Moving Picture, Audio and Data Coding
by Artificial Intelligence
www.mpai.community

MPAI Technical Specification

Neural Network Traceability
MPAI-NNT

V1.0

WARNING

Use of the technologies described in this Technical Specification may infringe patents, copyrights or intellectual property rights of MPAI Members or non-members.

MPAI and its Members accept no responsibility whatsoever for damages or liability, direct or consequential, which may result from the use of this Technical Specification.

Readers are invited to review Annex 2 – Notices and Disclaimers.

Neural Network Traceability
V1.0

1 Foreword

The international, unaffiliated, non-profit ***Moving Picture, Audio, and Data Coding by Artificial Intelligence (MPAI)*** organisation was established in September 2020 in the context of:

1. **Increasing** use of Artificial Intelligence (AI) technologies applied to a broad range of domains affecting millions of people
2. **Marginal** reliance on standards in the development of those AI applications
3. **Unprecedented** impact exerted by standards on the digital media industry affecting billions of people believing that AI-based data coding standards will have a similar positive impact on the Information and Communication Technology industry.

The design principles of the MPAI organisation as established by the MPAI Statutes are the development of AI-based Data Coding standards in pursuit of the following policies:

1. Publish upfront clear Intellectual Property Rights licensing frameworks.
2. Adhere to a rigorous standard development process.

3. Be friendly to the AI context but, to the extent possible, remain agnostic to the technology thus allowing developers freedom in the selection of the more appropriate – AI or Data Processing – technologies for their needs.
4. Be attractive to different industries, end users, and regulators.
5. Address five standardisation areas:
    1. *Data Type*, a particular type of Data, e.g., Audio, Visual, Object, Scenes, and Descriptors with as clear semantics as possible.
    2. *Qualifier*, specialised Metadata conveying information on Sub-Types, Formats, and Attributes of a Data Type.
    3. *AI Module* (AIM), processing elements with identified functions and input/output Data Types.
    4. *AI Workflow* (AIW), MPAI-specified configurations of AIMs with identified functions and input/output Data Types.
    5. *AI Framework* (AIF), an environment enabling dynamic configuration, initialisation, execution, and control of AIWs.
6. Provide appropriate Governance of the ecosystem created by MPAI Technical Specifications enabling users to:
    1. *Operate* Reference Software Implementations of MPAI Technical Specifications provided together with Reference Software Specifications
    2. *Test* the conformance of an implementation with a Technical Specification using the Conformance Testing Specification.
    3. *Assess* the performance of an implementation of a Technical Specification using the Performance Assessment Specification.
    4. *Obtain* conforming implementations possibly with a performance assessment report from a trusted source through the MPAI Store.

Today, the MPAI organisation operated on four solid pillars:

1. The MPAI Patent Policy specifies the MPAI standard development process and the Framework Licence development guidelines.
2. *Technical Specification: Artificial Intelligence Framework (MPAI-AIF)* specifies an environment enabling initialisation, dynamic configuration, and control of AIWs in the standard AI Framework environment depicted in Figure 1. An AI Framework can execute AI applications called AI Workflows (AIW) typically including interconnected AI Modules (AIM). MPAI-AIF supports small- and large-scale high-performance components and promotes solutions with improved explainability.

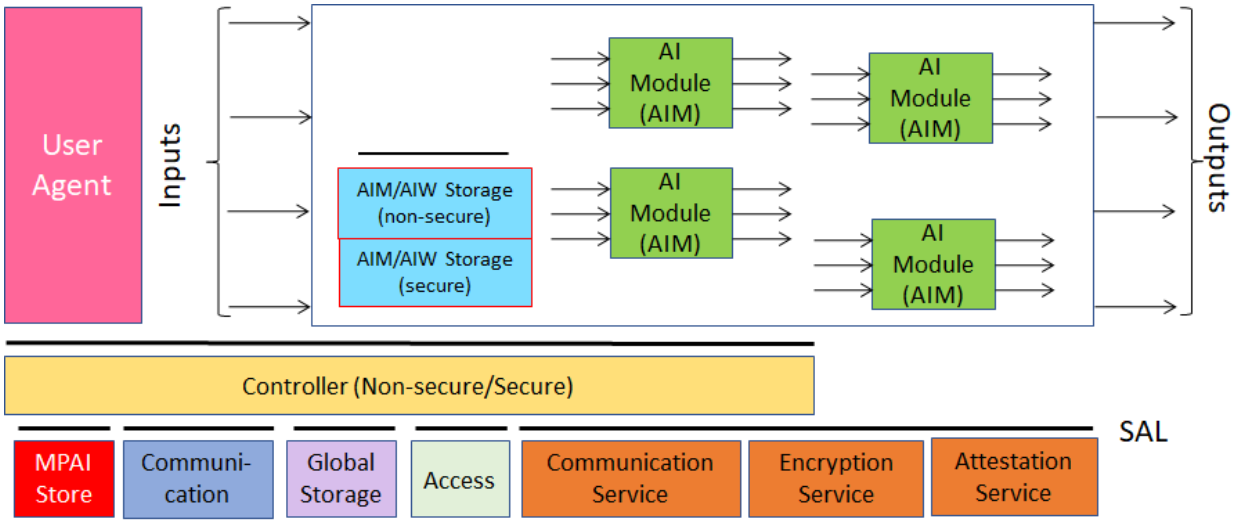

Figure 1 – The AI Framework (MPAI-AIF) V2 Reference Model

3. ***Technical Specification: Data Types, Formats, and Attributes (MPAI-TFA) V1.0*** specifies Qualifiers, a type of metadata supporting the operation of AIMs receiving data from other AIMs. Qualifiers convey information on Sub-Types (e.g., the type of colour), Formats (e.g., the type of compression and transport), and Attributes (e.g., semantic information in the Content). Although Qualifiers are human-readable, they are only intended to be used by AIMs. Therefore, Text, Speech, Audio, Visual, and other Data exchanged by AIWs and AIMs should be interpreted as being composed of Content (Text, Speech, Audio, and Visual as appropriate) and associated Qualifiers. Therefore a Text Object is composed of Text Data and Text Qualifier. The specification of most MPAI Data Types reflects this point.

4. ***Technical Specification: Governance of the MPAI Ecosystem (MPAI-GME) V1.1*** defines the following elements:
    1. Standards, i.e., the ensemble of Technical Specifications, Reference Software, Conformance Testing, and Performance Assessment.
    2. Developers of MPAI-specified AIMs and Integrators of MPAI-specified AIWS (Implementers).
    3. MPAI Store in charge of making AIMs and AIWs submitted by Implementers available to Integrators and End Users.
    4. Performance Assessors, independent entities assessing the performance of implementations in terms of Reliability, Replicability, Robustness, and Fairness.
    5. End Users.

The interaction between and among actors of the MPAI Ecosystem are depicted in Figure 2.

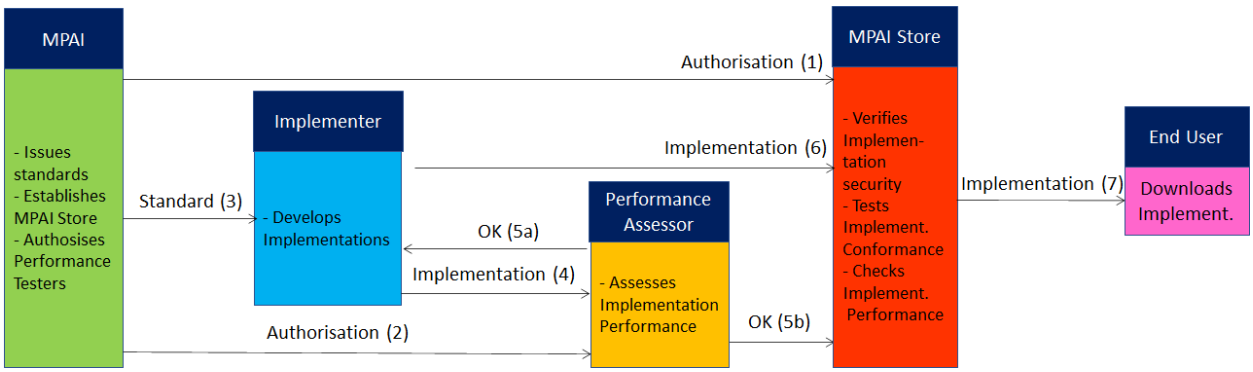

Figure 2 – The MPAI Ecosystem

2 Introduction (Informative)

Traceability enables identification of the origin or verification of the integrity of data. In the case of Neural Networks, Traceability enables understanding and tracking the development, use, and evolution of Neural Network models. A variety of methods have been developed for NN Traceability, and can be divided into two categories:

- Watermarking method, an Active method which alters the Weights of the NN to insert Traceability Data.
- Fingerprinting method, a Passive method which does not alter the Weights of the NN.

Numerous Traceability methods have been published since 2017 [6], especially for watermarking.

Technical Specification: Neural Network Watermarking (MPAI-NNW) V1.0 provides tools to evaluate Watermarking methods, for a given Payload, on three properties: Imperceptibility, Robustness, and Computational Cost.
Technical Specification: Neural Network Traceability (MPAI-NNT) V1.0 provides tools to evaluate both types of Traceability methods keeping the methods included in MPAI-NNW V1.0.

In all Chapters and Sections, Terms beginning with a capital letter are defined in Table 1 if they are specific to this Technical Specification and in Table 2 if they are common to all MPAI Technical Specifications. All Chapters, Sections, and Annexes are Normative unless they are labelled as Informative.

3 Scope of Standard

MPAI-NNT specifies methods to evaluate the following aspects of Active (Watermarking) and Passive (Fingerprinting) Neural Network Traceability Methods in terms of:

- Ability of a Neural Network Traceability Detector/Decoder/Matcher to detect/decode/match Traceability Data when the traced Neural Network has been modified.
- Computational cost of injecting, extracting, detecting, decoding, or matching Traceability Data.
- Specifically for active tracing methods, impact on the performance of a neural network with inserted Traceability Data and its inference.

The standard assumes that:

- The Neural Network Traceability Method to be evaluated according to this standard is known to the Tester.
- The performance of the Neural Network Traceability Method does not depend on specific Secret Keys.

This Technical Specification has been developed by the MPAI Neural Network Watermarking Development Committee (NNW-DC). As the Neural Network Traceability area is fast-evolving, MPAI expects it will produce future MPAI-NNT versions providing methods to cope with technology evolution.

4 Terms and Definitions

The terms used in this standard whose first letter is capital have the meaning defined in *Table 1*.
Note: in Table 1 and in the following, Neural Network may be abbreviated as NN.

Table 1 – Table of terms and definitions

Term	Definition
Active Traceability Method	A Traceability Method that alters the NN Weights.
Algorithmic Integrity	The equivalence of the Traceability Data extracted from a modified NN and those extracted from an unmodified NN.
Computational Cost	The cost of injecting, Detecting, Decoding or Matching Traceability Data.
Detection	The process of finding the presence of a known watermark in a NN.
Decoding	The process of extracting the Payload from a watermarked NN.
Extraction	The process of computing the fingerprint from an NN.
Imperceptibility	A difference in the performance of an NN before and after the

	watermark embedding process.
Matching	The process of finding a fingerprint in a database that correspond to the fingerprint computed from an NN.
Means	Procedure, tools, dataset or dataset characteristics used to evaluate one or more of Computational Cost, Imperceptibility, or Robustness of a NN Traceability method.
Modification	The result of an attack performed during NN Traceability testing.
Neural Network	or Artificial Neural Network, a set of interconnected data processing nodes whose connections are affected by Weights.
NN Fingerprinting Method	A NN Passive Traceability method that extracts NN identification data from the NN Weights and matches it to a known repository.
NN Traceability	The possibility to identify the source and/or a potential Modification of a NN.
NN Watermarking Method	A NN Active Traceability method that injects Traceability Data into the Weights or the activation function of a NN to subsequently enable a Decoder/Detector to decode/detect the injected Traceability Data.
Parameter	A set of values characterizing Type and Intensity of a Modification, as used in Table 3.
Passive Traceability Method	A Traceability Method that does not alter the NN Weights.
Payload	The Symbols carried by a watermark.
Robustness	The ability of a NN Traceability method to withstand a Modification in terms of Detection, Decoding or Matching capability.
Secret Key	The data that the Traceability method requires to be kept secret.
Symbol	A binary, numerical, or string element in a Payload.
Tester	The user who evaluates a NN Traceability Method according to this Technical Specification.
Traceability	The possibility to trace the origin of data.
Traceability Data	The data to be inserted by the Active Traceability method or the result of the application of a Detection algorithm to an NN for a Passive Traceability Method.
Weight	The value used to multiply the connection between two nodes of a NN.

The capitalized Terms used in this Technical Specification that are not already included in *Table 1* are defined in *Table 2*.

Table 2 – MPAI-wide Terms

Term	Definition
Access	Static or slowly changing data that are required by an application such as domain knowledge data, data models, etc.
AI Framework (AIF)	The environment where AIWs are executed.
AI Module (AIM)	A data processing element receiving AIM-specific Inputs and producing AIM-specific Outputs according to according to its Function. An AIM may be an aggregation of AIMs.
AI Workflow (AIW)	A structured aggregation of AIMs implementing a Use Case receiving AIW-specific inputs and producing AIW-specific outputs according to the AIW Function.

Application Standard	An MPAI Standard designed to enable a particular application domain.
Channel	A connection between an output port of an AIM and an input port of an AIM. The term "connection" is also used as synonymous.
Communication	The infrastructure that implements message passing between AIMs
Component	One of the 7 AIF elements: Access, Communication, Controller, Internal Storage, Global Storage, Store, and User Agent
Conformance	The attribute of an Implementation of being a correct technical Implementation of a Technical Specification.
Conformance Tester	An entity Testing the Conformance of an Implementation.
Conformance Testing	The normative document specifying the Means to Test the Conformance of an Implementation.
Conformance Testing Means	Procedures, tools, data sets and/or data set characteristics to Test the Conformance of an Implementation.
Connection	A channel connecting an output port of an AIM and an input port of an AIM.
Controller	A Component that manages and controls the AIMs in the AIF, so that they execute in the correct order and at the time when they are needed
Data Format	The standard digital representation of data.
Data Semantics	The meaning of data.
Ecosystem	The ensemble of actors making it possible for a User to execute an application composed of an AIF, one or more AIWs, each with one or more AIMs potentially sourced from independent implementers.
Explainability	The ability to trace the output of an Implementation back to the inputs that have produced it.
Fairness	The attribute of an Implementation whose extent of applicability can be assessed by making the training set and/or network open to testing for bias and unanticipated results.
Function	The operations effected by an AIW or an AIM on input data.
Global Storage	A Component to store data shared by AIMs.
Internal Storage	A Component to store data of the individual AIMs.
Identifier	A name that uniquely identifies an Implementation.
Implementation	1. An embodiment of the MPAI-AIF Technical Specification, or 2. An AIW or AIM of a particular Level (1-2-3) conforming with a Use Case of an MPAI Application Standard.
Implementer	A legal entity implementing MPAI Technical Specifications.
ImplementerID (IID)	A unique name assigned by the ImplementerID Registration Authority to an Implementer.
ImplementerID Registration Authority (IIDRA)	The entity appointed by MPAI to assign ImplementerID's to Implementers.
Interoperability	The ability to functionally replace an AIM with another AIW having the same Interoperability Level
Interoperability Level	The attribute of an AIW and its AIMs to be executable in an AIF Implementation and to: 1. Be proprietary (Level 1) 2. Pass the Conformance Testing (Level 2) of an Application Standard

	3. Pass the Performance Testing (Level 3) of an Application Standard.
Knowledge Base	Structured and/or unstructured information made accessible to AIMs via MPAI-specified interfaces
Message	A sequence of Records transported by Communication through Channels.
Normativity	The set of attributes of a technology or a set of technologies specified by the applicable parts of an MPAI standard.
Performance	The attribute of an Implementation of being Reliable, Robust, Fair and Replicable.
Performance Assessment	The normative document specifying the Means to Assess the Grade of Performance of an Implementation.
Performance Assessment Means	Procedures, tools, data sets and/or data set characteristics to Assess the Performance of an Implementation.
Performance Assessor	An entity Assessing the Performance of an Implementation.
Profile	A particular subset of the technologies used in MPAI-AIF or an AIW of an Application Standard and, where applicable, the classes, other subsets, options and parameters relevant to that subset.
Record	A data structure with a specified structure
Reference Model	The AIMs and theirs Connections in an AIW.
Reference Software	A technically correct software implementation of a Technical Specification containing source code, or source and compiled code.
Reliability	The attribute of an Implementation that performs as specified by the Application Standard, profile and version the Implementation refers to, e.g., within the application scope, stated limitations, and for the period of time specified by the Implementer.
Replicability	The attribute of an Implementation whose Performance, as Assessed by a Performance Assessor, can be replicated, within an agreed level, by another Performance Assessor.
Robustness	The attribute of an Implementation that copes with data outside of the stated application scope with an estimated degree of confidence.
Scope	The domain of applicability of an MPAI Application Standard
Service Provider	An entrepreneur who offers an Implementation as a service (e.g., a recommendation service) to Users.
Standard	The ensemble of Technical Specification, Reference Software, Conformance Testing and Performance Assessment of an MPAI application Standard.
Technical Specification	(Framework) the normative specification of the AIF. (Application) the normative specification of the set of AIWs belonging to an application domain along with the AIMs required to Implement the AIWs that includes: 1. The formats of the Input/Output data of the AIWs implementing the AIWs. 2. The Connections of the AIMs of the AIW. 3. The formats of the Input/Output data of the AIMs belonging to the AIW.
Testing Laboratory	A laboratory accredited to Assess the Grade of Performance of Implementations.

Time Base	The protocol specifying how Components can access timing information
Topology	The set of AIM Connections of an AIW.
Use Case	A particular instance of the Application domain target of an Application Standard.
User	A user of an Implementation.
User Agent	The Component interfacing the user with an AIF through the Controller.
Version	A revision or extension of a Standard or of one of its elements.
Zero Trust	A model of cybersecurity primarily focused on data and service protection that assumes no implicit trust.

5 Use cases (Informative)

This chapter provides a selection of NN Traceability use cases together with the types of actors playing roles in them. These are provided for information and are not intended to cover all possible uses of the standard.

The following use cases can relate to both watermarking and fingerprinting:

- *Identify an NN*
  In this use case, the Traceability Data gives information about the source of the NN.
- *Verify the Algorithmic Integrity of an NN*
  In this use case, the Traceability Data gives information about whether the NN Algorithmic Integrity is preserved or not.

The following use cases only relate to watermarking:

- *Identify the actors* (e.g., NN customer, NN end-user, NN owner, and NN watermarking provider) *of an NN*
  In this use case, the Payload conveys information about some or all actors.
- *Multiple watermarking*
  In this use case, the Payload conveys information about how multiple users may change the NN.

6 References

6.1 Normative references

MPAI-AIF normatively references the following documents:

1. MPAI; Technical Specification; AI Framework (MPAI-AIF) V2.1 https://mpai.community/standards/mpai-aif/

6.2 Informative references

2. MPAI; The MPAI Statutes; https://mpai.community/statutes/
3. MPAI; The MPAI Patent Policy; https://mpai.community/about/the-mpai-patent-policy/ .
4. MPAI; Framework Licence of the Artificial Intelligence Framework Technical Specification (MPAI-AIF); https://mpai.community/standards/mpai-aif/framework-licence/
5. Technical Specification: The Governance of the MPAI Ecosystem V1, 2021; https://mpai.community/standards/mpai-gme/
6. Y. Uchida, Y. Nagai, S. Sakazawa, and S. Satoh, “Embedding Watermarks into Deep Neural Networks,” Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval, pp. 269–277, Jun. 2017, doi: 10.1145/3078971.3078974.

7 Imperceptibility evaluation

This chapter will deal with two watermarking-related cases:

- NNs for which the watermark is added after the NNs model was created.
- NNs for which the watermark is added during the training of the NNs model.

7.1 Post-training watermark embedding

The Imperceptibility evaluation specifies the Means that enable a Tester to evaluate the differences in performance of a NN before and after the watermark embedding process. There are two cases:

1. The NN has the input and output data format with specified semantics.
2. The input and output data format of the NN do not have specified semantics.

7.1.1 NN with I/O data format has specified semantics

In this section, two actors are involved: the NN Watermarking provider requesting a Tester to evaluate the Imperceptibility performance of their NN Watermarking Method.

The Tester shall adopt the following procedure:

1. Define a pair of training and testing datasets with a size with at least an order of magnitude more entries than trainable Weights.
2. Select:
    a. A set of *M* unwatermarked NNs trained on the training dataset.
    b. *D* data Payloads corresponding to the pre-established Payload size.
3. Apply the NN Watermarking Method to the *M* NNs.
4. Process the training dataset and the *D* data Payload (if needed).
5. Feed the M unwatermarked NN with the test dataset
6. Measure the task-dependent quality of the produced inference.
7. Feed the M x D watermarked NN with the same test dataset
8. Measure the task-dependent quality of the produced inference, informative examples of quality evaluation are provided in Annex 3 .
9. Provide the task-dependent quality of the produced inference measured in 6 and 7.

7.1.2 NN with I/O data format has no specified semantics

In this section, two actors are involved: the NN Watermarking provider requesting a Tester to evaluate the Imperceptibility performance of their Watermarking Method.

The workflow of the process shall be the following:

1. Tester connects the NN to other NN until the input and output of the resulting configuration have input / output formats with specified semantics.
2. Tester applies all the steps in 6.1.1.

7.2 In-training watermark embedding

The Imperceptibility evaluation specifies the Means for evaluating the performance of a watermarked NN. The workflow of the process shall evaluate the watermarked NN as an NN.

8 Robustness evaluation

The Robustness evaluation specifies the Means to enable a Tester to evaluate the Robustness of the Traceability Data against a set of Modifications requested by one of the Actors.

8.1 Watermarking

The Tester evaluates the Decoder, and Detector of a NN Watermarking Method as specified in the following workflow:

1. Select:
    a. A set of *M* unwatermarked NNs trained on the training dataset.
    b. *D* data Payloads corresponding to the pre-established Payload size.
2. Apply the NN Watermarking Method to the *M* NNs with the *D* data Payloads
3. Produce a set of *M* x (*D* + 1) modified NNs (*M* unwatermarked NNs and *M* x *D* watermarked NNs), by applying one of the Modifications in Table 3 to a given Parameter value.
4. Evaluate the Robustness of the Detector:
    a. Find the presence of the watermark in all *M* x (*D* + 1) NNs using the Watermark Detector.
    b. Record the corresponding binary detection results (*Yes* – the mark is detected or *No* – the mark is not detected).
    c. Label the Yes/No outputs of the Watermark Detector as *true positive*, *true negative*, *false positive* (*false alarm*) and *false negative* (*missed detection*) according to the actual result.
    d. Count the total number of *false positives* and the total number of *false negatives*.
5. Evaluate the Robustness of the Decoder:
    a. Extract the Payload from all *M* x (*D* + 1) NNs using the Watermark Decoder.
    b. Count the number of different Symbols between the outputs of the Decoder and their corresponding original data Payloads.
    c. Compute the Symbol Error Rate (SER) for any of the *M* x (*D* + 1) NNs, as the ratio of the number of different Symbols to the number of the Symbols in the data Payload.
    d. Compute the average SER, as the average over the *M* x (*D* + 1) SER values computed in the previous step.
6. Provide the average values over the total number of tests:
    a. The ratio of the number of *false positives* to *M* x (*D* + 1),
    b. The ratio of the number of *false negatives* to *M* x (*D* + 1).
    c. The M x D number for tested NNs, and the average SER.
7. Repeat steps 3, 4, 5, and 6 for the requested set of Intensity Modifications of Table 3.
8. Repeat steps 3, 4, 5, 6, and 7 for the requested set of Type Modifications of Table 3.

Table 3 – List of modification with their parameters

Modification name	Parameter Type	Parameter Intensity
Gaussian noise addition**:** adding a zero-mean, *S* standard deviation Gaussian noise to a layer in the NN model. This noise addition can be simultaneously applied to a sub-set of layers.	- The layers to be modified by Gaussian noise - The ratio of *S* to standard deviation of the Weights in the corresponding layer.	- 1 to total number of layers. - 0.1 to 0.3.
L1 Pruning**:** delete the *P*% of the smallest Weights in a layer for each layer.	- The *P* percentage of the deleted Weights.	- 1% to 90%. - 1% to 99.99% when aiming one layer.
Random pruning**:** delete *R*% of randomly selected Weights, irrespective of their layers.	- The *R* percentage of the deleted Weights.	- 1% to 10%.

Quantizing: reduce to B the number of bits used to represent the Weights by 1. Reducing the number of bits based on a sequence of three operations: affine mapping from the interval the Weights belong to $(\mathbf{0}; \mathbf{2^B - 1})$ interval. 2. Rounding to the closest integer. 3. Backward affine mapping towards the initial interval the Weights belong to.	- The layers to be modified by quantization. - The value of *B.*	- 1 to total number of layers. - 32 to 2.
***Fine tuning / transfer learning*:** resume the training of the *M* watermarked NNs submitted to test, for *E* additional epochs.	- Ratio of *E* to the number of epochs in the initial training.	- Up to 0.5 time the total number of epochs.
Knowledge distillation: train a surrogate network using the inferences of the NN under test as training dataset	- The structure of the architecture. - The size of the dataset *D*. - The number of epochs *E*.	- Structures N. - 10,000 to 1,000,000. - 1 to 100.
Watermark overwriting: successively insert *W* additional watermarks, with random Payloads of the same size as the initial watermark	- *W* number of watermarks successively inserted.	- 2 to 4.

8.2 Fingerprinting

The Tester evaluates the capability of a NN Fingerprinting Method Matcher as specified in the following workflow:

1. Select a set of M_u NNs trained on the training dataset.
2. Compute the M_u fingerprints from the unmodified NNs.
3. Produce a set of M_m modified NNs, by applying one of the Modifications in Table 3 to a given Parameter value.
4. Evaluate the Robustness of the Matcher:
    a. Compute the fingerprint for any of the M_m NNs.
    b. Apply the Matcher to the M_m fingerprints obtained in 4.a and record its output (*Yes* – the matching found is correct or *No* – the matching found is not correct).
    c. Label the *Yes*/*No* outputs of 4.b as *true positive*, *true negative*, *false positive* (*false alarm*) and *false negative* (*missed detection*).
    d. Count the total number of *false positives* and the total number of *false negatives*.
5. Provide the average values over the total number of tests:
    a. The ratio of the number of *false positives* to M_m,
    b. The ratio of the number of *false negatives* to M_m.
    c. The M_m number for tested NNs, and the average BER.
6. Repeat steps 3, 4, and 5 for the requested set of Intensity Modifications of Table 3.
7. Repeat steps 3, 4, 5, and 6 for the requested set of Type Modifications of Table 3.

9 Computational Cost evaluation

The Computational Cost evaluation specifies the Means that enable a Tester to measure the Computational Cost of:

- Injecting, in terms of memory footprint, time to process an epoch, and number of epochs necessary to insert the watermark.
- Detecting, Decoding, or Extracting in terms of memory footprint and time for the Detector or the Decoder or the Extractor to produce the expected result.
- Matching in terms of memory footprint and time for the Matcher to produce the expected result.

9.1 Computational Cost of injecting a watermark

The Computational Cost evaluation specifies the Means that enable a Tester to measure the Computational Cost of the injection using NN Watermarking Method under testing.
The following four elements shall be used to measure the injection process:

1. The memory footprint.
2. The time to execute the operation required by one epoch normalized according to the number of batches processed in one epoch.
3. If the injection is done concurrently with the training of the network, the number of epochs required to insert the watermark.
4. The time for the watermarked NN to compute an inference.

The Tester shall measure the Computational Cost of the injection according to the following workflow:

1. Define a pair of training and testing datasets with a size with at least an order of magnitude more entries than trainable Weights.
2. Select:
    a. The training dataset (if needed).
    b. A set of *M* unwatermarked NNs trained on the training dataset.
    c. *D* data Payloads corresponding to the pre-established Payload size.
3. Apply the NN Watermarking Method to the *M* NNs using the *D* data Payloads.
4. Record the corresponding *M* x *D* set of Computational Costs.
5. Provide the average and 95% confidence limits of the Computational Costs divided by *M* x *D* for one of the informative Testing Environments of Table 4.

Table 4 – Testing Environments (informative)

	Testing environment
Medium	- Single GPU (16GB/6144 CUDA cores) - 8 cores CPU (2.6GHz)
Large	- Double GPU (32GB/12288 CUDA cores) - 16 cores CPU (3.4GHz)

9.2 Computational Cost of Detection, Decoding or Extraction

MPAI-NNT specifies the Means that enable a Tester to measure the Computational Cost of the Detection/Decoding/Extraction of a NN Traceability Method.

9.2.1 Detection and/or Decoding

The Computational Cost of Detection/Decoding is measured by the time and the memory footprint used by the process.

The Tester shall measure the Computational Cost according to the following workflow:

1. Select a set of *M* unwatermarked NNs, *D* data Payloads corresponding to the pre-established Payload size and, if needed, the training dataset.
2. Apply the NN Watermarking Method to the *M* NNs with the *D* data Payloads
3. Evaluate the Robustness of the Detector:
   a. Apply the Watermark Detector to any of the *M x D* NNs.
   b. Record the corresponding *M x D* set of values.
4. Evaluate the Robustness of the Decoder:
   a. Apply the Watermark Decoder to any of the *M x D* NNs.
   c. Record the corresponding *M x D* set of Computational Costs.
6. Provide the average and 95% confidence limits of the Computational Costs divided by *M x D* for one of the informative Testing Environments of Table 4.

9.2.2 Extraction

The Computational Cost of Extraction is measured by the time and the memory footprint used by the process.

The Tester shall evaluate the Computational Cost according to the following workflow:

1. Select a set of M_u NNs trained on the training dataset.
2. Compute the M_u fingerprints from the unmodified NNs.
3. Evaluate the Robustness of the NN Traceability Method:
   d. Apply the fingerprint Extractor to any of the M_m NNs.
   e. Record the corresponding M_m set of Computational Costs.
7. Provide the average and 95% confidence limits of the Computational Costs divided by M_m for one of the informative Testing Environments of Table 4.

9.3 Computational Cost of Matching

The Computational Cost of Matching is measured by the time and the memory footprint used by the process.

The Tester shall evaluate the Computational Cost according to the following workflow:

1. Select a set of M_u NNs trained on the training dataset.
2. Compute the M_u fingerprints from the unmodified NNs.
3. Evaluate the Robustness of the NN Traceability Method:
   a. Apply the fingerprint Matcher to any of the M_m NNs.
   b. Record the corresponding M_m set of Computational Costs.
4. Provide the average and 95% confidence limits of the Computational Costs divided by M_m for one of the informative Testing Environments of Table 4.

Annex 1 Notices and Disclaimers Concerning MPAI Standards (Informative)

The notices and legal disclaimers given below shall be borne in mind when downloading and using approved MPAI Standards.

In the following, "Standard" means the collection of four MPAI-approved and published documents: "Technical Specification", "Reference Software" and "Conformance Testing" and, where applicable, "Performance Testing".

Life cycle of MPAI Standards

MPAI Standards are developed in accordance with the MPAI Statutes. An MPAI Standard may only be developed when a Framework Licence has been adopted. MPAI Standards are developed by especially established MPAI Development Committees who operate on the basis of consensus, as specified in Annex 1 of the MPAI Statutes. While the MPAI General Assembly and the Board of Directors administer the process of the said Annex 1, MPAI does not independently evaluate, test, or verify the accuracy of any of the information or the suitability of any of the technology choices made in its Standards.

MPAI Standards may be modified at any time by corrigenda or new editions. A new edition, however, may not necessarily replace an existing MPAI standard. Visit the web page to determine the status of any given published MPAI Standard.

Comments on MPAI Standards are welcome from any interested parties, whether MPAI members or not. Comments shall mandatorily include the name and the version of the MPAI Standard and, if applicable, the specific page or line the comment applies to. Comments should be sent to the MPAI Secretariat. Comments will be reviewed by the appropriate committee for their technical relevance. However, MPAI does not provide interpretation, consulting information, or advice on MPAI Standards. Interested parties are invited to join MPAI so that they can attend the relevant Development Committees.

Coverage and Applicability of MPAI Standards

MPAI makes no warranties or representations of any kind concerning its Standards, and expressly disclaims all warranties, expressed or implied, concerning any of its Standards, including but not limited to the warranties of merchantability, fitness for a particular purpose, non-infringement etc. MPAI Standards are supplied "AS IS".

The existence of an MPAI Standard does not imply that there are no other ways to produce and distribute products and services in the scope of the Standard. Technical progress may render the technologies included in the MPAI Standard obsolete by the time the Standard is used, especially in a field as dynamic as AI. Therefore, those looking for standards in the Data Compression by Artificial Intelligence area should carefully assess the suitability of MPAI Standards for their needs.

IN NO EVENT SHALL MPAI BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO: THE NEED TO PROCURE SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR

TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE PUBLICATION, USE OF, OR RELIANCE UPON ANY STANDARD, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE AND REGARDLESS OF WHETHER SUCH DAMAGE WAS FORESEEABLE.

MPAI alerts users that practicing its Standards may infringe patents and other rights of third parties. Submitters of technologies to this standard have agreed to licence their Intellectual Property according to their respective Framework Licences.

Users of MPAI Standards should consider all applicable laws and regulations when using an MPAI Standard. The validity of Conformance Testing is strictly technical and refers to the correct implementation of the MPAI Standard. Moreover, positive Performance Assessment of an implementation applies exclusively in the context of the MPAI Governance and does not imply compliance with any regulatory requirements in the context of any jurisdiction. Therefore, it is the responsibility of the MPAI Standard implementer to observe or refer to the applicable regulatory requirements. By publishing an MPAI Standard, MPAI does not intend to promote actions that are not in compliance with applicable laws, and the Standard shall not be construed as doing so. In particular, users should evaluate MPAI Standards from the viewpoint of data privacy and data ownership in the context of their jurisdictions.

Implementers and users of MPAI Standards documents are responsible for determining and complying with all appropriate safety, security, environmental and health and all applicable laws and regulations.

Annex 2 The Governance of the MPAI Ecosystem (Informative)

Level 1 Interoperability

MPAI issues and maintains a Technical Specification – called MPAI-AIF – whose components are:

1. An environment called AI Framework (AIF) running AI Workflows (AIW) composed of inter-connected AI Modules (AIM) exposing standard interfaces.
2. A distribution system of AIW and AIM Implementation called MPAI Store from which an AIF Implementation can download AIWs and AIMs.

A Level 1 Implementation shall be an Implementation of the MPAI-AIF Technical Specification executing AIWs composed of AIMs able to call the MPAI-AIF APIs.

Implementers' benefits	Upload to the MPAI Store and have globally distributed Implementations of - AIFs conforming to MPAI-AIF. - AIWs and AIMs performing proprietary functions executable in AIF.
Users' benefits	Rely on Implementations that have been tested for security.
MPAI Store's role	- Tests the Conformance of Implementations to MPAI-AIF. - Verifies Implementations' security, e.g., absence of malware. - Indicates unambiguously that Implementations are Level 1.

Level 2 Interoperability

In a Level 2 Implementation, the AIW shall be an Implementation of an MPAI Use Case and the AIMs shall conform with an MPAI Application Standard.

Implementers' benefits	Upload to the MPAI Store and have globally distributed Implementations of - AIFs conforming to MPAI-AIF. - AIWs and AIMs conforming to MPAI Application Standards.
Users' benefits	- Rely on Implementations of AIWs and AIMs whose Functions have been reviewed during standardisation. - Have a degree of Explainability of the AIW operation because the AIM Functions and the data Formats are known.
Market's benefits	- Open AIW and AIM markets foster competition leading to better products. - Competition of AIW and AIM Implementations fosters AI innovation.
MPAI Store's role	- Tests Conformance of Implementations with the relevant MPAI Standard. - Verifies Implementations' security. - Indicates unambiguously that Implementations are Level 2.

Level 3 Interoperability

MPAI does not generally set standards on how and with what data an AIM should be trained. This is an important differentiator that promotes competition leading to better solutions. However, the performance of an AIM is typically higher if the data used for training are in greater quantity and more in tune with the scope. Training data that have large variety and cover the spectrum of all cases of interest in breadth and depth typically lead to Implementations of higher "quality".

For Level 3, MPAI normatively specifies the process, the tools and the data or the characteristics of the data to be used to Assess the Grade of Performance of an AIM or an AIW.

Implementers' benefits	May claim their Implementations have passed Performance Assessment.

Users' benefits	Get assurance that the Implementation being used performs correctly, e.g., it has been properly trained.
Market's benefits	Implementations' Performance Grades stimulate the development of more Performing AIM and AIW Implementations.
MPAI Store's role	- Verifies the Implementations' security - Indicates unambiguously that Implementations are Level 3.

The MPAI ecosystem

The following *Figure 1* is a high-level description of the MPAI ecosystem operation applicable to fully conforming MPAI implementations as specified in the Governance of the MPAI Ecosystem Specification [5]:

1. MPAI establishes and controls the not-for-profit MPAI Store.
2. MPAI appoints Performance Assessors.
3. MPAI publishes Standards.
4. Implementers submit Implementations to Performance Assessors.
5. If the Implementation Performance is acceptable, Performance Assessors inform Implementers and MPAI Store.
6. Implementers submit Implementations to the MPAI Store
7. MPAI Store verifies security and Tests Conformance of Implementation.
8. Users download Implementations and report their experience to MPAI.

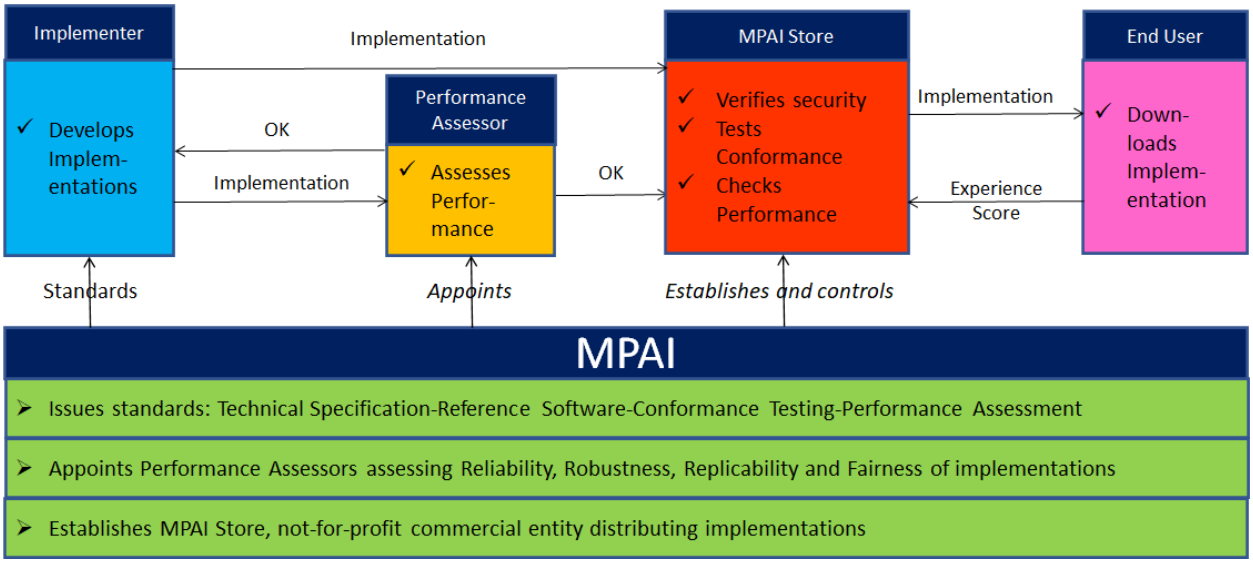

Figure 1 – The MPAI ecosystem operation

Annex 3 Imperceptibility evaluation (informative)

Classification task

The NN Watermarking state of the art studies consider the classification as a predilection task. Within this task, the inference of a Neural Network belongs to a fix set of predefined classes.

To evaluate the impact of injecting a watermark in a classification NN:

- Probability of false alarm: $P_{fa} = \frac{fp}{tp+fp+fn+tn}$ and Precision: $\frac{tp}{tp+fp}$
- Probability of missed detection: $P_{md} = \frac{fn}{tp+fp+fn+tn}$ and Recall: $\frac{tp}{tp+fn}$

As these measures are based on binary classification problem, for multiclass classifiers the average for all classes shall be computed.

Image/speech processing tasks

The inference of a Neural Network is a produced content. For example, a neural network for speech synthesis will return an artificial voice based on a text. Every qualitative/quantitative evaluation of a content can be use:

- Image: PSNR, SSIM, NCC, in addition to subjective test (e.g. as specified by ITU)
- Speech recognition: Word/Sentence error rate, Intent recognition rate, in addition to subjective test (e.g. as specified by ITU)

Image semantic segmentation

The inference of a Neural Network is a semantic-labelled. To evaluate this method, we propose:

- Precision $\frac{tp}{tp+fp}$
- Recall $\frac{tp}{tp+fn}$
- Intersection over Union $\frac{Area\ of\ overlap}{Area\ of\ Union}$