes are excluded from the scope of this document.

3. Qualifying Machine Learning within the ENIQ Framework

A ML system differs from a traditional system in that the inner logic of a trained network has limited transparency. While the model can be reviewed and various parts of the learned algorithm revealed, the models are typically far too complex to be reducible to a simplified decision process or model that could directly be used to reason about the capability or reliability of the system.

Qualifying a NDT system that contains a ML component may induce new demands for all of the components of an ENIQ qualification: input information, inspection procedure, technical justification (TJ), open trials and blind trials. These are all discussed in the following.

3.1 Input Information

The input information in an ENIQ qualification defines the types of flaws to be detected. In traditional qualification, this typically includes the damage mechanism (such as fatigue or stress corrosion cracking), the flaw sizes that shall be detected and limits to some features of the flaws, such as skew and tilt. These parameters are also used to define flaw simulations in the open and blind trial test pieces.

For an ML system, it is expected that the predictions will be good for samples that are similar to the learning data. While the models are trained and validated to generalise from the training data to previously unseen validation and verification data, the model predictions are expected to deteriorate as with decreasing similarity between training data and the final application data. Consequently, the depth and completeness of the learning data is vital for successful application of a ML model.

To be able to justify that the chosen training set fully represents the input information, the desired flaw properties need to be either defined directly in the input information or deduced from the given data in the TJ. The input information conventionally contains information on flaw properties, but ML systems may need more detailed specification.

3.2 Inspection Procedure

The inspection procedure will define the use of the ML algorithm within the wider NDT system. Several levels of ML use can be identified. In an ML-informed system, the human inspector is the primary agent of flaw detection, and the ML system is used in parallel to inform the human inspector from potentially missed flaws. A flaw is reported if either the human alone or informed by the ML system reports a flaw. Humans quickly learn to trust or disregard such a redundant ML system and the ML system might not actually work as intended. Also, such a ML system offers limited benefits over the current system.

Secondly, the ML system can be used to focus the inspector's effort. The acquired data is fed into the ML system, which produces a report indicating flaw locations, if any. The human inspector then

analyses and characterises these indications and decides the appropriate reporting and follow-up actions. A flaw is reported if both the ML system and subsequent human analysis judge the flaw to be reportable. Here, substantial time saving is achievable, especially in cases where flaw indications are rare. Existing automation solutions in, e.g., steam generator tube inspections are used in this way.

Thirdly, the ML system can be used as an independent system so that the indications reported by the ML system are reported directly, without human intervention. Such a system has even greater potential benefits, especially in cases where flaw indications are common and follow-up actions relatively cheap (such as abandoning a defective part).

3.3 Technical Justification

The role of the TJ is to show that the NDT system can achieve the required performance, based on theoretical arguments [4].

3.3.1 Justifying the used Machine Learning Model

There is a multitude of off-the-shelf ML models that can form the basis of a successful flaw detection model. The selection of the model in terms of sufficient capability needs to be justified in the TJ.

If the model is inadequate for the complexity of the problem in scope, the learning may fail even with ample training data. The remaining prediction error may be too large for the intended use. This is called “underfitting”. With modern models and computational power, underfitting is rarely a problem. It may cause problems when the model is intended to be used in an environment with limited computational power, such as an embedded system, or when the target problem is exceedingly complex.

Conversely, if the model is sufficiently complex, but the learning data is scarce, the model may learn the training data set with increasing precision, but fails to form a generalised model that would correctly work for unseen data. Thus, the results against unseen data may start to deteriorate. This is called “overfitting”. For this reason, it is important to check the model results against separate validation data which was not used in model training. Typically, the contemporary models are highly powerful in comparison to the available training data and thus overfitting usually needs to be addressed by specific model choices and techniques designed to reduce overfitting (i.e. model regularisation).

In order to justify, that the model is both sufficiently powerful and sufficiently generalised, results on previously unseen validation data needs to be shown and evaluated.

3.3.2 Justifying the Training and Validation Data

Justification of the selection of the training and validation data is an imperative. The accuracy of any ML system is expected to deteriorate quickly, if the input starts to deviate from the training data. Thus, the role of the training data is crucial. The data needs to contain sufficient number of flaw signals to account for the natural variation in different flaws across the size range of interest.

In particular, the following considerations need to be discussed and argued in the TJ:

- Data relevance and scope for the task,
- Data labelled correctly,
- Data sufficient in numbers,
- Data sufficient in range over the intended task.

During model development, typically, several different architectures are tried and hyper-parameters adjusted to find a model that learns the target problem well, avoids overfitting, and can learn efficiently from available data. During this development, the validation data is used repeatedly to check the model against overfitting. Thus, the architecture of the evolving model is influenced by the validation data and may evolve to give overly positive results on this validation set. It is also important to avoid a bias in the learning for the validation set. For this reason, the validation results cannot be used, to evaluate or validate the final model performance. Thus, additional evaluation on previously unseen test data needs to be included in the open and/or blind trials.

3.4 Open Procedure Trials

The role of the open trials is to test the claims made in the TJ and to show that sufficient performance can be attained by following the inspection procedure, as defined. For ML systems, the reasoning for the claims in the TJ on data analysis capability cannot be based on physical models to the same extent as for traditional systems. Thus, it is expected, that in qualifying ML systems, the procedure trials have a more important role than for traditional systems. This pertains only to the part of the procedure that uses ML. Other aspects of the procedure, e.g. variation of actual site conditions, are not affected and can make use of traditional physical reasoning.

To validate and verify the performance of a ML system, it needs to be evaluated against previously unseen verification data set(s). The emphasis of empirical evidence to qualify ML systems blurs the distinction between the TJ and open trials. Thus, it may be advisable to design the open trials and the TJ together so that redundant effort is avoided. In practice, this means, that the TJ is designed to be lighter than usual and defers some of the burden of proof to open trials, which then need to be more substantial than normal.

One consideration when developing the data set for validation (practical demonstration) is to also take into account a future revision and/or upgrade of the software and whether the same data set can be used for this as well. It is crucial, that the verification data has not in any way been used in the training or in the validation during the development of the ML system. Otherwise, the ML model may adjust to the specific data and may show overly positive results due to overfitting.

3.5 Blind Personnel Tests

ML systems reduce or remove human judgement from the inspection procedure. Consequently, parts of the inspection procedure that guide the human judgement and analysis are simplified and reduced. The primary role of the blind trial is to show, that the human inspectors can consistently apply the procedure and provide consistent results. As the procedure is simplified, there is less opportunity for inconsistency and thus less need for blind trials. Thus, it is expected that the role of the blind trials will decrease with increasing use of ML systems.

4. Acceptance Level and Performance Evaluation

The process of training and validation of a ML system will necessarily produce statistically representative data of the model performance, such as the false call rate and probability of detection (POD) curve, albeit within the scope of the validation or evaluation data. Such data is normally not available for human inspectors. Furthermore, the ML system is expected to have a very high repeatability as compared to traditional systems. When comparing the performance of human inspectors and ML systems, these benefits need to be considered, to enable meaningful comparison. This may result in different pass/fail criteria of ML systems and non-ML systems.

5. Discussion

The key features that distinguish NDT systems using ML from traditional systems are reduced human judgement and reduced opportunity for theoretical reasoning of decision criteria. Thus, qualifying a ML system will put more emphasis on the qualification parts that use empirical evidence, i.e., open trials and some parts of the TJ. Conversely, those parts of a qualification primarily needed to control and test variability of human judgement receive less emphasis. These include blind trials and the related chapters of the TJ. Overall, the inspection procedures can be expected to become simpler in terms of data evaluation, which in turn may simplify the whole qualification procedure.

As presented above, the ENIQ methodology offers a very flexible framework that uses a combination of theoretical reasoning and empirical evidence to build sufficient confidence that a NDT system can meet the specified reliability requirements. This framework can accommodate widely varying NDT systems by varying the respective weight and focus of these parts.

In the simplest case, a pre-trained and versioned supervised ML model is used to detect flaws, which are then confirmed, characterised and reported by a human inspector. Such a system uses the complementary skills of humans and machines and provides the speed, consistency and reliability of an automated system together with the case-specific expertise of the human analyst. The qualification of such a system fits well within the current ENIQ framework. The introduction of the ML component requires more thorough empirical evidence (e.g. in the open trial) to compensate for the reduced theoretical reasoning.

6. Conclusions

The following conclusions can be drawn from this study:

- The ENIQ framework is flexible enough to accommodate the qualification of NDT systems that make use of ML.
- Reasoning of the selection of the ML model and data sets will be a large part of the TJ.
- The qualification of NDT systems that change during operation, such as continuously learning ML models is non-trivial with the current framework.
- Qualifying ML systems may require more empirical evidence to compensate for the reduced theoretical reasoning.

REFERENCES

[1] Russell, S. J. and Norvig, P., 2016. *Artificial intelligence: A Modern Approach, Third edition*, Pearson Education Limited.

[2] Hao, K. 2018. *What is Machine Learning? MIT Technology review.* Available online: www.technologyreview.com/s/612437/what-is-machine-learning-we-drew-you-another-flowchart .

[3] Bengio, Y. and LeCun, Y., 2007. Scaling learning algorithms towards AI. In L. Bottou, O. Chapelle, D. DeCoste, & J. Weston (Eds.), *Large-scale kernel machines,* MIT Press.

[4] *ENIQ Recommended Practice 2: Strategy and Recommended Contents for Technical Justifications - Issue 3*, ENIQ Report no. 54, The NUGENIA Association, 2018.

[5] Bishop, C. M., 2006. *Pattern recognition and machine learning*, Springer.

[6] Shawe-Taylor, J. and Cristianini, N., 2004. *Kernel methods for pattern analysis*, Cambridge University Press.

[7] Cristianini, N. and Shawe-Taylor, J., 2004. *Support Vector Machines and other kernel-based learning methods*, Cambridge University Press.

[8] Scholkopf, B. and Smola, A.J., 2001. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press.

[9] Salucci, M. et al., 2016. Real-time NDT-NDE through an innovative adaptive partial least squares SVR inversion approach. *IEEE Transactions on Geoscience and Remote Sensing*, 54(11), pp. 6818-6832.

[10] Ahmed, S., Reboud, C., Lhuillier, P. E., Calmon, P. and Miorelli, R., 2019. An adaptive sampling strategy for quasi real time crack characterization on eddy current testing signals. *NDT & E International*, 103, 154-16.

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

[12] Virkkunen, I., Koskinen, T., Jessen-Juhler, O. and Rinta-Aho, J., 2019. Augmented Ultrasonic Data for Machine Learning. *arXiv*, 1903.11399v1.

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

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

[15] Munir, N., Kim, H.-J., Song, S.-J. and Kang, S.-S., 2018. Investigation of deep neural network with drop out for ultrasonic flaw classification in weldments, *Journal of Mechanical Science and Technology*, 32(7), pp. 3073-3080.

[16] Tang, Y., 2013. Deep learning using linear support vector machines. *arXiv preprint* arXiv:1306.0239.

ANNEX – MACHINE LEARNING TECHNICAL FUNDAMENTALS

1. Machine Learning Stages

Generally speaking, ML algorithms are based on a concatenation of different stages that, provided a given task, enable to carry out a suitable ML model. The set of stages that are matter of interest to develop a ML strategy regardless of the adopted ML model can be listed as follows:

- **Data gathering:** a set of data (labelled or not) is collected. The source of the data can be of various origins (e.g., numerical calculations, experimental databases of NDT signals, database of categories, etc.) depending on the task that one wants to solve. The quality and the quantity of this data is crucial to the performance of the ML model training.
- **Data preparation:** the data should be formatted and cleaned to avoid all non-spurious contents that have meaningless information from the physics point of view. In this stage, one can also add all data processing stages needed to format data (e.g., centring, sub-, under-sampling, etc.). Under this task one can collect all data and signal processing procedures.
- **Model choice or training phase:** once the data has been gathered and prepared one can fit a proper ML model on it. This phase is known as training phase and it is performed until the model converges towards a stable solution. The training phase is based on a given set of data (i.e., the training set) and is referred to as off-line phase. During this phase, a suitable set of model parameters is established based on the data provided to the model. The obtained outcome at the end of the process is the trained ML model.
- **Model evaluation or test phase:** once the model has been trained it can be tested on a set of data that has not been used during the training phase. This stage is known as test phase and it does not require any learning step. Therefore it is also sometimes called on-line phase. The main purpose of the evaluation phase is to assess the performance of the ML model before its deployment. It is worth mentioning that the input data provided at this stage should be prepared in a similar way as in the model choice stage (i.e., all the data preparation steps should be performed).
- **Prediction or inference:** the final trained model is then deployed and it can be fed by data with the purpose of classification or regression. As for the model evaluation stage, the input data should be prepared to the same format (e.g., resizing, data processing procedure, etc.) as the one provided in the training phase.

In the context of NDT, ML methods can be trained based on both labelled experimental and simulated data. The use of simulated data can be seen as a viable way to add an *a-priori* knowledge provided to the learning algorithm. In case of scarce labelled experimental data, this synthetic data can improve the generalisation capabilities of the ML model and, as a consequence, lead to a model less prone to errors.

2. Shallow Architectures

Under the term “shallow architectures” the ML community defines a set of ML methods that exploit the concept of kernel machines (KMs) [5] [6] [7] [8]. In most cases, supervised learning frameworks are used to perform classification (e.g., flaw(s)/anomaly detection, defect(s) classification) or regression tasks (parametric-flaw characterization). To cover the cardinality of NDT raw signals, very often the KMs are coped with a dimensionality reduction stage (part of data preparation phase) aiming to reduce the information content by reducing the redundancy in the signals to be used for the learning process and to mitigate the so-called curse-of-dimensionality issue [9]. This stage relies on very well-established algorithms in many communities (e.g., chemometric, signal processing, biomedical, etc.) from which one can establish statistical and/or geometrical properties associated to the data reduction stage. Loosely speaking, these methods can be grouped into two big families, the matrix

decomposition algorithms such as principal component analysis, independent component analysis etc. and the manifold learning families of algorithms such as isometric mapping (ISOMAP), locally linear embedding, etc.

Different learning methods can be collected under the term kernel machines (KMs). All KMs methods rely on the use of the so called kernel trick [5] [7] [8] to perform classification and regression tasks. The kernel trick enables to perform linear interpolations of non-linear data by fitting the model directly in the kernel space [6]. For sake of brevity, we mention just hereafter the most known and studied methods by providing a brief and concise applicative background associated to each of them.

It is worth mentioning that the most widely employed KMs in literature are based on the vectorisation of data (i.e., the matrices or tensors are reshaped into vectors). Therefore, any kind of spatial and/or temporal coherence within probed data is not preserved in the training and testing set.

The kernel ridge regression (KRR) is the kernel version of the well-known ridge regression [8]. KRR is obtained by formulating the ridge regression exploiting the kernel trick. KRR enables control of the regression performance through a regularisation coefficient that can be tuned to maximize the trade-off between variance and bias. Due to its statistical meaning, this penalisation coefficient can be a valuable degree of freedom to enhance the model performance in case of noisy measurements. This hyper-parameter, plus the one associated to the chosen kernel (very often just one) are the only parameters to be estimated in order to obtain a classification or regression (e.g., defect localization/characterization) model. The tuning of KRR hyper-parameters is often obtained via cross validation.

The Gaussian Process (GP) for classification and regression, also known as kriging in the geoscience community, is a statistical model that exploits the Bayesian framework in order to perform classification and regression tasks [5]. Even though the GP formulation shares many common points with KRR, its statistics enables access to the mean and the variance of the prediction, providing de-facto a measure of the model or epistemic uncertainties associated to the predictions (i.e., the classification or regression results). GP as KRR require the tuning of hyper-parameters associated to both the deployed kernel (i.e., the covariance function) and the regularization coefficient. This stage is often performed via minimisation of a suitable likelihood function.

Support Vector Machines (SVMs) for regression and classification tasks have been widely and successfully deployed in many different fields as of the late 1990s. SVMs are based on a mathematical background rooted in the statistical learning theory and structural risk minimisation. It enables providing theoretical limits to SVM models [6] [7]. As all the other kernel methods, SVM models require tuning of kernel hyper-parameters along with two parameters associated to the SVM algorithms. The physical meaning of SVM parameters and thus their choice, can be provided by the theory upon which the SVM is developed. The tuning of SVM hyperparameters is often obtained via cross validation. Compared to KRR and GP, a SVM model enables sparse predictions that turn into a computationally efficient model compared to the above-mentioned methods [10].

It is worth mentioning that other shallow architectures have been developed and published in literature. All these models can be considered as improvements, modifications or hybridisations of the aforementioned ones. Among the most known ones are co-kriging, universal-kriging and relevance vector machines, etc.

3. Deep Learning Methods

Deep learning (DL) methods showed their potential in the first decade of 21st century when they appeared to be able to provide the same or better performance than shallow architectures applied to supervised learning tasks in image classification problems. DL methods rely on the use of specific neural network architectures like multilayer perceptron, convolutional neural network, etc. More recently, the use of DL has been boosted by the increasing performances of graphical process units (GPUs) enabling more efficient model training. Research in DL have been motivated by the fact that DL methods aim to avoid feature engineering and kernel engineering stages that are often necessary

before training a KM model. This point makes DL an attractive tool to solve regression and classification problems for end users that are not experts in ML [11].

In neural networks, the learning function is formed by linear combination of a set of simple non-linear activation functions. The model is typically formed in layers, where the layers are connected though activation functions and results within a layer combined linearly with learned weights.

During learning, the input data is propagated through each layer to form the model result. This is then compared to the given label value and an error value computed using a specified function (the “cost function”). This error is then propagated backwards (back propagated) through the model and at each layer the weights are updated to improve the next prediction. With each iteration, the model weights of the whole network are updated to give, presumably, a better prediction.

For many problems, such as image classification, the location of the features sought are inconsequential. This location invariance can be introduced to the model by a convolution layer, where a small kernel is shifted through the data and the value computed at each location. The layers form “feature maps”, that encode location-invariant information about presence and relationships of specified features. This convolution effectively creates weight sharing that greatly reduces the number of learned parameters. The number of learned parameters can be further reduced by pooling layers, which combine activations for adjacent locations.

Deep convolutional neural networks (DCNNs) make use of convolutional and pooling layers, to encode source data (often an image) to increasingly abstract representations while reducing the dimensionality of the data with each subsequent layer. Such very deep models have proven very successful in many image classification tasks.

Virkkunen et al. [12] used DCNNs to successfully detect cracks in phased array ultrasonic data. Recently Meng et al. [13], Zhu et al. [14] and Munir et al. [15] used DCNNs for defect classification in ultrasonic and eddy-current data, respectively. In general, DCNNs are interesting for various flaw detection and classification tasks and various NDT signal data.

4. Hybrid Learning by Coupling shallow and deep Architectures Methods

Coupling between shallow and deep architectures is a valuable solution in order to take fully advantage of the solid mathematical background of shallow learning (SL) and the non-invasiveness (i.e., no-feature engineering stage required) associated to DL approaches. This kind of hybrid learning approach is widely studied and exploited by the ML community. Hybrid learning approaches have shown to perform better than SL or DL methods for complex classification tasks [16].

Contributors to Drafting and Editing

Iikka Virkkunen	Trueflaw Ltd.	Finland
Martin Bolander	Westinghouse Electric	Sweden
Heikki Myöhänen	Kiwa Inspecta	Finland
Roberto Miorelli	CEA	France
Oliver Martin	European Commission – Joint Research Centre	European Commission

The European Network for Inspection and Qualification (ENIQ) is a utility driven network working mainly in the areas of qualification of non-destructive testing (NDT) systems and risk-informed in-service inspection for nuclear power plants. Since its establishment in 1992 ENIQ has issued over 60 documents. Among them are the “European Methodology for Qualification of Non-Destructive Testing” and the “European Framework Document for Risk-Informed In-Service Inspection”. ENIQ is recognised as one of the main contributors to today’s global qualification guidelines for in-service inspection. ENIQ became Technical Area 8 of NUGENIA in 2012.