PoS PROCEEDINGS OF SCIENCE

Research on Hybrid Architecture System of Embedded and Cloud Computing Storage

Hengxiang He
School of Information and Communication Engineering, Harbin Engineering University
Harbin, 150001, China
E-mail: hehengxiang0@163.com

Chunyu Chen
School of Information and Communication Engineering, Harbin Engineering University
Harbin, 150001, China
E-mail: springrain@hrbeu.edu.cn

Yulong Qiao
The authors are with College of Information and Communication Engineering, Harbin Engineering University; National Natural Science Foundation of China under Grant 61371175
Harbin, 150001, China.
E-mail: qiaoyulong@hrbeu.edu.cn

In order to solve the problem that embedded devices are unable to run artificial intelligence and other computationally intensive tasks due to the poor performance, while high-performance computing devices can not be deployed on the application site because of the limitation of volume and power consumption, this paper presents a hybrid architecture with embedded and cloud computing storage. The system based on a robust network communication architecture can timely transmit data collected by embedded devices to the cloud, and can properly dispatch cloud computing node cluster for high-speed computing. After the actual test, the system fully meets the enormous requirements of computing resources for the embedded device to run artificial intelligence, greatly reducing the computational load of the embedded terminal. This hybrid architecture provides a large amount of computing resources for embedded devices, avoids resource waste and maintenance difficulties caused by the local deployment of computing resources, as well as effectively reduces the overall capital investment of the devic*e.*

ISCC2017
16-17 December 2017
Guangzhou, China

1.Introduction

In recent years, AI (artificial intelligence) has ushered with a rapid development, and the embedded system after decades of development brings in larger scope of application. Wang Hongqun and others from the Huazhong University of Science and Technology, successfully used embedded devices to control unmanned aerial vehicles for autonomous landing[1]. Combination of artificial intelligence and embedded system can create unmanned systems, make portable systems more intelligent, and enable problem solving via traditional method in an easier way, such as complex environment target segmentation and recognition problems. Liu Dawei and Han Ling adopted deep learning to achieve the classification of [2] remote sensing image. However, large-scale deployment of embedded devices also generates the problem of centralized management. How to better manage and comprehensively analyze the data collected by various embedded devices has become the hot topic of current researches. AI relies on extremely powerful computing capabilities, but embedded devices usually cannot meet the deep learning requirements due to their simple structure, low power consumption and low heat generation. Cloud server- workstation cluster, has a strong computing power and stable data storage capacity, can simultaneously introduce the multi-device common access, effectively improve the utilization of resources. Therefore, relying on the modern ubiquitous network environment, cloud computing and storage is now leading the development trend of AI run by embedded system . This paper will explore a hybrid architecture system of embedded data collection, cloud computing and storage.

2. The Introduction of Each Module of the System

According to the conventional approach, many computing resources are locally distributed, such as sensors, control devices and computing devices are usually located in the same device, which will lead to the significant increase of equipment volume, power consumption, heat and noise. However, this is unacceptable in many large-scale industrial applications. In this mode, each set of sensors need to be equipped with a set of computing devicesaccordingly. In that case, the computing device is in an intermittent working status, resulting in an unnecessary waste of computing resources and funds. Compared with the conventional method, the hybrid architecture system proposed in this paper concentrates on computing devices in the cloud, which can greatly improve the utilization rate of the devices and reduce the waste of computing resources and funds. This system has a tremendous advantage over the simple artificial intelligence computing with embedded system program, in the calculation speed. Atypical VGG16 network, for example, the calculation speed can be increased by several times.[3]

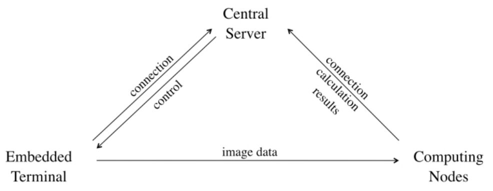


Figure 1: System architecture diagram

The system uses a central server as the communication core node, which dispatches all embedded devices and compute nodes to allocate computing resources to the embedded devices. The compute node is responsible for receiving the data collected by the embedded devices, running deep learning to get the calculation result, and finally pass results to the central server's database.

As a highly integrated device, embedded system has the advantages of small size, low heat, and stable operation over a long period of time, which makes it a widely used device in current smart home and automated production environments. Most embedded systems using ARM series SOC, in comparison with the traditional SCM, have higher computing power, and the ability to run the embedded operating system. Raspberry Pi (Raspberry Pi) is the current star of embedded products, with the latest model usingSOC 3B: Broadcom BCM2837, CPU for the ARM Cortex-A53, is a processor clocked at 1.2GHz 64-bit quad-core , equipped with 1GB of memory, onboard four USB2.0 ports and WiFi, Bluetooth module, and 40 GPIO interfaces. The pconfiguration for Smooth Operation Custom Linux and Windows loT systems, quad-core processors and large memory configurations make multi-threaded programming more efficient and rich peripheral interfaces make it easy to connect with a wide range of sensors. The above features make Raspberry Pi a good choice for the embedded architecture of the hybrid architecture described in this article[4].

A robust cloud computing platform requires a stable central service node. The cloud server is a simple, efficient, reliable and scalable computing service. Its redundancy backup and other disaster recovery measures can prevent the server from being powered off when a broken network and other unforeseen circumstances cause a fatal blow to the entire system. The system uses Aliyun server as the central node, which is configured for single-core 2.4GHz Xeon CPU, 2GB memory, 1Mb fixed bandwidth, for the operating system for Windows Server 2013.

Deep learning is in need of a large number of matrix operations, in this respect, GPGPU has a huge advantage over CPU. Deep learning relies on a lot of graphics to general purpose computing. The two commonly used frameworks of GPGPU are CUDA and OpenCL. Compared with open source OpenCL, Nvidia graphics card has CUDA programming framework, and it has certain strength on performance and vendor support. Therefore, most of our graphics use Nvidia in the field of deep learning. Workstations used by each computing node in the system are equipped with multiple Nvidia graphics cards, and multiple computing nodes work together to provide the system with plenty of computing power. A typical compute node is configured as a Core i7 / i9 or Ryzen 7 and above processors, 64GB of memory, 4 Titan Xp / 1080Ti graphics card, Raid 5 array of Western Digital red disk array as a data storage warehouse, operating system for Ubuntu 16.04 .

3. The Introduction of System Network Architecture

As a application-oriented hybrid architecture system, the system uses three platforms: embedded terminal, cloud server and workstation. The operating systems corresponding to the three platforms are: customized lightweight Linux, Windows and Ubuntu Operating system. How to make the three completely different hardware and software equipment to work together has become a major challenge. Socket as a cross-platform programming interface can well adapt to the system of communication among devices[5]. Socket has four operating modes: synchronous blocking mode, asynchronous blocking mode, synchronous non-blocking mode, and asynchronous non-blocking mode. Based on a comprehensive comparison, the synchronous blocking mode was selected as the inter-system communication mode.

3.1 Central Server

The central server is the core node of the system. It has a fixed IP address which is responsible for comprehensively leveraging the load of each computing node, controlling the working status of each embedded terminal, and reading and writing My SQL database. Due to the uncertainty of the connected devices, cloud servers are required to have flexibility on control, such as increasing or decreasing compute nodes anytime, anywhere, dynamically balancing compute node loads, increasing or decreasing embedded device connections. The cloud server separately establishes two Vector containers for the compute node and the embedded terminal. Through these two containers, it is convenient to dynamically manage the working status of each connected device.

The core of central server communication program is a Socket Server side, with each receiving a device connection request, and then create a new dedicated thread to deal with the device's communication. The program will firstly verify the identity of the connected device, if the identity or the information is incorrect, the communication with the device will be interrupted and the thread is to be released. After the authentication, information is passed, the type of the connected device is evaluated:

1. If the connection device is a computing workstation, the information of the computing node is further received. The information includes access mode, access address, maximum computing power and current load, among which access mode is two modes of fixed IP and domain name access. After receiving the above information, the device stores the information received by the Socket handle and above in a structure, and transfers the structure to the container of the computing node. Then, the device periodically shakes hands with the computing nodes, detects the working status of the computing nodes and the network communication status, updates the stored information in the container in real-time. Each computing node will return the calculation result to the central server by handshake communication, and the data will be stored in the My SQL server after the central server receives it.

2. If the connected device is embedded, similarly, the device needs to verify the identity information first to prevent unauthorized device connections. After authentication, the equipment information will be stored in the embedded server container, and then run the load balancing procedure to calculate the node information resource. The most abundant current is calculated, then the nodes of the network access mode and address sent to the embedded terminal. The node also keeps the handshake with the embedded terminal, receives the working status of the embedded terminal, and transfers the control instructions to the embedded terminal by handshaking communication if necessary.

Cloud server balancing program can monitor the working status of each computing node in real-time. When a work node growth time drops with no response, coping with error information and other situations in an abnormal way, cloud server can quickly detect that, and then shields the node, waiting for its recovery, and timely distributes the load of the node to the rest of the normal running nodes. Similarly, when a new computing node is added, the cloud server can assign the computing task to the node with priority, and when a network configuration of a computing node changes, the system has to respond in-time.

In order to comprehensively display the connection and status between the computing node and the embedded terminal, the central server program also has a status monitoring window to facilitate the management for observation.

3.2 Embedded Terminal

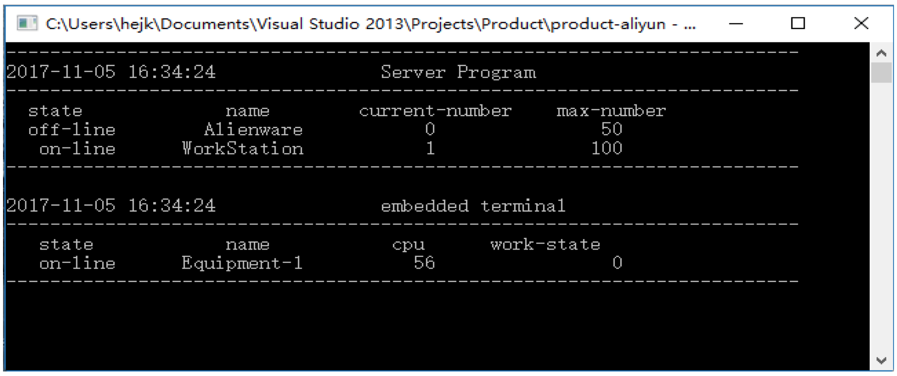
In this system, the embedded terminal is responsible for obtaining the 3D image of the target in real-time, and sending the data to the computing node through the network. Due to the fact that an embedded client needs a wide range of installation in the factory environment, the use of network cable connection will greatly increase the installation complexity, and messy alignment on the installation environment will have an extremely negative impact. Raspberry Pi has Wi-Fi module, so the system uses one or more routers to connect these embedded devices. As the heavy load calculation is done by the cloud-computing node, the embedded end of the computing load is small, can reduce the heat and power consumption of the device so to extend the lifetime of embedded devices.

After the embedded terminal system starts to run, a Socket Client end needs to be established in the first place, and then the preset IP and Port information is used to build up communication with the central server. After the mutual authentication information passes, it receives the network mode, connection address and so on, as information returned from the central server. With the adoption of multi-threading technology, a new Socket Client-side in the computing node network mode, connects with the corresponding mode with. After the connection, it starts running image acquisition and processing program.

After obtaining the 3D image data from the camera, the data preprocessing is firstly conducted on the embedded end, and the data with poor performance is removed to reduce the workload on the network and the computing node. The upload process will occupy the uplink bandwidth that is far less than the downlink bandwidth under the current prevailing network environment. When multiple embedded devices work together on the same network, the uplink bandwidth becomes more crowded. For large 3D images, single image size can reach 3.4MB. Direct uploading of the original 3D images will result in tremendous pressure on the network, which will eventually lead to uplink network congestion, data transmission capacity reduction. Therefore, it’s required to be embedded in the data-lossless compression, and then uploaded, so to greatly improve the network's data transmission capabilities. In addition, as the data needs to be transmitted on the public network to ensure the security of the data transmitted, it has to be encrypted.

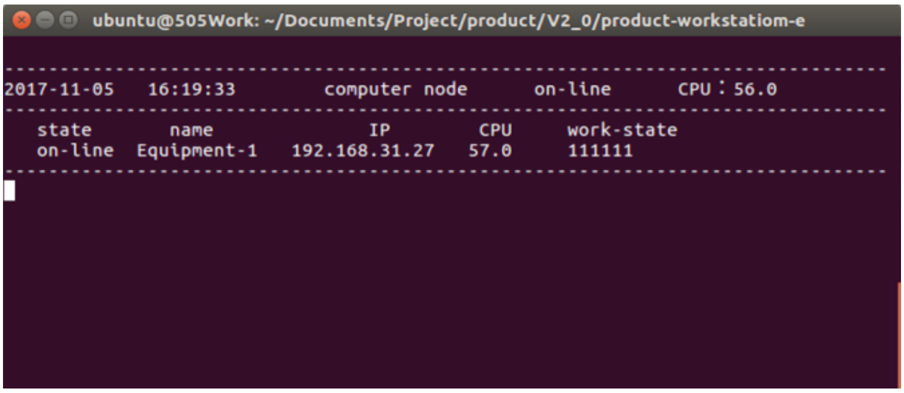
3.3 Computing Nodes

At the same time, computing nodes allow access to different network models. The access to fixed IP has advantages in stability, reliability and real-time performance, however at a very high price. Additionally, in some places, because of the network providers and other factors, the fixed IP cannot be accessed, so this system allows computing nodes in network mode. In this mode, it is necessary to use intranet penetration technology to map the internal network IP of the computing node to a fixed domain name, and the external device can communicate with the computing node through the domain name. This system uses peanut bars as network penetration equipment. The two working modes and network information need to be written into a cache file in advance. After the system starts, it will read the cache file first to obtain the information and complete the initialization.



C:\Users\hejk\Documents\Visual Studio 2013\Projects\Product\product-aliyun - ...

```
------------------------------------------------------------------------
2017-11-05 16:34:24                 Server Program
------------------------------------------------------------------------
  state            name        current-number    max-number
  off-line        Alienware          0               50
   on-line       WorkStation         1              100
------------------------------------------------------------------------
2017-11-05 16:34:24                 embedded terminal
------------------------------------------------------------------------
   state           name           cpu       work-state
   on-line       Equipment-1       56                0
------------------------------------------------------------------------
```



ubuntu@505Work: ~/Documents/Project/product/V2_0/product-workstatiom-e

```
--------------------------------------------------------------------------------
2017-11-05    16:19:33         computer node       on-line       CPU：56.0
--------------------------------------------------------------------------------
   state       name            IP          CPU      work-state
   on-line  Equipment-1   192.168.31.27   57.0      111111
--------------------------------------------------------------------------------
```

Figure 2 : GUI of server program **Figure 3 :** GUI of computer node

The calculation program contains several parts of nodes:

3.3.1 Communication Program

The communication program includes the Socket Server client that communicates with the central server and the one that communicates with the embedded client. When the program starts up, it initially builds up communication with the central server, then sends its own information and maintains the handshake communication with the server, while the calculation result will also be uploaded to the server database through the handshake communication. After starting the embedded terminal of the communication program, each receives an embedded terminal connection, the system will create a new thread to communicate with the device, with the terminal embedded at a high speed to accept the data. The various communication threads are independent from each other.

3.3.2 Deep Learning Program

A computing node configured with multiple high-performance graphics requires deep learning process to have the load distribution capabilities to ensure that multiple graphics cards can work well together. When the computing node receives the image data, it first decrypts the data and restores it to the 3D image without any loss, and then sends it to the deep learning network for calculation, after getting a certain amount of processing [5]. Since the result is obtained, the calculation result is sent to the central server by the communication program sin the database, and the original data received from the embedded terminal is simultaneously stored in the compute node's own hard disk array for subsequent use.

3.3.3 Comprehensive Scheduling and Monitoring Panel Program

Integrated scheduling and monitoring panel program charge for reading and writing a large amount of data of deep learning, comprehensive scheduling of resources for each thread to ensure the stable and reliable operation of each thread. Meanwhile, in order to facilitate the system administrator to monitor the system status, the compute node device program provides a status monitor panel.

4. System Performance Analysis

Under the condition that each computing node has a network bandwidth of 6Mb and a plurality of embedded terminals that share a 10Mb uplink bandwidth, the computing node can receive seven PNG images per second, with the image size of about 80KB. This speed reaches the network bandwidth limit. Computational stress test results are shown in the table below.

Net Device	Compute Node	Embedded Terminal
AlexNet	1500 frames per second	2 frames per second
Faster R-CNN	21 frames per second	unable to run

Table 1: Computing Capability Stress Test Table

5. Conclusion

With the in-depth application of AI in automation and unmanned scene, the massive data transmission and storage of embedded system has become a dominating research area. This paper describes a hybrid system architecture of embedded terminal and cloud computing storage which relies on a set of stable and reliable communication architecture. The embedded data can be transmitted to the computing nodes at a high speed and the embedded terminal load is reduced. At the same time, compute nodes with powerful computing capacity meet the needs of deep learning and computing, computing resources will then be concentrated on the cloud, in order to maximize the utilization of hardware resources, but also to greatly save the investment on embedded computing equipment.

References

[1] Otto C, Milenkovic A, Sanders C, Jovanov E (2006) *System architecture of a wireless body area sensor network for ubiquitous health monitoring*. J Mob Multimedia 1(4):307–326

[2] Evangelin E, Sam D (2014) *Wireless body area networks and its emerging technologies in real time applications.* IJESRT 3(1):309–313, ISSN: 2277–9655

[3] Fong EM, Chung WY (2013) *Mobile cloud-computing-based healthcare service by noncontact ecg monitoring.* Sensors 13(12):16451–16473, ISSN:1424–8220

[4] Ji Z, Ganchev I, O'Droma M, Zhang X, Zhang X (2014) *A cloud-based X73 ubiquitous mobile healthcare system: design and implementation.* Sci World J 2014:1–14

[5] Sum H, Sum X, Wang H. *Automatic target detection in high resolution remote sensing images using spatial sparse coding bag-of-words model*[J]. IEEE Geoscience and Remote Sensing Letters, 2012, 9(1): 109-113.