OGC® DOCUMENT: 23-008R5

External identifier of this OGC® document: http://www.opengis.net/doc/is/trainingdml-ai-1/1.0

Open Geospatial Consortium

OGC TRAINING DATA MARKUP LANGUAGE FOR ARTIFICIAL INTELLIGENCE (TRAININGDML-AI) PART 1: CONCEPTUAL MODEL STANDARD

STANDARD
Conceptual model

APPROVED

Version: 1.0.1
Submission Date: 2026-03-01
Approval Date: 2026-04-28
Publication Date: 2026-08-20
Editor: Peng Yue, Boyi Shangguan

CONTENTS

I ABSTRACT

The Training Data Markup Language for Artificial Intelligence (TrainingDML-AI) Standard aims to develop the UML model and encodings for geospatial machine learning training data. Training data plays a fundamental role in Earth Observation (EO) Artificial Intelligence Machine Learning (AI/ML), especially Deep Learning (DL). It is used to train, validate, and test AI/ML models. This Standard defines a UML model and encodings consistent with the OGC Standards baseline to exchange and retrieve the training data in the Web environment.

The TrainingDML-AI Standard provides detailed metadata for formalizing the information model of training data. This includes but is not limited to the following aspects:

- How the training data is prepared, such as provenance or quality;
- How to specify different metadata used for different ML tasks such as scene/object/pixel levels;
- How to differentiate the high-level training data information model and extended information models specific to various ML applications; and
- How to introduce external classification schemes and flexible means for representing ground truth labeling.

II KEYWORDS

The following are keywords to be used by search engines and document catalogues.

ogcdoc, OGC document, artificial intelligence, machine learning, deep learning, earth observation, remote sensing, training data, training sample, UML

SECURITY CONSIDERATIONS

No security considerations have been made for this Standard.

IV SUBMITTING ORGANIZATIONS

The following organizations submitted this Document to the Open Geospatial Consortium (OGC):

- Wuhan University
- Pixalytics Ltd
- WiSC Enterprises
- George Mason University
- Laboratoire d'Informatique de Grenoble
- South Digital Technology Co., Ltd
- Wuhan University of Technology
- Hubei University
- Chongqing Changan Automobile Co., Ltd

V SUBMITTERS

All questions regarding this submission should be directed to the editor or the submitters:

Name	Affiliation
Peng Yue	Wuhan University
Jianya Gong	Wuhan University
Ruixiang Liu	Wuhan University
Dayu Yu	Wuhan University
Samantha Lavender	Pixalytics Ltd
Jim Antonisse	WiSC Enterprises
Liping Di	George Mason University

Eugene Yu	George Mason University
Danielle Ziébelin	Laboratoire d'Informatique de Grenoble
Boyi Shangguan	South Digital Technology Co., Ltd
Lei Hu	South Digital Technology Co., Ltd
Liangcun Jiang	Wuhan University of Technology
Mingda Zhang	Hubei University
Kai Yan	Chongqing Changan Automobile Co., Ltd

VI ACKNOWLEDGEMENTS

Thanks to the members of the TrainingDML-AI Standards Working Group of the OGC as well as all contributors of change requests and comments. In particular: Scott Simmons, Carl Reed, Ivana Ivánová, Emily Daemen, Jon Duckworth, Zheheng Liang, Jibo Xie, Yuqi Bai, Winnie Shiu, Ignacio Correas, Chenxiao Zhang, Zhipeng Cao, Haofeng Tan, Yinyin Pan, Hanwen Xu, Shuaiqi Liu, Hao Li, Ming Wang, Kaixuan Wang, Haipeng Deng, Gobe Hobona, Chris Little, Kathi Schleidt, Rodolfo Orozco.

1

SCOPE

1 SCOPE

Training data is the building block of machine learning models. These models now constitute the majority of machine learning applications in Earth science. Training data is used to train AI/ML models, and to then validate model results. Formalizing and documenting the training data by characterizing the training data content, metadata, data quality, and provenance, and so forth is essential.

This OGC Training Data Standard describes work actions around training data:

- Documents the UML model with a target of maximizing the interoperability and usability of EO imagery training data;
- Defines different AI/ML tasks and labels in earth observation in terms of supervised learning, including scene level, object level and pixel level tasks;
- Describes the description of the permanent identifier, version, license, training data size, measurement or imagery used for annotation, and so on; and
- Specifies a description of quality (e.g., training data errors, training data unrepresentativeness, quality measures) and the provenance (agents who perform the labeling, labeling procedure).

2 CONFORMANCE

2 CONFORMANCE

This TrainingDML-AI Standard defines a conceptual model that is independent of any encoding or formatting technologies. The standardization targets for this Standard is:

- TrainingDML-AI Conceptual Model

Conformance with this Standard shall be checked using all the relevant tests specified in Annex A (normative) of this document. The framework, concepts, and methodology for testing, and the criteria to be achieved to claim conformance are specified in the OGC Compliance Testing Policies and Procedures and the OGC Compliance Testing website.

All requirements-classes and conformance-classes described in this document are owned by the standard identified.

3 NORMATIVE REFERENCES

3 NORMATIVE REFERENCES

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

Freed N, Borenstein N, Internet Engineering Task Force: IETF RFC 2046, *Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types*. RFC Publisher (1996). https://www.rfc-editor.org/info/rfc2046.

International Organization for Standardization (committee): ISO 19101-1:2014, *Geographic information — Reference model — Part 1: Fundamentals*. International Organization for Standardization, Geneva (2014). https://www.iso.org/standard/59164.html.

International Organization for Standardization (committee): ISO 19103:2024, *Geographic information — Conceptual schema language*. International Organization for Standardization, Geneva (2024). https://www.iso.org/standard/83454.html.

International Organization for Standardization (committee): ISO 19115-1:2014, *Geographic information — Metadata — Part 1: Fundamentals*. International Organization for Standardization, Geneva (2014). https://www.iso.org/standard/53798.html.

International Organization for Standardization (committee): ISO 19156:2023, *Geographic information — Observations, measurements and samples*. International Organization for Standardization, Geneva (2023). https://www.iso.org/standard/82463.html.

International Organization for Standardization (committee): ISO 19157-1, *Geographic information — Data quality — Part 1: General requirements*. International Organization for Standardization, Geneva https://www.iso.org/standard/78900.html.

4 TERMS AND DEFINITIONS

4 TERMS AND DEFINITIONS

This document uses the terms defined in OGC Policy Directive 49, which is based on the ISO/IEC Directives, Part 2, Rules for the structure and drafting of International Standards. In particular, the word “shall” (not “must”) is the verb form used to indicate a requirement to be strictly followed to conform to this document and OGC documents do not use the equivalent phrases in the ISO/IEC Directives, Part 2.

This document also uses terms defined in the OGC Standard for Modular specifications (OGC 08-131r3), also known as the ‘ModSpec’. The definitions of terms such as standard, specification, requirement, and conformance test are provided in the ModSpec.

For the purposes of this document, the following additional terms and definitions apply.

4.1. Artificial Intelligence (AI)

refers to a set of methods and technologies that can empower machines or software to learn and perform tasks like humans.

4.2. Machine Learning (ML)

is an important branch of artificial intelligence that gives computers the ability to improve their performance without explicitly being programmed to do so. ML processes create models from training data by using a set of learning algorithms, and then can use these models to make predictions. Depending on whether the training data include labels, the learning algorithms can be divided into supervised and unsupervised learning.

4.3. Deep Learning (DL)

is a subset of machine learning, which is essentially a neural network with three or more layers. The number of layers is referred to as depth. While a neural network with a single layer can still make approximate predictions, additional hidden layers can help to optimize and refine for accuracy.

4.4. Training Dataset

a collection of samples, often labelled with known terms or expected values for supervised learning. A training dataset can be divided into training, validation, and test sets. Training samples are different from samples in OGC Observations & Measurements (O&M). They are often collected in purposive ways that deviate from purely probability sampling, with known or expected results labeled as values of a dependent variable for generating a trained predictive model.

4.5. Label

refers to known or expected results annotated as values of a dependent variable in training samples. A training sample label is different from those on a geographical map, which are known as map labels or annotations.

Note 1 to entry: A training sample label is different from those on a geographical map, which are known as map labels or annotations.

4.6. Provenance

information about entities, activities, and people involved in producing a piece of data or thing, which can be used to form assessments about its quality, reliability or trustworthiness. In this standard provenance is a record of how training data were prepared.

4.7. Quality

degree to which a set of inherent characteristics fulfils requirements [ISO 9000:2005, definition3.1.1]. Quality of training data (such as data imbalance and mislabeling) can impact the performance of AI/ML models.

4.8. Earth Observation

data and information collected about our planet, whether atmospheric, oceanic or terrestrial. This includes space-based or remotely-sensed data, as well as ground-based or in situ data.

4.9. Scene Classification

task of identifying scene categories of images, on the basis of a training set of images whose scene categories are known.

4.10. Object Detection

task of recognizing objects such as cars from images. The objects are often localized using bounding boxes.

4.11. Semantic Segmentation

task of assigning class labels to pixels of images or points of point clouds.

4.12. Change Detection

recognition of changes between images acquired at different times.

4.13. 3D Model Reconstruction

task that builds 3D objects and scenes from multi-view images.

4.14. **Generative Model**

is one of the methods of large model training, which improve model performance through unsupervised pre-training. In the fine-tuning phase, labeled data plays a critical role in optimizing the model for specific vertical domains or tasks. By incorporating labeled data, the model can learn to accurately identify and extract relevant features, leading to better performance on specific downstream tasks. Overall, the combination of generative models and fine-tuning with labeled data can significantly improve the performance of large models in specialized domains or tasks.

5 CONVENTIONS

5 CONVENTIONS

This section provides details and examples for any conventions used in the document. Examples of conventions are symbols, abbreviations, use of XML schema, or special notes regarding how to read the document.

5.1. Identifiers

The normative provisions in this specification are denoted by the URI

http://www.opengis.net/spec/TrainingDML-AI-1/1.0

All requirements and conformance tests that appear in this document are denoted by partial URIs which are relative to this base.

5.2. Abbreviated Terms

In this document, the following abbreviations and acronyms are used or introduced:

AI	Artificial Intelligence
DL	Deep Learning
EO	Earth Observation
ISO	International Organization for Standardization
JSON	JavaScript Object Notation
LC	Land Cover
LU	Land Use
ML	Machine Learning
OGC	Open Geospatial Consortium
RS	Remote Sensing
TD	Training Data

UML	Unified Modeling Language
XML	Extensible Markup Language
MIME	Multipurpose Internet Mail Extensions

5.3. UML Notation

The Standard is presented in this document through diagrams using the Unified Modeling Language (UML) static structure diagram. The UML notations used in this Standard are described in the diagram in Figure 1.

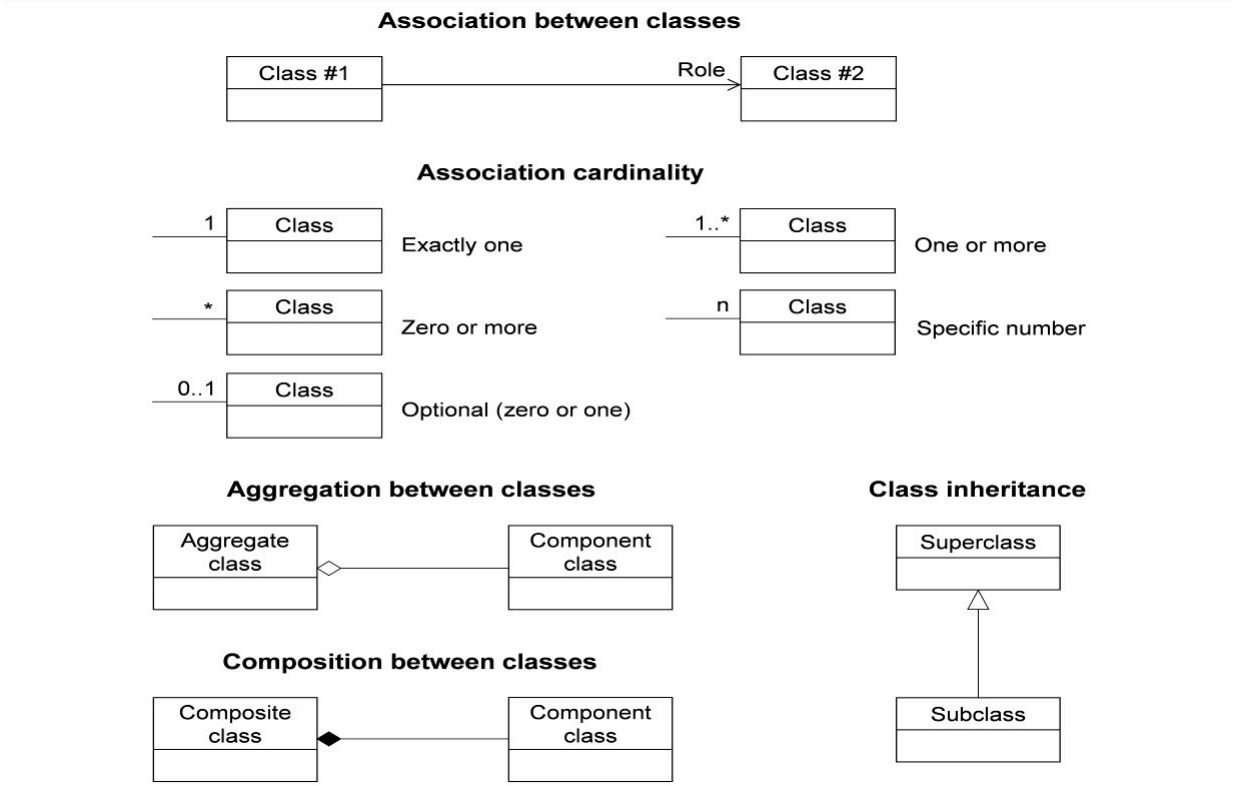

Figure 1 — UML notation (see ISO TS 19103, Geographic information — Conceptual schema language).

All associations between model elements in the TrainingDML-AI Conceptual Model are uni-directional. Thus, associations in the model are navigable in only one direction. The direction of navigation is depicted by an arrowhead. In general, the context an element takes within the association is indicated by its role. The role is displayed near the target of the association. If the graphical representation is ambiguous though, the position of the role has to be drawn to the element the association points to.

The following stereotypes are used in this model.

- «DataType» defines a set of properties that lack identity. A data type is a classifier with no operations, whose primary purpose is to hold information.
- «CodeList» enumerates the valid attribute values. In contrast to Enumeration, the list of values is open and, thus, not given inline in the TrainingDML-AI UML Model. The allowed values can be provided within an external code list.

6 OVERVIEW

6 OVERVIEW

The TrainingDML-AI Conceptual Model Standard defines how to represent and exchange ML training data. The conceptual model includes the most relevant training data entities from datasets, to instances (i.e. individual training samples), to labels. The conceptual schema specifies how and into which parts of the training data should be decomposed and classified.

The training data model defined in this document facilitates interoperability by enabling heterogeneous training datasets to conform to a unified representation and exchange form. It ensures that training data from different vendors can be consistently shared and interpreted, improving the accessibility and promoting the integration of geospatial AI/ML resources.

The TrainingDML-AI conceptual model (Clause 7) is formally specified using UML class diagrams, complemented by a data dictionary (Clause 8) providing the definitions and explanations of the object classes and attributes. This conceptual model provides the basis for specifying encoding implemented in languages such as JSON, or XML.

6.1. AI Tasks for EO

In recent years AI/ML is increasingly used in the EO domain. The new AI/ML algorithms frequently require large training datasets as benchmarks. AI/ML TD have been used in many EO applications to calibrate the performance of AI/ML models. Many efforts have been made to produce training datasets to make accurate predictions. As a result, a number of training datasets are publicly available, with new datasets being constantly released. In the EO domain, examples of AI/ML training datasets have been developed in various tasks including the following typical scenarios:

- Scene classification. These algorithms determine image categories from numerous pictures (e.g., agricultural, forest, and beach scenes). The training samples are a series of labeled pictures. The data can be either from satellite, drones, or aircrafts. The metadata of the datasets often includes the number of training samples, the number of classes, and the image size.
- Object detection. These algorithms detect and localize different objects (e.g., airplanes, cars and building) in a single image. The image can be optical or non-optical, such as Synthetic Aperture Radar (SAR). Recent work also suggests an increasing focus on object detection from street view imagery. The bounding boxes can also be either oriented or horizontal. The geometry of a bounding box can be expressed using top-left/bottom-right coordinates, coordinates of four corners, or center coordinates along with the length and width of the box.
- Semantic segmentation. In terms of Land cover (LC) and land use (LU) classification, this process assigns a LC/LU class label to a pixel (or groups of pixels) of RS imagery. Considering semantic segmentation of 3D point clouds, it is to classify points of a 3D

point cloud into categories. TDs are usually composed of RS images/point clouds, and the corresponding labeled value of each pixel/point recording its class.

- Change detection. These algorithms identify the difference between images acquired over the same geographical area but taken at different times. The TD comprise a set of pre-change and post-change RS images, with the corresponding ground truth map labeled changed and unchanged pixels. The image can be optical or SAR images.
- 3D model reconstruction. These algorithms infer the 3D geometry and structure of objects and scenes, mainly realized from the dense matching of multi-view images. The TD are usually composed of two-view or multi-view images, with the corresponding disparity map or depth maps as ground truth respectively.

6.2. Modularization

The TrainingDML-AI conceptual model provides models for the most important elements within TD. These elements have been identified to be either required or important in many different AI/ML tasks. However, implementations are not required to support the complete TrainingDML-AI model in order to be conformant to the Standard. Implementations may employ a subset of constructs according to their specific information needs. For this purpose, modularization is applied to the TrainingDML-AI.

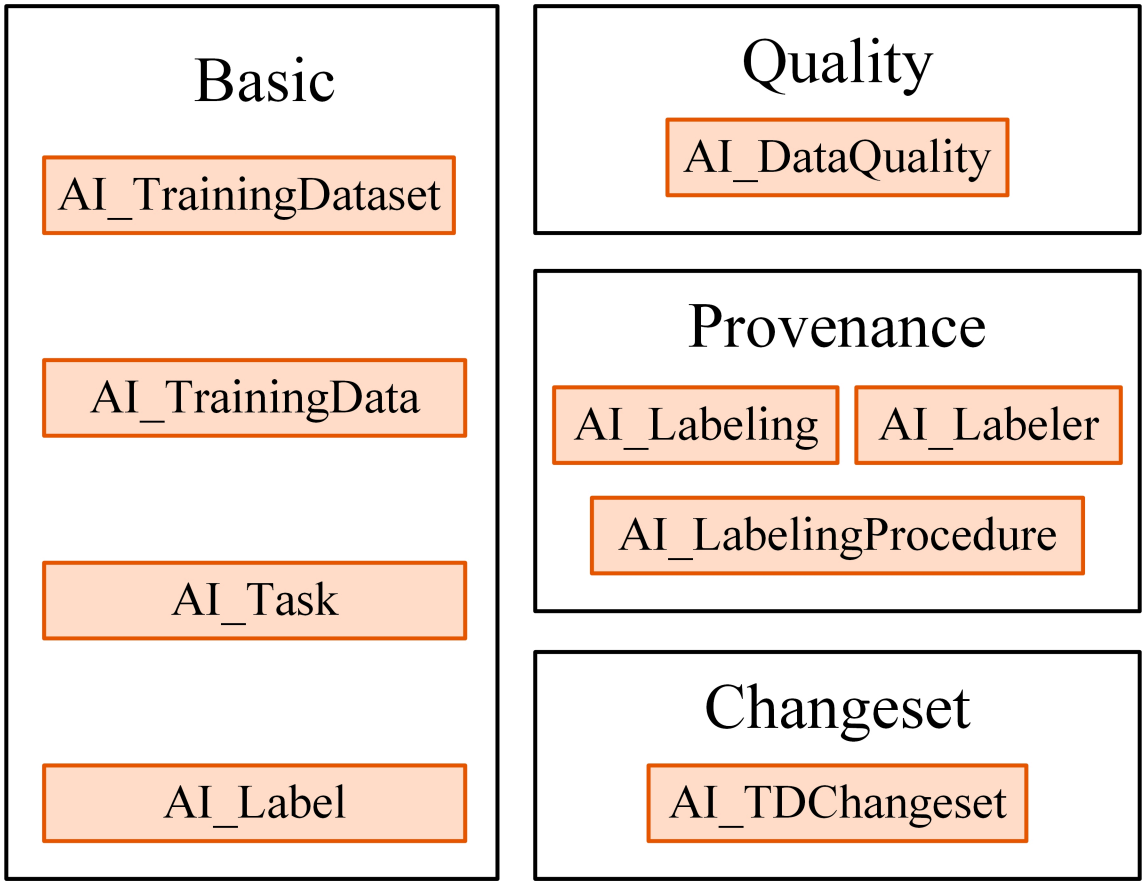

Figure 2 — TrainingDML-AI module overview.

As shown in Figure 2, the TrainingDML-AI conceptual model is thematically decomposed into a Basic module, a Provenance module, a Quality module and a Changeset module. The Basic module comprises the basic concepts and elements, including AI_TrainingDataset, AI_TrainingData, AI_Label, and AI_Task, of the TrainingDML-AI, and thus, must be implemented by any conformant system. The Provenance module provides a comprehensive definition of provenance by AI_Labeling, AI_Labeler, and AI_Labeling Procedure. The Quality module offers quality description of TD with AI_DataQuality elements. And the Changeset module defines AI_TDChangeset between versions of datasets.

6.3. General Modeling Principles

6.3.1. Element Modeling

The modeling of all elements in the TrainingDML-AI conceptual model has the following principles.

- Granularity. Two levels of granularity are differentiated in the conceptual model: The Training Dataset is used to refer to the collection level, and the Training Data is used to refer to the individual level.
- Label semantics. The training dataset will not be limited to one classification scheme. External classification schemes should be allowed to be linked into the Training Dataset to accommodate different cases in practice. The development of external classification schemas can follow the ISO 19144 series.
- Light-weight design. The lightweight designed conceptual model has a minimum set of metadata elements, provenance, or quality measures at the collection level instead of at the individual level. This is to facilitate the understanding of the dataset and improve the scalability for communicating large training datasets.
- Alignment. The modeling of elements in TDs can leverage existing efforts for wide adoption, such as for ISO 19109 Geographic information — Rules for application schema, ISO 19115-1 Geographic information — Metadata — Part 1: Fundamentals, ISO 19157-1 Geographic information — Data quality — Part 1: General requirements, and the OGC Geography Markup Language (GML) Standard. The conceptual model can be aligned with these existing standards and leverage capabilities fulfilled in part by other standards.
- Quality, bias, and ethics. Elements related to quality, or more specifically, bias that can be used to reduce the errors when using AI/ML. For example, any knowledge of the TD imbalance and mislabeling can be stored in TD quality. In addition, data ethics aims to safeguard the responsible use of TD, and it can be addressed by using the license property in the TD.
- Changeset. This will be an optional module in TD modeling. Changeset addresses how to capture changes in TD datasets. The change model considers the trend in TD collections to use the crowdsourcing platforms and borrow the change representation from the platforms such as OpenStreetMap.

6.3.2. Class Hierarchy and Inheritance of Properties and Relations

In the TrainingDML-AI conceptual model, the specific elements such as EO training datasets, EO training data, scene label, object label, and pixel label are defined as subclasses of more general higher-level classes. Hence, elements build a hierarchy along specialization / generalization relationships where more specialized elements inherit the properties and relationships of all their super classes along the entire generalization path to the topmost element.

6.3.3. Definition of the Semantics for all Classes, Properties, and Relations

The meanings of all elements defined in the TrainingDML-AI conceptual model are normatively specified in the data dictionary in Clause 8.

6.3.4. Data Integrity, Authenticity, and Non-repudiation

Sometimes training datasets can be downloaded, disseminated, and changed by anyone. The data integrity, authenticity, and non-repudiation are important to ensure unexpected bias propagation and distorted results. Currently the standard focuses on the information modeling, while data dissemination can be enriched with strategies from the general information domain by publishing hashes (e.g., MD5) and public-keys (e.g., RSA) after signing and encrypting.

6.4. Extending TrainingDML-AI

The TrainingDML-AI conceptual model is designed as a universal information model that defines elements and attributes which are useful for a broad range of AI/ML applications. In practical AI/ML applications, the elements within specific TDs will most likely contain attributes which are not explicitly modeled in TrainingDML-AI. Moreover, there might be TD elements which are not covered by the TrainingDML-AI thematic classes.

The model provides an abstract class-based method to support the exchange of such data. Elements not represented by the predefined thematic classes of the model may be modeled and exchanged by extending abstract class.

7 TRAININGDML-AI UML MODEL

7 TRAININGDML-AI UML MODEL

The TrainingDML-AI UML model is the normative definition of the TrainingDML-AI Conceptual Model. The tables and figures in this section were software generated from the UML model. As such, this section provides a normative representation of the TrainingDML-AI Conceptual Model.

7.1. ISO Dependencies

TrainingDML-AI builds on the ISO 19100 family of standards. The applicable standards are identified in Figure 3. Data dictionaries are included for all the ISO-defined classes explicitly referenced in the TrainingDML-AI UML model. These data dictionaries are provided for the convenience of the user. The ISO standards are the normative source.

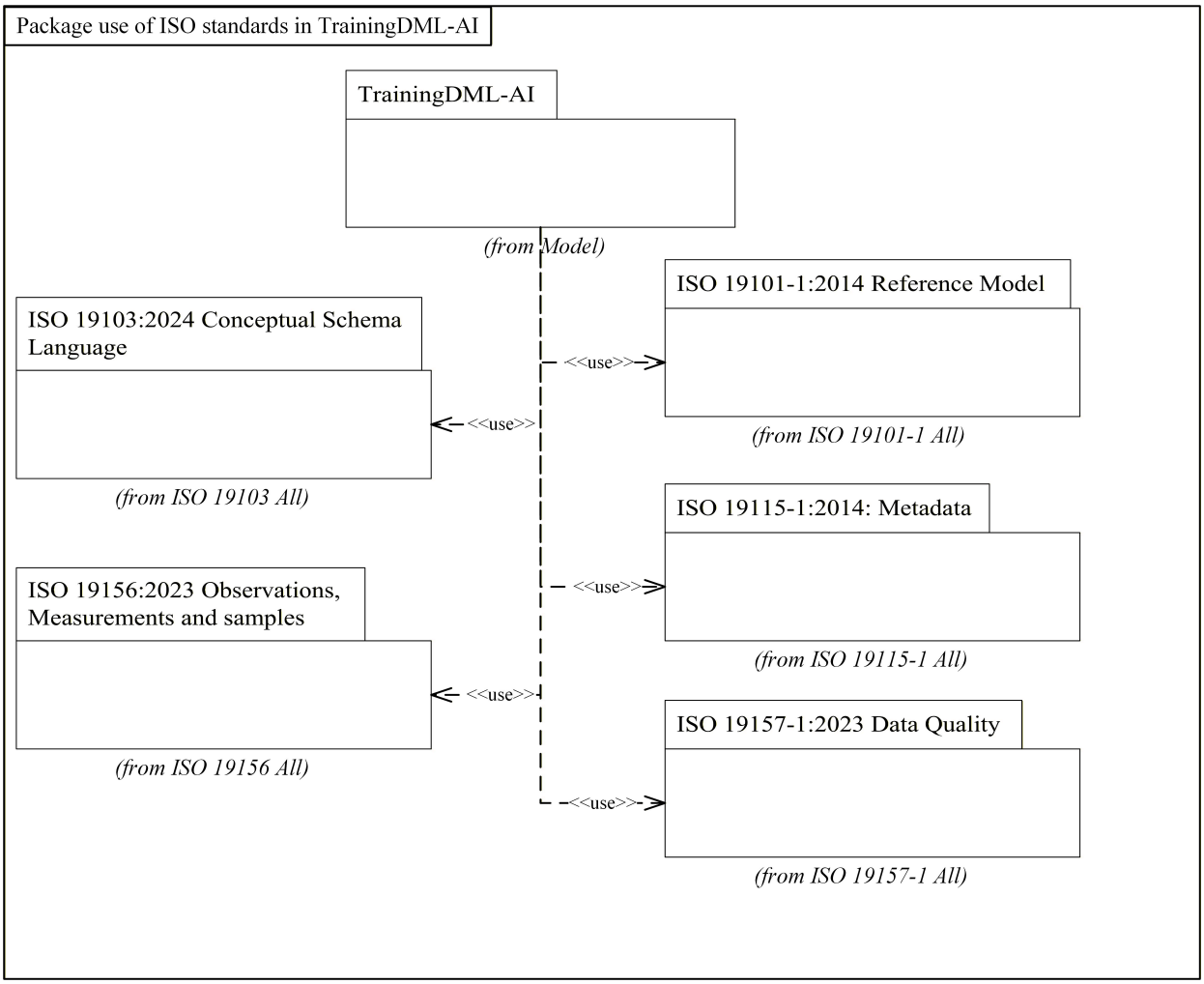

Figure 3 — Use of ISO Standards in TrainingDML-AI.

The ISO classes explicitly used in the TrainingDML-AI UML model are introduced in Table 1. Further details about these classes can be found in the Data Dictionary within Clause 8.

Table 1 — ISO Classes used in TrainingDML-AI

CLASS NAME	DESCRIPTION
Feature	Abstraction of real world phenomena.
MD_Band	Range of wavelengths in the electromagnetic spectrum.
MD_Scope	The target resource and physical extent for which information is reported.
MD_ReferenceSystem	Information about the reference system.
LI_Lineage	Information about the events or source data used in constructing the data specified by the scope or lack of knowledge about lineage.
EX_Extent	Extent of the resource.
CI_Citation	Standardized resource reference.
MD_Resolution	Level of detail expressed as a scale factor, a distance or an angle.
DataQuality	Quality information for the data specified by a data quality scope.
QualityElement	Aspect of quantitative quality information.

7.2. Overview of the UML Model

The UML model is presented with core concepts in Figure 4, followed by the concrete classes in Figure 5. The following describes the core concepts.

- AI_TrainingDataset: This concept represents a collection of training samples, i.e. a training dataset.
- AI_TrainingData: This concept is an individual training sample in a training dataset.
- AI_Task: This concept is used to identify the task that the training dataset is used for.
- AI_Label: This concept represents the label semantics for TD.
- AI_Labeling: This concept provides the provenance of how TD are created.

- AI_TDChangeset: This concept records types of TD changes between two versions of the training dataset.
- AI_DataQuality: This concept is associated with a training dataset to document its quality.

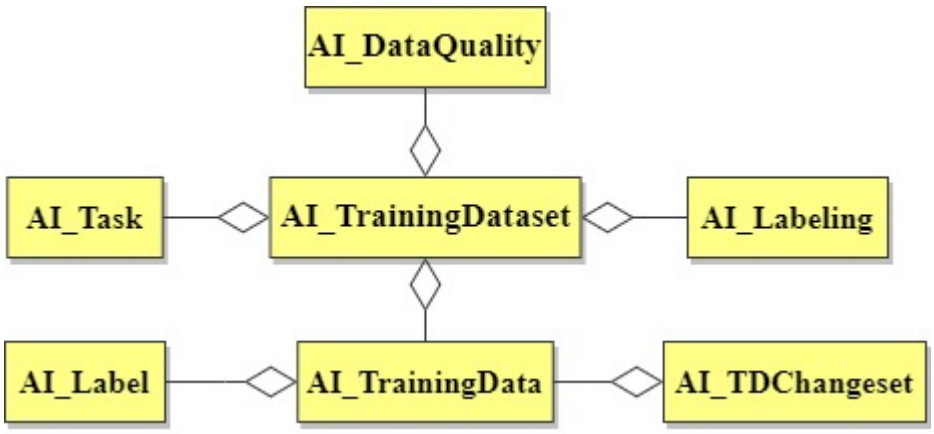

Figure 4 — Core concepts.

The full overview of concrete classes and attributes are presented in Figure 5. Concepts related to the EO AI/ML applications are defined as classes extended from abstract classes. Each core concept with related classes will be described in the rest subsections.

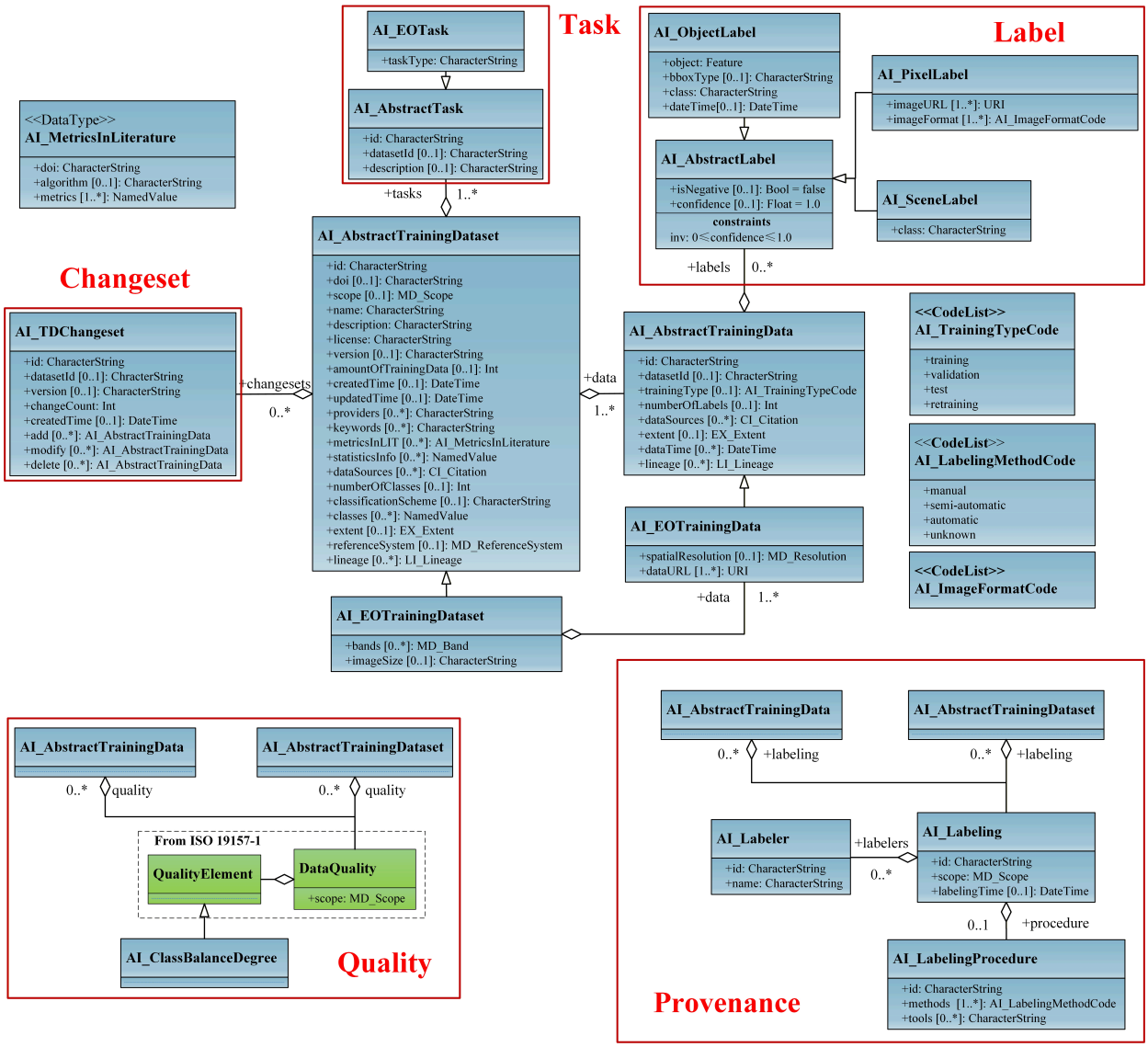

Figure 5 — Overview of the UML model.

7.3. AI_TrainingDataset

REQUIREMENTS CLASS 1	
IDENTIFIER	http://www.opengis.net/spec/TrainingDML-AI-1/1.0/req/aitrainingdataset
TARGET TYPE	Implementation Standard

REQUIREMENTS CLASS 1	
PREREQUISITES	ISO 19103:2024 ISO 19115-1:2014 ISO 19156:2023 /req/aitrainingdata /req/aitask /req/ailabeling /req/aidataquality /req/aitdchangeset

An AI training dataset is represented as AI_AbstractTrainingDataset. Each training sample in the dataset is represented as AI_AbstractTrainingData. A set of basic attributes are defined at the collection level for AI_AbstractTrainingDataset. These include the identification, authorship, time for creation and update, AI tasks, number of training samples (amountOfTrainingData), label semantics (classes recording the available semantic classes), number of label classes, and statistics of training samples in each class (statisticsInfo). These basic attributes at the collection level can also be seen as a minimum set of metadata to define a simple payload. In this context, payload means the actual content that will be transmitted on the Web. They can be mapped to the MD_Metadata entity properties in ISO 19115, and thus delegate the metadata description via a metadata association to existing metadata standards.

The AI_EOTrainingDataset is defined to convey attributes specific to EO. For example, the image size and band information are defined in AI_EOTrainingDataset.

The UML diagram of the AI_TrainingDataset is illustrated in Figure 6.

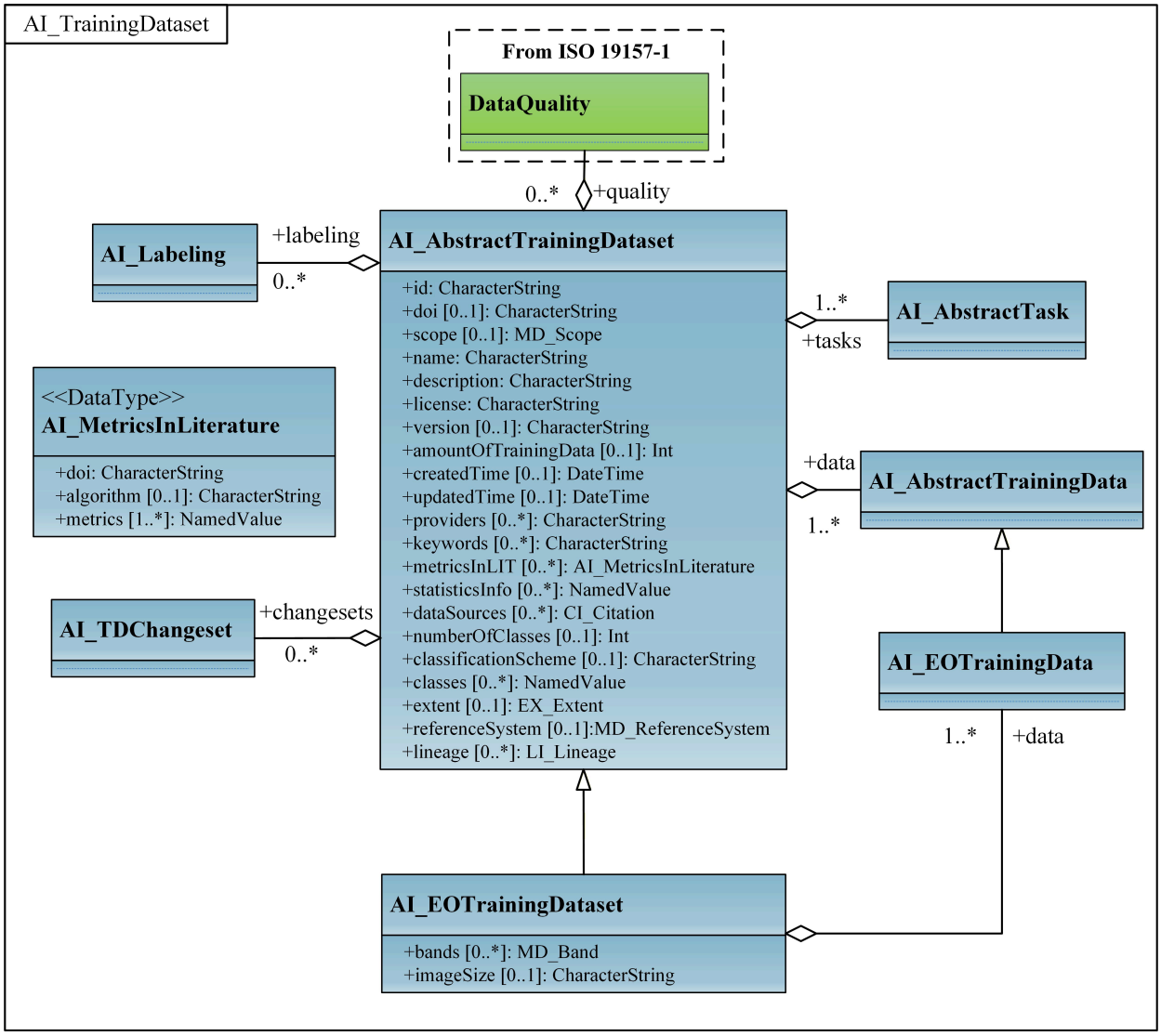

Figure 6 — UML diagram of AI_TrainingDataset.

7.3.1. Requirements

The following requirement defines the rules governing the implementation of the AI_TrainingDataset conceptual model as an implementation standard.

REQUIREMENT 1	
IDENTIFIER	`/req/aitrainingdataset/classes`
STATEMENT	For each UML class defined or referenced in AI_TrainingDataset conceptual model
A	Any implementation standard SHALL contain an element which represents the same concept as that defined for the UML class.

REQUIREMENT 1	
B	Any implementation standard SHALL represent associations with the same source, target, direction, roles, and multiplicities as those of the UML class.
C	Any implementation standard SHALL represent the attributes of the UML class including the name, definition, type, and multiplicity.
D	Any implementation standard SHALL represent the attributes of all super classes of the UML class including the name, definition, type, and multiplicity.
E	Any implementation standard SHALL represent the associations of all super classes of the UML class including the source, target, direction, roles, and multiplicity.
F	Any implementation standard SHALL specify how an implementation observes all constraints the Conceptual Model imposes on the UML class.

An implementing technology may not be able to support all the concepts defined in the TrainingDML-AI UML model. Alternately, some concepts may be inappropriate for the application domain for which the implementation standard was developed. In those cases, elements of the UML model may be mapped to null elements.

PERMISSION 1	
IDENTIFIER	`/per/aitrainingdataset/classes`
STATEMENT	For each UML class defined or referenced in AI_TrainingDataset
A	An implementation standard MAY represent that class as a null class with no attributes, associations, or definition.
B	An implementation standard MAY represent an association of the UML class with a null association.
C	An implementation standard MAY represent an attribute of the UML class with a null attribute.
D	Whenever a null element is used to represent a concept from the AI_TrainingDataset, the implementation standard SHOULD document that mapping and provide an explanation for why that concept was not implemented.

7.3.2. Class Definitions

Table 2 — Classes defined in AI_TrainingDataset

NAME	DESCRIPTION
AI_AbstractTrainingDataset	AI_AbstractTrainingDataset defines the basic concepts and components of different kinds of training dataset.

NAME	DESCRIPTION
AI_EOTrainingDataset	AI_EOTrainingDataset describes attributes specific to EO training dataset. For example, the data source, image size, band information, and spatial extent.

Table 3 — Data types defined in AI_TrainingDataset

NAME	DESCRIPTION
AI_MetricsInLiterature «DataType»	AI_MetricsInLiterature records results of performance metrics achieved by AI/ML algorithms in the peer-reviewed literature.

7.4. AI_TrainingData

REQUIREMENTS CLASS 2	
IDENTIFIER	`http://www.opengis.net/spec/TrainingDML-AI-1/1.0/req/aitrainingdata`
TARGET TYPE	Implementation Standard
PREREQUISITES	ISO 19103:2024 ISO 19115-1:2014 /req/aitrainingdataset /req/ailabel

The AI_AbstractTrainingData includes the source data and its corresponding labels, which focus on a compact modeling of individual training samples. When there is a need to identify the different uses/purposes, an optional attribute, the training type (training, validation, test, or retraining types), can be added at the individual level. The attribute datasetId helps identify samples from different training datasets.

The AI_EOTrainingData is defined to convey attributes specific to EO. It provides additional attributes, including spatial resolution and urls of the source EO inputs.

The UML diagram of the AI_TrainingData is illustrated in Figure 7.

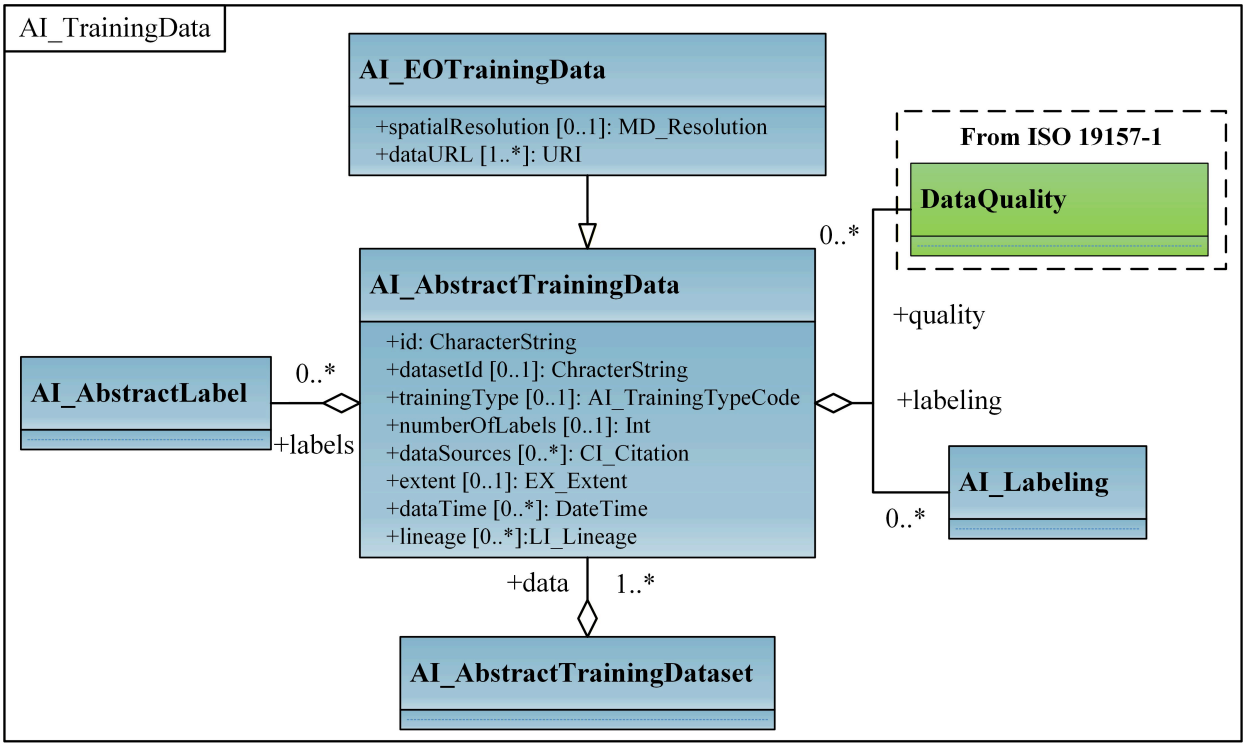

Figure 7 — UML diagram of AI_TrainingData.

The Code Lists provided for the AI_TrainingData are illustrated in Figure 8.

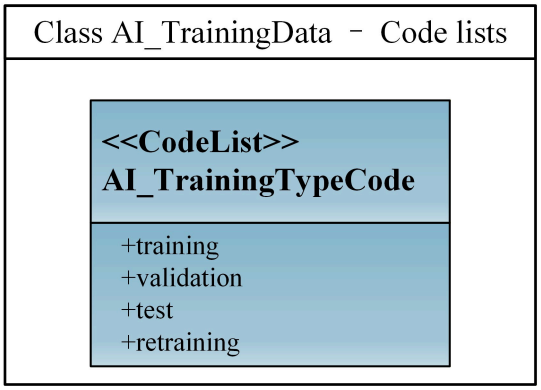

Figure 8 — Code lists from the AI_TrainingData.

7.4.1. Requirements

The following requirement defines the rules governing the implementation of the AI_TrainingData conceptual model as an implementation standard.

REQUIREMENT 2	
IDENTIFIER	`/req/aitrainingdata/classes`
STATEMENT	For each UML class defined or referenced in AI_TrainingData
A	Any implementation standard SHALL contain an element which represents the same concept as that defined for the UML class.
B	Any implementation standard SHALL represent associations with the same source, target, direction, roles, and multiplicities as those of the UML class.
C	Any implementation standard SHALL represent the attributes of the UML class, including the name, definition, type, and multiplicity.
D	Any implementation standard SHALL represent the attributes of all super classes of the UML class, including the name, definition, type, and multiplicity.
E	Any implementation standard SHALL represent the associations of all super classes of the UML class, including the source, target, direction, roles, and multiplicity.
F	Any implementation standard SHALL specify how an implementation observes all constraints the Conceptual Model imposes on the UML class.

PERMISSION 2	
IDENTIFIER	`/per/aitrainingdata/classes`
STATEMENT	For each UML class defined or referenced in AI_TrainingData
A	An implementation standard MAY represent that class as a null class with no attributes, associations, or definition.
B	An implementation standard MAY represent an association of the UML class with a null association.
C	An implementation standard MAY represent an attribute of the UML class with a null attribute.
D	Whenever a null element is used to represent a concept from the AI_TrainingData, the implementation standard SHOULD document that mapping and provide an explanation for why that concept was not implemented.

7.4.2. Class Definitions

Table 4 — Classes defined in AI_TrainingData

NAME	DESCRIPTION
AI_AbstractTrainingData	AI_AbstractTrainingData defines the basic concepts and components of different kinds of training data.
AI_EOTrainingData	AI_EOTrainingData describes attributes specific to EO training data. For example, spatial extent and date time of the source EO inputs.

Table 5 — Code list classes defined in AI_TrainingData

NAME	DESCRIPTION
AI_TrainingTypeCode «CodeList»	AI_TrainingTypeCode is a code list used to identify whether the individual data element is used for different purposes.

7.5. AI_Task

REQUIREMENTS CLASS 3	
IDENTIFIER	`http://www.opengis.net/spec/TrainingDML-AI-1/1.0/req/aitask`
TARGET TYPE	Implementation Standard
PREREQUISITES	ISO 19103:2024 /req/aitrainingdataset

Various AI/ML tasks in specific domains can have different organizing forms of TD, including source data and label representations. AI_EOTask is proposed by extending AI_AbstractTask to represent specific AI/ML tasks in the EO domain. The task type can refer to a particular type defined by an external category, such as scene classification or change detection. It specifies, for example, whether a remote sensing training dataset is used for scene classification, object detection, land use or land cover, or change detection scenarios.

The UML diagram of the AI_Task is illustrated in Figure 9.

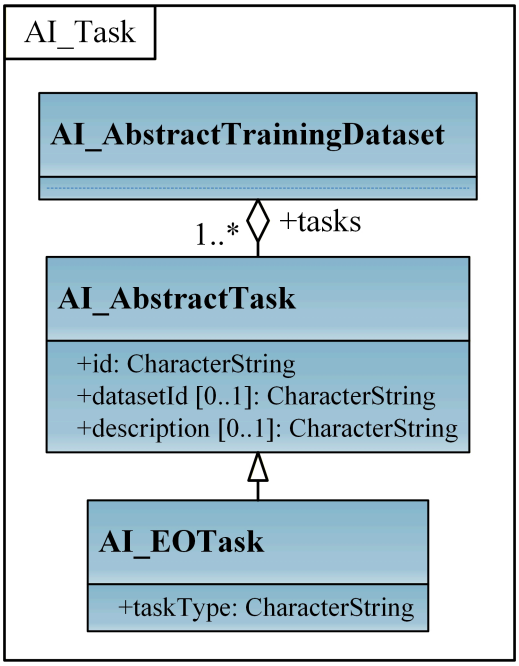

Figure 9 — UML diagram of AI_Task.

7.5.1. Requirements

The following requirement defines the rules governing the implementation of the AI_Task conceptual model as an implementation standard.

REQUIREMENT 3	
IDENTIFIER	`/req/aitask/classes`
STATEMENT	For each UML class defined or referenced in AI_Task
A	Any implementation standard SHALL contain an element which represents the same concept as that defined for the UML class.
B	Any implementation standard SHALL represent associations with the same source, target, direction, roles, and multiplicities as those of the UML class.
C	Any implementation standard SHALL represent the attributes of the UML class including the name, definition, type, and multiplicity.
D	Any implementation standard SHALL represent the attributes of all super classes of the UML class including the name, definition, type, and multiplicity.
E	Any implementation standard SHALL represent the associations of all super classes of the UML class including the source, target, direction, roles, and multiplicity.
F	Any implementation standard SHALL specify how an implementation observes all constraints the Conceptual Model imposes on the UML class.

PERMISSION 3	
IDENTIFIER	`/per/aitask/classes`
STATEMENT	For each UML class defined or referenced in AI_Task
A	An implementation standard MAY represent that class as a null class with no attributes, associations, or definition.
B	An implementation standard MAY represent an association of the UML class with a null association.
C	An implementation standard MAY represent an attribute of the UML class with a null attribute.
D	Whenever a null element is used to represent a concept from the AI_Task, the implementation standard SHOULD document that mapping and provide an explanation for why that concept was not implemented.

7.5.2. Class Definitions

Table 6 — Classes defined in AI_Task

NAME	DESCRIPTION
AI_AbstractTask	AI_AbstractTask defines the AI/ML tasks in specific domains that have different organizing forms of TD.
AI_EOTask	AI_EOTask extends AI_AbstractTask to represent specific AI/ML tasks in the EO domain.

7.6. AI_Label

REQUIREMENTS CLASS 4	
IDENTIFIER	`http://www.opengis.net/spec/TrainingDML-AI-1/1.0/req/ailabel`
TARGET TYPE	Implementation Standard
PREREQUISITES	ISO 19101-1:2014 ISO 19103:2024 IETF RFC 2046

REQUIREMENTS CLASS 4

/req/aitrainingdata

Labels for each individual training sample can be represented using a feature, coverage, or a semantic class from ontologies or shared vocabularies (external classification schemes and classes). A training sample typically relates pixels, objects, and scenes to semantic labels, where labels are encoded as coverage, geometric polygon, and text respectively. Therefore, the AI_AbstractLabel is extended to specify AI_SceneLabel, AI_ObjectLabel, and AI_PixelLabel respectively. There is no restriction for TD producers to describe label information with label classes. For example, labels of training dataset for object detection can be represented by either AI_ObjectLabel or AI_PixelLabel.

The UML diagram of the AI_Label is illustrated in Figure 10.

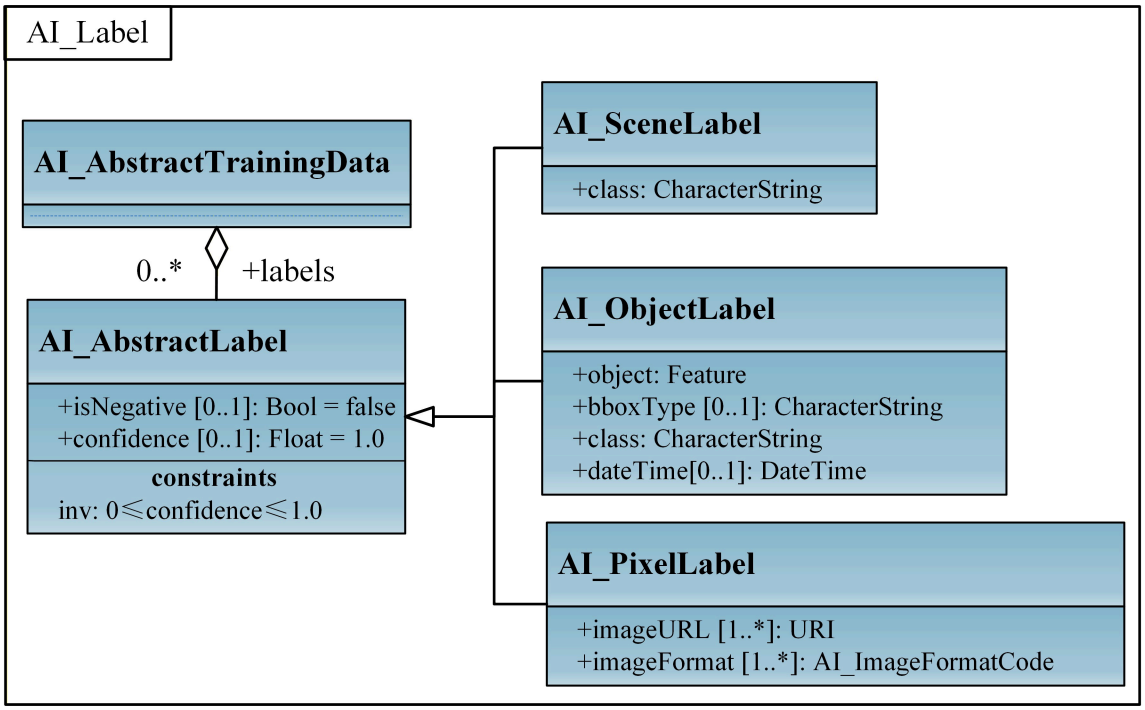

Figure 10 — UML diagram of AI_Label.

The Code Lists provided for the AI_Label are illustrated in Figure 11.

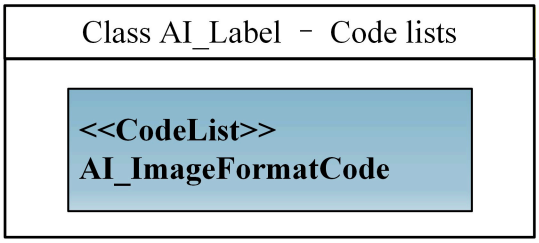

Figure 11 — Code lists from the AI_Label.

7.6.1. Requirements

The following requirement defines the rules governing the implementation of the AI_Label conceptual model as an implementation standard.

REQUIREMENT 4	
IDENTIFIER	`/req/ailabel/classes`
STATEMENT	For each UML class defined or referenced in AI_Label
A	Any implementation standard SHALL contain an element which represents the same concept as that defined for the UML class.
B	Any implementation standard SHALL represent associations with the same source, target, direction, roles, and multiplicities as those of the UML class.
C	Any implementation standard SHALL represent the attributes of the UML class including the name, definition, type, and multiplicity.
D	Any implementation standard SHALL represent the attributes of all super classes of the UML class including the name, definition, type, and multiplicity.
E	Any implementation standard SHALL represent the associations of all super classes of the UML class including the source, target, direction, roles, and multiplicity.
F	Any implementation standard SHALL specify how an implementation observes all constraints the Conceptual Model imposes on the UML class.

PERMISSION 4	
IDENTIFIER	`/per/ailabel/classes`
STATEMENT	For each UML class defined or referenced in AI_Label
A	An implementation standard MAY represent that class as a null class with no attributes, associations, or definition.
B	An implementation standard MAY represent an association of the UML class with a null association.
C	An implementation standard MAY represent an attribute of the UML class with a null attribute.
D	Whenever a null element is used to represent a concept from the AI_Label, the implementation standard SHOULD document that mapping and provide an explanation for why that concept was not implemented.

7.6.2. Class Definitions

Table 7 — Classes defined in AI_Label

NAME	DESCRIPTION
AI_AbstractLabel	AI_AbstractLabel defines a set of informative tags or metadata obtained by labelling training data. It can be represented using a feature, coverage, or a semantic class from ontologies or shared vocabularies.
AI_SceneLabel	AI_SceneLabel represents the scene level label using a semantic from ontologies or shared vocabularies.
AI_ObjectLabel	AI_ObjectLabel represents the object level label using a feature.
AI_PixelLabel	AI_PixelLabel represents the pixel level label using a coverage.

Table 8 — Code list classes defined in AI_Label

NAME	DESCRIPTION
AI_ImageFormatCode «CodeList»	AI_ImageFormatCode is a code list used to specify the MIME type of an image.

7.7. AI_Labeling

REQUIREMENTS CLASS 5	
IDENTIFIER	`http://www.opengis.net/spec/TrainingDML-AI-1/1.0/req/ailabeling`
TARGET TYPE	Implementation Standard
PREREQUISITES	ISO 19103:2024 ISO 19115-1:2014 /req/aitrainingdataset

AI_Labeling can be associated with AI_AbstractTrainingDataset or AI_AbstractTrainingData to record basic provenance information on how the training dataset or training samples are created. For example, AI_LabelingProcedure can be used to record the sampling strategy,

and dataSources from AI_AbstractTrainingDataset/ AI_AbstractTrainingData can refer to any additional geospatial data that was used as part of the sampling strategy.

AI_Labeling includes the labeler and the labeling procedure, which can be mapped to the agent and activity respectively in W3C PROV model. The labeler identifies the agent that creates the training dataset or individual samples, and the labeling procedure represents the process for data generation. AI_Labeling can also describe how the input data of the TD has been manipulated, such as resampled, color corrected, atmospherically corrected, and terrain corrected. For example, imagery with 10-cm and 15-cm resolution was resampled to 20-cm prior for labeling.

Alternatively, despite the simple payload in this TrainingDML-AI Standard, providing a comprehensive definition of provenance using the ISO 19115 lineage model through a metadata association in the data is also applicable.

The UML diagram of the AI_Labeling is illustrated in Figure 12.

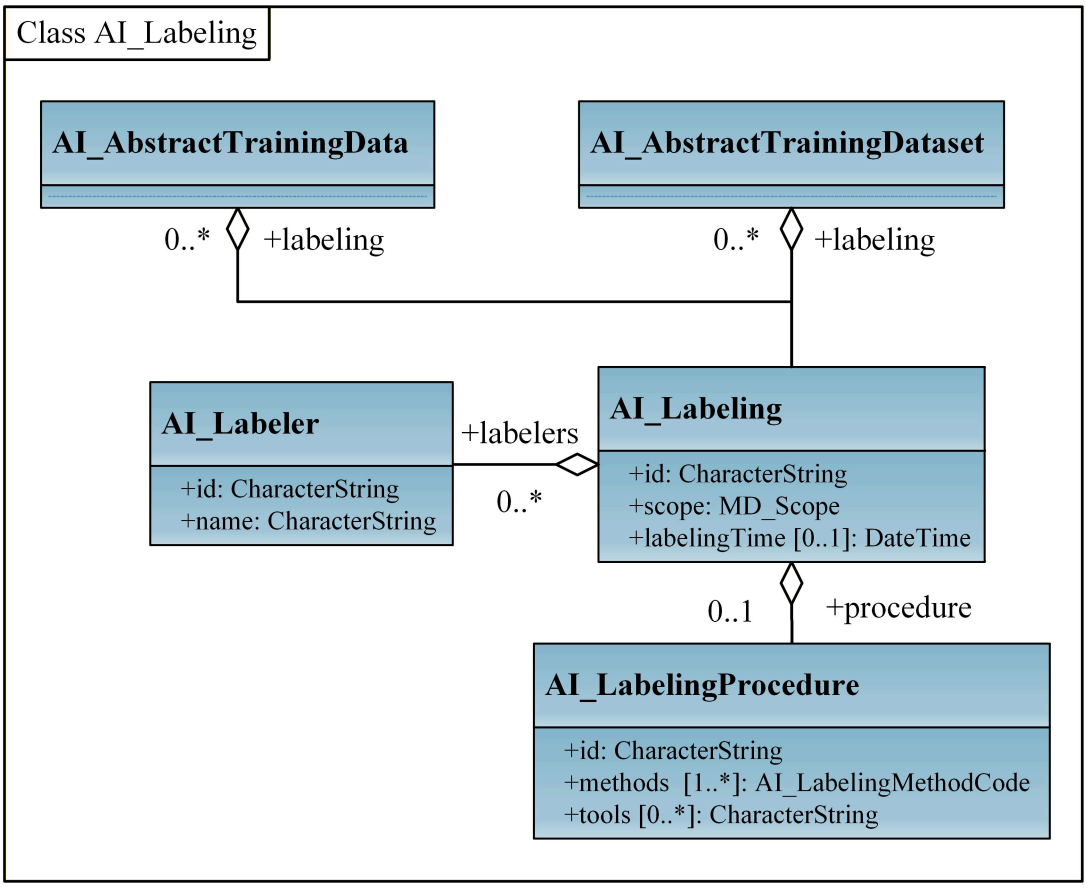

Figure 12 — UML diagram of AI_Labeling.

The Code Lists provided for the AI_Labeling are illustrated in Figure 13.

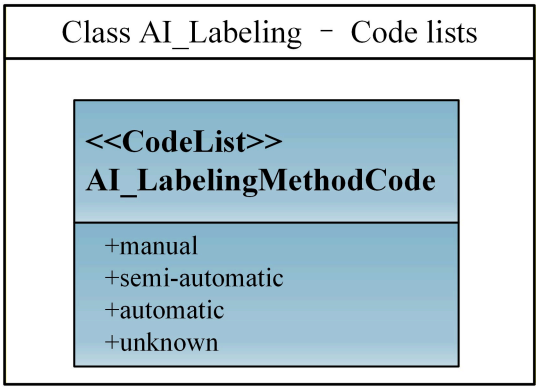

Figure 13 — Code lists from the AI_Labeling.

7.7.1. Requirements

The following requirement defines the rules governing the implementation of the AI_Labeling conceptual model as an implementation standard.

REQUIREMENT 5	
IDENTIFIER	`/req/ailabeling/classes`
STATEMENT	For each UML class defined or referenced in AI_Labeling Any implementation standard SHALL contain an element which represents the same concept as that defined for the UML class. Any implementation standard SHALL represent associations with the same source, target, direction, roles, and multiplicities as those of the UML class. Any implementation standard SHALL represent the attributes of the UML class including the name, definition, type, and multiplicity. Any implementation standard SHALL represent the attributes of all super classes of the UML class, including the name, definition, type, and multiplicity. Any implementation standard SHALL represent the associations of all super classes of the UML class, including the source, target, direction, roles, and multiplicity. Any implementation standard SHALL specify how an implementation observes all constraints the Conceptual Model imposes on the UML class.

PERMISSION 5	
IDENTIFIER	`/per/ailabeling/classes`
STATEMENT	For each UML class defined or referenced in AI_Labeling
A	An implementation standard MAY represent that class as a null class with no attributes, associations, or definition.

PERMISSION 5	
B	An implementation standard MAY represent an association of the UML class with a null association.
C	An implementation standard MAY represent an attribute of the UML class with a null attribute.
D	Whenever a null element is used to represent a concept from the AI_Labeling, the implementation standard SHOULD document that mapping and provide an explanation for why that concept was not implemented.

7.7.2. Class Definitions

Table 9 — Classes defined in AI_Labeling

NAME	DESCRIPTION
AI_Labeling	AI_Labeling defines basic provenance information on how to create the training dataset and includes the labeler and labeling procedure.
AI_Labeler	AI_Labeler identifies the agent that creates the labels and can be mapped to the agent in W3C PROV.
AI_LabelingProcedure	AI_LabelingProcedure represents the process for labeling and can be mapped to the activity in W3C PROV.

Table 10 — Code list classes defined in AI_Labeling

NAME	DESCRIPTION
AI_LabelingMethodCode «CodeList»	AI_LabelingMethodCode is a code list used to identify the methods used in the labeling procedure.

7.8. AI_TDChangeset

REQUIREMENTS CLASS 6	
IDENTIFIER	`http://www.opengis.net/spec/TrainingDML-AI-1/1.0/req/aitdchangeset`
TARGET TYPE	Implementation Standard

REQUIREMENTS CLASS 6	
PREREQUISITES	ISO 19103:2024 /req/aitrainingdata

AI_TDChangeset records changed training samples between two versions in the collection level, including added training samples, modified training samples and deleted training samples. AI_TDChangeset includes three kinds of changes (add, modify, and delete) for individuals.

The UML diagram of the AI_TDChangeset is illustrated in Figure 14.

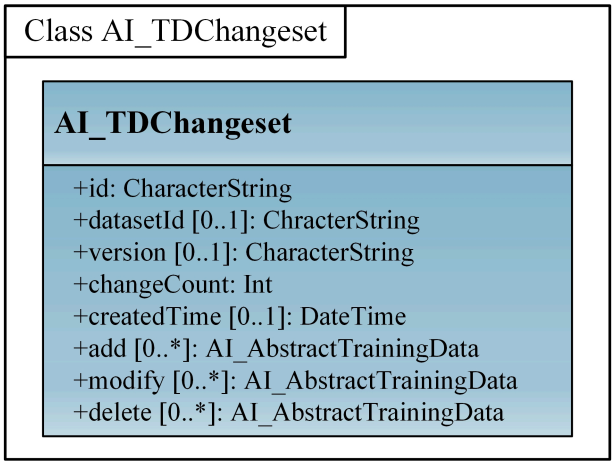

Figure 14 — UML diagram of AI_TDChangeset

7.8.1. Requirements

The following requirement defines the rules governing the implementation of the AI_TDChangeset conceptual model as an implementation standard.

REQUIREMENT 6	
IDENTIFIER	`/req/aitdchangeset/classes`
STATEMENT	For each UML class defined or referenced in AI_TDChangeset
A	Any implementation standard SHALL contain an element which represents the same concept as that defined for the UML class.
B	Any implementation standard SHALL represent associations with the same source, target, direction, roles, and multiplicities as those of the UML class.
C	Any implementation standard SHALL represent the attributes of the UML class, including the name, definition, type, and multiplicity.
D	Any implementation standard SHALL represent the attributes of all super classes of the UML class, including the name, definition, type, and multiplicity.

REQUIREMENT 6	
E	Any implementation standard SHALL represent the associations of all super classes of the UML class, including the source, target, direction, roles, and multiplicity.
F	Any implementation standard SHALL specify how an implementation observes all constraints the Conceptual Model imposes on the UML class.

PERMISSION 6	
IDENTIFIER	`/per/aitdchangeset/classes`
STATEMENT	For each UML class defined or referenced in AI_TDChangeset
A	An implementation standard MAY represent that class as a null class with no attributes, associations, or definition.
B	An implementation standard MAY represent an association of the UML class with a null association.
C	An implementation standard MAY represent an attribute of the UML class with a null attribute.
D	Whenever a null element is used to represent a concept from the AI_TDChangeset, the implementation standard SHOULD document that mapping and provide an explanation for why that concept was not implemented.

7.8.2. Class Definitions

Table 11 — Classes defined in AI_TDChangeset

NAME	DESCRIPTION
AI_TDChangeset	AI_TDChangeset represents a set of individual level changes (add, modify, and delete) of the training dataset.

7.9. AI_DataQuality

REQUIREMENTS CLASS 7	
IDENTIFIER	`http://www.opengis.net/spec/TrainingDML-AI-1/1.0/req/aidataquality`

REQUIREMENTS CLASS 7	
TARGET TYPE	Implementation Standard
PREREQUISITES	ISO 19157-1 /req/aitrainingdataset

TD quality description can use DataQuality from ISO 19157-1 to align with the existing efforts on geographic data quality. Data quality can be evaluated in terms of either collection or individual levels of the TD. Although most data quality elements are designed for datasets, it is sometimes needed to know the data quality at the individual level. For example, how is the positional uncertainty of an individual training sample measured by a GPS device in the field? Also, in terms of the object label, an individual training sample includes a source image and many object labels. In this case, the quality can be measured based on these labels, such as Completeness. Thus, the scope of the data quality can be used to specify the level and extent that identify the data on which data quality is to be established and evaluated. For TrainingDML-AI, AI_TrainingDataset/AI_TrainingData can be mapped to dataset/feature in MD_ScopeCode.

The quality description of the TD can also leverage the quality elements defined in QualityElement. For example, Completeness describes label commission and omission, Thematic accuracy describes class accuracy of labels. TD related quality elements like AI_ClassBalanceDegree can be defined by extending the QualityElement. AI_ClassBalanceDegree can help address the bias issue and evaluate whether the number of classes in the training dataset is imbalanced.

The UML diagram of the AI_DataQuality module is illustrated in Figure 15.

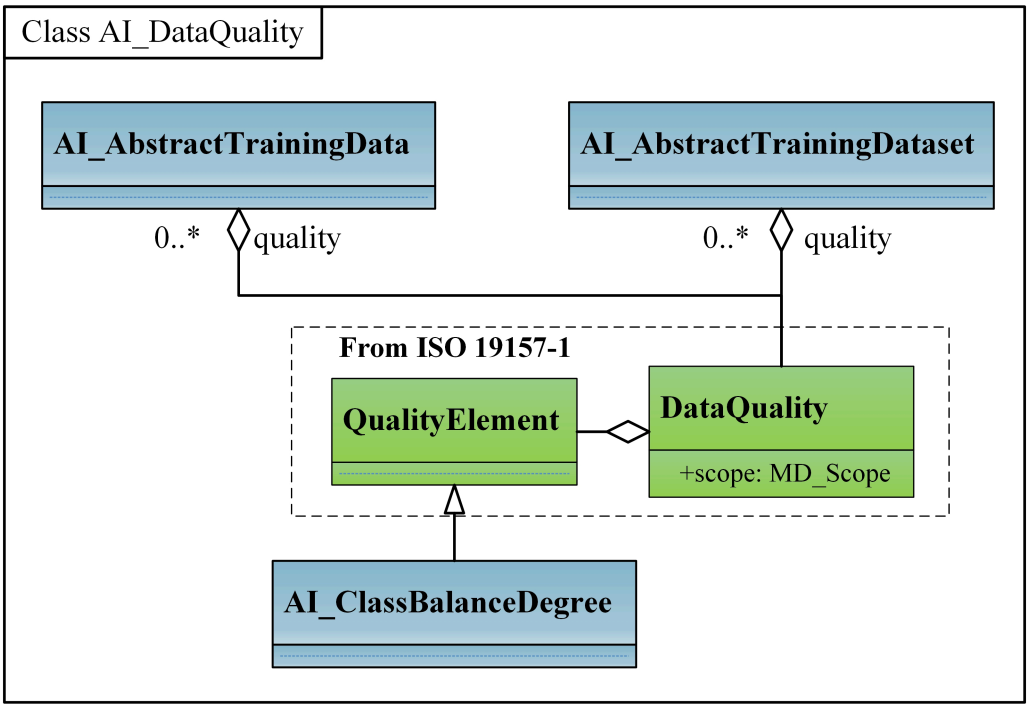

Figure 15 — UML diagram of AI_DataQuality.

7.9.1. Requirements

The following requirement defines the rules governing the implementation of the DataQuality conceptual model as an implementation standard.

REQUIREMENT 7	
IDENTIFIER	`/req/aidataquality/classes`
STATEMENT	For each UML class defined or referenced in DataQuality
A	Any implementation standard SHALL contain an element which represents the same concept as that defined for the UML class.
B	Any implementation standard SHALL represent associations with the same source, target, direction, roles, and multiplicities as those of the UML class.
C	Any implementation standard SHALL represent the attributes of the UML class, including the name, definition, type, and multiplicity.
D	Any implementation standard SHALL represent the attributes of all super classes of the UML class, including the name, definition, type, and multiplicity.
E	Any implementation standard SHALL represent the associations of all super classes of the UML class, including the source, target, direction, roles, and multiplicity.
F	Any implementation standard SHALL specify how an implementation observes all constraints the Conceptual Model imposes on the UML class.

PERMISSION 7	
IDENTIFIER	`/per/aidataquality/classes`
STATEMENT	For each UML class defined or referenced in DataQuality
A	An implementation standard MAY represent that class as a null class with no attributes, associations, or definition.
B	An implementation standard MAY represent an association of the UML class with a null association.
C	An implementation standard MAY represent an attribute of the UML class with a null attribute.
D	Whenever a null element is used to represent a concept from the DataQuality, the implementation standard SHOULD document that mapping and provide an explanation for why that concept was not implemented.

7.9.2. Class Definitions

Table 12 — Classes defined in AI_DataQuality

NAME	DESCRIPTION
AI_ClassBalanceDegree	AI_ClassBalanceDegree defines the quality elements for measuring degree of class balance of training datasets.

8

TRAININGDML-AI DATA DICTIONARY

8 TRAININGDML-AI DATA DICTIONARY

The TrainingDML-AI UML model is the normative definition of the TrainingDML-AI Conceptual Model. The Data Dictionary tables in this section were software generated from the UML model. As such, this section provides a normative representation of the TrainingDML-AI Conceptual Model.

8.1. ISO Classes

The following classes are defined in the ISO standards and used by the TrainingDML-AI Conceptual Model.

8.1.1. Feature (from ISO 19101-1:2014)

Table 13 — Metadata of Feature (Class)

Definition	Abstraction of real world phenomena
Subclass of	None
Stereotype	<<Type>>

8.1.2. MD_Band (from ISO 19115-1:2014)

Table 14 — Metadata of MD_Band (Class)

Definition	MD_Band is the range of wavelengths in the electromagnetic spectrum.
Subclass of	MD_SampleDimension
Stereotype	<<Class>>

8.1.3. MD_Scope (from ISO 19115-1:2014)

Table 15 — Metadata of MD_Scope (DataType)

Definition	The target resource and physical extent for which information is reported.
Subclass of	None
Stereotype	<<DataType>>

8.1.4. MD_ReferenceSystem (from ISO 19115-1:2014)

Table 16 — Metadata of MD_ReferenceSystem (Class)

Definition	Information about the reference system.
Subclass of	None
Stereotype	<<Class>>

8.1.5. LI_Lineage (from ISO 19115-1:2014)

Table 17 — Metadata of LI_Lineage (Class)

Definition	Information about the events or source data used in constructing the data specified by the scope or lack of knowledge about lineage.
Subclass of	None
Stereotype	<<Class>>

8.1.6. EX_Extent (from ISO 19115-1:2014)

Table 18 — Metadata of EX_Extent (DataType)

Definition	EX_Extent is the extent of the resource.
Subclass of	None
Stereotype	<<DataType>>

8.1.7. CI_Citation (from ISO 19115-1:2014)

Table 19 — Metadata of CI_Citation (Class)

Definition	CI_Citation is the standardized resource reference.
Subclass of	None
Stereotype	<<Class>>

8.1.8. MD_Resolution (from ISO 19115-1:2014)

Table 20 — Metadata of MD_Resolution (Class)

Definition	Level of detailed expressed as a scale factor, a distance or an angle.
Subclass of	None
Stereotype	<<Class>>

8.1.9. DataQuality (from ISO 19157-1)

Table 21 — Metadata of DataQuality (Class)

Definition	DataQuality is the quality information for the data specified by a data quality scope.
Subclass of	None
Stereotype	<<Class>>

8.1.10. QualityElement (from ISO 19157-1)

Table 22 — Metadata of QualityElement (Class)

Definition	QualityElement is the aspect of quantitative quality information.

Subclass of	None
Stereotype	<<Class>>

8.2. AI_TrainingDataset

Table 23 — Metadata of AI_TrainingDataset (ApplicationSchema)

Description	The AI_TrainingDataset module supports the modeling of different kinds of training dataset.
Parent package	TrainingDML-AI
Stereotype	<<ApplicationSchema>>

8.2.1. Classes

8.2.1.1. AI_AbstractTrainingDataset

Table 24 — Metadata of AI_AbstractTrainingDataset (Class)

Definition	AI_AbstractTrainingDataset defines the basic concepts and components of different kinds of training dataset.
Subclass of	None
Stereotype	<<Class>>

Table 25 — Attributes of AI_AbstractTrainingDataset (Class)

ATTRIBUTE	VALUE TYPE AND MULTIPLICITY	DEFINITION
id	CharacterString [1..1]	Identification of the AI training dataset.
doi	CharacterString [0..1]	Digital object identifier of the AI training dataset.

ATTRIBUTE	VALUE TYPE AND MULTIPLICITY	DEFINITION
scope	MD_Scope [0..1]	Description of the scope of the training dataset.
name	CharacterString [1..1]	Name of the AI training dataset.
description	CharacterString [1..1]	Description of the AI training dataset.
version	CharacterString [0..1]	Version number of the AI training dataset.
amountOfTrainingData	Int [0..1]	Total number of training samples in the AI training dataset.
createdTime	DateTime [0..1]	Time when the AI training dataset was created.
updatedTime	DateTime [0..1]	Time when the AI training dataset was updated.
license	CharacterString [1..1]	License description of the AI training dataset.
providers	CharacterString [0..*]	People or organizations who provide the AI training dataset.
keywords	CharacterString [0..*]	Keywords of the AI training dataset.
metricsInLIT	AI_MetricsInLiterature [0..*]	Results of performance metrics achieved by AI/ML algorithms in the peer-reviewed literature.
statisticsInfo	NamedValue [0..*]	Statistics results of training samples in each class.
dataSources	CI_Citation [0..*]	Citation of data sources.
numberOfClasses	Int [0..1]	Total number of classes in the AI training dataset.
classificationScheme	CharacterString [0..1]	Classification scheme for classes used in the AI training dataset.
classes	NamedValue [0..*]	Classes used in the AI training dataset.
extent	EX_Extent [0..1]	Spatial extent of the training dataset.
referenceSystem	MD_ReferenceSystem [0..1]	Reference system of the training dataset.
lineage	LI_Lineage [0..*]	Information about the provenance, source(s), and/or the production process(es) applied to the training dataset.

8.2.1.2. AI_EOTrainingDataset

Table 26 — Metadata of AI_EOTrainingDataset (Class)

Definition	AI_EOTrainingDataset describes attributes specific to EO training dataset. For example, the data source, image size, band information, and spatial extent.
Subclass of	AI_AbstractTrainingDataset
Stereotype	<<Class>>

Table 27 — Attributes of AI_EOTrainingDataset (Class)

ATTRIBUTE	VALUE TYPE AND MULTIPLICITY	DEFINITION
bands	MD_Band [0..*]	Description of the image bands used in the EO training dataset.
imageSize	CharacterString [0..1]	Size of the images used in the EO training dataset.

8.2.1.3. AI_MetricsInLiterature

Table 28 — Metadata of AI_MetricsInLiterature (DataType)

Definition	AI_MetricsInLiterature records results of performance metrics achieved by AI/ML algorithms in the peer-reviewed literature.
Subclass of	None
Stereotype	<<DataType>>

Table 29 — Attributes of AI_MetricsInLiterature (DataType)

ATTRIBUTE	VALUE TYPE AND MULTIPLICITY	DEFINITION
doi	CharacterString [1..1]	Digital object identifier of the peer-reviewed literature.
algorithm	CharacterString [0..1]	AI/ML algorithms used in the peer-reviewed literature.
metrics	NamedValue [1..*]	Metrics and results of AI/ML algorithms in the peer-reviewed literature.

8.3. AI_TrainingData

Table 30 — Metadata of AI_TrainingData (ApplicationSchema)

Description	AI_TrainingData module supports the modeling of an individual AI training sample.
Parent package	TrainingDML-AI
Stereotype	<<ApplicationSchema>>

8.3.1. Classes

8.3.1.1. AI_AbstractTrainingData

Table 31 — Metadata of AI_AbstractTrainingData (Class)

Definition	AI_AbstractTrainingData defines the basic concepts and components of different kinds of training data.
Subclass of	None
Stereotype	<<Class>>

Table 32 — Attributes of AI_AbstractTrainingData (Class)

ATTRIBUTE	VALUE TYPE AND MULTIPLICITY	DEFINITION
id	CharacterString [1..1]	Identification of an individual AI training sample.
datasetId	CharacterString [0..1]	Identification of the training dataset that the training sample belongs to.
trainingType	AI_TrainingTypeCode [0..1]	Training type of the individual AI training sample.
numberOfLabels	Int [0..1]	Total number of labels in the individual AI training sample.
dataSources	CI_Citation [0..*]	Citation of inputs to prepare a training sample.

ATTRIBUTE	VALUE TYPE AND MULTIPLICITY	DEFINITION
extent	EX_Extent [0..1]	Spatial extent of the individual training sample.
dataTime	DateTime [0..*]	Date and time at which the geospatial data were obtained.
lineage	LI_Lineage [0..*]	Information about the provenance, source(s), and/or the production process(es) applied to the training sample.

8.3.1.2. AI_EOTrainingData

Table 33 — Metadata of AI_EOTrainingData (Class)

Definition	AI_EOTrainingData describes attributes specific to EO training data. For example, spatial extent and date time of the source EO inputs.
Subclass of	AI_AbstractTrainingData
Stereotype	<<Class>>

Table 34 — Attributes of AI_EOTrainingData (Class)

ATTRIBUTE	VALUE TYPE AND MULTIPLICITY	DEFINITION
spatialResolution	MD_Resolution [0..1]	Spatial resolution of the individual EO training sample.
dataURL	URI [1..*]	URL of the EO data, including both relative and absolute paths, which can encompass local paths, network addresses, and more.

8.3.1.3. AI_TrainingTypeCode

Table 35 — Metadata of AI_TrainingTypeCode (CodeList)

Definition	AI_TrainingTypeCode is a code list used to identify whether the individual is used for different purposes.
Subclass of	None
Stereotype	<<CodeList>>

8.4. AI_Task

Table 36 — Metadata of AI_Task (ApplicationSchema)

Description	AI_Task module supports the modeling of different kinds of AI task.
Parent package	TrainingDML-AI
Stereotype	<<ApplicationSchema>>

8.4.1. Classes

8.4.1.1. AI_AbstractTask

Table 37 — Metadata of AI_AbstractTask (Class)

Definition	AI_AbstractTask defines the AI/ML tasks in specific domains that have different organizing forms of TD.
Subclass of	None
Stereotype	<<Class>>

Table 38 — Attributes of AI_AbstractTask (Class)

ATTRIBUTE	VALUE TYPE AND MULTIPLICITY	DEFINITION
id	CharacterString [1..1]	Identification of the task.
datasetId	CharacterString [0..1]	Identification of the training dataset the training sample belongs to.
description	CharacterString [0..1]	Description of the AI task.

8.4.1.2. AI_EOTask

Table 39 — Metadata of AI_EOTask (Class)

Definition	AI_EOTask extends AI_AbstractTask to represent specific AI/ML tasks in the EO domain.
Subclass of	AI_AbstractTask
Stereotype	<<Class>>

Table 40 — Attributes of AI_EOTask (Class)

ATTRIBUTE	VALUE TYPE AND MULTIPLICITY	DEFINITION
taskType	CharacterString [1..1]	Type description of the EO task.

8.5. AI_Label

Table 41 — Metadata of AI_Label (ApplicationSchema)

Description	AI_Label module supports the modeling of different kinds of AI Label.
Parent package	TrainingDML-AI
Stereotype	<<ApplicationSchema>>

8.5.1. Classes

8.5.1.1. AI_AbstractLabel

Table 42 — Metadata of AI_AbstractLabel (Class)

Definition	AI_AbstractLabel defines a set of informative tags or metadata obtained by labelling training data. AI_AbstractLabel can be represented using a feature, coverage, or a semantic class from ontologies or shared vocabularies.
Subclass of	None
Stereotype	<<Class>>

Constraint	valueBetween0and1:
	inv: 0≤confidence≤1.0

Table 43 — Attributes of AI_AbstractLabel (Class)

ATTRIBUTE	VALUE TYPE AND MULTIPLICITY	DEFINITION
isNegative	Bool [0..1]	Whether the training sample related to the label is a positive or negative sample.
confidence	Float [0..1]	Confidence score of the labeler.

8.5.1.2. AI_SceneLabel

Table 44 — Metadata of AI_SceneLabel (Class)

Definition	AI_SceneLabel represents the scene level label using a semantic type from ontologies or shared vocabularies.
Subclass of	AI_AbstractLabel
Stereotype	<<Class>>

Table 45 — Attributes of AI_SceneLabel (Class)

ATTRIBUTE	VALUE TYPE AND MULTIPLICITY	DEFINITION
class	CharacterString [1..1]	Class name that records the semantic type of the scene of the training sample.

8.5.1.3. AI_ObjectLabel

Table 46 — Metadata of AI_ObjectLabel (Class)

Definition	AI_ObjectLabel represents the object level label using a feature.
Subclass of	AI_AbstractLabel
Stereotype	<<Class>>

Table 47 — Attributes of AI_ObjectLabel (Class)

ATTRIBUTE	VALUE TYPE AND MULTIPLICITY	DEFINITION
object	Feature [1..1]	Feature that represents the position and attributes of the object.
bboxType	CharacterString [0..1]	Type of the bbox.
class	CharacterString [1..1]	Class that records the semantic of the object type.
dateTime	DateTime [0..1]	Created time of the object label.

8.5.1.4. AI_PixelLabel

Table 48 — Metadata of AI_PixelLabel (Class)

Definition	AI_PixelLabel represents the pixel level label using a coverage.
Subclass of	AI_AbstractLabel
Stereotype	<<Class>>

Table 49 — Attributes of AI_PixelLabel (Class)

ATTRIBUTE	VALUE TYPE AND MULTIPLICITY	DEFINITION
imageURL	URI [1..*]	URL of the images representing the label information.
imageFormat	AI_ImageFormatCode [1..*]	Image data format.

8.5.1.5. AI_ImageFormatCode

Table 50 — Metadata of AI_ImageFormatCode (CodeList)

Definition	AI_ImageFormatCode is a code list used to specify the MIME type of an image.
Subclass of	None
Stereotype	<<CodeList>>

8.6. AI_Labeling

Table 51 — Metadata of AI_Labeling (ApplicationSchema)

Description	AI_Labeling module supports the modeling of provenance information of the training dataset.
Parent package	TrainingDML-AI
Stereotype	<<ApplicationSchema>>

8.6.1. Classes

8.6.1.1. AI_Labeling

Table 52 — Metadata of AI_Labeling (Class)

Definition	AI_Labeling defines basic provenance information on how to create the training dataset. AI_Labeling includes the labeler and labeling procedure.
Subclass of	None
Stereotype	<<Class>>

Table 53 — Attributes of AI_Labeling (Class)

ATTRIBUTE	VALUE TYPE AND MULTIPLICITY	DEFINITION
id	CharacterString [1..1]	Identifier of the labeling.
scope	MD_Scope [1..1]	Description of the scope of the labeling.
labelingTime	DateTime [0..1]	Date and time at which the labelling procedure was carried out.

8.6.1.2. AI_Labeler

Table 54 — Metadata of AI_Labeler (Class)

Definition	AI_Labeler identifies the agent that creates the labels and can be mapped to the agent in W3C PROV.
Subclass of	None
Stereotype	<<Class>>

Table 55 — Attributes of AI_Labeler (Class)

ATTRIBUTE	VALUE TYPE AND MULTIPLICITY	DEFINITION
id	CharacterString [1..1]	Identifier of the labeler.
name	CharacterString [1..1]	Name of the labeler.

8.6.1.3. AI_LabelingProcedure

Table 56 — Metadata of AI_LabelingProcedure (Class)

Definition	AI_LabelingProcedure represents the process for labeling and can be mapped to the activity in W3C PROV.
Subclass of	None
Stereotype	<<Class>>

Table 57 — Attributes of AI_LabelingProcedure (Class)

ATTRIBUTE	VALUE TYPE AND MULTIPLICITY	DEFINITION
id	CharacterString [1..1]	Identifier of the labeling procedure
methods	AI_LabelingMethodCode [1..*]	Methods used in the labeling procedure.
tools	CharacterString [0..*]	Tools or software used in the labeling procedure.

8.6.1.4. AI_LabelingMethodCode

Table 58 — Metadata of AI_LabelingMethodCode (CodeList)

Definition	AI_LabelingMethodCode is a code list used to identify the methods used in the labeling procedure.
Subclass of	None
Stereotype	<<CodeList>>

8.7. AI_TDChangeset

Table 59 — Metadata of AI_TDChangeset (ApplicationSchema)

Description	AI_TDChangeset module supports the modeling of change information of the training dataset.
Parent package	TrainingDML-AI
Stereotype	<<ApplicationSchema>>

8.7.1. Classes

8.7.1.1. AI_TDChangeset

Table 60 — Metadata of AI_TDChangeset (Class)

Definition	AI_TDChangeset represents a set of individual level changes (add, modify, and delete) of the training dataset.
Subclass of	None
Stereotype	<<Class>>

Table 61 — Attributes of AI_TDChangeset (Class)

ATTRIBUTE	VALUE TYPE AND MULTIPLICITY	DEFINITION
id	CharacterString [1..1]	Identifier of the changeset.

ATTRIBUTE	VALUE TYPE AND MULTIPLICITY	DEFINITION
datasetId	CharacterString [0..1]	Identifier of the training dataset the changeset belongs to.
version	CharacterString [0..1]	Version of the training dataset that the changeset belongs to.
changeCount	Int [1..1]	Total number of changed training samples.
createdTime	DateTime [0..1]	Created time of the changeset.
add	AI_AbstractTrainingData [0..*]	Added training samples.
modify	AI_AbstractTrainingData [0..*]	Modified training samples.
delete	AI_AbstractTrainingData [0..*]	Deleted training samples.

8.8. AI_DataQuality

Table 62 — Metadata of AI_DataQuality (ApplicationSchema)

Description	AI_DataQuality module supports the modeling of quality description of different kinds of training datasets.
Parent package	TrainingDML-AI
Stereotype	<<ApplicationSchema>>

8.8.1. Classes

8.8.1.1. AI_ClassBalanceDegree

Table 63 — Metadata of AI_ClassBalanceDegree (Class)

Definition	AI_ClassBalanceDegree defines the quality elements for measuring degree of class balance of training datasets.
Subclass of	QualityElement

Stereotype	<<Class>>

Table 64 — Attributes of AI_ClassBalanceDegree (Class)

ATTRIBUTE	VALUE TYPE AND MULTIPLICITY	DEFINITION
measure	MeasureReference [1..1]	Reference to measure used
evaluationMethod	EvaluationMethod [1..*]	Evaluation information
result	QualityResult [1..*]	Value obtained from applying a data quality measure

A

ANNEX A (NORMATIVE) ABSTRACT TEST SUITE

ANNEX A (NORMATIVE) ABSTRACT TEST SUITE

A.1. Introduction

The test method for TrainingDML-AI conceptual model specified in this ATS is manual inspection. Automated methods may be used where they exist.

A.2. Conformance Class AI_TrainingDataset

CONFORMANCE CLASS A.1	
IDENTIFIER	`http://www.opengis.net/spec/TrainingDML-AI-1/1.0/conf/aitrainingdataset`
REQUIREMENTS CLASS	/req/aitrainingdataset

ABSTRACT TEST A.1	
IDENTIFIER	`/conf/aitrainingdataset/classes`
REQUIREMENT	Requirement 1: `/req/aitrainingdataset/classes`
TEST PURPOSE	To validate that the implementation standard correctly implements the UML Classes defined in the Conceptual Model.
TEST-METHOD-TYPE	Manual Inspection
TEST METHOD	For each UML class defined or referenced in the AI_TrainingDataset Package, validate as follows. 1. Validate that the implementation standard contains a data element which represents the same concept as that defined for the UML class.

ABSTRACT TEST A.1	
	2. Validate that the data element has the same relationships with other elements as those defined for the UML class. Validate that those relationships have the same source, target, direction, roles, and multiplicity as those documented in the Conceptual Model. 3. Validate that the data element has the same properties (attributes) as those specified for the UML class. Validate that those properties have the same name, definition, type, and multiplicity of those documented in the Conceptual Model. 4. Validate that the properties of the data element include those of all super classes of the UML class as documented in the Conceptual Model. Validate that those properties have the same name, definition, type, and multiplicity of those documented in the Conceptual Model. 5. Validate that the associations represented for the data element include those of all super classes of the UML class as documented in the Conceptual Model. Validate that those representations have the same source, target, roles, and multiplicity of those documented in the Conceptual Model. 6. Validate that the implementation standard enforces all constraints imposed on the UML class by the Conceptual Model.

A.3. Conformance Class AI_TrainingData

CONFORMANCE CLASS A.2	
IDENTIFIER	`http://www.opengis.net/spec/TrainingDML-AI-1/1.0/conf/aitrainingdata`
REQUIREMENTS CLASS	/req/aitrainingdata

ABSTRACT TEST A.2	
IDENTIFIER	`/conf/aitrainingdata/classes`
REQUIREMENT	Requirement 2: `/req/aitrainingdata/classes`
TEST PURPOSE	To validate that the implementation standard correctly implements the UML Classes defined in the Conceptual Model.
TEST-METHOD-TYPE	Manual Inspection
TEST METHOD	For each UML class defined or referenced in the AI_TrainingData Package, validate as follows. 1. Validate that the implementation standard contains a data element which represents the same concept as that defined for the UML class.

ABSTRACT TEST A.2	
	2. Validate that the data element has the same relationships with other elements as those defined for the UML class. Validate that those relationships have the same source, target, direction, roles, and multiplicity as those documented in the Conceptual Model. 3. Validate that the data element has the same properties (attributes) as those specified for the UML class. Validate that those properties have the same name, definition, type, and multiplicity of those documented in the Conceptual Model. 4. Validate that the properties of the data element include those of all super classes of the UML class as documented in the Conceptual Model. Validate that those properties have the same name, definition, type, and multiplicity of those documented in the Conceptual Model. 5. Validate that the associations represented for the data element include those of all super classes of the UML class as documented in the Conceptual Model. Validate that those representations have the same source, target, roles, and multiplicity of those documented in the Conceptual Model. 6. Validate that the implementation standard enforces all constraints imposed on the UML class by the Conceptual Model.

A.4. Conformance Class AI_Task

CONFORMANCE CLASS A.3	
IDENTIFIER	`http://www.opengis.net/spec/TrainingDML-AI-1/1.0/conf/aitask`
REQUIREMENTS CLASS	/req/aitask

ABSTRACT TEST A.3	
IDENTIFIER	`/conf/aitask/classes`
REQUIREMENT	Requirement 3: `/req/aitask/classes`
TEST PURPOSE	To validate that the implementation standard correctly implements the UML Classes defined in the Conceptual Model.
TEST-METHOD-TYPE	Manual Inspection
TEST METHOD	For each UML class defined or referenced in the AI_Task Package, validate as follows. 1. Validate that the implementation standard contains a data element which represents the same concept as that defined for the UML class.

ABSTRACT TEST A.3	
	2. Validate that the data element has the same relationships with other elements as those defined for the UML class. Validate that those relationships have the same source, target, direction, roles, and multiplicity as those documented in the Conceptual Model. 3. Validate that the data element has the same properties (attributes) as those specified for the UML class. Validate that those properties have the same name, definition, type, and multiplicity of those documented in the Conceptual Model. 4. Validate that the properties of the data element include those of all super classes of the UML class as documented in the Conceptual Model. Validate that those properties have the same name, definition, type, and multiplicity of those documented in the Conceptual Model. 5. Validate that the associations represented for the data element include those of all super classes of the UML class as documented in the Conceptual Model. Validate that those representations have the same source, target, roles, and multiplicity of those documented in the Conceptual Model. 6. Validate that the implementation standard enforces all constraints imposed on the UML class by the Conceptual Model.

A.5. Conformance Class AI_Label

CONFORMANCE CLASS A.4	
IDENTIFIER	`http://www.opengis.net/spec/TrainingDML-AI-1/1.0/conf/ailabel`
REQUIREMENTS CLASS	/req/ailabel

ABSTRACT TEST A.4	
IDENTIFIER	`/conf/ailabel/classes`
REQUIREMENT	Requirement 4: `/req/ailabel/classes`
TEST PURPOSE	To validate that the implementation standard correctly implements the UML Classes defined in the Conceptual Model.
TEST-METHOD-TYPE	Manual Inspection
TEST METHOD	For each UML class defined or referenced in the AI_Label Package, validate as follows. 1. Validate that the implementation standard contains a data element which represents the same concept as that defined for the UML class.

ABSTRACT TEST A.4	
	2. Validate that the data element has the same relationships with other elements as those defined for the UML class. Validate that those relationships have the same source, target, direction, roles, and multiplicity as those documented in the Conceptual Model. 3. Validate that the data element has the same properties (attributes) as those specified for the UML class. Validate that those properties have the same name, definition, type, and multiplicity of those documented in the Conceptual Model. 4. Validate that the properties of the data element include those of all super classes of the UML class as documented in the Conceptual Model. Validate that those properties have the same name, definition, type, and multiplicity of those documented in the Conceptual Model. 5. Validate that the associations represented for the data element include those of all super classes of the UML class as documented in the Conceptual Model. Validate that those representations have the same source, target, roles, and multiplicity of those documented in the Conceptual Model. 6. Validate that the implementation standard enforces all constraints imposed on the UML class by the Conceptual Model.

A.6. Conformance Class AI_Labeling

CONFORMANCE CLASS A.5	
IDENTIFIER	`http://www.opengis.net/spec/TrainingDML-AI-1/1.0/conf/ailabeling`
REQUIREMENTS CLASS	/req/ailabeling

ABSTRACT TEST A.5	
IDENTIFIER	`/conf/ailabeling/classes`
REQUIREMENT	Requirement 5: `/req/ailabeling/classes`
TEST PURPOSE	To validate that the implementation standard correctly implements the UML Classes defined in the Conceptual Model.
TEST-METHOD-TYPE	Manual Inspection
TEST METHOD	For each UML class defined or referenced in the AI_Labeling Package, validate as follows. 1. Validate that the implementation standard contains a data element which represents the same concept as that defined for the UML class.

ABSTRACT TEST A.5	
	2. Validate that the data element has the same relationships with other elements as those defined for the UML class. Validate that those relationships have the same source, target, direction, roles, and multiplicity as those documented in the Conceptual Model. 3. Validate that the data element has the same properties (attributes) as those specified for the UML class. Validate that those properties have the same name, definition, type, and multiplicity of those documented in the Conceptual Model. 4. Validate that the properties of the data element include those of all super classes of the UML class as documented in the Conceptual Model. Validate that those properties have the same name, definition, type, and multiplicity of those documented in the Conceptual Model. 5. Validate that the associations represented for the data element include those of all super classes of the UML class as documented in the Conceptual Model. Validate that those representations have the same source, target, roles, and multiplicity of those documented in the Conceptual Model. 6. Validate that the implementation standard enforces all constraints imposed on the UML class by the Conceptual Model.

A.7. Conformance Class AI_TDChangeset

CONFORMANCE CLASS A.6	
IDENTIFIER	`http://www.opengis.net/spec/TrainingDML-AI-1/1.0/conf/aitdchangeset`
REQUIREMENTS CLASS	/req/aitdchangeset

ABSTRACT TEST A.6	
IDENTIFIER	`/conf/aitdchangeset/classes`
REQUIREMENT	Requirement 6: `/req/aitdchangeset/classes`
TEST PURPOSE	To validate that the implementation standard correctly implements the UML Classes defined in the Conceptual Model.
TEST-METHOD-TYPE	Manual Inspection
TEST METHOD	For each UML class defined or referenced in the AI_TDChangeset Package, validate as follows. 1. Validate that the implementation standard contains a data element which represents the same concept as that defined for the UML class.

ABSTRACT TEST A.6	
	2. Validate that the data element has the same relationships with other elements as those defined for the UML class. Validate that those relationships have the same source, target, direction, roles, and multiplicity as those documented in the Conceptual Model. 3. Validate that the data element has the same properties (attributes) as those specified for the UML class. Validate that those properties have the same name, definition, type, and multiplicity of those documented in the Conceptual Model. 4. Validate that the properties of the data element include those of all super classes of the UML class as documented in the Conceptual Model. Validate that those properties have the same name, definition, type, and multiplicity of those documented in the Conceptual Model. 5. Validate that the associations represented for the data element include those of all super classes of the UML class as documented in the Conceptual Model. Validate that those representations have the same source, target, roles, and multiplicity of those documented in the Conceptual Model. 6. Validate that the implementation standard enforces all constraints imposed on the UML class by the Conceptual Model.

A.8. Conformance Class AI_DataQuality

CONFORMANCE CLASS A.7	
IDENTIFIER	`http://www.opengis.net/spec/TrainingDML-AI-1/1.0/conf/aidataquality`
REQUIREMENTS CLASS	/req/aidataquality

ABSTRACT TEST A.7	
IDENTIFIER	`/conf/aidataquality/classes`
REQUIREMENT	Requirement 7: `/req/aidataquality/classes`
TEST PURPOSE	To validate that the implementation standard correctly implements the UML Classes defined in the Conceptual Model.
TEST-METHOD-TYPE	Manual Inspection
TEST METHOD	For each UML class defined or referenced in the AI_DataQuality Package, validate as follows. 1. Validate that the implementation standard contains a data element which represents the same concept as that defined for the UML class.

ABSTRACT TEST A.7

2. Validate that the data element has the same relationships with other elements as those defined for the UML class. Validate that those relationships have the same source, target, direction, roles, and multiplicity as those documented in the Conceptual Model.

3. Validate that the data element has the same properties (attributes) as those specified for the UML class. Validate that those properties have the same name, definition, type, and multiplicity of those documented in the Conceptual Model.

4. Validate that the properties of the data element include those of all super classes of the UML class as documented in the Conceptual Model. Validate that those properties have the same name, definition, type, and multiplicity of those documented in the Conceptual Model.

5. Validate that the associations represented for the data element include those of all super classes of the UML class as documented in the Conceptual Model. Validate that those representations have the same source, target, roles, and multiplicity of those documented in the Conceptual Model.

6. Validate that the implementation standard enforces all constraints imposed on the UML class by the Conceptual Model.

B

ANNEX B (INFORMATIVE) EXAMPLE

B ANNEX B (INFORMATIVE) EXAMPLE

B.1. TrainingDataset Encoding Examples

B.1.1. WHU-RS19 Dataset

The WHU-RS19 dataset is widely used in scene classification of remote sensing images. This dataset is collected from Google Earth and has 19 classes including airport, beach, bridge, commercial, desert, farmland, football field, forest, industrial, meadow, mountain, park, parking, pond, port, railway station, residential, river, and viaduct. Each class contains around 50 images, with the image size 600×600 and a resolution of 0.5 m.

An example of JSON encoding of the WHU-RS19 dataset following the TrainingDML-AI UML model can be found in https://github.com/opengeospatial/TrainingDML-AI_SWG/tree/main/use-cases/examples/1.0/WHU-RS19.json.

B.1.2. DOTA-v1.5 Dataset

The DOTA-v1.5 dataset is a large-scale dataset for object detection in aerial images. The sources for content in the dataset include Google Earth, Gaofen-2, and Jilin-1 imagery provided by China Resources Satellite Data Center. The 16 classes in DOTA-v1.5 are plane, ship, storage tank, baseball diamond, tennis court, basketball court, ground track field, harbor, bridge, large vehicle, small vehicle, helicopter, roundabout, soccer ball field, swimming pool, and container crane. Compared with other aerial image object detection datasets, the dataset has the largest number of classes. The images in the dataset have various image sizes (from 800×800 to 2000×2000) and resolutions (Google Earth/0.1 m-1 m, Gaofen-2/1 m, Jilin-1/0.72 m).

An example of JSON encoding of the DOTA-v1.5 dataset following the TrainingDML-AI UML model can be found in https://github.com/opengeospatial/TrainingDML-AI_SWG/tree/main/use-cases/examples/1.0/DOTA-v1.5.json.

B.1.3. KITTI 2D Object Detection Dataset

The KITTI 2D object detection dataset is a novel open-access dataset and benchmark for road area and ego-lane detection. KITTI 2D consists of 7481 annotated training images of high variability from the KITTI autonomous driving platform by 2 PointGrey Flea2 color cameras,

capturing a broad spectrum of urban street views and road scenes. The eight (8) classes in the KITTI 2D object detection dataset are car, van, truck, pedestrian, person_sitting, cyclist, tram, and misc. Compared with other street view object detection datasets, this dataset compresses diverse scenarios and captures real-world traffic situations, ranging from freeways over rural areas to inner-city scenes with many static and dynamic objects.

An example of JSON encoding of the KITTI 2D object detection dataset following the TrainingDML-AI UML model can be found in https://github.com/opengeospatial/TrainingDML-AI_SWG/tree/main/use-cases/examples/1.0/KITTI.json.

B.1.4. GID Dataset

The GID dataset is one of start-of-art land cover classification datasets. This dataset has a large spatial coverage covering many provinces in China with a relatively high spatial resolution (2m). GID has two sets. One is the GID-5C. It has 150 images (image size 7200×6800) that are classified into 5 land cover classes. The other set is GID-15C. The images from GID-5C are sliced into 30,000 patches in GID-15C, which have three types of patch sizes (56×56, 112×112, 224×224) and are classified into 15 land cover classes.

An example of JSON encoding of the GID-5C dataset following the TrainingDML-AI UML model can be found in https://github.com/opengeospatial/TrainingDML-AI_SWG/tree/main/use-cases/examples/1.0/GID-5C.json.

B.1.5. Toronto3D Dataset

The Toronto3D dataset is a large urban outdoor point cloud dataset for segmentation collected by the Mobile Laser Scanning System. The dataset covers about 1 km of scene streets in Toronto, including four areas named L001, L002, L003, and L004, with a total of 78.3 million points. Each point in this dataset has 10 attributes representing the 3D position, RGB color, intensity, GPS time, scan angle rank, and category, respectively. This dataset has eight categories, including road, road mark, natural, building, utility line, pole, car, and fence.

An example of JSON encoding of the Toronto3D dataset following the TrainingDML-AI UML model can be found in https://github.com/opengeospatial/TrainingDML-AI_SWG/tree/main/use-cases/examples/1.0/Toronto_3D.json.

B.1.6. WHU-Building Dataset

The WHU-Building dataset is a change detection dataset collected from the Land Information New Zealand Data Service. The dataset is composed of images (with the resolution 0.2m) in 2012 and 2016, covering 20.5 km2. It includes 12,796 and 16,077 buildings respectively in 2012 and 2016.

An example of JSON encoding of the WHU-Building dataset following the TrainingDML-AI UML model can be found in https://github.com/opengeospatial/TrainingDML-AI_SWG/tree/main/use-cases/examples/1.0/WHU-building.json.

B.1.7. California Change Detection Dataset

The California Change Detection Dataset is composed of two images and a label image. The first image is a Landsat 8 acquisition covering Sacramento County, Yuba County and Sutter County, California, on 5 January 2017. It has nine channels covering the spectrum from deep blue to short-wave infrared, plus two long-wave infrared channels. The second image was acquired on 18 February 2017 by Sentinel-1A over the same area after the occurrence of a flood. The image is recorded in polarizations VV and VH and augmented with the ratio between the two intensities as a third channel. All these channels are log-transformed.

An example of JSON encoding of the California change detection dataset following the TrainingDML-AI UML model can be found in https://github.com/opengeospatial/TrainingDML-AI_SWG/tree/main/use-cases/examples/1.0/UiT_HCD_California_2017.json.

B.1.8. WHU MVS Dataset

The WHU MVS dataset is a synthetic aerial dataset created for large-scale and high-resolution Earth surface reconstruction. The basic training sample of the dataset is a multi-view unit consisting of five aerial images, and their corresponding depth maps are taken as ground truth. There are a total of 5680 pairs of five-view aerial images in the dataset. All the images are simulated from a 3D surface model, which is produced by Smart3D software using Unmanned Aerial Vehicle (UAV) images and refined by manual editing.

An example of JSON encoding of the WHU MVS dataset following the TrainingDML-AI UML model can be found in https://github.com/opengeospatial/TrainingDML-AI_SWG/tree/main/use-cases/examples/1.0/WHU_MVS.json.

B.2. DataQuality Encoding Example

B.2.1. WHU-RS19 Data Quality

An encoded data quality example of the WHU-RS19 datasets following the TrainingDML-AI UML model can be found in https://github.com/opengeospatial/TrainingDML-AI_SWG/tree/main/use-cases/examples/1.0/WHU-RS19-quality.json.

B.3. TDChangeset Encoding Examples

B.3.1. DOTA-v1.5 Changeset

DOTA-v1.5 uses the same images as DOTA-v1.0, but the extremely small instances (less than 10 pixels) are also annotated. Moreover, a new category “container crane” is added. It contains 403,318 instances in total. The number of images and dataset splits are the same as DOTA-v1.0. This version was released for the DOAI Challenge 2019 on Object Detection in Aerial Images in conjunction with IEEE CVPR 2019.

An encoded changeset example between the DOTA-v1.0 and DOTA-v1.5 datasets following the TrainingDML-AI UML model can be found in https://github.com/opengeospatial/TrainingDML-AI_SWG/tree/main/use-cases/examples/1.0/DOTA-v1.5-changeset.json.

C

ANNEX C (INFORMATIVE) REVISION HISTORY

ANNEX C (INFORMATIVE) REVISION HISTORY

Table C.1

DATE	RELEASE	EDITOR	PRIMARY CLAUSES MODIFIED	DESCRIPTION
2022-08-01	0.1	Peng Yue, Boyi Shangguan	All	Draft for internal review.
2022-09-09	0.2	Peng Yue, Boyi Shangguan, Carl Reed	Most	Merge edits and comments from Carl Reed.
2022-11-10	0.3	Peng Yue, Boyi Shangguan	Most	Revisions based on comments from Testbed-18 participants.
2023-01-13	0.6	Peng Yue, Boyi Shangguan	Most	Revisions based on comments from Testbed-18 ML TD ER.
2023-01-29	0.7	Peng Yue, Boyi Shangguan, Ruixiang Liu	General editorial	Internal review done and submission to request OAB review.
2023-04-23	0.8	Peng Yue, Boyi Shangguan, Ruixiang Liu	Chapter 6.3.4, 7.9, Annex A	Revisions after OGC-NA review and OAB review.
2023-05-17	0.9	Peng Yue, Boyi Shangguan, Ruixiang Liu	Chapter 3, 4, Annex D	Revisions after public comments and submission for TC vote.
2023-08-04	1.0	Peng Yue, Boyi Shangguan, Ruixiang Liu	General editorial	Revisions before publishing.
2026-02-28	1.0.1	Peng Yue, Baoxin Teng	Most	Revisions based on the OAB comments on Parts 2&3, and the DIS comments on ISO 19178-1.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Open Geospatial Consortium, n.d. How Our Compliance Program Works. https://www.ogc.org/how-our-compliance-program-works/

[2] Groth, P., Moreau, L., eds., 2013. PROV-Overview: An Overview of the PROV Family of Documents. W3C Note. https://www.w3.org/TR/prov-overview/

[3] Yue, P., Shangguan, B., Hu, L., Jiang, L., Zhang, C., Cao, Z., Pan, Y., 2022. Towards a training data model for artificial intelligence in earth observation. International Journal of Geographical Information Science, 1-25. https://doi.org/10.1080/13658816.2022.2087223

[4] ISO 9000:2005, Quality management systems — Fundamentals and vocabulary. https://www.iso.org/standard/42180.html

[5] ISO 19109, Geographic information — Rules for application schema. https://www.iso.org/standard/59193.html

[6] ISO 19144-1, Geographic information — Classification systems — Part 1: Classification system structure. https://www.iso.org/standard/32562.html

[7] ISO 19144-2:2023, Geographic information — Classification systems — Part 2: Land Cover Meta Language (LCML). https://www.iso.org/standard/81259.html