Journal of Computer and Communications, 2017, 5, 1-8
http://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Unified Platform for AI and Big Data Analytics

Sik Kim, Yongjin Kwon*

Department of Industrial Engineering, College of Engineering, Ajou University, Suwon, South Korea
Email: *yk73@ajou.ac.kr

How to cite this paper: Kim, S. and Kwon, Y.J. (2017) Unified Platform for AI and Big Data Analytics. *Journal of Computer and Communications*, **5**, 1-8.
https://doi.org/10.4236/jcc.2017.58001

Received: April 12, 2017
Accepted: June 9, 2017
Published: June 12, 2017

Abstract

This paper describes an integrated platform for machine learning and big data analysis. The integrated platform is configured in a way that builds a large distributed data processing environment in the computing environment that makes up the NVIDIA AI platform. In addition, this paper describes the background of this idea selection and the use of the software to build the unified platform. The technical details are shown in terms of how to create the proposed platform. In the anlaysis section, the methodology is provided and also the steps are explained as to how to use this integration platform. Finally, the expected effects are elaborated in the conclusion section.

Keywords

Integrated Platform, Hadoop Eco System, Ambari, Virtual OS, Jetson TX-1, Dev Box, SSH

1. Introduction

In general, artificial intelligence modeling requires a high-end computing environment. In particular, the modeling of AI, this is based on graphic processing capabilities such as NVIDIA, and requires the combination of high-end GPUs as well as CPUs [1] [2]. The inefficiency exists, however, in this high-end computing environment, if the computing power is only used for machine learning purposes. The valuable computing power can be better utilized, if a virtual server is made within the computer and used for the analysis of big data. In recent years, we have witnessed the massive increase of data streams generated by the unmanned systems, such as drones and autonomous vehicles. Those systems are increasingly integrated with machine learning algorithms, while generating and transmitting a large amount of data (*i.e.* image data, system parameter data, text data, and so on) in real-time. The onboard computers are conducting the processing for machine learning algorithms. However, the big data streams generated by the system itself also needs to be processed and analyzed simultaneously. In this regards, an integrated platform is proposed in this study that can efficiently and simultaneously perform the big data analysis as well as the machine learning processing (this function is our research purpose). This is achieved by creating distributed computing environment with the use of Hadoop EcoSystem [3] [4]. The details are explained in the following sections.

2. Idea Extraction Process

2.1. Idea Generation

When data storing of information gathered from drones or autonomous driving cars is made, the artificial intelligence modeling (that is, machine learning algorithms) and the big data processing have been performed on different platforms. This concept is illustrated in Figure 1.

This process depicts the inefficiency because the processing of big data is performed on a different computing platform. On the other hand, Figure 2 is depicting the unified platform that runs both machine learning algorithms and big data processing within a single PC.

2.2. Why Should Build a Distributed Computing Environment?

Figure 3 illustrates the performance between RDBMS and Distributed Computing Environment (Node 3 - Node 5). As one can see, the distributed computing environment is faster than RDBMS. Also, if there are more nodes, the data processing time becomes shorter.

Figure 4 illustrates the comparison of performance when using the Hadoop (distributed computing environment) and the SPARK, a more advanced technology that is called "in-memory" system.

As one can see from the figure, the Hadoop's Map Reduce performs better

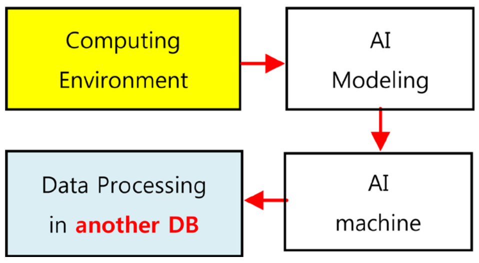


Figure 1. Previous AI modeling and big data processing.

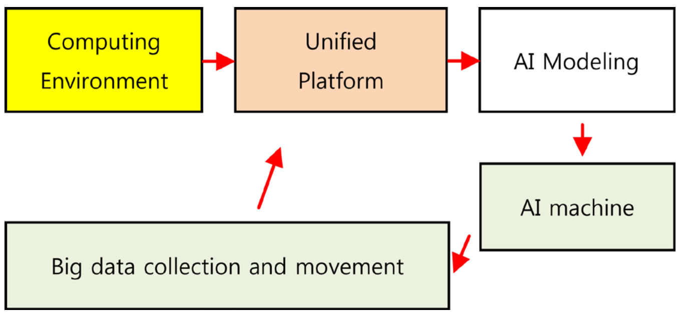


Figure 2. Unified platform model.

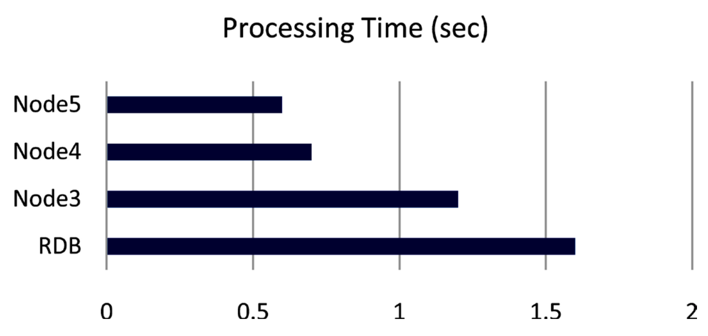


Figure 3. Comparison between distributed computing environment and existing RDBMS.

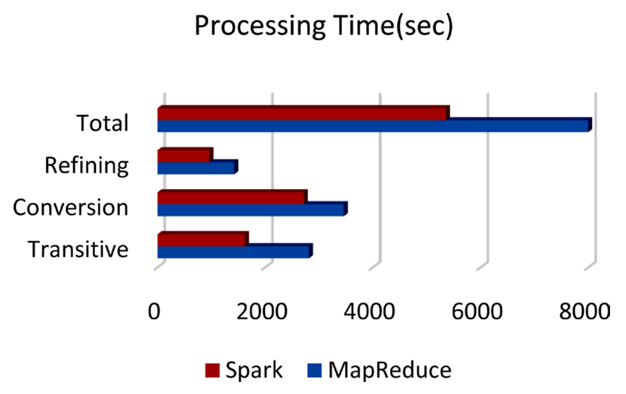


Figure 4. Comparison between SPARK and Hadoop (Map Reduce).

than RDBMS. The SPARK also performs better than the Hadoop's Map Reduce. This comparison shows the importance of using a distributed computing environment to handle big data.

3. NVIDIA Artificial Intelligence (AI) Platform

3.1. NVIDIA AI Platform—Host Server (Physical HW)

The host server is made up of HW equipment for AI machine learning such as NVIDIA's DevBox or Jetson TX1. These devices use the Linux Ubuntu 14.04 as the OS. The virtual server for the distributed environment also uses the same OS to enhance the compatibility. Overall, the distributed environment is made by using the host server and virtual server (built by using Oracle VirtualBox) [5] [6].

3.2. NVIDIA Digits (Image Training SW)

NVIDIA's representative image training software, Digits, is the SW that can be integrated with the autonomous vehicles and drones. It is basically SW that supports CUDA development environment developed by NVIDIA and is optimized for image training. The details are shown in Figures 5-7 [7] [8] [9].

4. Configuration of Network between Host and Slave Servers

After building the slave servers, we used the SSH network configuration to make it possible to access the slave server from the host server without sharing the information through the network configuration between the servers.

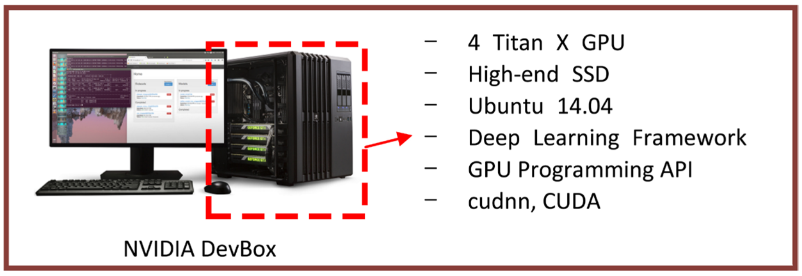
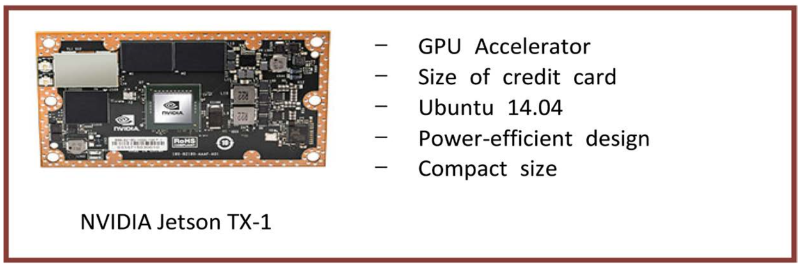


Figure 5. NVIDIAI AI modeling "DevBox" (HW).



1

Figure 6. NVIDIAI AI modeling machine "Jetson TX-1" (HW).

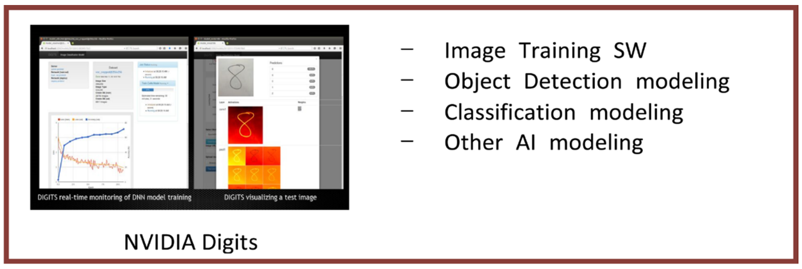

– Image Training SW
– Object Detection modeling
– Classification modeling
– Other AI modeling
DIGITS real-time monitoring of DNN model training
DIGITS visualizing a test image
NVIDIA Digits

Figure 7. NVIDIAI AI modeling SW "Digits".

4.1. SSH Key Generation and Share

SSH network configuration allows access from a host server to a virtual server (slave). However, a password is required to access another server in its current state. Therefore, it is necessary to create an SSH shared key and share it between the servers so that communication can be smoothly performed without a password. In other words, setting up SSH network allows the autonomous communication between the servers without a password. The Linux command code is given in Table 1 and Table 2.

4.2. After Network Configuration between Host and Slave

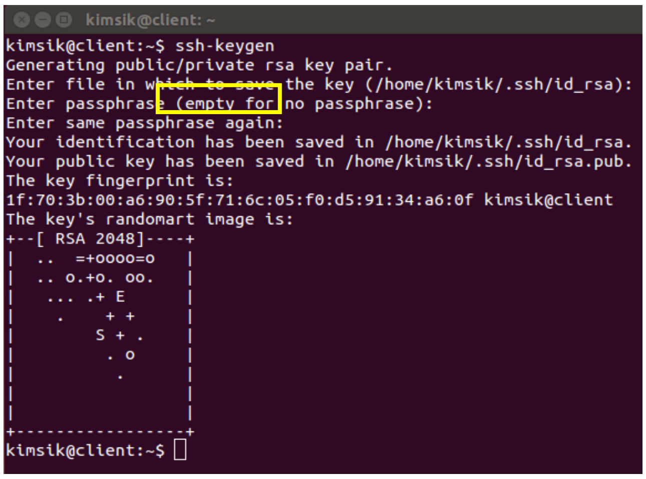
Figure 8 shows that the SSH key is generated. The SSH key acts as the connection identity of the platform that needs to be connected. For example, one system is given a specific key structure and this key is only unique to this system. By examining the SSH key, one can identify the each individual system. Once the key is created and given to each system (*i.e.*, server) and the connection is established, the host server and the slave server can freely access each other's resources. Then, the computing of bid data can be performed on each other's platform. Figure 9 shows the remote connection between the host server and the

Table 1. Linux command code for SSH network configuration in host server.

In Host Server	
generation of ssh key	root@client:~# ssh-keygen
confirmation of ssh key	root@client:~# ls －al ~/.ssh
firewall set-up	root@client:~# chmod 700 ~/.ssh root@client:~# chmod 600 ~/.ssh/id_rsa root@client:~# chmod 644 ~/.ssh/id_rsa.pub root@client:~# chmod 644 ~/.ssh/authorized_keys root@client:~# chmod 644 ~/.sshknown_hosts
copy ssh public key to Slave Server	root@client:~# scp~/.ssh/id_rsa.pub root@slave:id_rsa.pub

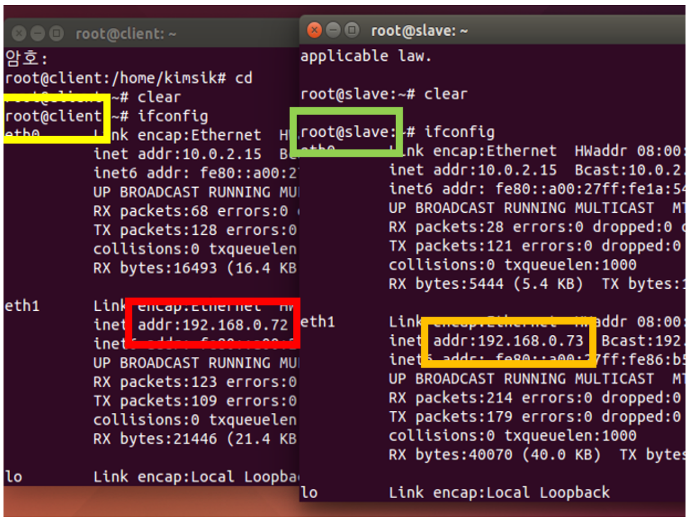
Table 2. Linux command code for SSH network configuration in slave server.

In Slave Server	
move ssh public key to ".ssh" dirictory	root@slave:~# cat id_rsa.pub >> ~/.ssh/authorized_keys
firewall set-up	root@slave:~# chmod 700 ~/.ssh root@slave:~# chmod 644 ~/.ssh/authorized_keys



```
kimsik@client:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/kimsik/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/kimsik/.ssh/id_rsa.
Your public key has been saved in /home/kimsik/.ssh/id_rsa.pub.
The key fingerprint is:
1f:70:3b:00:a6:90:5f:71:6c:05:f0:d5:91:34:a6:0f kimsik@client
The key's randomart image is:
+--[ RSA 2048]----+
|  ..   =+oooo=o  |
|  .. o.+o. oo.   |
|   ... .+ E      |
|    .    + +     |
|        S + .    |
|         . o     |
|          .      |
|                 |
|                 |
+-----------------+
kimsik@client:~$
```

Figure 8. Generations of SSH key.



```
암호:
root@client:/home/kimsik# cd
root@client:~# clear
root@client:~# ifconfig
eth0      Link encap:Ethernet  H
          inet addr:10.0.2.15  B
          inet6 addr: fe80::a00:2
          UP BROADCAST RUNNING MU
          RX packets:68 errors:0
          TX packets:128 errors:0
          collisions:0 txqueuelen
          RX bytes:16493 (16.4 KB

eth1      Link encap:Ethernet  H
          inet addr:192.168.0.72
          inet6 addr: fe80::a00:
          UP BROADCAST RUNNING MU
          RX packets:123 errors:0
          TX packets:109 errors:0
          collisions:0 txqueuelen
          RX bytes:21446 (21.4 KB

lo        Link encap:Local Loopba
```

```
applicable law.

root@slave:~# clear
root@slave:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:
          inet addr:10.0.2.15  Bcast:10.0.2.
          inet6 addr: fe80::a00:27ff:fe1a:54
          UP BROADCAST RUNNING MULTICAST  MT
          RX packets:28 errors:0 dropped:0 o
          TX packets:121 errors:0 dropped:0
          collisions:0 txqueuelen:1000
          RX bytes:5444 (5.4 KB)  TX bytes:1

eth1      Link encap:Ethernet  HWaddr 08:00:
          inet addr:192.168.0.73  Bcast:192.
          inet6 addr: fe80::a00:27ff:fe86:b5
          UP BROADCAST RUNNING MULTICAST  MT
          RX packets:214 errors:0 dropped:0
          TX packets:179 errors:0 dropped:0
          collisions:0 txqueuelen:1000
          RX bytes:40070 (40.0 KB)  TX bytes

lo        Link encap:Local Loopback
```

Figure 9. Remote access to virtual server.

slave server. It is shown that the IP address is different.

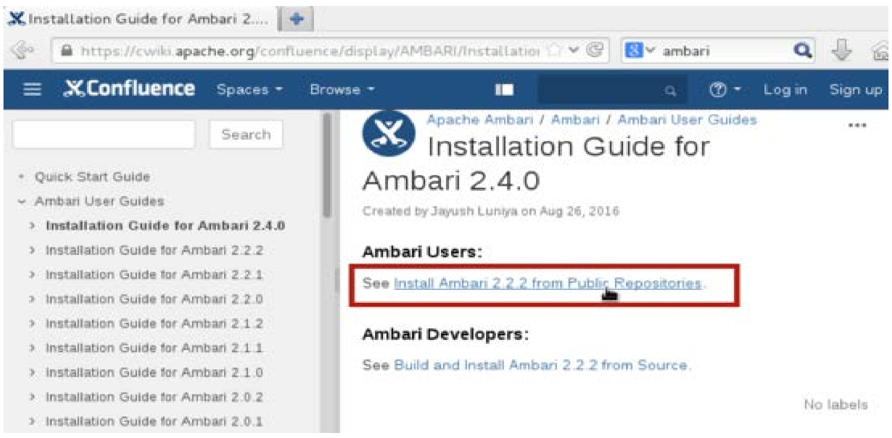
5. Creation of Hadoop Cluster on NVIDIA AI Platform

This section explains the building of a Hadoop cluster (distributed environment) on the AI platform. The framework called "Apache Ambari" has been used to build a Hadoop cluster.

5.1. Creation of Hadoop Cluster

Asone can see in Figure 10, Apache Ambari UI can be easily installed in Linux environment. After that, the NVIDIAAI platform (Client.com) is connected and the slave server (Slave.com) to the Hadoop cluster using the Ambari framework is constructed. In this way, we have created an environment for analyzing not only machine learning algorithms but also big data on the same platform. The Ambari allows you to install SPARK-like SWs in Hadoop clusters. You can also install and uninstall the SW after the Hadoop cluster is completed.

Figure 11 shows the completion of Hadoop Cluster and the completed inte-


Installation Guide for Ambari 2....
https://cwiki.apache.org/confluence/display/AMBARI/Installatio
ambari
Confluence
Spaces
Browse
Log in
Sign up
Search

Quick Start Guide
Ambari User Guides
Installation Guide for Ambari 2.4.0
Installation Guide for Ambari 2.2.2
Installation Guide for Ambari 2.2.1
Installation Guide for Ambari 2.2.0
Installation Guide for Ambari 2.1.2
Installation Guide for Ambari 2.1.1
Installation Guide for Ambari 2.1.0
Installation Guide for Ambari 2.0.2
Installation Guide for Ambari 2.0.1

Apache Ambari / Ambari / Ambari User Guides

Installation Guide for Ambari 2.4.0

Created by Jayush Luniya on Aug 26, 2016

Ambari Users:

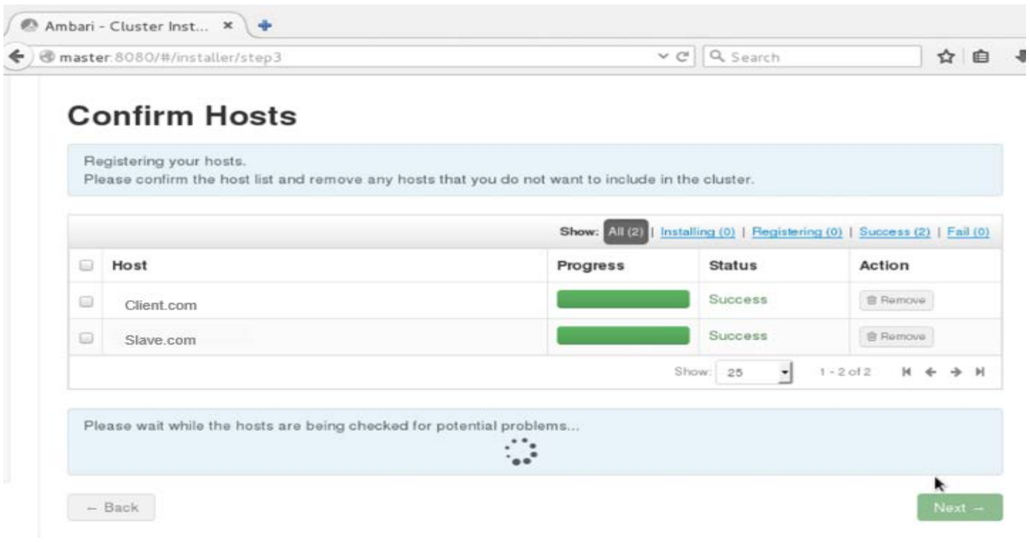
See Install Ambari 2.2.2 from Public Repositories.

Ambari Developers:

See Build and Install Ambari 2.2.2 from Source.

No labels

Figure 10. Apache Ambari installation guide.



Confirm Hosts

Registering your hosts.
Please confirm the host list and remove any hosts that you do not want to include in the cluster.

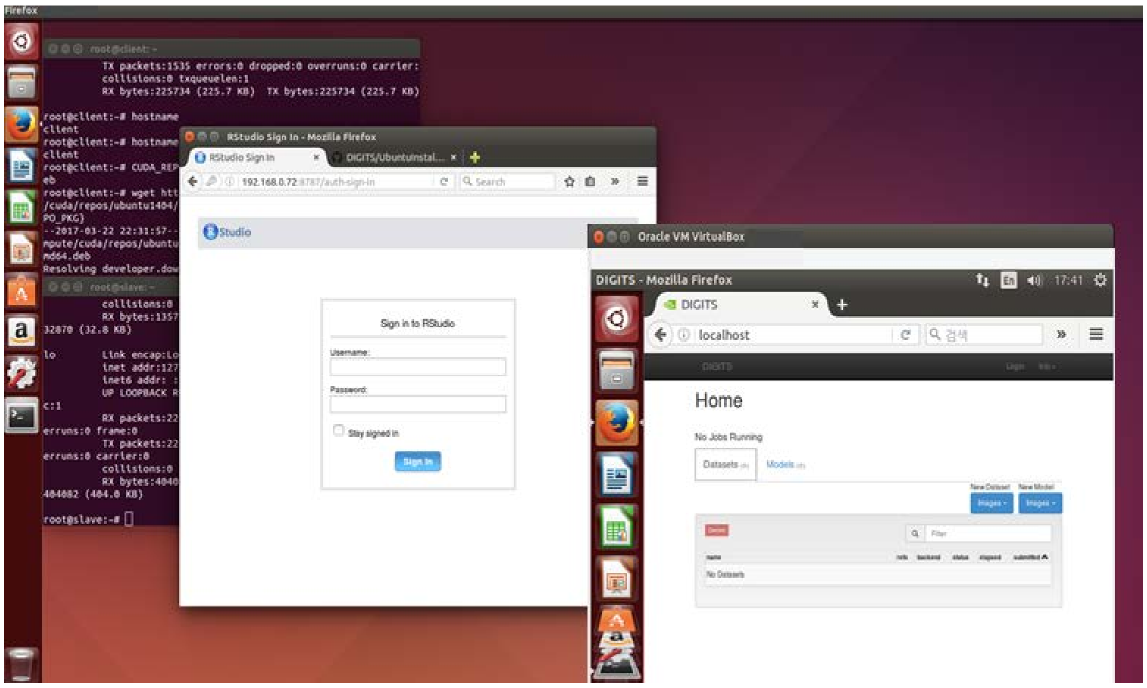
Show: All (2) | Installing (0) | Registering (0) | Success (2) | Fail (0)

Host	Progress	Status	Action
Client.com	🟩	Success	Remove
Slave.com	🟩	Success	Remove

Show: 25 1 - 2 of 2

Please wait while the hosts are being checked for potential problems...

Figure 11. Completion of making Hadoop cluster.



```
            TX packets:1535 errors:0 dropped:0 overruns:0 carrier:
            collisions:0 txqueuelen:1
            RX bytes:225734 (225.7 KB)  TX bytes:225734 (225.7 KB)

root@client:~# hostname
client
root@client:~# hostname
client
root@client:~# CUDA_REP
eb
root@client:~# wget htt
/cuda/repos/ubuntu1404/
PO_PKG}
--2017-03-22 22:31:57--
mpute/cuda/repos/ubuntu
md64.deb
Resolving developer.dow
```

```
            collisions:0
            RX bytes:1357
32870 (32.8 KB)

lo          Link encap:Lo
            inet addr:127
            inet6 addr: :
            UP LOOPBACK R
c:1
            RX packets:22
erruns:0 frame:0
            TX packets:22
erruns:0 carrier:0
            collisions:0
            RX bytes:4040
404082 (404.0 KB)

root@slave:~# 
```

Figure 12. Unified platform for AI and big-data processing.

grated platform. The figure describes the completion of distributed computing environment by using Ambari UI. If one wants to use another SW (*i.e.*, SPARK and PIG) to process big data, one can also download them after the Hadoop Cluster is made.

6. Conclusions

In Figure 12, one can see the finished unified platform for AI and big data analytics. The UI represents the Digits SW accessed from the slave server (NVIDIA AI platform). The Digits SW is mainly for image training and machine learning purposes. The UI also represents the R-studio connected to the host server. The R-studio mainly processes the statistical analysis of big data. By creating this platform, the performance of the computing speed and the processing time can be significantly improved, as opposed to the conventional system that has been explained in Figure 1.

Therefore, in this integrated platform, it is possible to process the big data as well as the artificial intelligence algorithms using the same GPU accelerator. The development of this platform maximizes the utilization of the AI platform. Then a high-performance computing environment will improve efficiency. Therefore, it is not necessary to add additional computers for the big data analysis for the information gathering devices, such as drones and autonomous vehicles. This kind of technology will be very useful in the near future, where we expect the introduction of huge amount of autonomous devices.

Acknowledgements

This work was supported by the Ajou University research fund.

References

[1] Kim, J.W., Kim, J.H. and Kim, I. (2015) SPQUSAR: A Large-Scale Qualitative Spatial Reasoner Using Apache Spark. *KIISE Transactions on Computing Practices*, **21**, 774-779. https://doi.org/10.5626/KTCP.2015.21.12.774

[2] Chen, L., Ko, J.H. and Yeo, J.M. (2014) Performance Comparison of DW System Tajo Based on Hadoop and Relational DBMS. *KIPS Tr. Software and Data Eng*, **3**, 349-354 https://doi.org/10.3745/KTSDE.2014.3.9.349

[3] Lim, Y.-H. (1995) ComBiStation: A Computer Platform for a Distributed Multimedia Computing Environment. *Journal of KIISE*, **2**, 160-181.

[4] Lee, W.-H. and Lee, B.-H. (2012) Service Delivery Time Improvement Using HDFS in Desktop Virtualization. *Journal of KIICE*, **16**, 913-921. https://doi.org/10.6109/jkiice.2012.16.5.913

[5] NVIDIA (2017) Jetson. http://www.nvidia.com/object/embedded-systems-dev-kits-modules.html

[6] NVIDIA (2017) DevBox. https://developer.nvidia.com/devbox

[7] NVIDIA (2017) Digits. https://developer.nvidia.com/digits

[8] ORACLE (2016) Virtual Box. https://www.oracle.com/virtualization/virtualbox/index.html

[9] Apache Ambari (2016). https://ambari.apache.org/