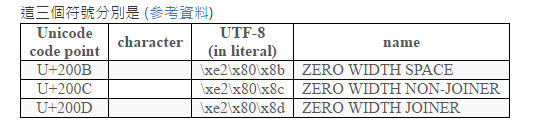
1. **过滤<U+200B><U+200C><U+200D>等不可见字符。**

文本中出现上述不可见字符：



这些字符其实就是排版过程中产生的，而排版使用的规范是Unicode编码标准。



通过如下方法过滤：

sed 's/\xe2\x80\x8b//g' inputfile

sed 's/\xe2\x80\x8c//g' inputfile

sed 's/\xe2\x80\x8d//g' inputfile

<http://www.ruanyifeng.com/blog/2007/10/ascii_unicode_and_utf-8.html>

(**字符编码笔记：ASCII，Unicode 和 UTF-8**)

2. 打开urllib2的调试。有些时候调用接口请求怎么都不对，可以打开urllib2的调试日志，查看发送请求的详细信息来确定问题：

httpHandler = urllib2.HTTPHandler(debuglevel=1)

opener = urllib2.build\_opener(httpHandler)

urllib2.install\_opener(opener)

response = urllib2.urlopen(urllib2.Request(url=base\_url, data=data))

请求详细信息如下：

send: 'POST /ocrimagex HTTP/1.1\r\nAccept-Encoding: identity\r\nContent-Length: 78\r\nHost: libaiocr.jd.local\r\nContent-Type: application/x-www-form-urlencoded\r\nConnection: close\r\nUser-Agent: Python-urllib/2.7\r\n\r\n[{"url":"http://img.aimer.com.cn/product\_pic/AM/AM11EX1/AM11EX1\_t\_1\_1.jpg"}]\r\n'

reply: 'HTTP/1.1 200 OK\r\n'

header: Date: Fri, 10 Aug 2018 09:35:58 GMT

header: Content-Length: 1241

header: Content-Type: text/html; charset=UTF-8

header: Server: TornadoServer/4.5.1

header: Connection: close

3. vi 隔行合并：

1,.g/^/join

使用sed:

sed -n '{N;s/\n//p}' sentences\_embeddings.csv>111

4.通过subshell并行执行命令：

# 通过subshell并行执行，否则太慢了！

for i in $(seq 1 2)

do

if (( $i == 1 )); then

(

# 模型v4

spark-submit --files $HIVE\_CONF\_DIR/hive-site.xml --master yarn --deploy-mode cluster --queue root.bdp\_jmart\_srd\_union.bdp\_jmart\_srd\_formal --driver-memory=25G --num-executors=50 --executor-memory=30G --executor-cores=8 --conf spark.driver.maxResultSize=20G --conf spark.executor.memoryOverhead=5000 --conf spark.network.timeout=3000s --conf spark.executor.heartbeatInterval=1800s --conf spark.sql.broadcastTimeout=1200 --conf spark.executorEnv.yarn.nodemanager.container-executor.class=DockerLinuxContainer --conf spark.yarn.appMasterEnv.yarn.nodemanager.docker-container-executor.image-name=bdp-docker.jd.com:5000/wise\_mart\_srd:latest --conf spark.yarn.appMasterEnv.yarn.nodemanager.container-executor.class=DockerLinuxContainer --conf spark.yarn.appMasterEnv.PYSPARK\_PYTHON=python3 --conf spark.yarn.executorEnv.PYSPARK\_PYTHON=python3 --conf spark.executorEnv.yarn.nodemanager.docker-container-executor.image-name=bdp-docker.jd.com:5000/wise\_mart\_srd:latest task\_ypzj\_generate\_desc\_v4.py app.app\_srd\_jwriter\_ypzj\_candidate\_sentence\_new tmp.app\_srd\_jwriter\_ypzj\_inventory\_desc\_new\_da\_v4 tmp.app\_srd\_jwriter\_ypzj\_inventory\_desc\_new\_da ${BUFFALO\_ENV\_BCYCLE}

)&

fi

if (( $i == 2 )); then

(

# 模型v5

spark-submit --files $HIVE\_CONF\_DIR/hive-site.xml --master yarn --deploy-mode cluster --queue root.bdp\_jmart\_srd\_union.bdp\_jmart\_srd\_formal --driver-memory=25G --num-executors=20 --executor-memory=30G --executor-cores=10 --conf spark.driver.maxResultSize=20G --conf spark.executor.memoryOverhead=10000 --conf spark.network.timeout=3000s --conf spark.executor.heartbeatInterval=1800s --conf spark.sql.broadcastTimeout=1200 --conf spark.executorEnv.yarn.nodemanager.container-executor.class=DockerLinuxContainer --conf spark.yarn.appMasterEnv.yarn.nodemanager.docker-container-executor.image-name=bdp-docker.jd.com:5000/wise\_mart\_srd:latest --conf spark.yarn.appMasterEnv.yarn.nodemanager.container-executor.class=DockerLinuxContainer --conf spark.yarn.appMasterEnv.PYSPARK\_PYTHON=python3 --conf spark.yarn.executorEnv.PYSPARK\_PYTHON=python3 --conf spark.executorEnv.yarn.nodemanager.docker-container-executor.image-name=bdp-docker.jd.com:5000/wise\_mart\_srd:latest task\_ypzj\_generate\_desc\_v5.py app.app\_srd\_jwriter\_ypzj\_candidate\_sentence\_new tmp.app\_srd\_jwriter\_ypzj\_inventory\_desc\_new\_da\_v5 tmp.app\_srd\_jwriter\_ypzj\_inventory\_desc\_new\_da ${BUFFALO\_ENV\_BCYCLE}

)&

fi

done

# 等subshell任务执行完成

wait