Intel® Select Solutoin for BigDL with Apache Spark

Installation Overview with Intel® Xeon® Scalable Processors

Contents

[1、Architecture 2](#_Toc534699444)

[1.1 4 worker Nodes Architecture 2](#_Toc534699445)

[1.2 1 Master Node Architecture 2](#_Toc534699446)

[1.3 Network Infrastructure 2](#_Toc534699447)

[1.4 Cluster Requirements 3](#_Toc534699448)

[1.5 Firmware Settings 3](#_Toc534699449)

[2、OS Preparation 4](#_Toc534699450)

[2.1 Network 4](#_Toc534699451)

[2.2 Firewall&SELinux\* 4](#_Toc534699452)

[2.3 Prepare Data Drives 4](#_Toc534699453)

[2.4 Limit configuration 4](#_Toc534699454)

[2.5 Set hostname map 5](#_Toc534699455)

[3、Open Source Hadoop Installation 5](#_Toc534699456)

[3. 1 Download 5](#_Toc534699457)

[3.1.1 Apache Hadoop 2.7.3 5](#_Toc534699458)

[3.1.2 Apache Spark 2.2.0 5](#_Toc534699459)

[3.1.3 JDK 1.8 5](#_Toc534699460)

[3.2 Environment Configuration (for spark standalone) 5](#_Toc534699461)

[4、BigDL – Inception v1 7](#_Toc534699462)

[4.1 Install BigDL 0.7 7](#_Toc534699463)

[4.2 Data preparation – ImageNet\*-2012 (for Inception v1) 7](#_Toc534699464)

[4.3 Run Inception v1 8](#_Toc534699465)

[4.3.1 Train Inception V1 8](#_Toc534699466)

[4.3.2 Inference Inception V1 9](#_Toc534699467)

[5、BigDL Benchmark Results 11](#_Toc534699468)

# 1、Architecture

The system is comprised of 4 Spark worker nodes(NF5280M5) and 1 master node(NF5280M5). The following section details the architecture for the compute nodes, network infrastructure and storage infrastructure.

## 1.1 4 worker Nodes Architecture

|  |  |
| --- | --- |
| **Hardware**  | **Description** |
| CPU | 2x Intel® Xeon® 8160CPU  |
| Memory | 384 GB (12x 32 GB @ 2666 MHz DDR4 ECC RDIMM)  |
| Boot Drive | 240 GB (1x Intel® SSD DC S4510 Series)  |
| Storage | 1.92TB(1x Intel® SSD DC S4510 Series)  |
| Data Network | 10 GbE  |
| Mgmt Network | 1 GbE  |

## 1.2 1 Master Node Architecture

|  |  |
| --- | --- |
| **Hardware**  | **Description** |
| CPU | 2x Intel® Xeon® 6150CPU  |
| Memory | 384 GB (12x 32 GB @ 2666MHz DDR4 ECC RDIMM)  |
| Boot Drive | 240 GB (1x Intel® SSD DC S4510 Series)  |
| Storage | 1.92TB(1x Intel® SSD DC S4510 Series)  |
| Data Network | 10 GbE  |
| Mgmt Network | 1 GbE  |

## 1.3 Network Infrastructure

|  |  |  |  |
| --- | --- | --- | --- |
|

|  |
| --- |
| Network  |

 |

|  |
| --- |
| 10 Gbps 48x port switch  |

 |
|

|  |
| --- |
| Management  |

 |

|  |
| --- |
| 1 Gbps 48x port switch  |

 |

## 1.4 Cluster Requirements

The following technologies are required to run the benchmark.

|  |  |  |  |
| --- | --- | --- | --- |
|

|  |
| --- |
| **Software**  |

 |

|  |
| --- |
| **Version**  |

 |
| Linux\* Distribution  | Fedora 27 |
| Apache Spark | 2.2.0 |
| Apache Hadoop | 2.7.3 |
| JDK | Oracle jdk 1.8.0 update 191 |
| BigDL  | 0.7 |

inspur testing as of Jan 23, 2019.

Base configuration: One master node: 2 x Intel® Xeon® Platinum 6150 processor, Inspur® NF5280M5, total memory: 192 GB, 12 slots/32 GB/2666 megatransfers per second (MT/s) DDR4 RDIMM, Intel® Hyper-Threading Technology (Intel® HT Technology) disabled, Intel® Turbo Boost Technology enabled; storage (boot): 1 x 240 GB Intel® SSD DC S4510; storage:1x 1.92TB Intel® SSD DC S4510; network devices: 1 x 10 Gb Intel® 82599ES, network speed: 10GbE , OS/software: Fedora 27;

Four worker nodes: 2 x Intel® Xeon® Platinum 8160 processor, Inspur® NF5280M5, total memory: 384 GB, 12 slots/32 GB/2666 megatransfers per second (MT/s) DDR4 RDIMM, Intel® Hyper-Threading Technology (Intel® HT Technology) disabled, Intel® Turbo Boost Technology enabled; storage (boot): 1 x 240 GB Intel® SSD DC S4510; storage:1x 1.92 TB Intel® SSD DC S4510; network devices: 1 x 10 Gb Intel® 82599ES, network speed: 10GbE, OS/software: Fedora 27；

## 1.5 Firmware Settings

Except for HT、Turbo、CPU Power & Performance Policy, the rest are default。

Hyper Threading Technology: Disabled

Turbo Mode: Enabled

CPU Power & Performance Policy: Max Performance

PCIE Hot Plug: Enabled

PCI-E ASPM Support (Global) : Disabled

Uncore Freq Scaling (UFS) : Enabled

SpeedStep(Pstates) : Enabled

Turbo Mode: Enabled

Hardware P-States: Native Mode

EPP Enable: Enabled

Autonomous Core C-State: Disabled

CPU C6 report: Disabled

Enhanced Halt State (C1E) : Disabled

Package C State: C0/C1 State

VMX: Enabled

SMX: Disabled

Hardware Prefetcher: Enabled

Adjacent Cache Prefetch: Enabled

DCU Streamer Prefetcher: Enabled

DCU IP Prefetcher: Enabled

LLC Prefetch: Disabled

DCU Mode:32KB 8Way Without ECC

Extended APIC: Enabled

# 2、OS Preparation

## 2.1 Network

|  |  |  |
| --- | --- | --- |
| Hostname | Mgmt Network | Data Network |
| Master | 10.7.9.161 | 192.168.99.101 |
| Slave1 | 10.7.9.162 | 192.168.99.102 |
| Slave2 | 10.7.9.163 | 192.168.99.103 |
| Slave3 | 10.7.9.164 | 192.168.99.104 |
| Slave4 | 10.7.9.165 | 192.168.99.105 |

## 2.2 Firewall&SELinux\*

Because this test cluster is behind a hardware firewall on a safe network, we can disable firewall and SELinux to simplify configuration.

## 2.3 Prepare Data Drives

Format 1.92T SSD drives with the ext4 and mount to /bigdldata.

## 2.4 Limit configuration

Add the following lines to /etc/security/limits.conf

\* soft memlock unlimited

\* hard memlock unlimited

\* soft nofile 65535

\* hard nofile 65535

\* soft nproc 65535

\* hard nproc 65535

## 2.5 Set hostname map

set the mapping of hostname and Ip in /etc/hosts for all nodes.

192.168.99.101 master

192.168.99.102 slave1

192.168.99.103 slave2

192.168.99.104 slave3

192.168.99.105 slave4

# 3、Open Source Hadoop Installation

## 3. 1 Download

### 3.1.1 Apache Hadoop 2.7.3

Download the Hadoop binary package from http://hadoop.apache.org/releases.html, select the Hadoop 2.7.3 binary package.

### 3.1.2 Apache Spark 2.2.0

Download the binary package from http://spark.apache.org/downloads.html, select the Spark 2.2.0 version and Prebuilt for Apache Hadoop 2.7 or later.

### 3.1.3 JDK 1.8

Download the JDK 1.8 from [http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads- 2133151.html](http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-%202133151.html).

## 3.2 Environment Configuration (for spark standalone)

1. Configure in $HADOOP\_HOME/etc/hadoop/core-site.xml

<configuration>

<property>

<name>fs.defaultFS</name>

<value>hdfs://master:8020</value>

</property>

</configuration>

2. Configure in $HADOOP\_HOME/etc/hadoop/hdfs-site.xml

<configuration>

<property>

<name>dfs.namenode.name.dir</name>

<value>file:///bigdldata/nn</value> <!-- set the path of the dfs folder -->

</property>

<property>

<name>dfs.datanode.data.dir</name>

<value>file:///bigdldata/dn </value> <!-- set the path of the dfs folder -->

</property>

</configuration>

3. Configure the workers’ hostname or IP in

$HADOOP\_HOME/etc/hadoop/slaves and $SPARK\_HOME/conf/slaves

Slave1

Slave2

Slave3

Slave4

4. Configure in $SPARK\_HOME/conf/spark-defaults.conf

spark.authenticate=false

spark.eventLog.enabled=true

spark.shuffle.service.enabled=true

spark.shuffle.service.port=7337

spark.ui.killEnabled=true

spark.eventLog.dir=file:///bigdldata/spark-events

spark.driver.extraLibraryPath=$HADOOP\_HOME/lib/native/

spark.executor.extraLibraryPath=$HADOOP\_HOME/lib/native/

spark.master=spark://master:7077 # set your master’s hostname

5. Configure in ~/.bashrc

export JAVA\_HOME= /usr/local/jdk1.8.0\_191

export HADOOP\_HOME= /home/bigdl/hadoop-2.7.3

export HADOOP\_CONF\_DIR=$HADOOP\_HOME/etc/hadoop/

export SPARK\_HOME= /home/bigdl/spark-2.2.0-bin-hadoop2.7

export PATH=$HADOOP\_HOME/bin/:$JAVA\_HOME/bin:$PATH:$SPARK\_HOME/bin

6. Copy the Hadoop folder, Spark folder, JDK folder and ~/.bashrc to the same directory on each worker and master node

7. source ~/.bashrc on each server

8. Format the namenode on the master node

$HADOOP\_HOME/bin/hadoop namenode –format

9. Start the service on the master node

$HADOOP\_HOME/sbin/start-dfs.sh

$SPARK\_HOME/sbin/start-master.sh

$SPARK\_HOME/sbin/start-slaves.sh

$SPARK\_HOME/sbin/start-history-server.sh

#If you failed to start the history server here, check that you have created the folder spark.eventLog.dir set in spark-defaults.conf

10. Check the environment by spark-submit pi

***Note:*** Specify the arguments according to your environment.

#!/bin/sh

master\_hostname=`master`

executor\_cores=20

worker\_num=4

total\_executor\_cores=$(( ${executor\_cores} \* ${worker\_num} ))

spark-submit --class org.apache.spark.examples.SparkPi \

--master spark://${master\_hostname}:7077 \

--executor-cores ${executor\_cores} \

--total-executor-cores ${total\_executor\_cores} \

--executor-memory 160g \

$SPARK\_HOME/examples/jars/spark-examples\_2.11-2.2.0.jar 1500

# 4、BigDL – Inception v1

## 4.1 Install BigDL 0.7

Download the BigDL 0.6 package through https://bigdl-project.github.io/0.7.0/#release-download/, choose Spark 2.2.0.

mkdir /home/bigdl/bigdl

unzip dist-spark-2.2.0-scala-2.11.8-all-0.7.0-dist.zip -d /home/bigdl/bigdl

## 4.2 Data preparation – ImageNet\*-2012 (for Inception v1)

1. Download the Imagenet-2012 training image data from http://www.image-net.org/challenges/LSVRC/2012/nnoupb/ILSVRC2012\_img\_train.tar

Download the *Imagenet-2012* validation image data from http://www.image-net.org/challenges/LSVRC/2012/nnoupb/ILSVRC2012\_img\_val.tar

2. Transfer the images to a sequence file (only for Imagenet-2012)

***Note:*** Execute the script in the folder where ILSVRC2012\_img\_train.tar and ILSVRC2012\_img\_val.tar are located. Prepare at least 1 TB of space for this part.

mkdir train

mv ILSVRC2012\_img\_train.tar train/

cd train

tar -xvf ILSVRC2012\_img\_train.tar

mv ILSVRC2012\_img\_train.tar ..

find . -name "\*.tar" | while read CLASS\_NAME ; do mkdir -p "${CLASS\_NAME%.tar}"; tar -xvf "${CLASS\_NAME}" -C "${CLASS\_NAME%.tar}"; done

cd ../

mkdir val

mv ILSVRC2012\_img\_val.tar val/

cd val

tar -xvf ILSVRC2012\_img\_val.tar

cat $BIGDL\_HOME/scripts/classes.lst | while read CLASS\_NAME; do mkdir -p ${CLASS\_NAME}; done

cat $BIGDL\_HOME/scripts/img\_class.lst | while read PARAM; do mv ${PARAM/ n[0-9]\*/} ${PARAM/ILSVRC\*JPEG /}; done

mv ILSVRC2012\_img\_val.tar ..

cd ..

rm -f train/\*.tar

mkdir sequence

spark-submit --class com.intel.analytics.bigdl.models.utils.ImageNetSeqFileGenerator $BIGDL\_HOME/lib/bigdl-\*-jar-with-dependencies.jar -f . -o ./sequence/ -p 20

3. check the sequence file to hdfs

hadoop fs –ls /user/bigdl/sequence

## 4.3 Run Inception v1

### 4.3.1 Train Inception V1

Note：Executor number is 4 and executor cores number is 48

1. Execute the following script

#!/bin/sh

BIGDL\_HOME=/home/bigdl/bigdl # set your bigdl home

JAR=${BIGDL\_HOME}/lib/bigdl-SPARK\_2.2-0.7.0-jar-with-dependencies.jar # update the jar’s name if needed

CLASS=com.intel.analytics.bigdl.models.inception.TrainInceptionV1

check\_point\_folder=/bigdldata/ckp/inceptionv1 # set the checkpoint folder

CORES=48 # set the core number

MASTER="spark://master:7077" # set the spark mode

EXECUTORS=4 # set the executors number, 10-node

DRIVER\_MEM=60G # set the driver memory

EXE\_MEM=300G # set the executor memory

TOTAL\_CORES=$((${CORES} \* ${EXECUTORS}))

MULTIPLIER=4

BATCH\_SIZE=$(( ${TOTAL\_CORES} \* ${MULTIPLIER}))

LEARNING\_RATE=0.0896

DATA=hdfs:/// user/bigdl/sequence/ #

if [ ! -d ${check\_point\_folder} ]; then

echo "check point dir does not exist, make it..."

mkdir -p ${check\_point\_folder}

fi

spark-submit --class ${CLASS} \

--master ${MASTER} \

--executor-cores=${CORES} \

--total-executor-cores=${TOTAL\_CORES} \

--driver-memory=${DRIVER\_MEM} \

--executor-memory=${EXE\_MEM} \

${JAR} \

--batchSize ${BATCH\_SIZE} \

--learningRate ${LEARNING\_RATE} \

-f ${DATA} \

--checkpoint ${check\_point\_folder}

Result：

use following command to get an average training throughput:

grep Throughput bigdl.log | awk '{sum+=$20} END {print sum/NR}'

448.9

use following command to get an average training Accuracy:

grep Top5Accuracy bigdl.log

2019-01-24 05:20:19 INFO DistriOptimizer$:152 - [Epoch 38 188928/1281167][Iteration 61999][Wall Clock 127084.854223706s] Top5Accuracy is Accuracy(correct: 219055, count: 249999, accuracy: 0.8762235048940196)

### 4.3.2 Inference Inception V1

Enlarge val set to fit the cluster size, it is recommended making 8 copies of val set, that is, 100K validation images per node.

#!/bin/sh

BIGDL\_HOME=/home/bigdl/bigdl # set your bigdl home

JAR=${BIGDL\_HOME}/lib/ bigdl-SPARK\_2.2-0.7.0-jar-with-dependencies.jar # update the jar’s name if needed

CLASS=com.intel.analytics.bigdl.models.inception.Test

CORES=48 # set the core number

MASTER="spark://master:7077" # set the spark mode

EXECUTORS=4 # set the executors number

DRIVER\_MEM=60G # set the driver memory

EXE\_MEM=300G # set the executor memory

TOTAL\_CORES=$((${CORES} \* ${EXECUTORS}))

MULTIPLIER=4

BATCH\_SIZE=$(( ${TOTAL\_CORES} \* ${MULTIPLIER}))

DATA=hdfs:///user/bigdl/sequence/val # set your val data path

MODEL\_FILE=/bigdldata/ckp/inceptionv1/20190122\_175822/model.62000# set path of model file in checkpoint folder

start=`date +%s.%3N`

spark-submit --class ${CLASS} \

--master ${MASTER} \

--executor-cores=${CORES} \

--total-executor-cores=${TOTAL\_CORES} \

--driver-memory=${DRIVER\_MEM} \

--executor-memory=${EXE\_MEM} \

${JAR} \

--batchSize ${BATCH\_SIZE} \

-f ${DATA} \

--model ${MODEL\_FILE} | tee -a inference.log

end=`date +%s.%3N`

duration=`echo "scale=2; $end - $start" | bc`

image\_num=`grep Accuracy inference.log | tail -1 | awk -F ' |,' '{print $7}'`

echo "Inference ${image\_num} images, time is ${duration} s, inference throughput is `echo "scale=2; ${image\_num} / ${duration}" | bc` images/sec." | tee -a inference.log

result：

cat inference.log

Top1Accuracy is Accuracy(correct: 302383, count: 449999, accuracy: 0.6719637154749233)

Top5Accuracy is Accuracy(correct: 394331, count: 449999, accuracy: 0.8762930584290187)

Inference 449999 images, time is 339.136 s, inference throughput is 1326.89 images/sec.

5、BigDL Benchmark Results

This Apache Spark BigDL Solution was setup and tested with the GoogleNet application and ImageNet-2012 datasets. The Inception v1 training for GoogLeNet is performed on 1 master node plus 4 worker nodes configuration. The number of iterations is maintained at 62K, which is BigDL’s default configuration. The Batch size is 768.

The average throughput is 448.9 images/sec ,the Top5 Accuracy is 87.6%,the max throughput is above 500 images/sec during training.

The inference throughput is 1337.69 images/sec，Inference 449999 images, time is 339.136 s.