Considerations of deploying AI services in a distributed approach

draft-hong-nmrg-ai-deploy-03

Y-G. Hong (Daejeon Univ.), S-B. Oh (KSA), J-S. Youn (DONG-EUI Univ),
S-J. Lee (Korea University/KT), S-W. Hong (ETRI), H-S. Yoon (ETRI)

nmrg Meeting@IETF 116 – Yokohama

March 29. 2023

History and status

- 00 : draft-hong-nmrg-ai-deploy-00 (Mar. 2022)
- 1st revision : draft-hong-nmrg-ai-deploy-01 (Jul. 2022)
  - 1st presentation
- 2nd revision : draft-hong-nmrg-ai-deploy-02 (Oct. 2022)
  - 2nd presentation
- **3rd revision : draft-hong-nmrg-ai-deploy-03 (Mar. 2023)**
  - **3rd presentation**

Motivations

- Deployment of AI services
  - Focus : training (learning) -> inference (prediction)
  - For inference, not only high-performance servers, but also small hardware, microcontroller, low-performance CPUs, and AI chipsets are optimal target device (due to cost)
- Configuration of the system/network in terms of AI inference service
  - For training : accuracy of the model
  - For inference :
    - Target device : Local, edge, cloud
    - Objectives : Accuracy, Latency, Network traffic, Resource utilization, etc.
    - Considerations : Network configurtion, AI model, Serving framework, Communication method, device capacity, inference data, etc.
- Accelerate the study AI issues and find some possible standardization items in the nmrg

Comments in last meeting

- Question by Alexander Clemm
  - more system management or network management? considerations for federated learning, transfer learning? would like to see more "network-side"
- Question by Jeff Tantsura
  - mentioned "vertical" hierarchy (remote/close edge) vs. "horizontal" hierarchy (ability to run the inference on one or multiple servers)
- Comment by Jeferson Campos Nobre
  - It would be nice to have the similarities between this draft and the research challenges one pointed out

Updates after last meeting (1/2)

– Add section 3.5 “AI inference service on horizontal multiple servers”

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.

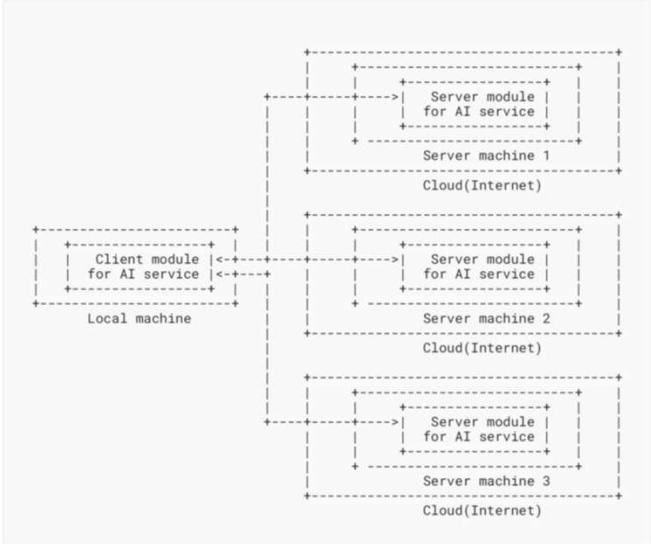


Figure 6: AI inference service on horizontal multiple servers

Updates after last meeting (2/2)

– Add section 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.

Network configuration structure to provide AI services

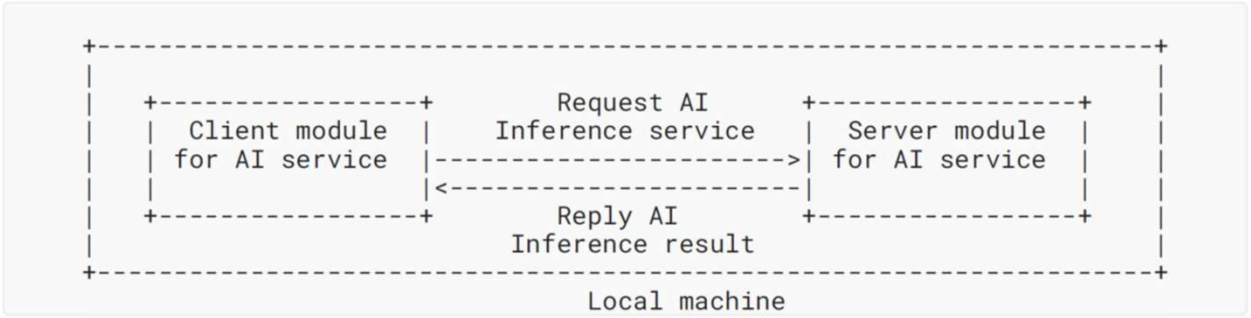


Figure 2: AI inference service on Local machine

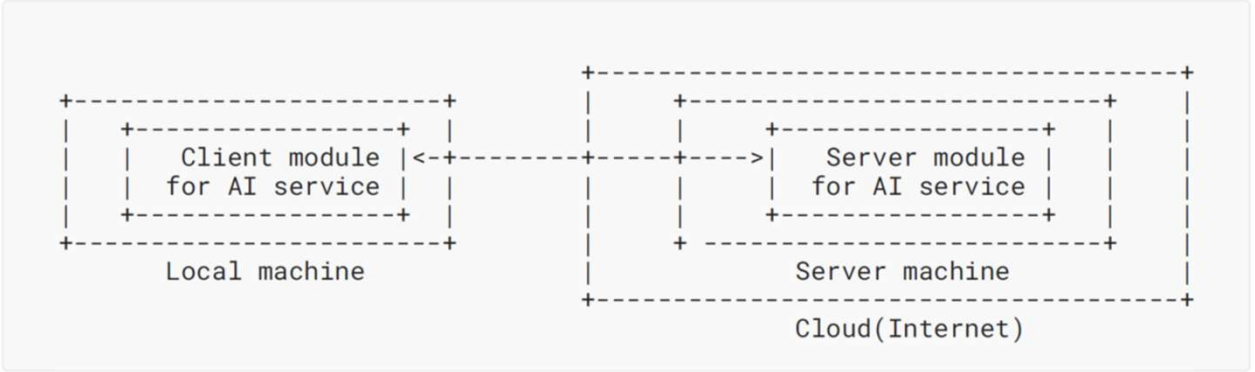


Figure 3: AI inference service on Cloud server

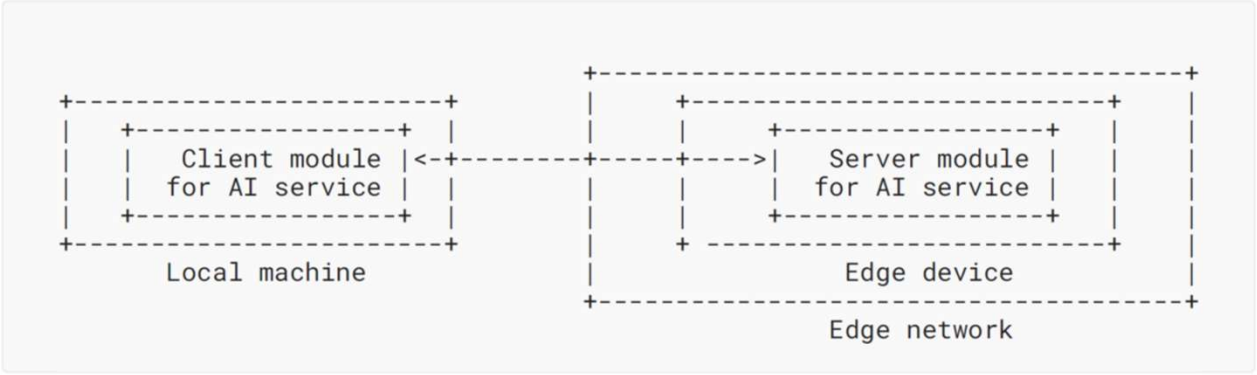


Figure 4: AI inference service on Edge device

AI inference service on vertical/ horizontal servers

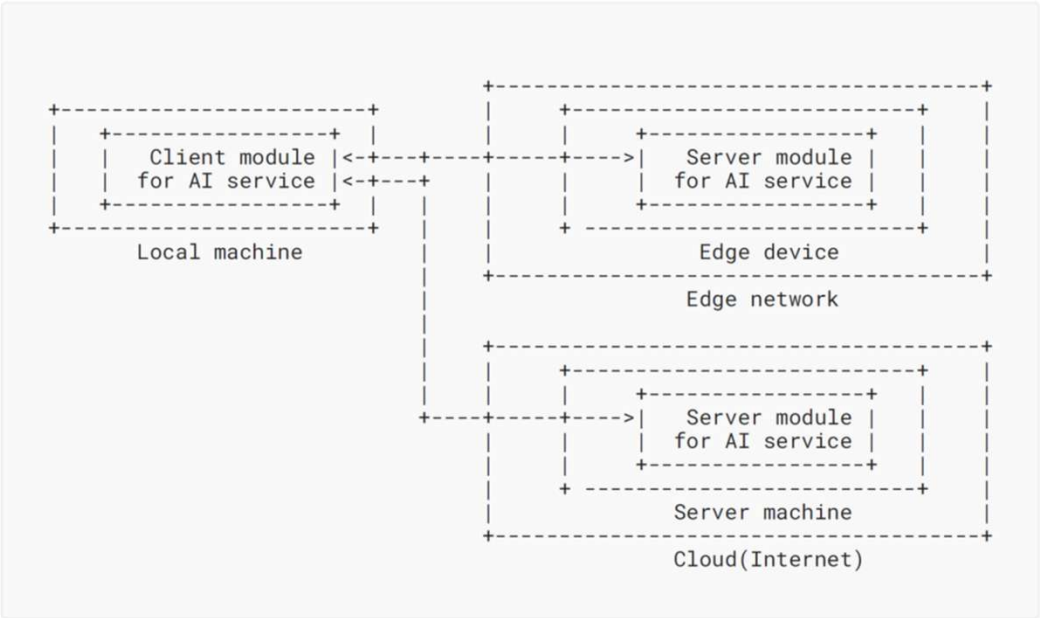


Figure 5: AI inference service on Cloud sever and Edge device

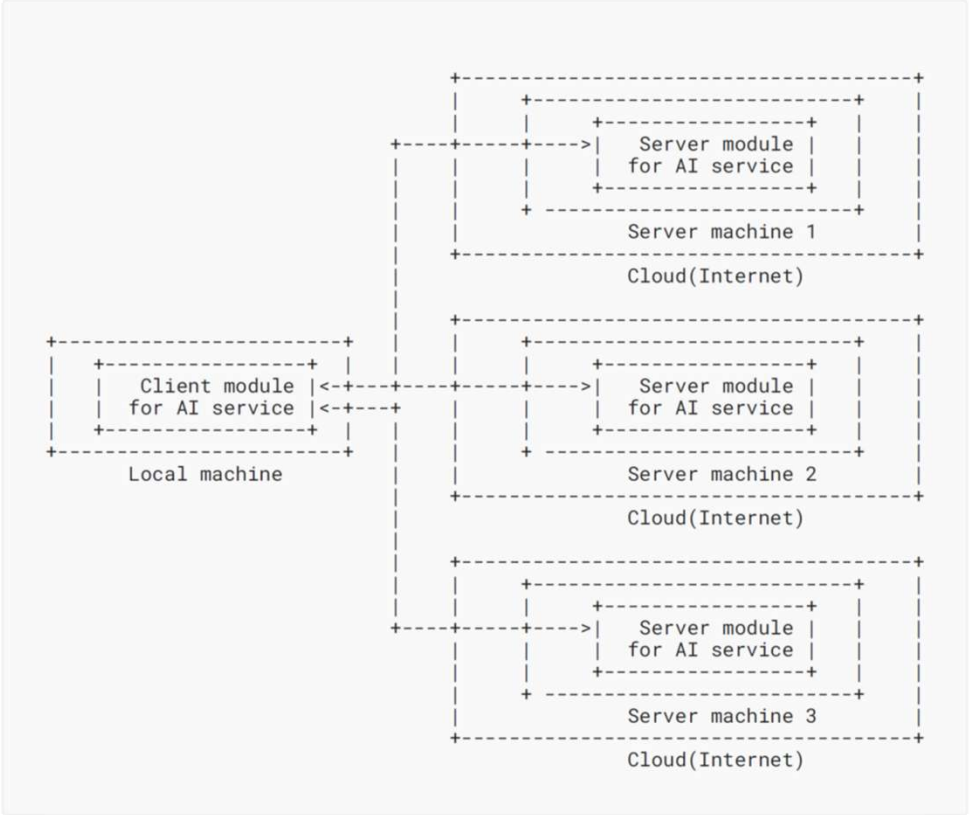


Figure 6: AI inference service on horizontal multiple servers

Considerations according to the functional characteristics of the hardware (1/2)

- (Reference) ETSI Group Specification MEC-IEG 006 V1.1.1 (2017-01) "Mobile Edge Computing; Market Acceleration; MEC Metrics Best Practice and Guidelines"
  - It describes various metrics which can potentially be improved through deploying a service on a MEC platform
  - It can be identified in order to highlight the benefits of deploying MEC for various services and applications
  - Functional metrics
    - latency (both end-to-end, and one-way), energy efficiency, throughput, goodput, loss rate (number of dropped packets), jitter, number of out-of-order delivery packets, QoS, and MOS
  - Non-functional metrics
    - service lifecycle (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 ME host load, number of API request (more generally number of events) processed/second on ME host, delay to process API request (north and south), number of failed API request

Considerations according to the functional characteristics of the hardware (2/2)

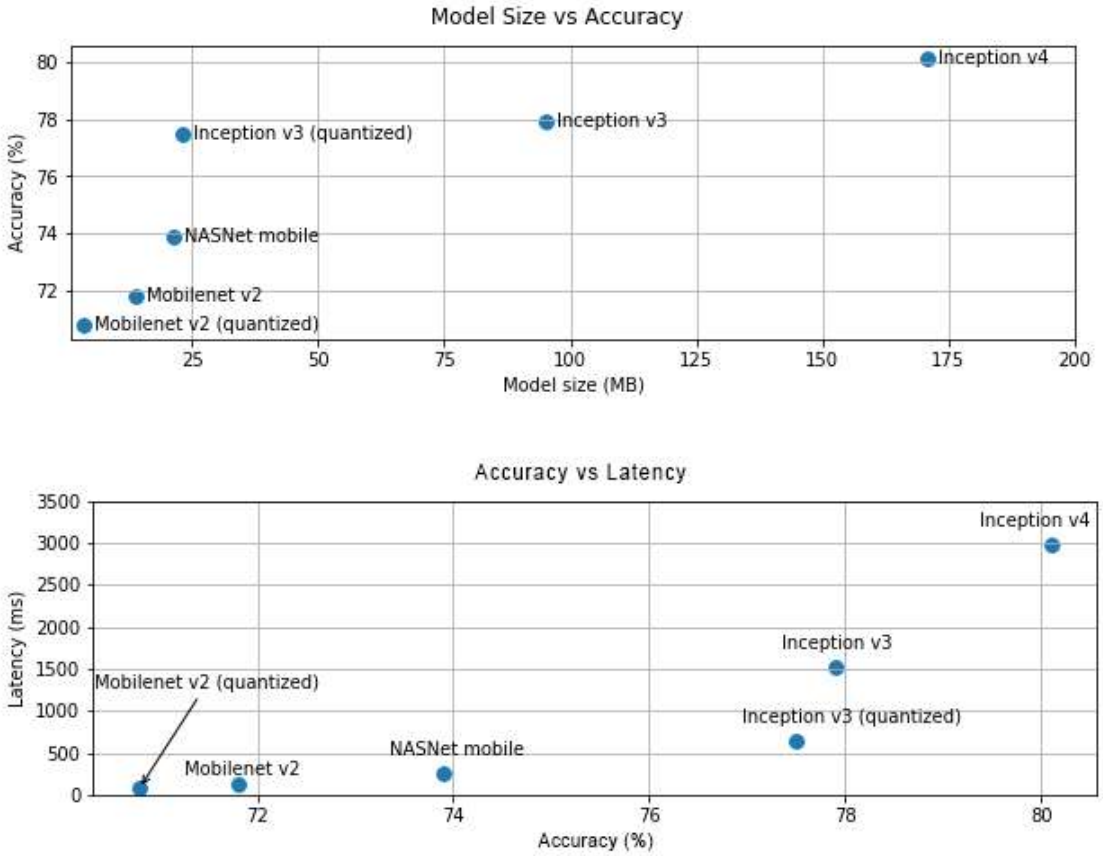
– The performance of AI inference service varies depending on how the hardware such as CPU, RAM, GPU, and network interface is configured for each cloud server and edge device.

– AI inference service can be deployed in the following locations
  - Distant cloud server : High performance and high cost
  - Near edge device : Medium performance and medium cost
  - Local machine : Low performance and low cost

– AI inference service result in (assumption: same AI model)
  - Distant cloud server : High accuracy, short inference time, and long delay to transmit
  - Near edge device : Medium accuracy, medium inference time, and medium delay to transmit
  - Local machine : Low accuracy, long inference time, and short delay to transmit

Considerations according to the characteristics of the AI model (1/2)

– Model size vs. Accuracy vs. Latency



[Source : Google Tensorflow]

Considerations according to the characteristics of the AI model (2/2)

- AI inference service can be deployed in the following locations
  - Distant cloud server : Heavy AI model, high accuracy, Big size, long inference time
  - Near edge device : Medium AI model, medium accuracy, medium size, medium inference time
  - Local machine : Light AI model, low accuracy, small size, short inference time

- AI inference serving framework
  - Traditional web server : ex) FastAPI, Flask, and Django
    - It can be operated on low performance machines
  - Specialized serving framework : ex) Tensorflow serving
    - It can provide high performance.

Considerations according to the characteristics of the communication method

- AI inference service can be utilized
  - Traditional REST method
    - Common and easily deployed
  - Specified communication method (e.g., gRPC)
    - Better performance but need some works

- AI Inference data can be classified
  - Real-time vs. Batch
  - Secure & non-secure

Relationship to “Challenge document” (1/2)

- This draft is also related to the “Challenge document” and some texts can be added or merged
  - Distributed AI service
  - Lightweight AI service
  - Deployment of AI service

- This draft can be developed as a different document to focus on AI inference (Deployment of AI services)
  - The “Challenge document” includes many items

Relationship to “Challenge document” (2/2)

1. Introduction 2
2. Procedure to provide AI services 4
3. Network configuration structure to provide AI services . . . 6
3.1. AI inference service on Local machine 6
3.2. AI inference service on Cloud server 7
3.3. AI inference service on Edge device 7
3.4. AI inference service on Cloud server and Edge device . . 8
3.5. AI inference service on horizontal multiple servers . . . 9
3.6. Network-side utilization for AI learning 11
4. Considerations for configuring a network to provide AI services 11
4.1. Considerations according to the functional characteristics of the hardware 11
4.2. Considerations according to the characteristics of the AI model 12
4.3. Considerations according to the characteristics of the communication method 13
5. IANA Considerations 14
6. Security Considerations 14
7. Acknowledgements 14
8. Informative References 14
Authors' Addresses 15

<This draft>

1. Introduction 3
2. Conventions and Definitions 5
3. Acronyms 5
4. Difficult problems in network management 5
5. High-level challenges in adopting AI in NM 8
6. AI techniques and network management 10
6.1. Problem type and mapping 10
6.1.1. Sub-challenge: Suitable Approach for Given Input . . 11
6.1.2. Sub-challenge: Suitable Approach for Desired Output 11
6.1.3. Sub-challenge: Tailoring the AI Approach to the Given Problem 12
6.2. Performance of produced models 13
6.3. Lightweight AI 15
6.4. Distributed AI 17
6.4.1. Network management for efficient distributed AI . . . 17
6.4.2. Distributed AI for network management 18
6.5. AI for planning of actions 19
7. Network data as input for ML algorithms 21
7.1. Data for AI-based NM solutions 21
7.2. Data collection 22
7.3. Usable data 24
8. Acceptability of AI 25
8.1. Explainability of Network-AI products 25
8.2. AI-based products and algorithms in production systems . 27
8.3. AI with humans in the loop 28
9. Security Considerations 29
10. IANA Considerations 29

<Challenge document>

Thanks!!

Questions & Comments