# 文字查重系统使用说明书

## 版本0.05

## 1. 安装

### 1.1 准备

#### 1.1.1服务器准备

LINUX Ubuntu 14 64位或 WINDOWS均可

#### 1.1.2安装git

##### LINUX

sudo apt-get update

sudo apt-get install git

##### WINDOWS

请参考<https://github.com/progit/progit/blob/master/zh/01-introduction/01-chapter1.markdown>

#### 1.1.3 安装jdk1.8

##### LINUX

$sudo add-apt-repository ppa:webupd8team/java

$ sudo apt-get update

$ sudo apt-get install oracle-java8-installer

##### WINDOWS

<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

#### 1.1.4 安装gradle

##### LINUX

sudo apt-get install gradle

sudo gedit /usr/bin/../share/gradle/bin/gradle

把下面一行注释掉:  
export JAVA\_HOME=....

##### WINDOWS

<http://wenku.baidu.com/view/d109245c6c85ec3a87c2c58f.html>

### 1.2 获取git源码

创建合适的目录,例如 xx然后到该目录下执行下面的命令

git clone <https://github.com/lxh5147/Zeen>

会创建Zeen/PlagiarismChecker,该目录称之为安装目录

### 1.3 使用DOCKER

安装DOCKER请参考相关文档，例如https://docs.docker.com/installation/ubuntulinux/

然后执行：

sudo docker pull lxh5147/plagiarism\_checker

### 2. 索引2.1 导出论文库

新建一个目录，将论文库全文（用PDFBox提取的文本）导出到该目录下。每篇论文对应一个文本文件，文件名规则为：[论文编号].txt。论文编号为整数。该目录称为PDF文本库目录。

PDF文本库目录从而把PDF文本。注意，此时要PDF全文

### 2.2准备论文库

打开终端,执行下面的命令:

gradle buildArticleRepository -PappArgs="['--pdfTextFileFolders', '*pdfTextFileFolder1,pdfTextFileFolder2*','--articleRepositoryFolder', *'articles'*,'--overwrite']"

参数说明：

pdfTextFileFolders：PDF文本库目录目录列表，多个PDF文本库目录目录用英语半角逗号分开。

articleRepositoryFolder': 论文库目录

overwrite: 可选参数，如果指定该参数，则论文库目录下同名的文件将被改写

**注意，用合适的参数替换上述命令行中的红色部分。上述命令执行将对全部PDF文本库目录下的文本文件做段落提取操作，并将提取的结果保存为论文库目录下一个同名但无.txt扩展名的压缩的文本文件。**

### 2.3 建索引

打开终端,执行下面的命令:

gradle buildIndex -PappArgs="['--articleRepositoryFolders', '/home/lxh5147/git/PlagiarismChecker/articles','--contentAnalyzers','SimpleContentAnalizerWithSimpleTokenizer, SegmentContentAnalizerWithSimpleSegmentSplitter ','--indexPath','/home/lxh5147/git/PlagiarismChecker/indexes','--capability','100000','--batchSize','1000','--parallelism','2']"

#### 2.2.1 参数说明

articleRepositoryFolders: 论文库目录列表,多个论文库目录用英文半角逗号分开(”,”),例如’ /home/lxh5147/git/PlagiarismChecker/articles, /home/lxh5147/git/PlagiarismChecker/articles2'.论文库目录可以为绝对路径或者相对路径(相对于执行该命令时所在的目录)

--contentAnalyzers:对段落进行的哪些种类的指纹提取,多个提取操作名称用英文半角逗号隔开.目前支持如下的指纹提取操作,其复杂程度依次递增. 测试阶段建议选取示例所给的两种做测试.

SimpleContentAnalizerWithSimpleTokenizer

SimpleContentAnalizerWithOpenNLPTokenizer

SegmentContentAnalizerWithSimpleSegmentSplitter

ShallowContentAnalizerWithSimpleTokenizer

ShallowContentAnalizerWithOpenNLPTokenizer

BagOfWordsContentAnalizerWithSimpleTokenizer

BagOfWordsContentAnalizerWithOpenNLPTokenizer

关于：SegmentContentAnalizerWithSimpleSegmentSplitter

它将段落切分成句子，为每个句子建一个指纹。如果该句子少于16个单词，将从后续的句子中依次从头取若干单词，凑齐16个。

--indexPath: 指纹索引保存的目录.将在该目录下为每种指纹提取类别,建一个同名的索引文件. 如果该文件已经存在,将被重写(先被清空)

--capability: 和指纹提取种类对应的每个索引最多允许多少索引项.目前每篇论文的每个段落将占用一个索引项,后期一个段落可能占用多个索引项. 如果有100万论文，每篇论文平均100个段落，则该参数应设为 1亿。

--batchSize：批量进行索引的指纹数目，一般可设置为capability的百分之一或千分之一。内存足够的情况下，可酌情增大该参数值。

--parallelism：将启用多少个线程同时对批量传入的指纹进行索引。可将其设为服务器的内核总数-2，例如对8核的服务器，可将值设置为6.

#### 2.3.2 多服务器并行索引

如果有多台服务器，可在每台服务器上同时执行索引操作。

这些服务器执行几乎完全一样的命令，但每个索引论文库的不同部分，例如：

两台服务器A和服务器B:

服务器A:

gradle buildIndex -PappArgs="['--articleRepositoryFolders', '/home/lxh5147/git/PlagiarismChecker/articlesA','--contentAnalyzers','SimpleContentAnalizerWithSimpleTokenizer,BagOfWordsContentAnalizerWithOpenNLPTokenizer','--indexPath','/home/lxh5147/git/PlagiarismChecker/indexesA','--capability','100000','--batchSize','1000','--parallelism','2']"

服务器B:

gradle buildIndex -PappArgs="['--articleRepositoryFolders', '/home/lxh5147/git/PlagiarismChecker/articlesB','--contentAnalyzers','SimpleContentAnalizerWithSimpleTokenizer, SegmentContentAnalizerWithSimpleSegmentSplitter ','--indexPath','/home/lxh5147/git/PlagiarismChecker/indexesB','--capability','100000','--batchSize','1000','--parallelism','2']"

说明，还有其他并行索引的方式，例如不同服务器共享同一个论文库，但执行不同的指纹提取。

## 3. 检测服务

假定已经建好索引，索引目录为/home/lxh5147/git/PlagiarismChecker/indexesA和/home/lxh5147/git/PlagiarismChecker/indexesB。论文库目录为/ home/lxh5147/git/PlagiarismChecker/articlesA和home/lxh5147/git/PlagiarismChecker/articlesB。

打开终端执行下面的命令：

gradle startService -PappArgs="['--articleRepositoryFolders', '/ home/lxh5147/git/PlagiarismChecker/articlesA, / home/lxh5147/git/PlagiarismChecker/articlesB ','--contentAnalyzers','SimpleContentAnalizerWithSimpleTokenizer,BagOfWordsContentAnalizerWithOpenNLPTokenizer','--indexPaths','/home/lxh5147/git/PlagiarismChecker/indexesA, /home/lxh5147/git/PlagiarismChecker/indexesB ']"

这将启动检索服务，检索服务监听8080端口。

参数：

--articleRepositoryFolders：同上。

--contentAnalyzers：同上。将为输入的文本做该参数指定的一个或多个指纹提取，并在对应的指纹库中查找是否有相同的指纹。

--indexPaths：索引目录；多个索引目录用英文半角逗号分开。这些目录下的和指纹提取种类对应的索引将被用以查询

验证服务

打开浏览器，输入：<http://localhost:8080/check?paragraph=some%20test,%20one%20line%20one%20paragraph>.

命令格式为： check?paragraph=…

将返回一个JSON数组，数组的每个元素对应一个疑似抄袭的段落：

[{"articleId":1,"hittedContentAnalizerTypes":["SimpleContentAnalizerWithSimpleTokenizer","BagOfWordsContentAnalizerWithOpenNLPTokenizer"],"paragraphContent":"some test, one line one paragraph.","paragraphId":0}]

字段含义：

articleId: 论文ID

paragraphId:段落ID

paragraphContent：段落文本

hittedContentAnalizerTypes：命中的指纹类别列表

说明：为改变缺省端口8080，可以定义如下的环境变量：

export PlagiarismCheckerServicePortNumber=88888; gradle startService….

## 使用预配置好DOCKER

打开终端执行：

sudo docker run -t -i -p 8080:8080 ubuntu14\_java\_gradle\_checker /bin/bash

cd /Zeen/PlagiarismChecker

上述命令将启动DOCKER容器，为容器打开端口8080，并将容器的端口映射到宿主机器的8080端口。

在容器中执行：

gradle startService -PappArgs="['--articleRepositoryFolders', 'articles','--contentAnalyzers','SimpleContentAnalizerWithSimpleTokenizer,BagOfWordsContentAnalizerWithOpenNLPTokenizer,SegmentContentAnalizerWithSimpleSegmentSplitter','--indexPaths','indexes']"

然后在宿主机器打开浏览器，输入<http://localhost:8080/check?paragraph=some%20test,%20one%20line%20one%20paragraph>.