NUGENIA
Nuclear Generation II&III Alliance

ENIQ RECOMMENDED PRACTICE

ENIQ Recommended Practice 13

Qualification of Non-Destructive Testing Systems that Make Use of Machine Learning

Issue 1

ENIQ Report No. 65

Technical Area 8

European Network for Inspection & Qualification

June 2021

ENIQ
European Network for
Inspection & Qualification
NUGENIA Technical Area 8

SNETP Association
c/o EDF
Avenue des Arts 53B, 1000 Brussels, Belgium
Email: secretariat@snetp.eu
Website: www.snetp.eu

This document was prepared by the SNETP Association.

LEGAL NOTICE

Neither SNETP nor any person acting on behalf of SNETP is responsible for the use which might be made of this publication.

Additional information on SNETP is available on the Internet. It can be accessed through the SNETP website (www.snetp.eu).

Brussels: The SNETP Association

ISBN 978-2-919313-28-0

Index

Executive Summary

This Recommended Practice (RP) has been developed as a consensus document amongst the members of NUGENIA Technical Area 8 (TA8) – European Network for Inspection and Qualification (ENIQ). The main objective of this RP is to support licensees, qualification bodies and inspection vendors to produce and assess inspection procedures that use machine learning (ML) for automated data analysis. For the most part, the qualification of non-destructive testing (NDT) systems that utilize ML is similar to qualifying more traditional NDT systems. This document provides guidance on the specific considerations related to the use of ML in the qualification process.

1. Introduction

The European methodology for the qualification of non-destructive testing [1] is intended to provide a general framework for the qualification of inspections of specific components to ensure that they are performed in a coherent and consistent way while still allowing qualifications to be tailored in detail to meet different national requirements.

This ENIQ Recommended Practice (RP) will assist those involved in the qualification of non-destructive testing (NDT) systems that use machine learning (ML) models as part of the data analysis procedure. This RP is relevant to any inspection method and builds on the previously published ENIQ position paper “Qualification of an Artificial Intelligence / Machine Learning Non-Destructive Testing System” [2].

2. Objectives

The main objectives of this RP are to:

- Identify the specific challenges related to the use of ML data analysis as part of qualified NDT systems;
- Show the differences to a qualification of a conventional NDT system;
- Promote the harmonisation of practices in qualifying such systems; and
- Provide guidance on how to address the specific challenges and how to qualify such systems and procedures.

Although this document is developed specifically for in-service inspection (ISI) of nuclear power plant (NPP) components, the principles given here can also be applied to the qualification of manufacturing inspections or ISI performed for non-nuclear applications.

3. Machine Learning as Part of a NDT System

3.1. 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 [3]. This is a vast field with long tradition and includes, for example, various expert systems that seek answers from databases based on a set of questions or adaptive sequence of questions. In recent times, the bulk of attention within AI has focused on ML and deep learning systems (see Figure 1).

Various algorithms have been developed to implement ML behaviour. Each of these algorithms provides a wide area of potential applications 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.

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 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 open flawed and unflawed samples including e.g. position, type and orientation of each flaw.

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.

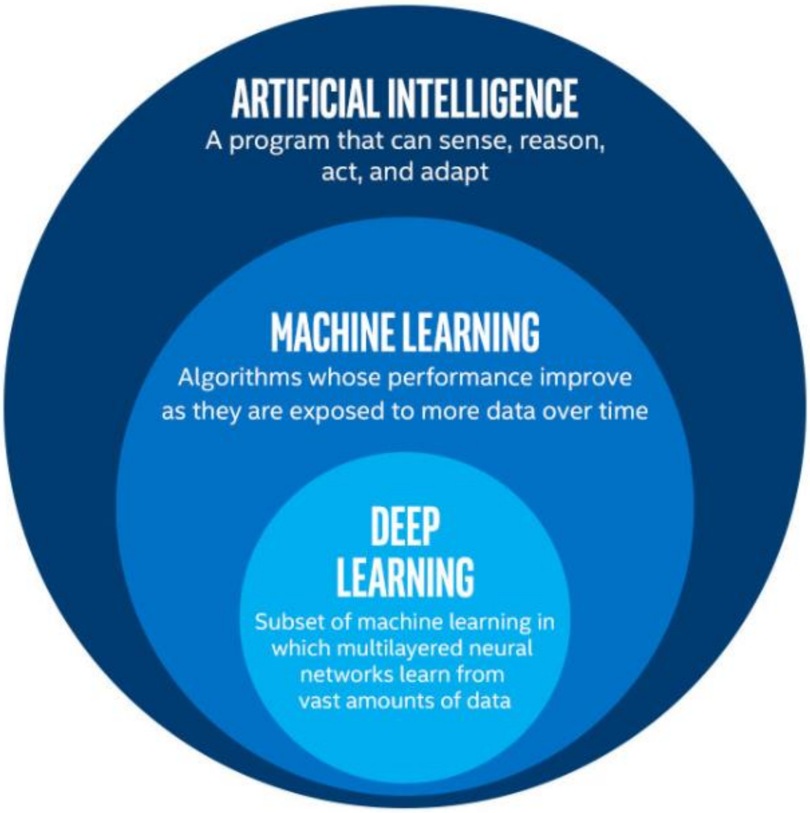
ARTIFICIAL INTELLIGENCE
A program that can sense, reason, act, and adapt
MACHINE LEARNING
Algorithms whose performance improve as they are exposed to more data over time
DEEP LEARNING
Subset of machine learning in which multilayered neural networks learn from vast amounts of data

Figure 1: A schematic overview of AI sub items [4]

In reinforcement learning, the ML algorithm takes active action and learns to optimize its actions to maximize some “reward”, i.e. desired outcome, in a continuous development of itself. 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 dataset.

While all of these ML models offer potential benefits in the field of NDT, supervised learning models can be most easily integrated within the current ENIQ framework. The applicable models can be trained with a controlled, verifiable training dataset and versioned (i.e. the learning frozen) so that an immutable trained model can be qualified with predictable results. Chapter 4 focuses on such supervised learning models while the Appendix provides some high-level descriptions of shallow and deep architectures [5], and the associated methods that are widely deployed 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 software uses reinforced learning after the qualification, this could potentially be justified by freezing 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 utilized in the next qualified revision of the software. Such continuous learning schemes are excluded from the scope of this document.

Analysis of NDT data can be divided in three tasks that need to be solved:

1. Detection needs to be assured by proper criteria for the identification, i.e. false calls excluded.
2. Characterization reveals the nature of the indication (e.g. orientation, surface breaking / embedded, volumetric / planar) within the limitations of the inspection technique.
3. Sizing of indications provides the possibility of further assessment, e.g. fracture mechanics.

3.2. Cases of Use and Extent of Qualification

Data analysis models developed by ML algorithms can be applied with different approaches. The qualification needs to be seen in conjunction with the responsibility and role of the human analyst and for which task the model is supposed to be used. The following cases of use are discussed in terms of the expected extent of qualification:

(1) Primary and classical analysis by humans is independently screened by a ML model.

Obviously, the most conservative approach is using a ML model as a secondary analysis instance. Detection, positioning, characterization and/or sizing could be executed by a ML model. This might reveal false calls of flaws or confirm the classical analysis result. It could be concluded that qualification of the ML model is not required since classical analysis is still performed. On the other hand, it should be considered that humans could be influenced and biased by the ML model results, although the primary and secondary analysis is meant to be independent. This would blur the boundaries to the following case of use and would then require qualification.

(2) Classical analysis by humans is supported by a ML model.

The human analyses the complete dataset, but is alerted in areas where flaw-like patterns are identified by the ML model. The responsibility can be ascribed to the human analyst. Nevertheless, qualification of the ML model is necessary since the human analyst is influenced by the ML model result.

(3) ML model substitutes part of the analysis by humans.

The human analysis is reduced to areas identified by the ML model as suspicious. Human analysis of the entire data would be possible but not necessary and therefore the whole process is more time efficient. The qualification of the ML model needs to focus on the detection and positioning of flaws, since the responsibility for areas without identified indications is taken over by the ML model. If the ML model performs characterisation and/or sizing then qualification of these is necessary since the human analyst may be influenced.

Another example for a substitute is a partly autonomous analysis by the ML model without human interaction or decision in cases where humans' logical reasoning or experience cannot accomplish the task. This might be e.g. a data processing prior to further human analysis. The impact on detection, positioning, characterization and sizing is supposed to be beneficial for the overall performance. This requires qualification of all aspects that are influenced by the task accomplished by the ML model.

(4) Completely autonomous analysis by the ML model.

This requires qualification of all aspects of data analysis: detection, positioning, characterization and sizing.

The above considerations lead to the conclusion that qualification is necessary no matter how a ML model is used.

3.3. Challenge and Opportunity of Machine Learning

For a traditional (non-ML) data analysis the procedure describes analysis criteria, which were determined during the development and demonstration process of the NDT system and is justified using physical reasoning. Additionally, the experience of inspection personnel provides useful and meaningful criteria for detection, positioning, characterization and sizing. Not all of them can be entirely described in a procedure document. Beside the procedure, the data analyst still needs experience for a successful qualification and examination.

The relevant parameters of flaws can be directly observed and suitable test flaws can be justified. In a traditional qualification, it is often possible to cover the scope of a procedure with a limited number of flaws concentrated around worst-case defects, together with the technical justification (TJ) justifying the capability over the complete defect range via physical reasoning and modelling.

Experience and open literature with ML in the NDT community is so far limited. Training of the ML system that provides the analysis criteria may therefore be more obscure and challenging to accept than the experience of the human inspectors we are used to. Thus, more evidence may be required for the ML system in the TJ, as detailed in Section 4.2. In addition, the worst-case defects used in traditional qualifications may need to be adjusted for the ML system.

At the same time, ML systems provide considerable opportunity for improving the performance and reliability of ISI. It is well known that even the highly experienced and qualified inspectors exhibit variations in performance due to “human factors” [6] [7] [8]. Although much care is taken to make the qualification trials as close to the real inspections as possible, there are necessarily differences that may affect the apparent inspector performance [9].

In contrast, ML systems are expected to provide high repeatability and consistency in their application. Their performance is directly observable in the qualification and expected to carry on unaltered in the actual inspection setting.

4. Qualifying Machine Learning NDT Systems

4.1. Example of ML Qualification Flow Chart

The flow chart in Figure 2 highlights the areas in the qualification process where ML has an impact.

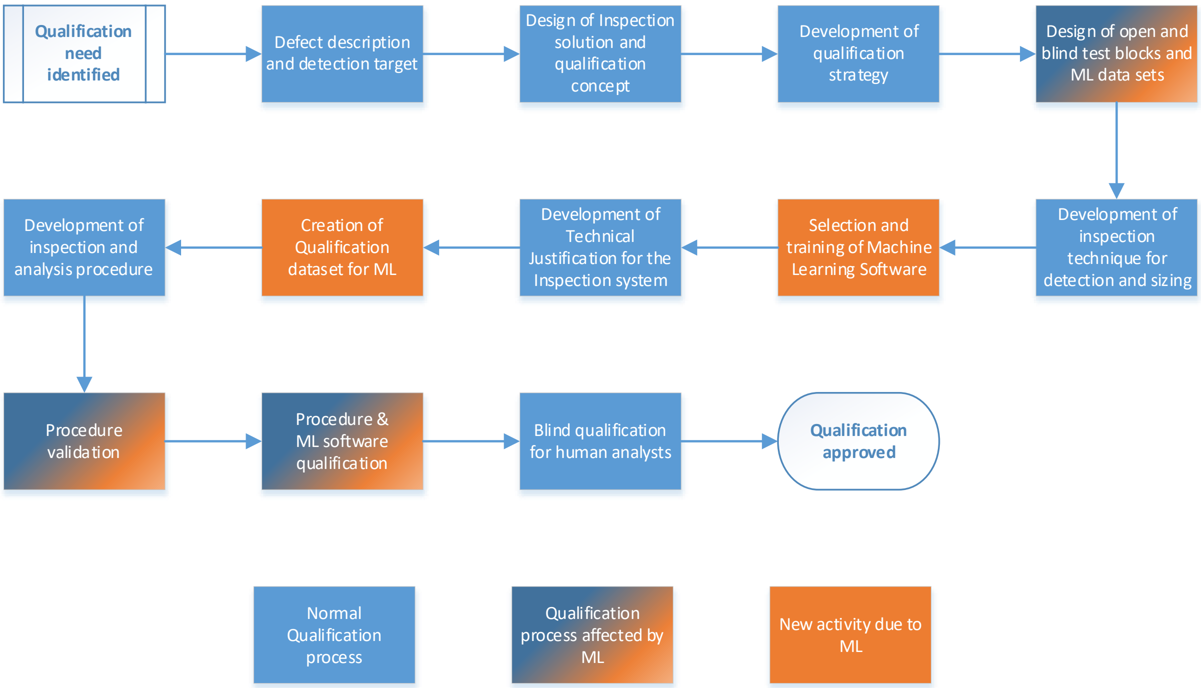
Qualification need identified
Defect description and detection target
Design of Inspection solution and qualification concept
Development of qualification strategy
Design of open and blind test blocks and ML data sets
Development of inspection technique for detection and sizing
Selection and training of Machine Learning Software
Development of Technical Justification for the Inspection system
Creation of Qualification dataset for ML
Development of inspection and analysis procedure
Procedure validation
Procedure & ML software qualification
Blind qualification for human analysts
Qualification approved
Normal Qualification process
Qualification process affected by ML
New activity due to ML

Figure 2: Flow chart highlighting areas in the qualification process where ML has an impact

4.2. Demonstrating the Performance of a ML System

The capability of a ML system is demonstrated in the same manner as for a conventional NDT system using the ENIQ methodology, by a TJ and practical qualification trial. The justification of the capability of the ML elements of the NDT system should be incorporated into the main TJ, although a separate TJ could be produced specifically for the ML elements if required.

It is expected that ML is used in the context of data analysis and the guidelines below are written accordingly.

4.2.1. Specific Aspects for the Technical Justification

The TJ for the NDT system should follow the structure and guidelines of ENIQ RP2 [10] and include justification of the acquisition, human data analysis and ML sub-systems. ML should be considered in the following sections of the TJ:

Section 3: Overview of the NDT System

The NDT system should be presented, detailing the role of the ML system. A summary of the ML system should be given including the reasoning for its use, chosen algorithms and expected results.

Section 4: Analysis of the Influential Parameters

Due to the complexity of any ML system, it is not expected that every parameter is defined and justified. Once the ML system has been fully commissioned and frozen, it can be treated as any other software package. Key parameters along with the software version number and management of revision should be included.

It is anticipated that the justification for a ML system will rely more heavily on experimental evidence than physical reasoning. Thus, the evidence to justify the influential parameters should be sufficient in scope and depth to understand how it impacts the ML decisions concerning the inspection outcome.

ML related parameters to include, but not limited to, are:

- Choice of algorithm

  While many different algorithms may provide a solution for a given inspection case, they may differ in their potential failure modes and the needed justification. The chosen algorithm should be introduced in sufficient detail to allow assessing the potential failure modes and needed justification.

- Dataset

  The quality and size of the dataset used in training and validating the ML model is crucial for overall performance. Thus, the used dataset should be described in sufficient detail to allow assessment of its sufficiency. In particular, the dataset should

  - Cover the inspection target laid out in the input information,
  - Contain representative flaws and representative non-flaws to sufficient extent and
  - Not include any bias that could adversely affect the inspection results.

  It is expected that data scarcity will be one of the key issues in ML development and various data-augmentation schemes and simulated data may be used to expand the training data. Any use of such schemes should be presented.

  During the development of a ML system, the data is commonly divided into:

  1. Training datasets used directly for training the ML model,
  2. Test datasets used to monitor and evaluate the training, and
  3. Qualification datasets used to evaluate the final performance of the model.

  The selection and independence of these datasets have significant effect on the ML performance and should be considered an influential parameter.

Section 5: Physical Reasoning

It is expected that most if not all the essential parameters detailed in Section 4 can be justified by experimental evidence (Section 7). Physical reasoning may be used to justify the use of simulated signals or other data-augmentation schemes.

It is expected that the ML system as a whole can be justified later in Section 9 as the ML element of the NDT system can be treated as inspection software.

Section 7: Experimental Evidence

It is expected that the justification of performance will require use of suitable statistical performance metrics, but this will depend on the system and the way ML is applied. The performance metrics should be selected and justified to suit the particular system to be qualified. The metrics may include hit/miss probability of detection (POD) [11], receiver operating characteristics (ROC) [12] or some other suitable metric.

The confidence of the performance estimates should be assessed. This can be included in the standard methodology applied (e.g. POD) or may be specific to the justification. The performance of a ML model can be assessed by considering the uncertainties linked to the model predictions (model variability). Such uncertainties (also known as epistemic uncertainty) are due to the model variability fit on the provided dataset. Changing the training dataset may impact the ML model online performance. The estimation of this variability can be provided either by using probabilistic ML models or by evaluating different ML models for a given training dataset (e.g., by cross-validation procedure).

Qualification criteria are normally focused on correct detection or characterization/sizing. False call performance often receives secondary attention. In the case of a ML system, the false call rate may serve as a leading indicator of potential performance problems. If an inspection involving ML is applied to data that was not sufficiently represented in the training data, the performance is expected to deteriorate. Since, in the case of NDT, the opportunities of making false calls are much more prevalent than opportunities for making missed calls, it can be expected that deteriorating performance is first observed as an increase in the false call rate. Thus, the false call rate can be used as an early warning signal of a ML system and it may be advisable to screen the ML false call rate closely.

A ML system may be susceptible to a failure mode, where the model performs well on the training and validation data used during development, but fails to generalize properly to unseen defects. This is called overfitting and the absence of overfitting needs to be justified, typically by testing the ML model on previously unseen data.

Section 8: Parametric Studies

Parametric studies may be used, for example, to give further justification for used data-augmentation schemes and to demonstrate the representativeness of simulated data.

Section 9: Equipment, Data Analysis and Personnel Requirements

The ML system should be detailed in this section. Justification should be provided for the influential and essential parameters defined in Section 4. Evidence should be provided on how the system has been commissioned and version controlled.

Section 10: Review of Evidence Presented

A summary should be provided detailing the capability of the ML system demonstrated in the previous sections.

Section 11: Conclusions and Recommendations

The capability of the ML system should be stated with any recommendations for improvement if the full specification could not be achieved.

4.2.2. Qualification Practical Trials

Traditionally, practical trials are conventionally divided into open trials to validate the procedure operations and blind trials to confirm operator performance. The open trial samples may contain flaws smaller than the detection target to gain further confidence on the method performance.

In case of a ML system, the same basic structure is followed. However, the following changes and specific practices are recommended.

For a ML system, the separation into “procedure performance” and “operator performance” is less significant since the role of the operator is diminished. Consequently, it is recommended that the open and blind practical trials are combined into a single blind trial which serves both purposes. Thus, the practical procedure trial should be blind in the sense that the ML system has no prior information about the datasets that will be used or the details of the performance of the system. This is important in order to keep the blind trials truly independent of the model development, previous justification and the TJ. The practical trials should nevertheless contain flaws designed to test the “procedures capability”, similar to conventional open trials. The trial data may, e.g., contain flaws smaller than the detection target to allow the qualification (QB) to assess “performance margin” with respect to the set detection target, even if the detection of these flaws is not required.

The criteria used by the QB should be set for the purpose of performance check, as in a non-ML qualification, and may include, e.g., 100% detection result of flaws greater than or equal to the detection target, allowed root-mean squared (RMS) error and allowed maximum under/oversizing limits for sizing and positioning.

The practical trials typically utilize worst-case defects, meaning that some of the flaws in the practical trials are selected to be especially challenging, to gain additional confidence on the method performance. In practice, these worst-case defects are typically those that result in poor detectability due to physical reasons of the NDT system, and thus are not specific to human inspector judgement. Such use of worst-case defects is also valid for ML systems.

The ML system may exhibit an additional failure mode, where an indication relevant to a human inspector is nevertheless missed or miss-characterized by the ML system. This failure mode would indicate overfitting. To exclude overfitting, the model should be tested on unseen data to assess its performance. The blind practical trials serve as an important final check against overfitting. The definition of worst-case defects for the ML system may introduce considerations that were previously not included for NDT systems with manual data analysis. E.g. flaw types that were under-represented in the training data may be considered worst-case defects for the ML system.

In traditional qualification, it is sometimes possible to refer to previous evidence in the TJ and to complete qualification without practical trials. This may be the case for a ML system as well, e.g. when the ML system has already been used in similar settings and it can be demonstrated that the data the ML system receives has not changed.

5. Summary

ML is well suited to be utilized in all inspection techniques where data can be digitalized. ML systems provide considerable opportunity for improving the performance and reliability of ISI, as even the highly experienced and qualified inspectors exhibit variation in performance. In contrast, ML systems are expected to provide high repeatability and consistency in their application. The ML performance is directly observable during the qualification and expected to carry on unaltered in the actual inspection setting.

For the QBs the main change versus conventional qualifications is the design and maintenance of relevant qualification test blocks and datasets. ML systems may require additional flaws. At the same time, it is recommended that ML systems are qualified using blind trials (in contrast to open and blind for the traditional), which may reduce mock-up requirements.

Qualifications must be done utilizing frozen software (i.e. the learning has stopped) to avoid changes in performance during and after the qualification. The main challenge of inspection vendors is the development of a solid justification presenting the functionality of the frozen ML software.

For the actual inspectors using the qualified ML software, whether as an assisted analysis, detection tool or other, there is no change in the use of the conventional automated analysis as both are versioned software in the eyes of the operator.

Further information is available in industry journal articles relating to the state-of-the-art of ML development, targeting the nuclear inspection industry (see [13] as an example).

References

[1] *The European Methodology for Qualification of Non-Destructive Testing – Issue 4*, ENIQ report no. 61, The NUGENIA Association, 2019.

[2] *ENIQ Position Paper: Qualification of an Artificial Intelligence / Machine Learning Non-Destructive Testing System*, ENIQ report no. 64, the NUGENIA Association, 2020.

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

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

[5] 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.

[6] McGrath, B., 2008. Programme for the assessment of NDT in industry, PANI 3. Retrieved from www.hse.gov.uk/research/rrpdf/rr617.pdf on May 7, 2019.

[7] Bertovic, M., 2015. Human Factors in Non-Destructive Testing (NDT): Risks and Challenges of Mechanised NDT. Technical University Berlin.

[8] Cumblidge, S., D'Agostino, A., Morrow, S., Franklin, C., & Hughes, N., 2017. Review of Human Factors Research in Nondestructive Examination. Retrieved from www.nrc.gov/docs/ML1705/ML17059D745.pdf.

[9] Virkkunen, I., Koskinen, T., & Jessen-Juhler, O. Virtual round robin – a new opportunity to study NDT reliability. To be published.

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

[11] ASTM E2862-18, Standard Practice for Probability of Detection Analysis for Hit/Miss Data, American Society for Testing of Materials (ASTM), 2018.

[12] Fawcett, T., 2005. An introduction to ROC analysis. Pattern Recognition Letters, 27, pp. 861-874.

[13] Miorelli, R., Skarlatos, A., and Reboud, C., 2019. A machine learning approach for classification tasks of ECT signals in steam generator tubes nearby support plate. *AIP Conference – Proceedings*, 2102(1) , 090004, https://doi.org/10.1063/1.5099822 .

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

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

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

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

[18] Salucci, M., Anselmi, N., Oliveri, G., Calmon, P., Miorelli, R., Reboud, C., & Massa, A., 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.

[19] Ahmed, S., Reboud, C., Lhuillier, P. E., Calmon, P., & Miorelli, R., 2019. An adaptive sampling strategy for quasi real time crack characterization on eddy current testing signals. *NDT & E International*, 103, pp. 154-165.

[20] Virkkunen, I., Koskinen, T., Jessen-Juhler, O., & Rinta-Aho, J., 2021. Augmented Ultrasonic Data for Machine Learning. *Journal of Nondestructive Evaluation*, 40(1), pp. 1-11.

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

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

[23] Munir, N., Kim, H.-J., Song, S.-J., & 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.

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

[25] Theodoridis, S., 2015. *Machine learning: a Bayesian and optimization perspective*. Academic Press.

[26] *ENIQ Glossary of Terms – Issue 3*, ENIQ report no. 62, The NUGENIA Association, 2019.

Appendix: Machine Learning Fundamentals

The Machine Learning Stages

Generally speaking, ML algorithms consist of the following stages:

- **Stage 1, 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. The quality and the quantity of the training data is crucial for the performance of the ML model.
- **Stage 2, Data preparation:** The data is formatted and cleaned to remove spurious content that has meaningless information from a physics point of view. At this stage, data and signal processing procedures can be applied as required. These may include e.g. image registration or under sampling.
- **Stage 3, Model choice or training phase:** Once the data has been gathered and prepared, the ML model can be fitted to the data. This phase is known as the training phase and is performed until the model converges towards the stable solution. The training phase is based on a given set of data (i.e., the training dataset) and is referred to as the off-line phase. During this stage, 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.
- **Stage 4, Model evaluation or test phase:** Once the model has been trained it can be tested on a set of data that was not considered in the training phase. This stage is known as the test phase. It does not require any learning and is therefore also known as the on-line phase. The main purpose of this stage is to assess the performance of the ML model before deploying it. The input data used for this stage should be collected and processed following the steps defined in Stages 1 and 2.
- **Stage 5, Prediction or inference:** The final trained model is deployed. As for Stage 4, the input data should be collected and prepared following the steps defined in Stages 1 and 2.

Shallow architectures

Shallow architecture refers to the set of ML methods that exploit the concept of kernel machines (KMs) [14] [15] [16] [17]. In most cases, a supervised learning framework is used to perform classification (e.g., flaw(s)/anomaly detection, defect(s) classification) or regression tasks (parametric-flaw characterization). In order to deal with the cardinality of NDT raw signals, very often the KMs are coped with a dimensionality reduction stage (part of data preparation phase) aiming at shrinking the information content by reducing the redundancy on the learning signals and mitigate the so-called curse-of-dimensionality issue [18]. This stage relies on well-established algorithms in many communities (e.g., chemometric, signal processing, biomedical, etc.) from which one can establish statistical and/or geometrical properties associated with 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 ISOMAP[1], locally linear embedding, etc.

Different learning methods can be collected under the name of KMs. All KM methods rely on the use of the so-called kernel trick [14] [16] [17] in order to perform classification and regression tasks. The kernel trick enables linear interpolations of non-linear data by fitting the model directly in the kernel

[1] ISOMAP is a nonlinear dimensionality reduction method.

space [15]. For the 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 deployed KMs in the literature repose on the vectorization 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 dataset.

The kernel ridge regression (KRR) is the kernel version of the well-known ridge regression [17]. KRR is obtained by formulating the ridge regression exploiting the kernel trick. KRR enables the control on the regression performance through a regularization coefficient that can be tuned in order to maximize the trade-off between variance and bias. Due to its statistical meaning, this penalization coefficient can be a valuable degree of freedom to enhance the model performance in case of noisy measurements. This hyper-parameter, plus the ones associated to the chosen kernel (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 under the name of Kriging in the geoscience community, is a statistical model that exploits the Bayesian framework in order to perform classification and regression tasks [14]. Even though the GP formulation shares many common points with KRR, its statistics enables the 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 requires 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 minimization of a suitable likelihood function.

Support Vector Machines (SVMs) for regression and classification tasks are widely and successfully deployed in many different fields from late nineties. SVMs are based on a mathematical background rooted in the statistical learning theory and structural risk minimization. It allows for theoretical limits to be applied to the SVM model [16][17]. As all the other kernel methods, a SVM model requires tuning of the 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 hyper-parameters is often obtained via cross validation. Compared to KRR and GP, SVM model enables sparse predictions that turns into a computationally efficient model compare to the above-mentioned methods [19].

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

Deep Learning Methods

Deep learning (DL) methods have shown 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 kernel machine model. This makes DL an attractive tool to solve regression and classification problems for end-users that are non-experts in ML [4].

In neural networks, the learned 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 through activation functions and results within a layer are 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 a value is computed at each location. The layers form "feature maps" that encode location-invariant information about the presence and relationships of specified features. This convolution effectively creates weight sharing that significantly 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. [20] used DCNNs to successfully detect cracks in phased array ultrasonic data. Recently Meng et al. [21], Zhu et al. [22] and Munir et al. [23] used DCNNs for defect classification in ultrasonic and EC-data, respectively. In general, the DCNNs are interesting for various flaw detection and classification tasks and various NDT signal data.

Hybrid Learning by Coupling Shallow and Deep Architecture Methods

Coupling between shallow and deep architectures is a valuable solution in order to fully take advantage of the solid mathematical background of KMs and the non-invasive (i.e., no-feature engineering stage required) associated with DL approaches. This kind of hybrid learning approach is widely studied and exploited by the ML community. It has been shown that hybrid learning approaches perform better than KMs and DL methods for complex classification tasks [24].

Decision Trees

Decision trees (DTs) for classification and regression are non-parametric supervised methods that can be applied to classification or regression problems [25]. The learning model is created by inferring a decision rule based on data features. The decision chain is started from the root containing all features and by successively splitting them while moving to the root children (called leaf node). The algorithm stops when the whole tree depth is explored and convergence is achieved. Classification and Regression Trees (CART) are the most widely used DT algorithms, which may also be used for sensitivity analysis purposes.

The main strength of DTs consists in the fact that decisions are intuitive and easy to interpret (one can visualize the decision graph). For this reason they are considered as white box models. Furthermore, there is very little need to prepare data (scaling features) and the algorithm scales logarithmically with respect to the training dataset size and can handle numerical and categorical data. The main disadvantages of DTs are that they are prone to overfitting and that they are unstable with respect to small variations in input data (a completely different tree can be obtained). These disadvantages can be mitigated by deploying a ML method known as ensemble learning.

Ensemble Learning

Ensemble learning (EL) is based on the concept that the aggregation of different ML methods (classifiers or regressors) may give better prediction performance [25]. A group of predictions is referred to as an ensemble, thus this technique is referred as ensemble learning or method.

Within EL different and possibly heterogeneous ML algorithms can be combined together. E.g. to mitigate the disadvantages of DTs, one can combine different DTs together based on different input subsets in order to obtain the prediction of all the DTs together. Such an algorithm is known as random forest. In the recent past the ML scientific community has developed different EL algorithms, most notably voting, bagging, boosting, stacking, etc. [25].

Design of Datasets

In the last decades ML research community have developed many ML models aiming to resolving different kind of tasks ranging from image recognition, outlier detection, speech recognition, etc. Generally speaking, among the best performer ML methods, one cannot a-priori infer that a ML model with good performance in terms of accuracy and efficiency can be obtained for different problems. This is also true for the NDT domain. Addressing ML problems, i.e., detection, classification or regression, may require different ML models for each task. Furthermore, in order to obtain the desired degree of performance for a given task, different ML models could be used for different inspection methods and techniques. Evaluation of a batch of different candidate ML models is a robust and feasible approach for a given inspection problem.

Apart from the choice of a suitable ML model, the availability of labelled data to train the ML model is crucial. Indeed, the performance of a ML method depends significantly on the input data of the algorithm. The input data of a supervised learning strategy is commonly referred to as training dataset. The training dataset is composed of a set of input data (i.e., NDT probed signals) and output (or targets) (e.g., flaw(s) categories, flaw detection, flaw(s) parameters estimation, etc.) data, which represent all the information available to the ML method in order to establish a suitable model that links these two datasets. In order to evaluate the ML model it must be tested for an unseen dataset referred to as a test dataset. In order to evaluate the ML model performance, it is mandatory that the test dataset must not contain any data points of the training dataset. In case of DL models, one should also account for the creation of a validation dataset which is used to check the DL model accuracy in the training phase.

In the context of NDT, ML methods can be trained based on both labelled experimental and simulated data. The former is referred to as data-driven approach, the latter is referred to as physics-driven approach, since the data is generated by a numerical model. A mix of mode- and data-driven approaches can also be deployed. The generation of a training dataset based on a sub-set of simulated signals can be seen as a viable way to add *a-priori* knowledge to the ML algorithm. In case of scarce labelled experimental data, the synthetic data can improve the generalization capabilities of the ML model and can give a model that is less prone to errors.

Training Dataset

One of the most important issues in developing a supervised ML system is the choice of the training and test samples to properly fit and test the ML model. The training dataset should be representative enough to contain a meaningful set of inspection configurations for a given inspection case (e.g., inspection of corrosion via eddy current testing, weld inspection via multi-element probe, etc.). The suitable number of training samples depends on the deployed ML algorithm (shallow, deep architectures) as well as on the ML model parameters. The correct number of training samples cannot be fixed a-priori. Instead of fixing a given number of samples, one should demonstrate that a ML algorithm is able to increase its performance when more labelled input data is available. Doing so they can infer on the convergence of the model with respect to the deployed training sample.

The choice of the training dataset has an impact on the choice of the ML model to be deployed. Working with labelled data (i.e., supervised learning) enables access to classes of membership or to continuous (physical) values. The ML model should be able to deal with different training datasets in terms of input and target cardinality and volume, like e.g. prediction of discrete (i.e. classes) or continuous, multi-variate values. The type of the targets impacts the performance of the ML model in terms of accuracy training (and prediction) time. E.g. one can expect that working with A-scan, B-scan or C-scan signals will not require the same computational effort due to the cardinality of the treated problem. This implies that some ML models are more suitable than others.

Test Dataset

The test phase should contain the needed number of samples to check the robustness of the deployed ML system. The test dataset should only be used to infer the potential of the ML model to generalization capabilities (i.e., its robustness) on unseen test data. The experiments should be representative for a variety of cases belonging to the same family of problems evaluated in the training phase. To avoid bias in the obtained results, none of the test samples should be used in the training phase. The number of test samples should be large enough to empirically provide statistical insight (mean value and variance of predictions) of the obtained predictions.

Qualification Dataset

The qualification dataset has the same requirements as the test dataset, but with the caveat that this dataset is designed and controlled by the qualification body (QB) and can only be used for qualification purposes in order to avoid bias.

Guidelines on the Choice of ML Model and their Application in NDT

The performance of a ML model significantly depends upon the data to which the ML model is fitted. For a given inspection method the generalization capability is linked to both the information content of the training dataset and its cardinality. A given ML model may provide better performance than another one depending on the data, i.e. the considered signals / measurements. In NDT, these signals depend upon the inspection method and technique. In general we classify these signals as scalars, time varying (A-scans, B-scans, and C-scans), 2D-images, 3D-images, etc. Such kind of signals may be found in different inspection methods and techniques such as ultrasonic testing, eddy-current testing, etc. Based on the application case the following recommendation can be given about ML-models to choose.

- **Classification based on scalar signals (e.g., flaw(s) detection tasks)**: ML methods such as shallow and deep architectures should provide thorough performance.
- **Classification based on images-like signals (e.g., flaw(s) classification tasks)**: ML methods such as shallow and deep architectures should provide thorough performance.
- **Classification based on time-domain-like signals (e.g., flaw(s) classification tasks)**: Off-the-shelf DL architectures may have an edge compared to traditional shallow architectures for which careful data preparation, i.e., feature engineering, may be needed
- **Regression based on images-like signals (e.g., flaw(s) sizing tasks)**: ML methods such as shallow and deep architectures should provide thorough performance. For high-dimensional regression problems (e.g., more than fifteen dimensions) DL architectures may provide an edge in performance due to their learning procedure.

- **Regression based on time-domain-like signals (e.g., flaw(s) sizing tasks)**: Off-the-shelf DL architectures may have an edge compared to traditional shallow architectures for which a careful data preparation, i.e., feature engineering, may be needed.

According to the research in the field of ML it can be concluded that DL architectures often required a high amount of training data in order to achieve good accuracy levels. Therefor the choice between shallow or DL architectures also depends on the amount of available training data for a given task.

Model Evaluation

Assessing the performance of a ML system is crucial. To address this issue the ML research community has provided a wide set of metrics that can be used to quantitatively compare different ML systems. Two major families of metrics can be distinguished with respect to the problem that needs to be solved, regression or classification.

In case of a regression problem, one can rely on different error metrics that are commonly used to evaluate the performance of a ML model. The most common metrics are provided in **Error! Reference source not found.**. In case of classification along with an accuracy estimation, one would like to carry out a study of the ML model performance itself. This concept is much related to the evaluation of the performance of an inspection procedure via POD studies. For ML systems one rarely relies on scalar inputs as for POD studies. Thus, the concept of detection threshold cannot be applied for flaw detection in a straightforward manner. To address this issue tools like ROC curve, Precision/Recall curves and other quantities that can be derivate from Fawcett [12] are used instead.

ROC curves have been developed to evaluate the detection performance in early radar systems in noisy environments and have been applied for a long time in biomedical and ML domains to evaluate the presence of possible diseases or pathologies. ROC curves are in particular useful in case of with skewed/ unbalanced class distribution and to evaluate cost-sensitive learning. Such kind of problems is quite common in real inspection for which very few anomalies are detected among many other signal indications.

Table 1: Common error metrics to assess the performance of a ML model for regression purposes

Error metric	
Mean-squared error	$\mathrm{MSE} = \frac{1}{N}\sum_{i=0}^{N}(p_i - a_i)^2$
Root mean-squared error	$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=0}^{N}(p_i - a_i)^2}$
Mean-absolute error	$\mathrm{MAE} = \frac{1}{N}\sum_{i=0}^{N}\lvert p_i - a_i\rvert$
Relative-squared error*	$\mathrm{RSE} = \frac{\sum_{i=0}^{N}(p_i - a_i)^2}{\sum_{i=0}^{N}(a_i - \bar{a})^2}$
Root relative-squared error*	$\mathrm{RRSE} = \sqrt{\frac{\sum_{i=0}^{N}(p_i - a_i)^2}{\sum_{i=0}^{N}(a_i - \bar{a})^2}}$
Relative-absolute error*	$\mathrm{RAE} = \frac{\sum_{i=0}^{N}\lvert p_i - a_i\rvert}{\sum_{i=0}^{N}\lvert a_i - \bar{a}\rvert}$
Coefficient of determination**	$\mathrm{R}^2 = 1 - \frac{\sum_{i=0}^{N}(a_i - p_i)^2}{\sum_{i=0}^{N}(a_i - \bar{a})^2}$
Where in (*) $\bar{a}$ is the mean value over the training data and in (**) $\bar{a}$ is over the test data. p_i stands for the i-th predicted instance and a_i is its actual counterpart.	

Glossary

The general definitions in the ENIQ Glossary [26] are applicable to this RP. In addition, the following definitions apply.

Artificial Intelligence	Computer systems that mimic human intelligence or perform tasks that have been thought to require human intelligence.
Machine learning (ML) method	Algorithm to create and improve the performance of a model through training data.
Machine learning (ML) model	Programme for the prediction of a result based on input data. The prediction can be related to regression or classification problems.
Training dataset	A digital set of NDT data, representative of the inspection configuration, used to develop and teach a ML system.
Test dataset	A digital set of NDT data, representative of the inspection configuration, kept independent to the training data and used to verify functionality outside the training data.
Qualification dataset	A digital set of NDT data, representative of the inspection configuration, controlled and managed by the qualification body and only used for the qualification of the ML system.

Contributors to Drafting and Editing

Iikka Virkkunen	Trueflaw Ltd.	Finland
Martin Bolander	Westinghouse Electric	Sweden
Heikki Myöhänen	Kiwa Inspecta	Finland
Roberto Miorelli	CEA	France
Ola Johansson	Swedish Qualification Centre (SQC)	Sweden
Philip Kicherer	Swiss Association for Technical Inspections (SVTI)	Switzerland
Chris Curtis	Jacobs / Inspection Validation Centre (IVC)	Great Britain
Oliver Martin	European Commission – Joint Research Centre	European Commission

ENIQ

European Network for
Inspection & Qualification
NUGENIA Technical Area 8

ABOUT ENIQ AND SNETP

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 (RI-ISI) for nuclear power plants (NPPs). Since its establishment in 1992 ENIQ has issued over 60 documents. Among them are the “European Methodology for the 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 is the technical area 8 of NUGENIA, one of the three pillars of the Sustainable Nuclear Energy Technology Platform (SNETP) that was established in September 2007 as a R&D&I platform **to support technological development for enhancing safe and competitive nuclear fission in a climate-neutral and sustainable energy mix.** Since May 2019, SNETP has been operating as an international non-profit association (INPA) under the Belgian law pursuing a networking and scientific goals. It is recognised as a European Technology and Innovation Platform (ETIP) by the European Commission.

The international membership base of the platform includes industrial actors, research and development organisations, academia, technical and safety organisations, SMEs as well as non-governmental bodies.

secretariat@snetp.eu www.snetp.eu SNETP SNE_TP

9 782919 313280