**3GPP TSG-SA WG4 Meeting #127S4-240210r1**

**Sophia-Antipolis, France, 29 January - 2 February 2024**

**Source: Samsung Electronics Co., Ltd., InterDigital Finland Oy, HUAWEI TECHNOLOGIES Co. Ltd.**

**Title: [FS\_AI4Media] pCR on Metadata for model delivery and split inferencing**

**Spec: 3GPP TS 26.927 v0.5.0**

**Agenda item: 9.6**

**Document for: Agreement**

**1. Introduction**

In TR 26.927 v0.5.0 and the related PD we have identified certain procedures related to services for complete/basic AI model delivery, split AI/ML operation and distributed/federated learning.

The metadata clause in TR 26.927 v.0.5.0 is missing aspects related AI model delivery and split AI/ML operation, we suggest relevant text in this contribution. A placeholder table for distributed/federated learning is also provided for merging the current text.

**2. Reason for Change**

There is a missing section for metadata related to AI model delivery and split inferencing in the TR.

**3. Proposal**

We propose the changes into TR 26.927 v0.5.0.

\* \* \* First Change \* \* \* \*

## 6.5 Metadata

### 6.5.1 Introduction

Metadata for AI media services may include information describing AI models, inference requirements, endpoint capabilities (UE or network) and information more specific to the configuration, control and management of the basic AI service scenarios (AI model delivery, split AI/ML operation and distributed/federated learning).

NOTE: The delivery of the metadata described in this clause is not specified.

### 6.5.2 Common AI model information

AI model information metadata is used to describe the characteristics of AI models which may be used for an AI media service. This information may be common to all three AI service scenarios, and may be used in the selection of a suitable AI model by the UE or network, given an AI media service.

|  |  |  |  |
| --- | --- | --- | --- |
| **Metadata category** | **Metadata type** | **Definition** | **Metadata type description (Examples)** |
| **Model information** | **Model identifier** | An identifier for an AI model (or variants of it) specified for a certain AI media service. The identifier may be a name, a number, a combination thereof, a hash value. The identifier is defined during the configuration stage. | model\_1, model\_2 |
| **Number of parameters** | Total number of parameters in the neural network. | 11 million |
| **Model size** | The size of the AI model file in megabytes. | 40MB |
| **Input size** | The maximum size of the input data supported by the AI model in kilobytes. | 256 KB |
| **Output size** | The maximum size of the output data supported by the AI model in kilobytes. | 256 KB |
| **Accuracy** | The trained accuracy of the AI model as a percentage. | 85% |
| **Target inference latency** | The target inference latency specified for a given AI model in milliseconds. Such latency is measured between the input and output layers of the AI model at inference. This value is related to the service inference latency requirement of the service for which the AI model is provided, as well as the typical hardware capabilities of an entity performing the inference of the model. | 20ms |
| **Format/ framework** | The format or framework used to express the AI model, including its version number. | Pytorch 2.0 ONNX 1.15.0 |
| **Processing capabilities** | Estimated capabilities for processing the model including the computational power such as the computational cost (in FLOPS), the computational complexity (in MAC operations). It also includes the temporary memory to store model parameters. | NPU 10TFLOPS, MEM 10GB |

### 6.5.3 AI model information for split AI/ML operations

AI model information metadata for split AI/ML operations is used to describe the characteristics of AI models for split inference service scenarios. This information may be used in the selection of a split point (from which a multiple may be predefined by the service provider for a certain AI media service). A trained model can be represented as a directed acyclic graph model represented by a collection of nodes interconnected with edges (e.g. ONNX). A split point may happened before or after a graph node identified by its name or a number.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metadata category** | **Metadata type** | | | **Definition** | **Metadata type description (Examples)** | |
| **Split model information** | **Split points** | | | The number of predefined split points at which a certain model can be divided into two for split inferencing. | 2 | |
| **Split point information** | **Split point identifier** | | | An identifier of the split point in a description of a computing graph, may be generated by a neural network description language such as ONNX/NNEF. Identifiers must guarantee unique identification of a specific split point. | Nb:10, 75 Name: Layer\_10, | |
| **Split point intermediate data size** | | | The size of the intermediate data resulting from the give split point, in kilobytes. Intermediate data size is typically dependent on the tensor size at the given split point. | 1086KB | |
|  | | **Split point number** | The number of the split point where the split occurs. The number may belong to set of identified numbers defined at the configuration stage. | | | 10 |
| **Split point name** | The name of the split point where the split occurs. The name may belong to set of identified split point names defined at the configuration stage. | | | conv2d\_1234 |
| **Split point flag** | An information on whether to consider the split point before the split point identifier or after. The convention on whether it is before or after may be defined at the configuration stage. | | | before, after |

### 6.5.3 Intermediate data information for split AI/ML operations

Intermediate data information identifies the structure of intermediate data output from a first endpoint that need to be retrieved to feed the inference of the second endpoint after transmission of the intermediate data over the network.

|  |  |  |  |
| --- | --- | --- | --- |
| **Metadata category** | **Metadata type** | **Definition** | **Metadata type description (Examples)** |
| **Intermediate data information** | Tensor structure information | The exact underlying tensor structure of the intermediate data tensors including the exact version of it. | PyTorch 2.0,  Tensor flow v2.13.0, NumPy v1 .25 |
| Tensor shape | The tensor shape(s) when the output is intermediate data. Tensor shape is a tuple of positive integers, where the size of the tuple represents the dimension of the tensor, and each value represents the size in each dimension. | [1,64,64,64]. |
| Tensor element data type | The data type of each output intermediate data tensor | :int64, Float32 |
| Data direction | This defines the direction of transmitted data, either uplink (from UE endpoint to network endpoint) or downlink (From a network endpoint to the UE endpoint). This information may be useful to configure an intermediate data delivery session | Upstream, Downstream |
| Compression algorithm | Identifies the compression algorithm(s) that can be applied to the intermediate data. When the connectivity condition between the UE and the network is insufficient to transmit the original intermediate data, a compression algorithm may be applied. | NONE, FC\_VCM, SNAPPY, … |

### 6.5.4 Service requirement information

Service requirement information metadata is used to describe the latency and processing requirements for the AI media service. Such information may be used in the selection of an AI model for the service, and/or the selection of a split point for a certain AI model for split inferencing.

|  |  |  |  |
| --- | --- | --- | --- |
| **Metadata category** | **Metadata type** | **Definition** | **Metadata type description (Examples)** |
| **Service requirement information** | **Maximum service inference latency** | The maximum inference latency requirement specified for a given AI media service, in milliseconds. In the case of split inferencing, this requirement includes the delivery latency of the intermediate data between the first and second split inference entities. | 100ms |
| **Minimum service inference accuracy** | The minimum accuracy specified for a given AI media service. | 80% |
| **Service type identifier** | An identifier for the service type to be supported by the AI/ML model, such as ASR (Automatic Speech Recognition), TTS (Text To Speech), Translation (with the indication of input and output languages). | TTS, ASR, Trans-EN-to-ZH |
| **Service accuracy** | The expected service accuracy | 85% |

### 6.5.5 Endpoint capability information

The endpoint capability information includes the capabilities of the endpoint (UE or network) for processing and transmitting the AI/ML model and intermediate data. Such information can be updated due to the change of the endpoint’s work load or the network conditions. It can be used for the selection of AI model, split inference, intermediate data compression, progressive model delivery.

|  |  |  |  |
| --- | --- | --- | --- |
| **Metadata category** | **Metadata type** | **Definition** | **Metadata type description (Examples)** |
| **Endpoint capability information** | **Processing capabilities** | The available resources for processing AI/ML model including the computational power (in FLOPS), the memory to store model parameters and perform the inference. | NPU 10TFLOPS, MEM 10GB |
| **Supported AI Framework** | The AI framework(s) supported by the endpoint. | TensorFlow 2.0 |
| **Supported compression algorithms** | The supported compression algorithm(s) for intermediate data compression. | NONE, FC\_VCM, SNAPPY, … |
| **Connection capabilities** | This indicates the available bandwidth in bit/s between the UE and the network for transmitting the AI model and/or the intermediate data. | 256 kb/s |

### 6.5.6 Distributed/Federated learning information

Editor's note: Placeholder for merging current text on distributed/federated learning metadata.

|  |  |  |
| --- | --- | --- |
| **Metadata category** | **Metadata type** | **Definition** |
| **Control information** |  |  |
| **Synchronization information** |  |  |
| **Device eligibility information** |  |  |
| **Model evaluation information** |  |  |
| **Model update information** |  |  |
| **Failure reporting information** |  |  |

#### 6.5.5.1 Control information

##### 6.5.5.1.2 General

This clause describes a set of possible control information for managing the training process, synchronization the training rounds, and defining the selection criteria for participating devices, or monitoring the convergence of the training process, in federated learning.

#### 6.5.5.2 Synchronization information

##### 6.5.5.2.1 Definition

Synchronization information may be used to ensure that all devices start the training process simultaneously and progress at the same pace. For example, the server may send a synchronization information to all UEs to start a new round of training.

##### 6.5.5.2.2 Behavior

The network application sends synchronization information to all UE applications to start a new round of training at the same time as described in step 1 of figure 5.2.4-2. The information contains the round number and may also contain a timestamp indicating when the training round should begin.

##### 6.5.5.2.3 Parameters

The possible parameters are:

- The Round\_number indicates the training round in a model training.

- The Start\_time indicates the start time of the training.

- The Duration indicates the desirable duration of the training. This value just shows an indication of the desirable time for completing the training round.

#### 6.5.5.3 Device eligibility information

##### 6.5.5.3.1 Definition

Device eligibility information may be used to define the criteria for selecting the devices that will participate in the training process. For example, the server may send a device eligibility information to all devices that belong to the defined group by the application.

##### 6.5.5.3.2 Behavior

The Federated learning engine sends a device eligibility information to the AI model training engine to select the devices that meet certain criteria defined by the application as described in step 4 of figure 5.2.4-2. Depending on the number of criteria met, the application assigns a group id to the device. For example, the criteria could contain information about the device's operating system, processor speed, available memory, available image library (number of images…), geographical location of the device, language setting, and other attributes.

##### 6.5.5.3.3 Parameters

The possible parameters are:

- The Group\_id is used to assign a new id for the devices that meet the eligibility criteria of this information. If the device is eligible, it uses this value as one of its group ids and from now on, it reacts to information with the same group id.

- The Application\_group\_id, is assigned by the application on the device and if that value is equal to the value of this field, then the device is eligible.

- The Hardware, Location, and Language parameters define the hardware, location, and language eligibility criteria respectively for the device.

- The Data\_library\_id defines the data library an eligible device shall have.

Note: if more than one eligibility field exists, the device needs to meet all criteria to become eligible.

#### 6.5.5.4 Model evaluation information

##### 6.5.5.4.1 Definition

Model evaluation information may be used to evaluate the performance of the global model for each device and make decisions about the training process. After running the learning phase, a device sends a model evaluation information to the server that measures the accuracy of the model. The server can then decide whether to continue training for another round or stop.

Alternatively, this information may be used by the server to request the device to perform an evaluation of a newly downloaded global model.

##### 6.5.5.4.2 Behavior

For Federated learning engine sends the model evaluation information to the AI model training engine in the UE containing the metrics to be used for evaluation such as accuracy or precision as described in step 7 of figure 5.2.4-2.

##### 6.5.5.4.3 Parameters

The possible parameters are:

- The Round\_number shows the round after which the evaluation is performed.

- The Metric\_number shows the number of metrics included in this information body.

- The Metric is one or more of the Name-Value pairs showing the name of the metric and the corresponding value obtained in the evaluation.

#### 6.5.5.5 Model update information

##### 6.5.5.5.1 Definition

Model update information may be used to update the model parameters on the devices after each round of training. For example, the server may send a model update information to all devices to update the global model with the new model parameters.

Model update information may also be used to update the global model on the server with the new parameters updated by the local training on the device.

##### 6.5.5.5.2 Behavior

The server may send a model update information to all devices to update the AI/ML model with the new model parameters as described in step 5 of figure 5.2.4-2. The information contains the model id of the AI/ML model to be updated, the updated model parameters that the UE will use to train the model in the next round, and the new model id when the parameters are updated.

After running the training locally, each AI Model training Engine in the UEs may send a model update information to the server with the updated parameters as described in step 15 of figure 5.2.4-2. Together with the received model evaluation information, the server can decide if the global model needs to be updated or not. The model update information then only contains the model id of the AI/ML model used for local training and the updated parameters.

##### 6.5.5.5.3 Parameters

The possible parameters are:

- The Parameters includes the new model vector of values.

- The New\_model\_id is the id of the new model when the server sends the model to one or more devices.

#### 6.5.5.6 Failure reporting information

##### 6.5.5.6.1 Definition

Error information may be used to handle unexpected errors or exceptions that may occur during the training process. For example, the server may send an error information to all devices to handle a device failure or network disruption.

##### 6.5.5.6.2 Behavior

The server sends a request to all devices to report a device failure or network disruption as described in step 6 of figure 5.2.4-2. For example, if a device fails to send its model parameters back to the server, the device should notify the server so that the device has been removed from the training process.

The AI Model training engine in the UE sends a failure information to the Federated learning engine in the server if a failure occurs as described in step 15 of figure 5.2.4-2.

##### 6.5.5.6.3 Parameters

The information describes the reason for the failure.

\* \* \* End of Changes \* \* \* \*