KVectors 运维手册

杭州福强科技有限公司

2025-10-23

Contents

KVectors 的安装与部署

KVectors 是 Scala/Java 程序，最简洁的使用方式就是`java -jar kvectors.jar`，因为程序编译打包之后，已经打包成了一个可执行的 fat-jar/one-jar 格式。

但是，假如客户公司有 docker 容器化的需求，也支持以 docker image 的方式安装和部署，下面是一个可供参考的 Dockerfile 定义：

```
# 多阶段构建 Dockerfile
# 阶段1: 构建阶段
FROM ghcr.io/graalvm/graalvm-community:25-ol9 AS builder

# 设置工作目录
WORKDIR /kvectors

# 安装Maven
RUN microdnf install -y maven

# 复制Maven配置文件
COPY pom.xml .

# 下载依赖（利用Docker缓存层）
RUN mvn dependency:go-offline -B

# 复制源代码
COPY src ./src

# 构建应用程序，跳过测试以加快构建速度
RUN mvn clean package -DskipTests

# 阶段2: 运行阶段 - 使用GraalVM完整镜像
FROM ghcr.io/graalvm/graalvm-community:25-ol9

# 创建非root用户
RUN groupadd -r kvectors && useradd -r -g kvectors kvectors

# 设置工作目录
WORKDIR /kvectors

# 从构建阶段复制JAR文件
COPY --from=builder /kvectors/target/*.jar kvectors.jar

# 更改文件所有者
RUN chown -R kvectors:kvectors /kvectors
```

```
37
38 # 切换到非root用户
39 USER kvectors
40
41 # 暴露端口
42 EXPOSE 1980
43
44 # 设置JVM参数
45 ENV JAVA_OPTS="--enable-native-access=ALL-UNNAMED --add-modules
       jdk.incubator.vector"
46
47 # 设置JAR参数（传递给应用程序的参数）
48 ENV JAR_OPTS=""
49
50 # 启动应用程序
51 ENTRYPOINT ["sh", "-c", "java $JAVA_OPTS -jar $JAR_OPTS kvectors.jar
       $@", "--"]
```

这里主要有三个需要重点关注的点：

1. 最好使用 graalvm，因为它可以给 scala 程序带来 30% 的性能提升；（当然，即使使用 OpenJDK，对于程序正常运行也没有任何影响，但推荐使用 graalvm）
2. 以上 Dockerfile 定义采用多阶段构建的定义，构建最简洁的 docker image
3. 开放了 JAVA_OPTS 和 JAR_OPTS，允许用户自定义 java 虚拟机参数以及程序运行参数。

建议部署环境：

1. Java：推荐使用 Java25 LTS 版本。
2. JDK：GraalVM
3. OS（操作系统）：MacOS 或者 Linux (原则上来说，Windows 也可以，但没有进行测试)
4. Memory(内存)：无严格要求，当然最好不小于 2G，一般根据系统负载进行增减，内存越多，KVectors 可以发挥空间越大，尤其是可以获得更低时延 (Latency)。
5. CPU（中央处理器）：建议支持 SIMD 指令的 CPU 型号，不支持也没关系，但没有 SIMD 的性能加成。
6. Disk（磁盘）：建议 SSD（HDD 也支持），存储容量根据应用情况选择就可以了，比如 100 万 1024 维向量，存储空间需至少 4G，供参考以此类推。
7. GPU（显卡/图形处理器）：无强制需求，KVectors 不依赖 GPU 资源。

KVectors 的配置与参数

KVectors 的配置和参数相对较少，用户大部分情况下使用默认值就可以了，对于 java 虚拟机运行期的参数，可以根据通用文档进行配置。

对于 KVectors 应用方面的参数配置，默认配置如下：

```
web.admin.enabled=false

# api endpoints config
api.server.host=localhost
api.server.port=1980
api.server.access.token=...
```

唯一需要配置的参数，原则上只有 api.server.access.token，我们可以在程序启动的时候，通过-Dapi.server.access.token=xxx的方式进行配置：

- 如果选择直接的命令行启动，需要在-jar与jar参数之间指定，比如：java -jar -Dapi.server.access.token=xxx kvectors.jar
- 如果选择 Docker 启动，直接以 JAR_OPTS 的环境变量形式提供即可，比如：docker run ... -e JAR_OPTS="-Dapi.server.access.token=xxx"...

生产环境部署，除了要明确指定api.server.access.token参数（不建议使用默认值，容易造成安全漏洞），还有一个参数是必须（或者强烈建议）的，那就是-Dprofile=production。

-Dprofile=production参数的目的与api.server.access.token类似，都是基于安全考虑，因为有些开发和测试期间为了便利而开放的访问 token，如果在生产环境不屏蔽它们的话，也同样容易造成安全漏洞。

这是 KVectors 在应用一级的配置重点。

再深入一级，就是对 KVectors 中的各个 Vector Collection 进行配置，这个需要根据具体的 Vector Collection 实现类型决定，比如 AnnIndexKVectorCollection，它的构造参数是这样的：

```
class AnnIndexKVectorCollection(val name: String, val repositoryDir:
    File, maxNumOfIndexArchives: Int = 11)
```

只是在构造的时候指定一下 Vector Collection 的 name 就可以了，repositoryDir 会自动配置（虽然这里没有给默认值）。

至于 maxNumOfIndexArchives，更多是指定要保留多少索引的存档。因为 AnnIndexKVectorCollection 支持定期全量构建索引，所以，如果不定期清理，会占用不必要的磁盘空间，所

以，通过这个参数，程序会自动清理不再需要的索引存档。

大部分情况下，AnnIndexKVectorCollection 可以满足 90% 的客户场景需求。

KVectors 的监控与集成

监控无非三个关键途径和指标：

1. logs
2. metrics
3. traces

KVectors 采用 slf4j 和 logback 打印日志，会定期 rollover，客户公司可以根据需要，通过自己的 log agent 本地刮取日志上传到自己的日志分析平台。

KVectors 集成了 dropwizard metrics 和 open telemetry 两种 metrics 注册与统计方式，客户公司可以根据需要选择使用。

使用 open telemetry，只要在 kvectors.jar 启动运行的时候，指定 open telemetry 自带的 javaagent 就可以了，比如：

```
java -javaagent:./opentelemetry-javaagent.jar \
  -Dotel.service.name=my-hybrid-app \
  -Dotel.metrics.exporter=prometheus \
  ...
```

至于 traces，open telemetry 同样有支持。

假如客户公司有需要，后续版本可以再添加更多监控基础设施的支持。

KVectors 的错误与诊断

这也更多依赖前面提到的 logs/metrics/traces 三件套的支持。

大部分情况下，KVectors 经过测试发布之后，不会有太大的错误和 bugs 存在。

客户公司如果自身实在搞不定，我们也提供在线或者线下支持。

KVectors 备份与恢复

KVectors 的备份和恢复目前采用最简单的文件系统备份和恢复策略。

所有的 KVectors 的数据状态默认都存放在`${HOME}/.kvectors`下面

在`${HOME}/.kvectors`下面，又按照 Vector Collection 进行排布，每个 Vector Collection 有自己的数据目录：

- 要全量备份，就直接备份`${HOME}/.kvectors`整个目录（比如用 rsync 或者拷贝到冷热备份设备）
- 要局部备份，就直接备份指定的 Vector Collection 的数据目录就可以了。

KVectors 面向单节点的设计初衷决定了，它可以最简单的方式完成数据的备份与恢复。

如果愿意，也可以备份到云 OSS 对象服务。

> TIP
>
> 可以通过配置文件或者启动参数的形式更改 kvectors 默认的数据目录（dataDir）地址，比如：
>
> ```
> java -jar -Ddata.dir="{{SOME_WHERE_YOU_LIKE}}"kvectors.jar
> ```
>
> 不过，如果更改了默认的数据目录位置，备份和恢复的时候也需要同步使用这个数据目录。

KEEVOL

In Tech We Trust

https://keevol.cn