Workgroup: Internet Research Task Force
Internet-Draft: draft-hong-nmrg-ai-deploy-03
Published: 13 March 2023
Intended Status: Informational
Expires: 14 September 2023
Authors: Y-G. Hong S-B. Oh J-S. Youn
Daejeon University KSA DONG-EUI Univ
S-J. Lee S-W. Hong H-S. Yoon
Korea University/KT ETRI ETRI

Considerations of deploying AI services in a distributed approach

Abstract

As the development of AI technology matured and AI technology began to be applied in various fields, AI technology is changed from running only on very high-performance servers with small hardware, including microcontrollers, low-performance CPUs and AI chipsets. In this document, we consider how to configure the network and the system in terms of AI inference service to provide AI service in a distributed approach. Also, we describe the points to be considered in the environment where a client connects to a cloud server and an edge device and requests an AI service.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 14 September 2023.

Table of Contents

1. Introduction

In the Internet of Things (IoT), the amount of data generated from IoT devices has exploded along with the number of IoT devices due to industrial digitization and the development and dissemination of new devices. Various methods are being tried to effectively process the explosively increasing IoT devices and data of IoT devices. One of them is to provide IoT services in a place located close to IoT devices and users, away from cloud computing that transmits all data generated from IoT devices to a cloud server [I-D.irtf-t2trg-iot-edge].

IoT services also started to break away from the traditional method of analyzing IoT data collected so far in the cloud and delivering the analyzed results back to IoT objects or devices. In other words, AIoT (Artificial Intelligence of Things) technology, a combination of IoT technology and artificial intelligence (AI) technology, started to be discussed at international standardization organizations such as ITU-T. AIoT technology, discussed by the ITU-T

CG-AIoT group, is defined as a technology that combines AI technology and IoT infrastructure to achieve more efficient IoT operations, improve human-machine interaction, and improve data management and analysis [CG-AIoT].

The first work started by the IETF to apply IoT technology to the Internet was to research a lightweight protocol stack instead of the existing TCP/IP protocol stack so that various types of IoT devices, not traditional Internet terminals, could access the Internet [RFC6574][RFC7452]. These technologies have been developed by 6LoWPAN working group, 6lo working group, 6tisch working group, core working group, t2trg group, etc. As the development of AI technology matured and AI technology began to be applied in various fields, just as IoT technology was mounted on resource-constrained devices and connected to the Internet, AI technology is also changed from running only on very high-performance servers with the old GPU installed. The technology is being developed to run on small hardware, including microcontrollers, low-performance CPUs and AI chipsets. This technology development direction is called On-device AI or TinyML[tinyML].

In this document, we consider how to configure the network and system in terms of AI inference service to provide AI service in the IoT environment. In the IoT environment, the technology of collecting sensing data from various sensors and delivering it to the cloud has already been studied by many standardization organizations including the IETF and many standards have been developed. Now, after creating an AI model to provide AI services based on the collected data, how to configure this AI model as a system has become the main research goal. Until now, it has been common to develop AI services that collect data and perform inferences from the trained servers, but in terms of the spread and spread of AI services, it is not appropriate to use expensive servers to provide AI services. In addition, since the server that collects and trains data mainly exists in the form of a cloud server, there are also many problems in proceeding in the form of requesting AI service by connecting a large number of terminals to these cloud servers to provide AI services. Therefore, when an AI service is requested to an edge device located at a close distance, it may have effects such as real-time service support, network traffic reduction, and important data security rather than requesting an AI service to an AI server located in a distant cloud. [I-D.irtf-t2trg-iot-edge]

Even if an edge device is used to serve AI services, it is still important to connect to an AI server in the cloud for tasks that take a lot of time or require a lot of data. Therefore, an offloading technique for properly distributing the workload between the cloud server and the edge device is also a field that is being

actively studied. In this contribution, in the following proposed network structure, the points to be considered in the environment where a client connects to a server and an edge device and requests an AI service are derived and described. That is, the following considerations and options could be derived.

*AI inference service execution entity

*Hardware specifications of the machine to perform AI inference services

*Selection of AI models to perform AI inference services

*A method of providing AI services from cloud servers or edge devices

*Communication method to transmit data to request AI inference service

2. Procedure to provide AI services

Since research on AI services has been started for a long time, there may be shapes to provide various types of AI services. However, due to the nature of AI technology, in general, a system for providing AI services consists of the following steps [AI_inference_archtecture] [Google_cloud_iot].

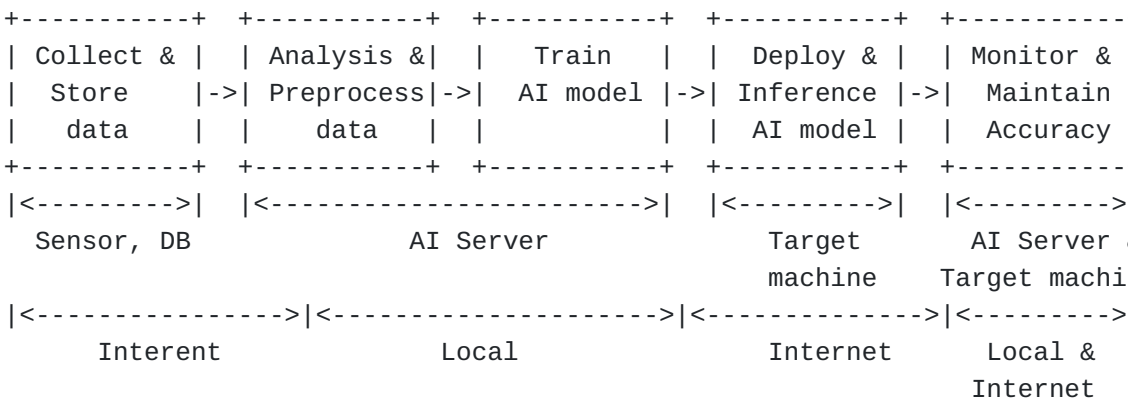

Figure 1: AI service workflow

*Data collection & Store

*Data Analysis & Preprocess

*AI Model Training

*AI Model Deploy & Inference

*Monitor & Maintain Accuracy

In the data collection step, data required for training is prepared by collecting data from sensors and IoT devices or by using data stored in a database. Equipment involved in this step includes sensors, IoT devices and servers that store them, and database servers. Since the operations performed at this step are conducted through the Internet, many IoT technologies studied by the IETF so far have developed technologies suitable for this step.

In the data analysis and pre-processing step, the features of the prepared data are analyzed and pre-processing for training is performed. Equipment involved in this step includes a high-performance server equipped with a GPU and a database server, and is mainly performed in a local network.

In the model training step, a training model is created by applying an algorithm suitable for the characteristics of the data and the problem to be solved. Equipment involved in this step includes a high-performance server equipped with a GPU, and is mainly performed on a local network.

In the model deploying and inference service provision step, the problem to be solved (e.g., classification, regression problem) is solved using AI technology. Equipment involved in this step may include a target machine, a client, a cloud, etc. that provide AI services, and since various equipment is involved in this stage, it is conducted through the Internet. This document summarizes the factors to be considered at this step.

In the accuracy monitoring step, if the performance deteriorates due to new data, a new model is created through re-training, and the AI service quality is maintained by using the newly created model. This step is the same as described in the model training, model deploying, and inference service provision steps described in the previous step because re-training and model deploying are performed again.

3. Network configuration structure to provide AI services

In general, after training the AI model, the AI model can be built on a local machine for AI model deploying and inference services to provide AI services. Alternatively, we can place AI models on cloud servers or edge devices and make AI service requests remotely. In addition, for overall service performance, some AI service requests to the cloud server and some AI service requests to edge devices can be performed through appropriate load balancing.

3.1. AI inference service on Local machine

The following figure shows a case where a client module requesting AI service on the same local machine requests AI service from an AI server module on the same machine.

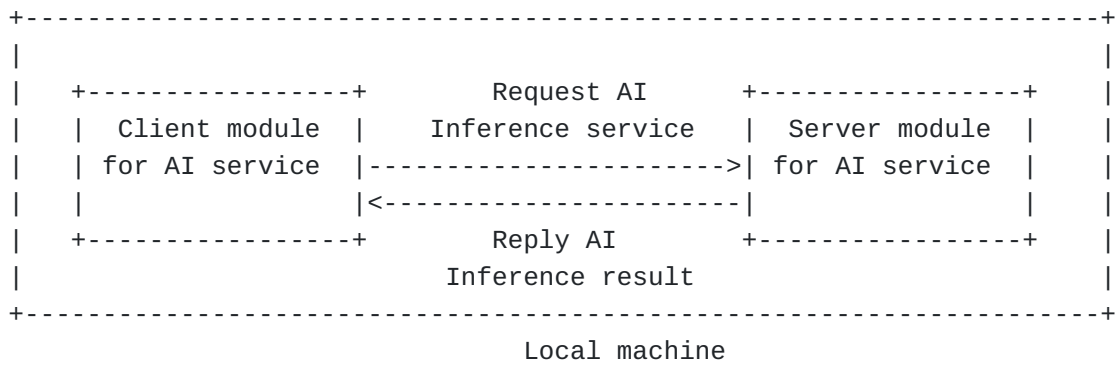

Figure 2: AI inference service on Local machine

This method is often used when configuring a system focused on training AI models to improve the inference accuracy and performance of AI models without considering AI services or AI model deploying and inference in particular. In this case, since the client module that requests the AI inference service and the AI server module that directly performs the AI inference service are on the same machine, it is not necessary to consider the communication/network environment or service provision method too much. Alternatively, this method can be used when we want to simply decorate the AI inference service on one machine without changing the AI service in the future, such as an embedded machine or a customized machine.

In this case, a high level of hardware performance is not required to train the AI model, but hardware performance sufficient to run the AI inference service is required, so it is possible on a machine with a certain amount of hardware performance.

3.2. AI inference service on Cloud server

The following figure shows the case where the client module that requests AI service and the AI server module that directly performs AI service run on different machines.

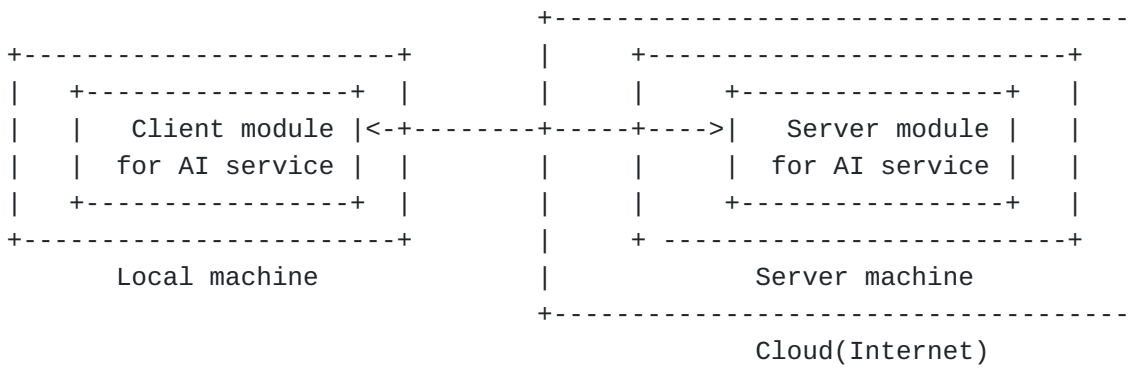

Figure 3: AI inference service on Cloud server

In this case, the client module requesting the AI inference service runs on the local machine, and the AI server module that directly performs the AI inference service runs on a separate server machine, and this server machine is in the cloud network. In this case, the performance of the local machine does not need to be high because the local machine simply needs to request the AI inference service and, if necessary, deliver only the data required for the AI service request. For the AI server module that directly performs AI inference service, we can set up our own AI server, or we can use commercial clouds such as Amazon, Microsoft, and Google.

3.3. AI inference service on Edge device

The following figure shows the case where the client module that requests AI service and the AI server module that directly performs AI service are separated, and the AI server module is located in the edge device.

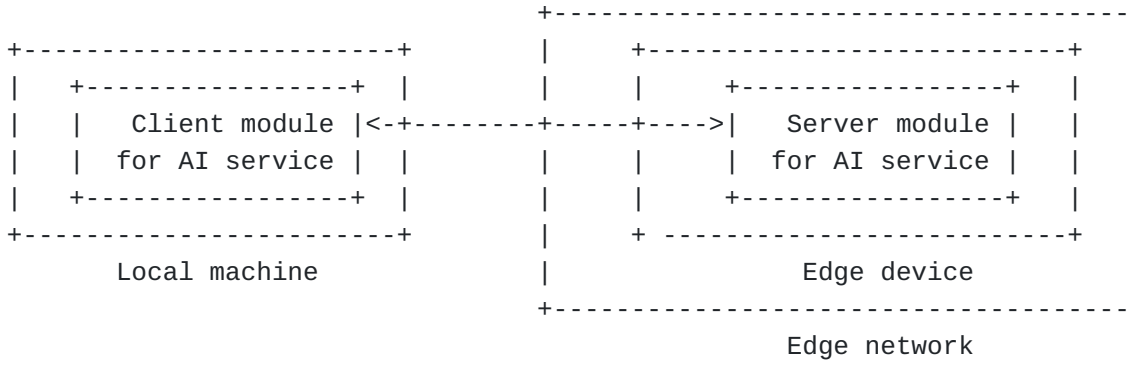

Figure 4: AI inference service on Edge device

Even in this case, the client module that requests the AI inference service runs on the local machine, the AI server module that directly performs the AI inference service runs on the edge device, and the edge device is in the edge network. Even in this case, the client module that requests the AI inference service runs on the local machine, the AI server module that directly performs the AI inference service runs on the edge device, and the edge device is in the edge network. The AI module that directly performs the AI inference service on the edge device can directly configure the edge device or use a commercial edge computing module.

The difference from the above case where the AI server module is in the cloud is that the edge device is usually close to the client, whereas the performance is lower than that of the server in the cloud, so there are advantages in data transfer time and inference time, but in unit time Inference service performance is poor.

3.4. AI inference service on Cloud server and Edge device

The following figure shows the case where AI server modules that directly perform AI services are distributed in the cloud and edge devices.

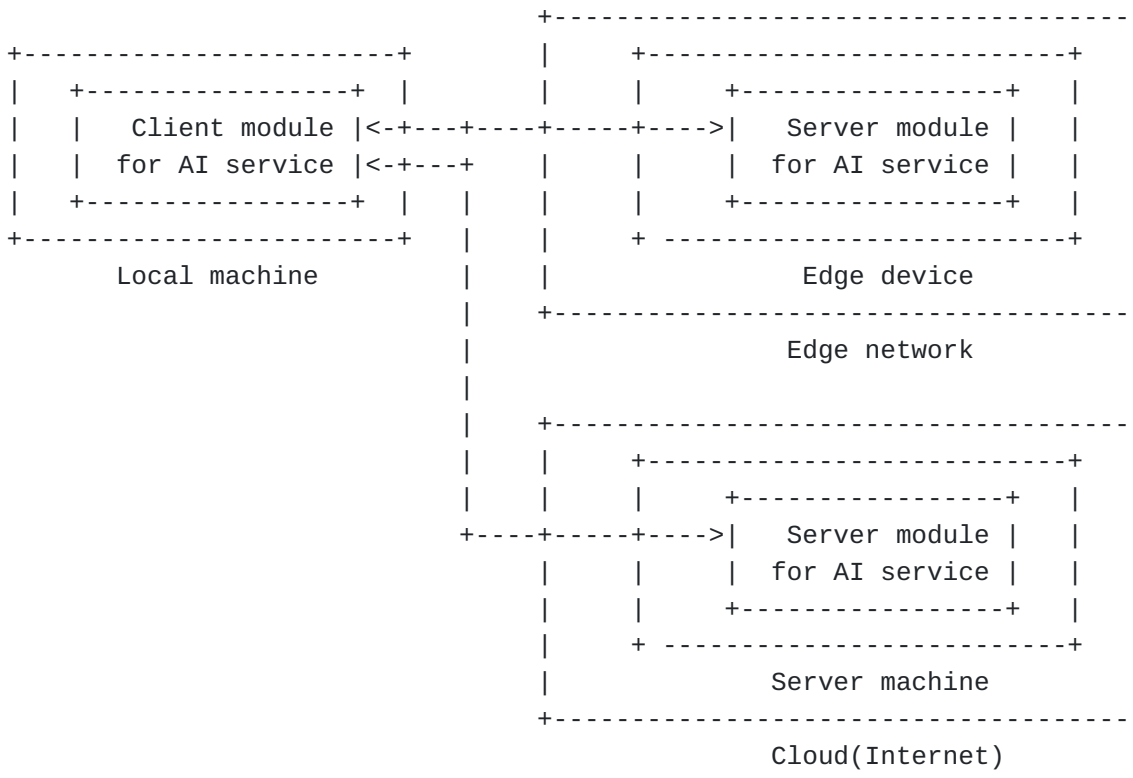

Figure 5: AI inference service on Cloud sever and Edge device

There is a difference between the AI server module performed in the cloud and the AI server module performed on the edge device in terms of AI inference service performance. Therefore, the client requesting the AI inference service may request by distributing the AI inference service request to the cloud and edge device appropriately in order to perform the desired AI service. In other words, in the case of an AI service with low inference accuracy but short inference time, we can request an AI inference service to the edge device.

3.5. AI inference service on horizontal multiple servers

In the previous section, to provide AI inference service, the network configuration that consisted of local machines, edge devices, and cloud servers is a kind of vertical hierarchy. Because the capabilities of each machine are different, the overall performance of the network using vertical hierarchy is dependent of each machine. Generally, a cloud server has a most powerful performance and then an edge device has the second powerful performance.

In this network configuration, AI service may have different performance according to the load level of the server, computing capability of the server machine and link-state between the local machine and the server machines of the horizontal level. Thus, to look for the server machine that can support the best AI service, it is necessary for the network element that can monitor network link-state and current state of the computing capability of the server machines and the network load-balance that can perform a scheduling policy of load balancing. The following figure shows the case where the local machine that requests AI service to horizontal multiple cloud servers.

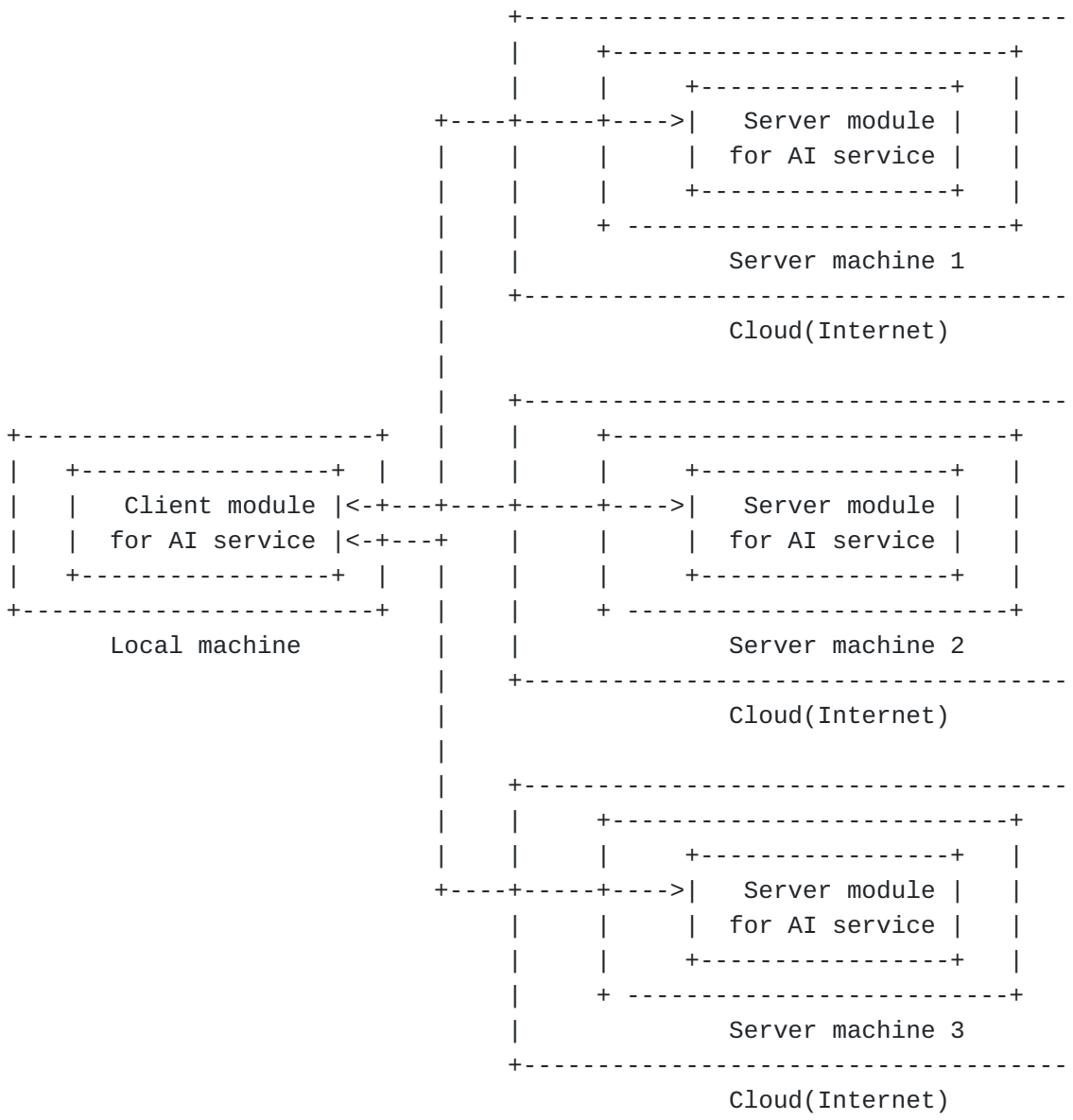

Figure 6: AI inference service on horizontal multiple servers

3.6. Network-side utilization for AI learning

Collecting and preprocessing of data and training an AI model requires a high-performance resource such as CPU, GPU, Power, and Storage. To mitigate this requirement, we can utilize a network-side configuration. Typically, federating learning is a machine learning technique that trains an AI model across multiple decentralized servers. It is a contrast to traditional centralized machine learning techniques where all the local datasets are uploaded to one

server. In this federated learning, it enables multiple network nodes to build a common machine learning model.

And, transfer learning is a machine learning technique that focuses on storing information gained while solving one problem and applying it to a different but related problem. In this transfer learning, we can utilize a network configuration to transfer common information and knowledge between different network nodes.

4. Considerations for configuring a network to provide AI services

As described in the previous chapter, the AI server module that directly performs AI inference services by utilizing AI models can be performed on a local machine or a cloud server or an edge device.

In theory, if AI inference service is performed on a local machine, AI service can be provided without communication delay time or packet loss, but a certain amount of hardware performance is required to perform AI service inference. So, in the future environment where AI services become popular, such as when various AI services are activated and AI services are disseminated, the cost of a machine that performs AI services is important and this case would not that many.

If so, whether the AI inference service will be performed on the cloud server or the discount price on the edge device can be a determining factor in the system configuration.

4.1. Considerations according to the functional characteristics of the hardware

When AI inference service request is made to a distant cloud server, it may take a lot of time to transmit, but it has the advantage of being able to perform many AI inference service requests in a short time, and the accuracy of AI service inference increases. Conversely, when an AI service request is made to a nearby edge device, the transmission time is short, but many AI inference service requests cannot be performed at once, and the accuracy of AI service inference is lowered.

Therefore, by analyzing the characteristics and requirements of the AI service to be performed, it is necessary to determine where to perform the AI inference service on a local machine, a cloud server, or an edge device.

The hardware characteristics of the machine performing the AI service varies. In general, machines on cloud servers are viewed as machines with higher performance than edge devices. However, the performance of AI inference service varies depending on how the hardware such as CPU, RAM, GPU, and network interface is configured

for each cloud server and edge device. If we do not think about cost, it is good to configure a system for performing AI services with a machine with the best hardware performance, but in reality, we should always consider the cost when configuring the system. So, according to the characteristics and requirements of the AI service to be performed, the performance of the local machine, cloud server, and edge device must be determined.

Performance evaluation is possible through the performance matrix presented in the standard of ETSI[MEC.IEG006]. The performance metrics suggested by the ETSI standard are as follows. These metrics is divided into two groups, namely Functional metrics, which assess the user performance and include some classical indexes such as latency in task execution, device energy efficiency, bit-rate, loss rate, jitter, Quality of Service (QoS), etc.; and Non-functional metrics that, instead, focus on the MEC(Mobile Edge Computing) network deployment and management. Non-functional metrics include the following indexes. Service life-cycle(instantiation, service deployment, service provisioning, service update (e.g. service scalability and elasticity), service disposal), service availability and fault tolerance (aka reliability), service processing/ computational load, global mobile equipment host load, number of API request (more generally number of events) processed/second on mobile equipment host, delay to process API request (north and south), number of failed API request. The sum of service instantiation, service deployment, and service provisioning provide service boot-time.

4.2. Considerations according to the characteristics of the AI model

According to the characteristics of the AI service, although not directly related to communication/network, the biggest influence on AI inference services is the AI model to be used for AI inference service. For example, in AI services such as image classification, there are various types of AI models such as ResNet, EfficientNet, VGG, and Inception. These AI models differ in AI inference accuracy, but also in AI model file size and AI inference time. AI models with the highest inference accuracy typically have very large file sizes and take a lot of AI inference time. So, when constructing an AI service system, it is not always good to choose an AI model with the highest AI inference accuracy. Again, it is important to select an AI model according to the characteristics and requirements of the AI service to be performed.

Experimentally, it is recommended to use an AI model with high AI inference accuracy in the cloud server, and use an AI model that can provide fast AI inference service although the AI inference accuracy is slightly lower for the fast AI inference service in the edge device.

It might be a bit of an implementation issue, but we should also consider how we deliver AI services on cloud servers or edge devices. With the current technology, a traditional web server method or a server method specialized for AI service inference (e.g., Google's Tensorflow Serving) can be used. Traditional web server methods such as Flask and Django have the advantage of running on various types of machines, but since they are designed to support general web services, the service execution time is not fast. Tensorflow Serving uses the features of Tensorflow to make AI service inference services very fast and efficient. However, older CPUs that do not support AVX cannot use the Tensorflow serving function because Google's Tensorflow does not run. Therefore, rather than unconditionally using the server method specialized in AI service inference, it is necessary to decide the AI server module method that provides AI services in consideration of the hardware characteristics of the AI system that can be built.

4.3. Considerations according to the characteristics of the communication method

The communication method for transferring data to request AI inference service is also an important decision in constructing an AI system. Using the traditional REST method, it can be used for various machines and services, but its performance is inferior to Google's gRPC. There are many advantages to using gRPC for AI inference services because Google's gRPC enables large-capacity data transfer and efficient data transfer compared to REST.

Cloud-edge collaboration-based AI service development is actively underway. In particular, in the case of AI services that are sensitive to network delays, such as object recognition and autonomous vehicle services, (micro)services for inference are placed on edge devices to obtain fast inference results and provide services. As such, in the development of intelligent IoT services, various devices that can provide computing services within the network, such as edge devices, are being added as network elements, and the number of IoT devices using them is rapidly increasing. Therefore, a new function for computing resource management and operation is required in terms of providing computing services within the network.

5. IANA Considerations

There are no IANA considerations related to this document.

6. Security Considerations

When AI service is performed on a local machine, there is no security issue, but when AI service is provided through a cloud

server or edge device, IP address and port number may be known to the outside can attack. Therefore, when providing AI services by utilizing machines on the network such as cloud servers and edge devices, it is necessary to analyze the characteristics of the modules to be used well, identify vulnerabilities in security, and take countermeasures.

7. Acknowledgements

TBA

8. Informative References

[RFC6574] Tschofenig, H. and J. Arkko, "Report from the Smart Object Workshop", RFC 6574, DOI 10.17487/RFC6574, April 2012, <https://www.rfc-editor.org/info/rfc6574>.

[RFC7452] Tschofenig, H., Arkko, J., Thaler, D., and D. McPherson, "Architectural Considerations in Smart Object Networking", RFC 7452, DOI 10.17487/RFC7452, March 2015, <https://www.rfc-editor.org/info/rfc7452>.

[I-D.irtf-t2trg-iot-edge] Hong, J., Hong, Y., de Foy, X., Kovatsch, M., Schooler, E., and D. Kutscher, "IoT Edge Challenges and Functions", Work in Progress, Internet-Draft, draft-irtf-t2trg-iot-edge-08, 16 January 2023, <https://datatracker.ietf.org/doc/html/draft-irtf-t2trg-iot-edge-08>.

[CG-AIoT] "ITU-T CG-AIoT", <https://www.itu.int/en/ITU-T/studygroups/2017-2020/20/Pages/ifa-structure.aspx>.

[tinyML] "tinyML Foundation", <https://www.tinyml.org/>.

[AI_inference_archtecture] "IBM Systems, AI Infrastructure Reference Architecture", <https://www.ibm.com/downloads/cas/W1JQBNJV>.

[Google_cloud_iot] "Bringing intelligence to the edge with Cloud IoT", <https://cloud.google.com/blog/products/gcp/bringing-intelligence-edge-cloud-iot>.

[MEC.IEG006] ETSI, "Mobile Edge Computing; Market Acceleration; MEC Metrics Best Practice and Guidelines", Group Specification ETSI GS MEC-IEG 006 V1.1.1 (2017-01), January 2017.

Authors' Addresses

Yong-Geun Hong

Daejeon University
62 Daehak-ro, Dong-gu
Daejeon
34520
South Korea

Phone: +82 42 280 4841
Email: yonggeun.hong@gmail.com

SeokBeom Oh
KSA
Digital Transformation Center, 5
Teheran-ro 69-gil, Gangnamgu
Seoul
06160
South Korea

Phone: +82 2 1670 6009
Email: isb6655@korea.ac.kr

Joo-Sang Youn
DONG-EUI University
176 Eomgwangno Busan_jin_gu
Busan
614-714
South Korea

Phone: +82 51 890 1993
Email: joosang.youn@gmail.com

SooJeong Lee
Korea University/KT
2511 Sejong-ro
Sejong City
30019
South Korea

Email: ngenius@korea.ac.kr

Seung-Woo Hong
ETRI
218 Gajeong-ro Yuseong-gu
Daejeon
34129
South Korea

Phone: +82 42 860 1041
Email: swhong@etri.re.kr

Ho-Sun Yoon

ETRI
218 Gajeong-ro Yuseong-gu
Daejeon
34129
South Korea

Phone: +82 42 860 5329
Email: yhs@etri.re.kr