软件栈

su huihuang

2021

软件栈

目录

[软件栈 1](#_Toc95813907)

[1. 目录规划 3](#_Toc95813908)

[1.1. 目录说明 3](#_Toc95813909)

[1.2. 软件版本 3](#_Toc95813910)

[2. 一键安装说明 4](#_Toc95813911)

[2.1. 功能说明 4](#_Toc95813912)

[3. 系统源 5](#_Toc95813913)

[3.1. nginx 5](#_Toc95813914)

[3.2. 配置源 5](#_Toc95813915)

[3.3. 客户端的配置文件 5](#_Toc95813916)

[4. 深度学习框架 5](#_Toc95813917)

[4.1. Anaconda 3 6](#_Toc95813918)

[4.2. TensorFlow 1.13 6](#_Toc95813919)

[4.3. PyTorch 1.7.0 7](#_Toc95813920)

[5. 智能加速卡 9](#_Toc95813921)

[5.1. B卡驱动安装 9](#_Toc95813922)

[5.2. caffe 模型转换 11](#_Toc95813923)

[6. 事件分析系统 12](#_Toc95813924)

[6.1. 数据库安装配置 12](#_Toc95813925)

[6.2. 预警模块配置 12](#_Toc95813926)

[6.3. JDK 1.8 12](#_Toc95813927)

[6.4. elasticsearch 5.6.1 13](#_Toc95813928)

[6.5. apache-tomcat 7.0.47 14](#_Toc95813929)

[6.6. 后端配置 15](#_Toc95813930)

[7. 图数据库 16](#_Toc95813931)

[7.1. neo4j-community 3.4.17 16](#_Toc95813932)

[7.2. Janusgraph 0.4 16](#_Toc95813933)

[8. 大数据体系 17](#_Toc95813934)

[8.1. Zookeeper 17](#_Toc95813935)

[8.1.1. 解压安装 17](#_Toc95813936)

[8.1.2. 修改配置文件 17](#_Toc95813937)

[8.1.3. 添加 myid 17](#_Toc95813938)

[8.1.4. 安装包分发并修改 myid 的值 18](#_Toc95813939)

[8.1.5. 三台机器启动 zookeeper 服务 18](#_Toc95813940)

[8.2. Hadoop 19](#_Toc95813941)

[8.2.1 安装前准备 19](#_Toc95813942)

[8.2.2 安装方法： 19](#_Toc95813943)

[8.2.3 Hadoop 启动方式： 19](#_Toc95813944)

[8.2.4 Hadoop 配置文件 20](#_Toc95813945)

[8.3. Scala 22](#_Toc95813946)

[8.3.1 安装 Scala： 22](#_Toc95813947)

[8.4. Spark 22](#_Toc95813948)

[8.4.1. spark 安装 22](#_Toc95813949)

[8.4.2. spark 集群启动方式 23](#_Toc95813950)

[8.4.3. 访问spark的 webUI 界面 23](#_Toc95813951)

[8.4.4. Spark 配置文件 24](#_Toc95813952)

# 目录规划

## 目录说明

|  |  |  |
| --- | --- | --- |
| 序号 | 目录 | 说明 |
| 1 | /app/soft | 所有软件目录 |
| 2 | /app/conf | 配置文件目录 |
| 3 | /app/public | 公共资料安装目录 |
| 4 | /app/script | 脚本目录 |
| 5 | /app/mirrors | 系统源目录 |
| 6 | /app/biaozhu | 事件系统目录 |

## 软件版本

|  |  |  |
| --- | --- | --- |
| 序号 | 软件名称 | 版本 |
| 1 | Nginx | 1.18 |
| 2 | JDK | 1.8.0\_27 |
| 3 | Mysql | 5.7 |
| 4 | Conda (Anaconda3) | 4.5.12 |
| 5 | Python | 3.7 |
| 6 | Python (test) | 3.6 |
| 7 | TensorFlow (python 3.6 CPU ) | 1.13 |
| 8 | PyTorch (python 3.6 CPU) | 1.7.0 |
| 9 | Elasticsearch | 5.6.1 |
| 10 | Kibana | 5.6.1 |
| 11 | neo4j-community | 3.4.17 |
| 12 | Janusgraph | 0.4 |
| 13 | Caffe-ssd |  |
| 14 | Hadoop |  |
| 15 | Slaca |  |
| 16 | spark |  |

# 一键安装说明

一键安装，为了更安全的操作，所有的操作都以普通用户（kylin）执行，脚本会对系统进行检测当前用户，如果用户不存在，创建用户；如果用户存在，则切换到kylin用户。

另外，为了标准化，所有的安装文件都安装在 /app 目录下面，所有目录严格按照上面的目录规划。

## 功能说明

使用方法：

cd /app/scripts

./install

Usage: install [anaconda|nginx|jdk|mysql|pytorch|tensorflow]

例如：安装 anaconda

cd /app/scripts

./install anaconda

# 系统源

## nginx

nginx 作为系统源的发布服务器，安装二进制安装包版本。

cd /app/soft

tar xf nginx1.18\_linux\_arm.tar.gz -C /app/public

ln -sf /app/public/nginx1.18/sbin/nginx /usr/local/sbin/nginx

cp /app/conf/nginx/nginx.service /lib/systemd/system/nginx.service

systemctl daemon-reload

systemctl enable nginx.service

systemctl restart nginx.service

## 配置源

解压源包到nginx发布目录

cd /app/mirrors/

tar xf ubuntu-ports.tar.gz -C /app/public/nginx1.18/html/ubuntu18.04

## 客户端的配置文件

请修改对应的IP地址

#deb file:///arm\_mirror/ubuntu-ports bionic main universe multiverse restricted

deb http://IP/ubuntu1804 bionic main universe multiverse restricted

# 深度学习框架

深度学习框架，tensoflow 1.13 和 pytorch 1.7.0 都是以python 3.6 环境下编译的CPU版本。

## Anaconda 3

我们采用的是移植 anaconda打包，因为我们把需要的库文件和依赖都安装好。如果你有需要安装全新的anaconda，我们也提供了基于ARM的 anaconda安装包（/app/soft/Archiconda3-0.2.2-Linux-aarch64.sh）。

举粟子：

激活python 3.6 环境：

source /app/public/archiconda3/etc/profile.d/conda.sh

conda activate test

注意：

如果与我的目录不相同，需要修改命令和第一行声明，把你自己的路径写上，如下：

#!/app/public/archiconda3/bin/python

修改成如下：

#!/你的安装路径/archiconda3/bin/python

## TensorFlow 1.13

TensorFlow 使用的是Python 3.6 CPU版本，只需要激活python 3.6 环境，然后引用 tensorflow 即可。具体操作如下：

tensorflow 位置：

/app/public/tensorflow1.13.1-py36

激活 tensorflow 环境：

cd /app/scripts

source start\_tensorflow.sh

激活环境脚本示意：

source /app/public/archiconda3/etc/profile.d/conda.sh

conda activate test

export PYTHONPATH=$PYTHONPATH:/app/public/tensorflow1.13.1-py36/lib/python3.6/site-packages/

## PyTorch 1.7.0

PyTorch 与TensorFlow一样，使用的是 python 3.6 CPU版本，只需要激活 python 3.6环境，然后引用 pytorch 即可。具体操作如下：

pytorch 位置：

/app/public/pytorch1.7.0-py36

激活 pytorch 环境：

Cd /app/scripts

source start\_pytorch.sh

激活环境脚本示意：

cat /app/scripts/start\_pytorch.sh

source /app/public/archiconda3/etc/profile.d/conda.sh

conda activate test

export PYTHONPATH=$PYTHONPATH:/app/public/pytorch1.7.0-py36/lib/python3.6/site-packages

export LD\_LIBRARY\_PATH=$LD\_LIBRARY\_PATH:/app/public/openblas-FT2000+/lib

FAQ：

1. 库文件版本太低

ImportError: /app/public/archiconda3/envs/test/bin/../lib/libstdc++.so.6: version `GLIBCXX\_3.4.26' not found (required by /app/public/pytorch1.7.0-py36/lib/python3.6/site-packages/torch/lib/libtorch\_python.so)

解决办法：

cp /app/public/pytorch1.7.0-py36/lib/libstdc++.so.6.0.28 /app/public/archiconda3/envs/test/lib/libstdc++.so.6

# 智能加速卡

## B卡驱动安装

B 卡驱动安装之前，请确认B卡是否已经安装。如未安装，驱动将安装不上。

查看B 卡设备：

lspci | grep BM1684

查看驱动是否安装：

ls /dev/bm-sophon\*

lsmod | grep bms

安装驱动：

tar xf bmnnsdk2-bm1684\_v2.0.3.tar.gz -C /app/public

cd bmnnsdk2-bm1684\_v2.0.3/scripts

./install\_driver\_arm\_pcie.sh

# 安装库文件

./install\_lib.sh nntc

以下可以不装，X86上做模型转换需要

# 安装 ufw

source envsetup\_pcie.sh ufw

# 安装 bmnetu

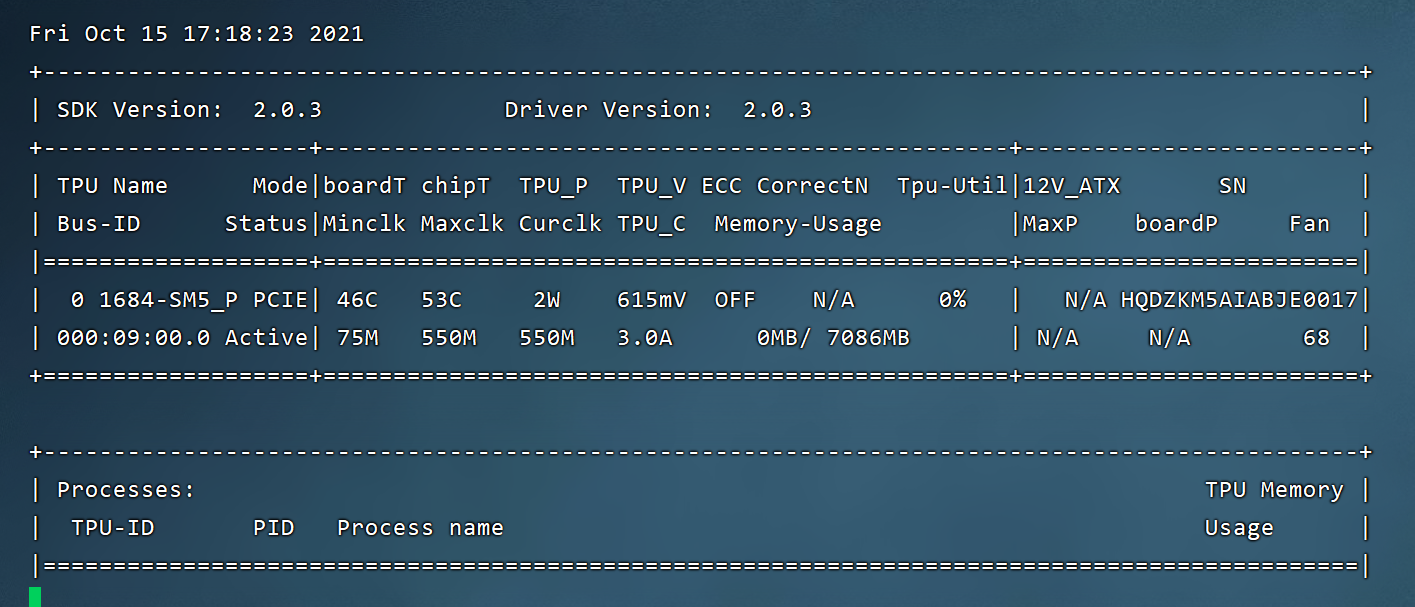
source envsetup\_pcie.sh bmnetu

检查B 卡状态：

cd /app/public/bmnnsdk2-bm1684\_v2.0.3/scripts

source envsetup\_arm\_pcie.sh

bm-smi



返回成功，即安装成功！

查看多节点B 卡的温度：

Yhdo cn[0-15] cat /proc/bmsophon/bmsophon0/chip\_temp

## caffe 模型转换

为什么要将caffe模型移植到咱们的定制卷积加速器上呢？

因为咱们平台上选用了这款加速器，并且它是一款自主可控、能效比高的加速器。这款加速器主要是在推理阶段对模型进行优化并实现高效运行。

而模型量化可以在精度损失微乎其微的的前提下，减少模型参数与计算量，我们的加速器提供了一套完整的量化工具链来实现模型的量化，并且加速器拥有定制计算单元，可以实现量化模型推理阶段的高效运行。

为了更好的适配我们的加速器的硬件设计，我们需要将训练好的caffemodel转换成面向加速器的深度神经网络模型文件。

caffe-ssd 位置：

/app/soft/caffe-ssd.tar.gz

tar xf caffe-ssd.tar.gz -C /app/public/

使用方法：

引入环境变量(source activate\_ft.sh)

cat activate\_ft.sh

export LD\_LIBRARY\_PATH=/app/public/caffe-ssd/build/lib:/usr/local/lib/3libopencv:$LD\_LIBRARY\_PATH

export PYTHONPATH=/app/public/caffe-ssd/python:$PYTHONPATH

# 事件分析系统

## 数据库安装配置

使用系统源来安装数据库：

apt-get install mysql-server mysql-client

在使用系统源安装之前，请先同步时间。

## 预警模块配置

模型所需的模块已安装到 conda 环境中，解压后，直接启动即可。

tar xf action\_prediction.tar.gz -C /app/biaozhu/

cp /app/conf/python/predic.service /lib/systemd/system/

systemctl daemon-reload

systemctl enable predic.service

systemctl start predic.service

## JDK 1.8

只要在arm服务器上需要安装arm版本的JDK。

cd /app/soft

tar xf jdk-8u271-linux-aarch64.tar.gz -C /app/public/

mv /app/public/jdk1.8.0\_271/ /app/public/jdk1.8

sed -i.ori '$a export JAVA\_HOME=/app/public/jdk1.8\nexport PATH=$JAVA\_HOME/bin:$JAVA\_HOME/jre/bin:$PATH\nexport CLASSPATH=.$CLASSPATH:$JAVA\_HOME/lib:$JAVA\_HOME/jre/lib:$JAVA\_HOME/jre/lib:$JAVA\_HOME/lib/tools.jar' /etc/profile

source /etc/profile

## elasticsearch 5.6.1

Elasticsearch 必须要普通用户才能运行，请先确定你使用哪个用户运行elasticsearch。

useradd pdl

cd /app/soft

tar xf elasticsearch-5.6.1.tar.gz -C /app/biaozhu/

chown -R pdl.pdl /app/biaozhu

cp /app/conf/elasticsearch/elasticsearch.service /lib/systemd/system/

systemctl daemon-reload

systemctl enable elasticsearch.service

systemctl start elasticsearch.service

FAQ：

报错日志：

[1]: max file descriptors [4096] for elasticsearch process is too low, increase to at least [65536]

[2]: max virtual memory areas vm.max\_map\_count [26214] is too low, increase to at least [262144]

问题 1 解决办法：

echo “\* soft nofile 65536” >>/etc/security/limits.conf

echo “\* hard nofile 65536” >>/etc/security/limits.conf

问题2 解决办法：

echo “vm.max\_map\_count = 262144” >>/etc/sysctl.conf

sysctl -p

## apache-tomcat 7.0.47

tomcat 用于发布前端应用，只需要修改配置文件IP地址(这里在脚本中替换IP地址)，然后启动即可。

cd /app/soft

tar xf apache-tomcat-7.0.47.tar.gz -C /app/biaozhu/

IP=`ip a|grep enp|grep inet|awk -F '[ /]' '{print $6}'`

sed -i s#10.107.17.70#$IP#g /app/biaozhu/apache-tomcat-7.0.47/webapps/event-anno/resource/httpRequest.js

sed -i s#10.107.17.70#$IP#g /app/biaozhu/apache-tomcat-7.0.47/webapps/event-anno/views/event-wall/app.a9ed35ad.js

chown -R pdl.pdl /app/biaozhu

cp /app/conf/tomcat/tomcat.service /lib/systemd/system/

systemctl daemon-reload

systemctl enable tomcat.service

systemctl start tomcat.service

## 后端配置

后端应用，静态资源在配置文件中写绝对路径，需要注意，所用于依赖于前面的服务，所以必需最后启动。

cp /app/soft/zlxsfs-0.0.1-SNAPSHOT.jar /app/biaozhu/

tar xf annotation.tar.gz -C /app/biaozhu/

cp /app/conf/event/event.service /lib/systemd/system/

cp /app/conf/event/event\_start.sh /app/biaozhu/

systemctl enable event.service

systemctl start event.service

注意： Jar 包的静态资源和服务的启动顺序。

# 图数据库

## neo4j-community 3.4.17

neo4j-community 为社区版本。需要先安装JDK。

neo4j 安装：

cd /app/soft

tar xf neo4j-community-3.4.17-unix.tar.gz -C /app/public/

cd /app/public/neo4j-community-3.4.17/bin

./neo4j start

## Janusgraph 0.4

Janusgraph 目前版本迭代比较快，这里只提供一个比较新的安装包。Janusgraph 是基于java 开发，所有只需要安装JDK即可。对于一些周边工具请查看对应的版本所对应工具集版本号：

<https://github.com/JanusGraph/janusgraph/releases>

Janusgraph 的安装包位置：

/app/soft/janusgraph/

<https://github.com/JanusGraph/janusgraph/tags>

# 大数据体系

## Zookeeper 3.4.5

### 解压安装

cd /app/soft/Hadoop

tar xf zookeeper-3.4.5-cdh5.14.2.tar.gz -C /app/hadoop/

### 修改配置文件

cd /app/hadoop/zookeeper-3.4.5-cdh/conf

cp zoo\_sample.cfg zoo.cfg

mkdir -p /app/hadoop/zookeeper-3.4.5-cdh/zkdatas

vim zoo.cfg

dataDir=/app/hadoop/zookeeper-3.4.5-cdh/zkdatas

autopurge.snapRetainCount=3

autopurge.purgeInterval=1

server.1=node01:2888:3888

server.2=node02:2888:3888

server.3=node03:2888:3888

### 添加 myid

在第一台机器的/app/hadoop/zookeeper-3.4.5-cdh/zkdatas/

这个路径下创建一个文件，文件名为myid ,文件内容为1

echo 1 > /app/Hadoop/zookeeper-3.4.5-cdh/zkdatas/myid

### 安装包分发并修改 myid 的值

安装包分发到其他机器

第一台机器上面执行以下两个命令

scp -r /app/hadoop/zookeeper-3.4.5-cdh/ node02:/app/hadoop/

scp -r /app/hadoop/zookeeper-3.4.5-cdh/ node03:/app/hadoop/

第二台机器上修改myid的值为2

直接在第二台机器任意路径执行以下命令

echo 2 > /app/hadoop/zookeeper-3.4.5-cdh/zkdatas/myid

第三台机器上修改myid的值为3

直接在第三台机器任意路径执行以下命令

echo 3 > /kkb/install/zookeeper-3.4.5-cdh/zkdatas/myid

### 三台机器启动 zookeeper 服务

三台机器启动zookeeper服务

这个命令三台机器都要执行:

/app/hadoop/zookeeper-3.4.5-cdh/bin/zkServer.sh start

查看启动状态

/app/hadoop/zookeeper-3.4.5-cdh/bin/zkServer.sh status

## Hadoop 2.7.7

Hadoop 作为大数据体系基础支撑，是必须安装软件。

### 安装前准备

需要在 /etc/hosts 下做解析，合理规划你的主机，若是有多台做集群，需要先安装好zookeeper。如：

10.10.10.1 hadoop01

10.10.10.2 hadoop02

### 安装方法：

cd /app/scripts/

./install hadoop

安装过程中会安装，JDK，scala，Hadoop。并且已经启动。

### Hadoop 启动方式：

/app/hadoop/hadoop\_2.7.7/sbin/start-all.sh

### Hadoop 配置文件

* core-site.xml

<configuration>

<property>

<name>fs.defaultFS</name>

<value>hdfs://hadoop01:8020</value>

</property>

<property>

<name>io.file.buffer.size</name>

<value>131072</value>

</property>

<property>

<name>hadoop.tmp.dir</name>

<value>/app/hadoop/hadoop\_2.7.7/hadoopDatas/tmpDatas</value>

</property>

</configuration>

* hdfs-site.xml

<configuration>

<property>

<name>dfs.namenode.secondary.http-address</name>

<value>hadoop01:50090</value>

</property>

<property>

<name>dfs.namenode.http-address</name>

<value>hadoop01:50070</value>

</property>

<property>

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

<value>/app/hadoop/hadoop\_2.7.7/hadoopdatas/namenodeDatas</value>

</property>

<property>

<name></name>

<value>/app/hadoop/hadoop\_2.7.7/hadoopdatas/datanodeDatas</value>

</property>

<property>

<name>dfs.replication</name>

<value>2</value>

</property>

</configuration>

* mapred-site.xml

<configuration>

<property>

<name>mapreduce.framework.name</name>

<value>yarn</value>

</property>

<property>

<name>mapreduce.jobhistory.address</name>

<value>hadoop01:10020</value>

</property>

<property>

<name>mapreduce.jobhistory.webapp.address</name>

<value>hadoop01:19888</value>

</property>

</configuration>

* yarn-site.xml

<configuration>

<property>

<name>yarn.nodemanager.aux-services</name>

<value>mapreduce\_shuffle</value>

</property>

<property>

<name>yarn.resourcemanager.hostname</name>

<value>hadoop01</value>

</property>

<!-- <property>

<name>yarn.nodemanager.vmem-pmem-ratio</name>

<value>5</value>

</property> -->

</configuration>

* slaves

hadoop01

* hadoop-env.sh

export JAVA\_HOME=/app/public/jdk1.8.0\_271

## Scala 2.11.8

Scala 需要JDK 的支持，只需要解压即可安装完成。另外在配置环境变量。

### 安装 Scala：

cd /app/soft/Hadoop/

unzip scala-2.11.8.zip -d /app/Hadoop/

ln -s /app/Hadoop/scala-2.11.8/bin/\* /usr/local/bin/

scala -version

## Spark 2.4.3

目前Spark 安装需要 Hadoop 和 zookeeper ,所以安装spark 之前需要先安装 Hadoop 和 zookeeper。

### spark 安装

cd /app/scripts

./install spark

创建spark日志文件夹

hdfs dfs -mkdir -p /spark\_log

### spark 集群启动方式

* 先启动 ZK 和 Hadoop
* 启动 Spark 集群
* 可以在任意一台服务器执行（条件：需要任意二台机器之间实现ssh 免密登录

/app/hadoop/spark-2.4.3/sbin/start-all.sh

/app/hadoop/spark-2.4.3/sbin/start-history-server.sh

* 在哪里启动脚本，就在当前该机器启动一个 Master 进程
* 整个集群的 Worker 进程的启动由 Slaves 文件
* 后期可以在其他机器单独独在启动 Master

/app/hadoop/spark-2.4.3/sbin/start-master.sh

* 停止 spark 集群

在处于 active Master 主节点执行

/app/hadoop/spark-2.4.3/sbin/stop-all.sh

在处于 standby Master 主节点执行

/app/hadoop/spark-2.4.3/sbin/stop-master.sh

### 访问spark的 webUI 界面

* 访问Master主节点web管理界面: <http://IP:8080>
* 访问historyserver 历史任务访问界面：[http://IP:4000](http://hadoop01:4000)

### Spark 配置文件

配置文件目录：/app/hadoop/spark-2.4.3/conf

spark-env.sh

export JAVA\_HOME=/app/public/jdk1.8.0\_271

export SPARK\_HISTORY\_OPTS="-Dspark.history.ui.port=4000 -Dspark.history.retainedApplications=3 -Dspark.history.fs.logDirectory=hdfs://hadoop01:8020/spark\_log"

export SPARK\_DAEMON\_JAVA\_OPTS="-Dspark.deploy.recoveryMode=ZOOKEEPER -Dspark.deploy.zookeeper.url=node01:2181,node02:2181,node03:2181 -Dspark.deploy.zookeeper.dir=/spark"

slaves

hadoop01

hadoop02