|  |
| --- |
|  |

|  |  |
| --- | --- |
| Input Contribution | |
| Meeting ID\* | RDM#55 |
| Title:\* | Key issue on data labeling |
| Source:\* | JaeSeung Song, Sejong University, jssong@sejong.ac.kr |
| Date:\* | 2022-07-11 |
| Input related to\* | TR-0068 (Rel-5) |
| Intended purpose of  document:\* | Decision  Discussion  Information  Other <specify> |
| Impacted other TS/TR(s) |  |
| Decision requested or recommendation:\* | Agree for inclusion in TR-0068 |
| Template Version: January 2017 (Do not modify) | |

**oneM2M Notice**

The document to which this cover statement is attached is submitted to oneM2M. Participation in, or attendance at, any activity of oneM2M, constitutes acceptance of and agreement to be bound by terms of the Working Procedures and the Partnership Agreement, including the Intellectual Property Rights (IPR) Principles Governing oneM2M Work found in Annex 1 of the Partnership Agreement.

# Introduction

This contribution introduces a key issue on data labling for AI/ML application.

Comments on R02

* Having a single datalabel for multiple images.. (add another potential solution)
* Think of optimization (maybe add an editors note to remember)
* Add more example to type (need support from Andrew)

### -----------------------Start of change 1-------------------------------------------

## 8.x Key Issue & Possible Concept x – Data labelling

### 8.x.1 Key Issue

Many Artificial Intelligence (AI) and Machine Learning (ML) applications use data collected in IoT platforms to train their model. IoT platform (including oneM2M) is a place holder to collect and manage various data (e.g., image, text, and sensory).

Data labelling is an essential step in a supervised machine learning task. Data labelling is the process of identifying raw data (e.g., images, text files, videos) and adding one or more meaningful and informative labels to provide a context for data. Data labelling is a task that requires much manual work. If an IoT platform provides a means to support providing data labelling, developers can save time and resources. For example, labels on data indicate whether data contains the temperature of a room or if an x-ray contains a tumour.

At the moment, there is no single standard format to annotate raw data. Below are several available annotation formats used by AI/ML developers:

* COCO: COCO has five annotation types: object detection, keypoint detection, stuff segmentation, panoptic segmentation, and image captioning. The annotations are stored using JSON.
* Pascal VOC: Pascal VOC stores annotation in an XML file.
* YOLO: In YOLO labelling format, a .txt file with the same name is created for each image file in the same directory. Each .txt file contains the annotations for the corresponding image file (i.e., object class, object coordinates, height and width).

AI/ML developers can define a customized labelling format depending on which data needs annotation.

Here are several annotation types used in AI/ML:

* Bounding boxes: use rectangular boxes to define the target object's location.
* Polygonal segmentation: complex polygons are used to define the shape and location of the object in a much more precise way.
* Semantic segmentation: a pixel-wise annotation technique, where every pixel in the image is assigned to a class
* 3D cuboids: similar to bounding boxes but with additional depth information about the object
* Lines and splines: annotation is created by using lines and splines
* Key-point and landmark: detect small objects and shape variations by creating dots across the image
* Customization: various annotations can be used depending on the data type. For example, if the network packet is data, header field information (base address, offset, field description) can be used for labelling.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| Bounding boxes | Polygonal | Semantic segmentation | 3D cuboids | Lines and spines |

oneM2M currently provides semantics and ontology-related functions. These functions have the potential to be used for artificial intelligence and machine learning but are not directly utilized for training data. Suppose the oneM2M platform provides a new function of data labelling used for AI/ML using the semantics and ontology functions. In that case, the reuse of the learning data becomes possible, and AI application developers can create more accurate models.



Figure 8.x-1: An example structure of [*dataLabel*] resource

### 8.2.2 Possible Solution

*Editor’s Note: Optimization of the possible solution is FFS. For example, multiple container resources can be referred to by a single <dataLabel> resource. Also, it is FFS how to use semantics and ontologies in data labelling.*

Suppose oneM2M systems support functions that allow developers to annotate labelling information to training datasets using ontologies. In that case, an AI/ML data labelling tool can create and manage resource(s) to annotate labelling information to training data (set).

This can be done by introducing a new resource called the <dataLabel> to hold the information required to perform data labelling. The <dataLabel> resource can have the following attributes (see Figure 8.x-1):

* trainingData: confirms that this data is for training
* labelingType: describes labelling type, e.g. square, polygon, line.
* labelAnnotationFormat : there exist many labelling format such as COCO, YOLO, Pascal VOC, custom
* labelAnnotationContents: contains actual annotation contents following the given format (e.g., XML or JSON)
* refOntology: reference ontology used in data label annotation

For this purpose, we can assume that an oneM2M platform holds data set for AI/ML training to build a model (see Figure 8.x-2). First, a labelling annotation tool (oneM2M application) discover resources for training. Then the labelling tool creates a resource(s) for data labelling. The labelling application requests to update data labelling resources. Another AI/ML application then uses the training data set with label information and builds a model for AI/ML service.



Figure 8.x-2: An example procedure for data labelling

### -----------------------End of change 1-------------------------------------------