**3GPP TSG-SA WG4 Meeting #132 *S4-250973\_Rev3***

**Fukuoka, Japan, 19 - 23 May 2025**

**Source: Interdigital Finland Oy, Nokia**

**Title: [FS\_AI4Media] pCR on definitions to TR 26.927**

**Agenda item:** **9.6**

**Document for: Agreement**

**1. Introduction**

This contribution provides changes to TR 26.927 aiming at better readability and consistency between different clauses as some definitions are either spread in different sections of the document or missing.

* AI/ML data is defined in “5.3.1 AIML data components” but should defined in 3.1. This includes AI/ML model data, intermediate data, inference input data, inference output data, output media data, training input data, training results data, metadata.
* There are different definitions for the same purpose: AI/ML Task, AI/ML-based media processing, and AI/ML media processing.
* Both AI/ML model subset, AI/ML sub-model definition are used and mentioned in the document
* Missing definitions to AI/ML operations.

**2. Reason for Change**

Following changes in clause 3.1, clause 5.3.1 and clause 6.3.3 are proposed:

* move and refine definition of components of AI/ML data above from 5.3.1 to clause 3.1,
* combine AI/ML model subset and AI/ML sub-model in the same definition.
* define and keep a common definition for AI/ML Task and AI/ML media processing and further replace AI/ ML-based media processing.
* Align clause 6.3.3 with the definition in clause 3.1 regarding single and multi-branch operations
* add other missing definitions related to AI/ML operations.

**3. Proposal**

It is proposed to discuss and agree and document the following changes to 3GPP TR 26.927

\* \* \* first Changes \* \* \* \*

# 3 Definitions of terms, symbols and abbreviations

## 3.1 Terms

For the purposes of the present document, the terms given in 3GPP TR 21.905 [1] and the following apply. A term defined in the present document takes precedence over the definition of the same term, if any, in 3GPP TR 21.905 [1].

**Autoencoder:** Type of artificial neural network used for unsupervised learning, primarily for dimensionality reduction, feature learning, and data reconstruction.

**AI/ML model:** A mathematical representation that has been trained to recognize patterns or make decisions based on data.

**AI/ML model data**: Internal components of a trained neural network, including its architecture, weights, biases, and parameters that together define how the inputs are processed to generate the outputs of the AI/ML model.

**Model inference:** Process by which a deployed machine learning model generates a result [5].

**AI/ML task, AI/ML media processing**: A software process that executes an ML algorithm to accomplish a specific function.

**Inference engine:** Functionality that provides runtime environment for a machine learning model and exposes corresponding machine learning model inference capability [5].

**AI/ML model subset, AI/ML sub-model:** A discrete component of a larger AI/ML model that can perform inference independently.

**AI/ML model composition:** The composition of an AI/ML model into one or more AI/ML model subsets.

**AI/ML model split points:** The points in an AI/ML model where the model can be split into multiple distinct AI/ML model subsets.

**AI/ML inference endpoint:** A UE or Network inference engine that executes an AI/ML model or a model subset

**Split operation:** Operation which consists in splitting an AI/ML model into AI/ML model subsets, a head subset that takes the same input as the full AI/ML model, and a tail subset that takes the same output as the full AI/ML model.

**Split AI/ML model:** An AI/ML model composed of subsets) (e.g. head and tail) that can be distributed across different endpoints.

**Split AI/ML model inference:** Distributed inference performed on subsets of an AI/ML model across different endpoints.

**Tensor**: A multi-dimensional array of numerical data used in AI/ML, defined by its shape (number and size of dimensions) and data type (e.g., float32, int64).

**Input media data:** Raw or preprocessed media data captured from a data source (e.g. camera, microphones), including images, video, text, audio and speech that serves as the input for AI/ML models or AI/ML model subsets.

**Output media data:** Inference output data that has been post-processed to meet the specific requirements of the targeted AI/ML model and application expectations.

**Inference input data:** Input media data pre-processed to meet the specific requirements of a target AI/ML model, used as input by the AI/ML model or its head subset to perform inference.

**Inference output data:** The results produced by the AI/ML model or its tail subset after performing inference.

**Intermediate data:** Output from the inference process of a head AI/ML model subset, which serves as input to the tail AI/ML model subset and is not the final inference result of the full AI/ML model.

**Training input data**: Data feeding the AI/ML training process on a device e.g. for federated learning. Such data is typically created on or exists in UEs**.**

**Training results data:** Output result of the AI/ML training process. e.g. such data is typically delivered by a UE device to a federated learning entity which aggregates data from multiple UE devices to update and train a model.

**Model update:** A partial or full update of a trained model which may include changes to its internal structure and/or related parameters (e.g. weight, biases).

**AI/ML data**: Any of AI/ML model data, intermediate data, inference input data, inference output data, output media data, training input data, training results data, metadata.

**Split point configuration:** Settings comprising the necessary interoperable information to configure an endpoint for a particular split point.

\* \* \* End of first change \* \* \* \*

\* \* \* second Changes \* \* \* \*

### 5.3.1 AI/ML data components

AI/ML user plane data is defined in clause 3.1 as AI/ML data

- AI/ML model data,

- Intermediate data,

- Inference output data,

- Inference input data,

- Training input data,

- Training results data,

- User-plane metadata that corresponds to contextual and additional information to the data payload being transmitted.

\* \* \* End of second changes \* \* \* \*

\* \* \* Third Changes \* \* \* \*

### 6.3.3 Operations for splitting AI/ML models

For split configurations, clause 5.1, an AI/ML model is split in two subsets consisting of a head subset or a part 1 running on a first endpoint (UE resp. Network) and a tail subset or part 2 running on a second endpoint (Network resp. UE). The first endpoint provides intermediate data to the second endpoint over the network.

The computational graph of a model above may be represented by a directed acyclic graph with existing formats or frameworks such as ONNX, NNEF, TensorFlow and PyTorch as described in clause 6.2.5 and 6.4.

.

A split operation on a particular node in such a graph is called (see clause 3.1):

* A single branch split, when the output of the head subset is composed of only one tensor. Intermediate data is then composed of one tensor.
* A multi-branch split when the output of the head subset is composed of several tensors. Intermediate data are then composed of several tensors.

Certain specific aspects may need to be considered when applying a split operation to divide a model into two or more subsets:

- A model graph may start taking input media data into account at a node later than the first node. In this case, if the split occurs before the node requiring the input media data, this input media data needs to be provided to the tail subset running in the second endpoint, in addition to the intermediate data generated by the head subset.

- A model graph may comprise multiple nodes producing partial results. In this case, a split occurring after a node producing partial results will require to collect partial results from both endpoints to compute the final consolidated results.

ONNX is an interoperable format available for frameworks 6.4 and for a large number of models (https://github.com/onnx/models). Splitting a single branch is straightforward using extract\_model function (https://onnx.ai/onnx/api/utils.html). A generic multi-branch split using the same function may also be performed after parsing the model to obtain the list of inputs and outputs required by each model subset.

Different endpoints may split an ONNX model down to the split node or from the split node to the end by using the same node identification and parsing rules to apply to the graph. For example, a first endpoint may build the head subset from the full ONNX model and the second endpoint may build the tail subset from the same full ONNX model.

One endpoint may split model from an ONNX model file in two subsets and deliver the required subset to an endpoint

For dynamic split point reselection, an endpoint may prepare different model subsets for the different candidate split point configurations or build the split model subset ‘on demand’ from a full ONNX model when it is required.

\* \* \* End of third changes \* \* \* \*