**Source: China Telecomunication Corp., Nokia Corporation, Interdigital Finland Oy**

**Title: [FS\_AI4Media] Updates on AI/ML work in MPEG WGs**

**Agenda Item: 9.7**

**Document for: Agreement**

# 1 Introduction

This contribution proposes several updates to AI/ML work in MPEG WGs (section 2.2) and Examples of split point references (section 8.2.1) in the latest version of the permanent document (v0.7).

# 2 Changes

## 2.2 AI/ML work in MPEG WGs

MPEG currently has two working groups studying coding technologies optimized for machine vision tasks: Feature Compression for Video Coding for Machines (FC-VCM) and Video Coding for Machines (VCM). In the following the source content is referred to as video but the system can also be used with still images. The scope of these two groups differs in the inputs/outputs: the inputs to the encoder of VCM are videos or images, while the inputs and outputs to the FC-VCM codec are features extracted from the images or videos, which corresponds to the split-inference pipeline considered in this document.

VCM has issued a Call for Proposals (CfP) in Apr. 2022, and is currently performing Core Experiments (CE) to decide what should be included in the reference software. FC-VCM, on the other hand, is in a relatively earlier stage. It issued a CfP in April. 2023, the responses will be evaluated in October 2023.

### 2.2.1 MPEG Feature Compression for Video Coding for Machines (FC-VCM)

In the MPEG Requirements Working Group which explores new market needs, an ad-hoc group has been created to study the optimization of the Compression of Features in the context of Video Coding for Machine tasks (FC-VCM).

Intermediate data can consist of large tensors of floating-point values, which would require very large bitstream over 5G to enable split inference between the network and the UE. Therefore, compression may be required in this scenario. The FC-VCM encoder and the FC-VCM decoder would then be part of the intermediate delivery function and intermediate access function, respectively.

Figure 2.2.1-1 illustrates the considered pipeline where, like in the current study, a first part of the Neural-Network-based algorithm is split into two parts. The intermediate features are first encoded on the sender side and embedded in a bitstream, which is decoded at the receiver before inferring the second part of the Neural Network.



Figure 2.2.1-1: FC-VCM pipeline

This standard, which targets use-cases matching the proposed Intermediate data transfer, is expected to be finalized by the end of 2025.

The current baseline considers the use of traditional video compression methods, e.g., the latest H.266/(Versatile Video Coding (VVC) standard, to encode the features that are processed and packed into input frames to the codec. The activity has just started, and new methods are going to be proposed. As the AI models considered in this study rely on Neural Networks, it can be envisioned to optimize the compression of the intermediate features using trained auto-encoders as well, to minimize the size of the bitstreams to be transmitted over 5G, while conserving an acceptable accuracy of the inferred models.

2.2.2 MPEG Video Coding for Machines (VCM)

In the MPEG Video Working Group which explores video coding technologies, an ad-hoc group has been created to study the optimization of the Video Coding for Machine tasks (VCM).

Traditional coding methods aim for the best video reconstruction under certain bit-rate constraints for human consumption. However, with the rise of machine learning applications, along with the abundance of sensors, many intelligent platforms have been implemented with massive data requirements including scenarios such as connected vehicles, video surveillance, and smart city.

The sheer quantity of data being produced constantly leads previous methods with a human in the pipeline to be inefficient, and unrealistic in terms of latency and scale. There are additional concerns in transmission and archive systems which require a more compact data representation and low latency solution.

Figure 2.2.1-1 illustrates the considered pipeline where, like in the current study, videos are embedded in a bitstream, which is decoded to either a reconstructed video or a representation of the input video before inferring the task neural network.

Figure 2.2.2-1: VCM pipeline

#### 8.2.1.1 Feature Maps used in MPEG FC-VCM (Feature Compression for Video Coding for Machines)

The pipeline that is considered for feature compression for video coding for machines is described in Figure 8.2.1.1-1.



Figure 8.2.1.1-1: FC-VCM pipeline

The video or image is first analyzed to extract the feature maps, which will be compressed by FC-VCM. For the standardization process, a so-called anchor model has been defined, to which proponents will compare to evaluate the responses of the Call for Proposal and the upcoming reference software. It corresponds to the implementation of the pipeline of Figure 8.2.1.1-1 using exiting tools and standards. The state-of-the-art H.266/MPEG VVC codec is then used to compress the feature maps. Some pre-processing maybe needed, for example to pack all channels of the feature map into a single atlas map and 10-bit quantization to fit VVC input format. FC-VCM proposals will be measured against this basic approach.

Task networks and corresponding split points that are currently used in the Common Test Conditions for FC\_VCM are:

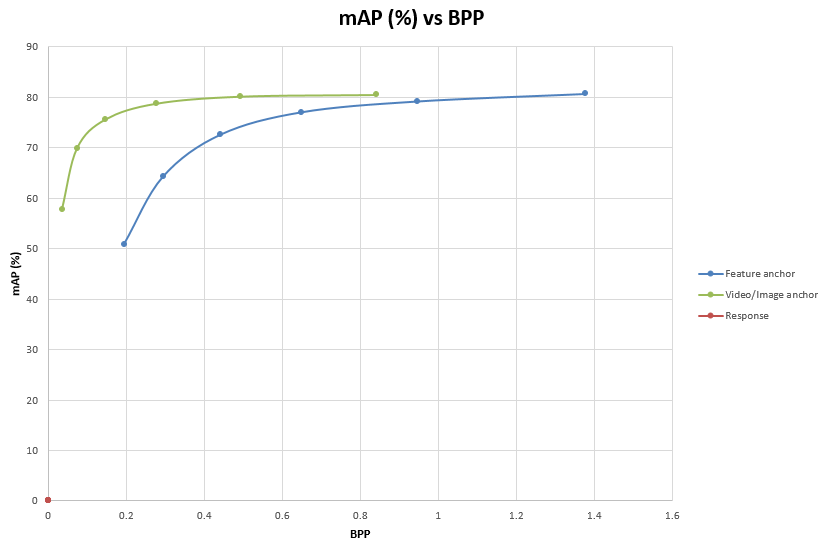
* Mask R-CNN p-layer split point for object segmentation,
* Faster R-CNN p-layer split point for object detection, and
* JDE-1088x608 Darknet-53 split point for object tracking.

Mask R-CNN and Faster R-CNN implementations are part of the detectron2 framework that can be found at <https://github.com/facebookresearch/detectron2>. JDE (Joint Detection and Embedding) is a multiple object tracker that can be accessed at https://github.com/Zhongdao/Towards-Realtime-MOT.

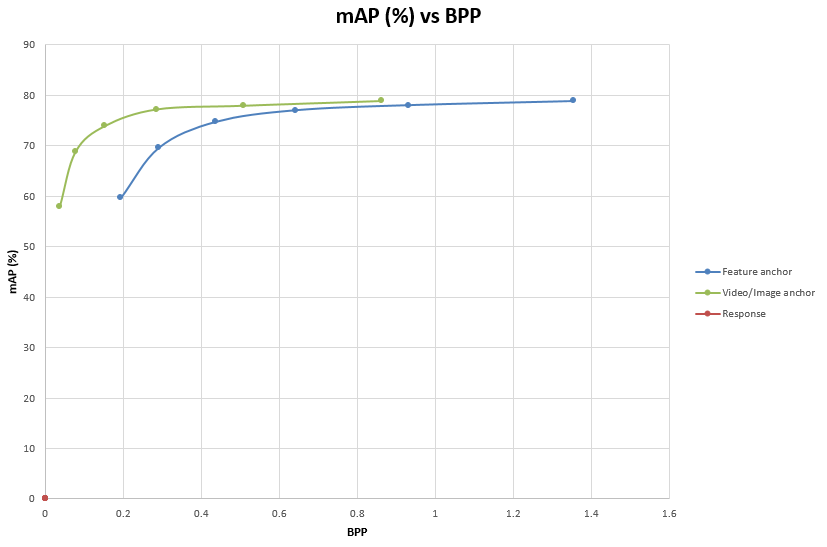
Anchor results for each task and split point are described by the following graphs. For both object detection and instance segmentation, mean Average Precision (mAP) is used to measure the performance of the network, while MOTA(Multi-Object Tracking Accuracy) (https://pub.towardsai.net/multi-object-tracking-metrics-1e602f364c0c) is used for object tracking. The performance of the corresponding split-inference is then measured as a model accuracy using one of the above metrics, as a function of the bitrate of the bitstream containing the compressed intermediate data. In the following figures, the bitrate is measured using Bits Per Pixel (BPP), i.e., the total number of bits of divided by the number of pixels in the source images/videos or kilo bits per second (Kbps) in the context of videos.

Two curves are shown in the graphs, which correspond to the “feature anchor”, i.e., the compression of intermediate data using VVC as described above, and the “Video/Image anchor” which correspond to the encoding of input videos/images using VVC and running the entire task network on decoded images/videos as performed in MPEG VCM test configurations. The red curve is empty for now as this template will be used by the proponents to the call for proposals to compare their response with the anchors.

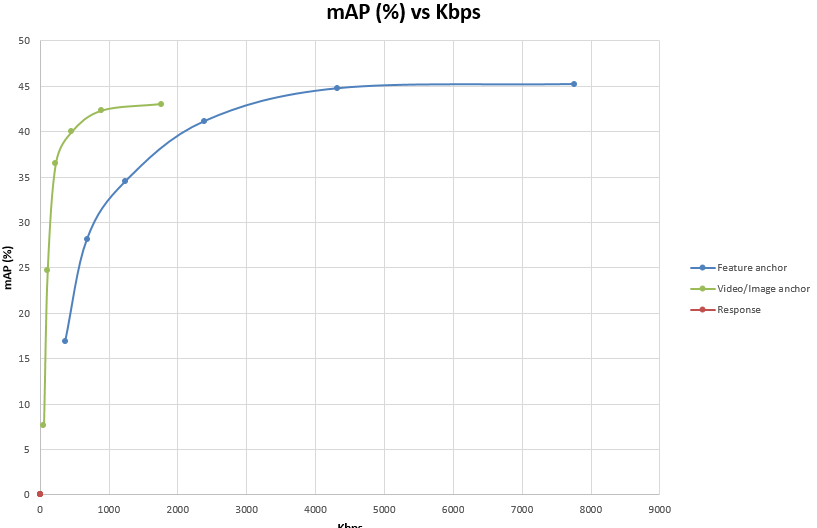
As the extracted features at the proposed split points are larger than input images/videos and VVC is optimized to compress pixel content, i.e., content with strong spatial correlation and predictable motion whereas feature maps can be more erratic noisy tensors, one can note that the naïve approach used for feature anchors currently underperforms the compression of input images/videos. New feature compression technologies will be proposed to the CfP and analyzed at the October 2023 meeting.



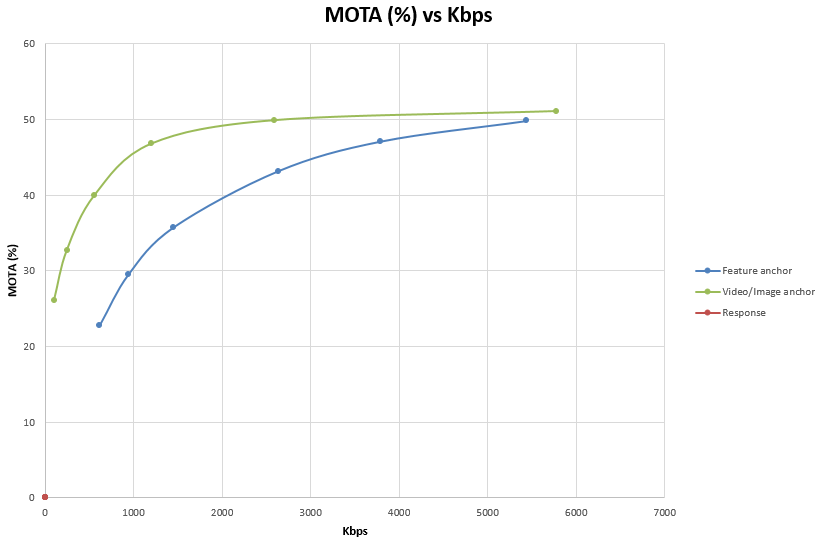
8.2.1.1 – Figure 1: Instance Segmentation on OpenImages dataset using Mask R-CNN P-layer split point



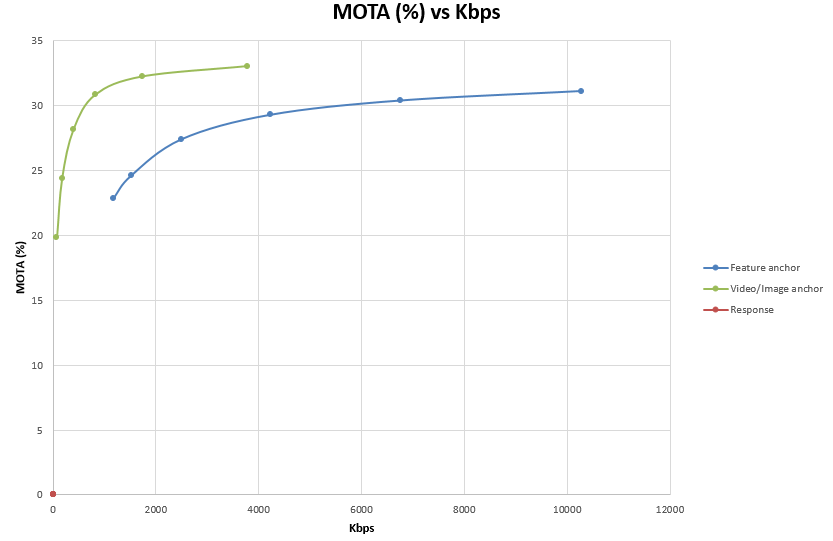
8.2.1.1 – Figure 2: Object Detection on OpenImages Dataset using Faster R-CNN P-layer split point



8.2.1.1 – Figure 3: Object Detection on the SFU Dataset (http://multimedia.fas.sfu.ca/data/) using Faster R-CNN P-layer split point (Detectron 2)



8.2.1.1 – Figure 4: Object Tracking on the Tencent Video Dataset TVD (https://multimedia.tencent.com/resources/tvd) using JDE Darknet-53 split point



8.2.1.1 – Figure 5: Object Tracking on Hieve (http://humaninevents.org/) Dataset using JDE Darknet-53 split point



# 3 Proposal

We propose to include the updates in section 2 and 8 of this contribution into the next version of the permanent document.