传统方法进行文本分析

东北大学自然语言处理实验室

今天讲啥？

https://github.com/NLP-lecture/part-1

- part-1
  - 虚拟环境的搭建
  - 文件读取、写入
  - 数据预处理（中英文分词、6种数据清洗方式）
- part-2
  - 利用TF-IDF模型、词袋模型进行关键词提取
  - 词频词典构建
- part-3
  - 文本相似度——最小编辑距离
  - 用fairseq框架从数据开始训练一个翻译模型（GPU may be required）

今天就是在跟文本数据打交道

基础：虚拟环境搭建

https://github.com/NLP-lecture/part-1

- 不用虚拟环境的话，可能遇到一些问题……
  - 使用一些python包往往需要不同的python版本？
  - 很多包需要用到科学计算的numpy，但是不同的包用到的numpy版本也可能不同？
  - 不同包的依赖是冲突的？

我们真的需要虚拟环境！

Anaconda

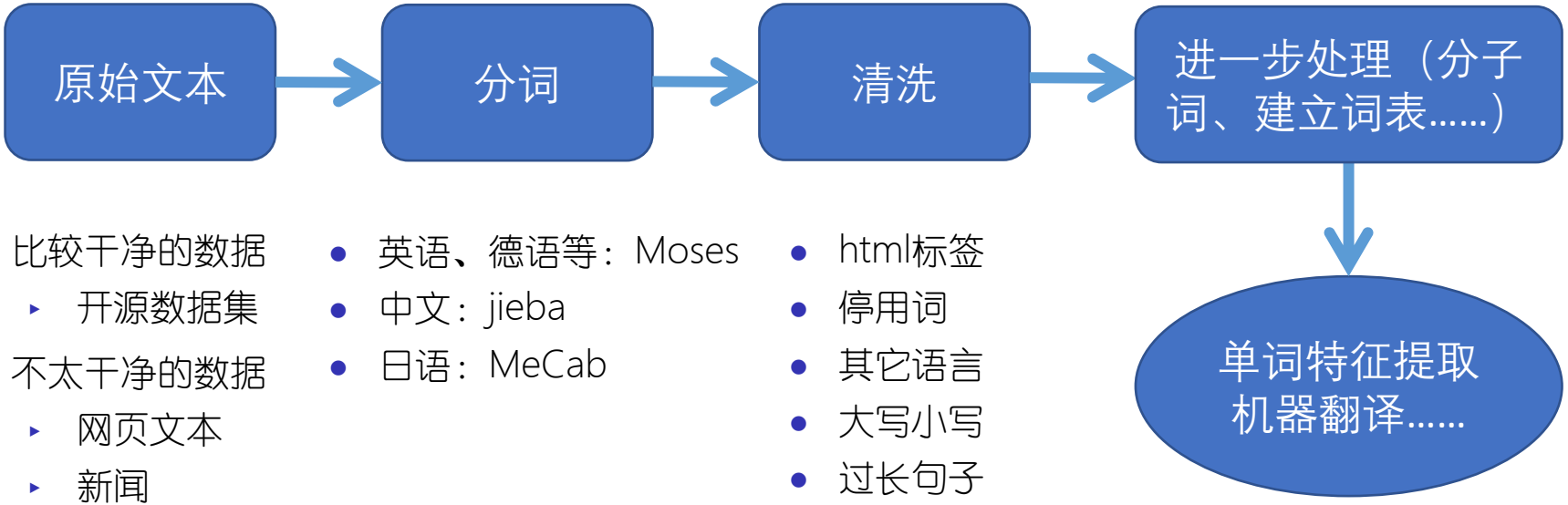
基础：虚拟环境搭建

https://github.com/NLP-lecture/part-1

- 安装anaconda：https://www.jianshu.com/p/edaa744ea47d
- 使用命令：
  - 创建虚拟环境 conda create –n 环境名 python=3.6
  - 进入虚拟环境 conda activate 环境名
  - 安装包 pip install 包名 / conda install 包名
  - 添加清华镜像源：
    https://mirrors.tuna.tsinghua.edu.cn/help/anaconda/
- 本次学习的环境
  - Linux服务器
  - python 3.X

数据处理流程

https://github.com/NLP-lecture/part-1



- 比较干净的数据
  - 开源数据集
- 不太干净的数据
  - 网页文本
  - 新闻
  - 报告

- 英语、德语等：Moses
- 中文：jieba
- 日语：MeCab

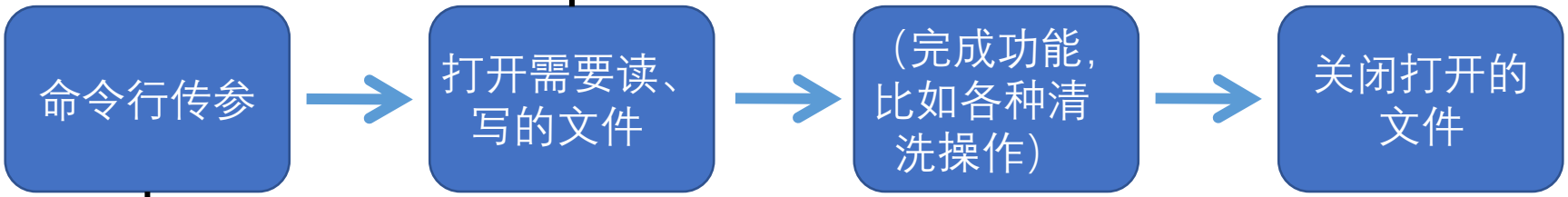
- html标签
- 停用词
- 其它语言
- 大写小写
- 过长句子
- 特殊字符
- ……

文本读取、写入

https://github.com/NLP-lecture/part-1

```
file_name = open(“file_path”, “r/w”, encoding=“utf-8”)
file_name.close()

with open(file_name, 'r', encoding='utf-8') as file_name:
```




```
python test.py readFile.txt writeFile.txt
```

分词：中文分词-jieba

https://github.com/NLP-lecture/part-1

https://github.com/fxsjy/jieba

分词：中文分词-jieba

https://github.com/NLP-lecture/part-1

安装说明

代码对 Python 2/3 均兼容

- 全自动安装：`easy_install jieba` 或者 `pip install jieba` / `pip3 install jieba`
- 半自动安装：先下载 http://pypi.python.org/pypi/jieba/ ，解压后运行 `python setup.py install`
- 手动安装：将 jieba 目录放置于当前目录或者 site-packages 目录
- 通过 `import jieba` 来引用
- 如果需要使用paddle模式下的分词和词性标注功能，请先安装paddlepaddle-tiny， `pip install paddlepaddle-tiny==1.6.1` 。

```
(Lecture) [                                                    ]$ pip install jieba
Looking in indexes: https://mirrors.bfsu.edu.cn/pypi/web/simple
Collecting jieba
  Downloading https://mirrors.bfsu.edu.cn/pypi/web/packages/c6/cb/18eeb235f833b726522d7ebed54f2278ce28ba9438e3135ab0278d9792a2/jieba-0.42.1.tar.gz (19.2 MB)
     |                                | 19.2 MB 5.1 MB/s
Building wheels for collected packages: jieba
  Building wheel for jieba (setup.py) ... done
  Created wheel for jieba: filename=jieba-0.42.1-py3-none-any.whl size=19314476 sha256=bc79814a79a80b4623119e106d1ec8dcded9d06719ae352a78db19505b572cfc
  Stored in directory: /          /.cache/pip/wheels/4a/b6/f8/40a5e7c93827f0b2d618cb48ec18d13c518138d08ad211dda5
Successfully built jieba
Installing collected packages: jieba
Successfully installed jieba-0.42.1
```

分词：中文分词-jieba

https://github.com/NLP-lecture/part-1

特点

- 支持四种分词模式：
  - 精确模式，试图将句子最精确地切开，适合文本分析；
  - 全模式，把句子中所有的可以成词的词语都扫描出来, 速度非常快，但是不能解决歧义；
  - 搜索引擎模式，在精确模式的基础上，对长词再次切分，提高召回率，适合用于搜索引擎分词。
  - paddle模式，利用PaddlePaddle深度学习框架，训练序列标注（双向GRU）网络模型实现分词。同时支持词性标注。paddle模式使用需安装paddlepaddle-tiny，`pip install paddlepaddle-tiny==1.6.1`。目前paddle模式支持jieba v0.40及以上版本。jieba v0.40以下版本，请升级jieba，`pip install jieba --upgrade`。
    PaddlePaddle官网
- 支持繁体分词
- 支持自定义词典
- MIT 授权协议

分词：中文分词-jieba

https://github.com/NLP-lecture/part-1

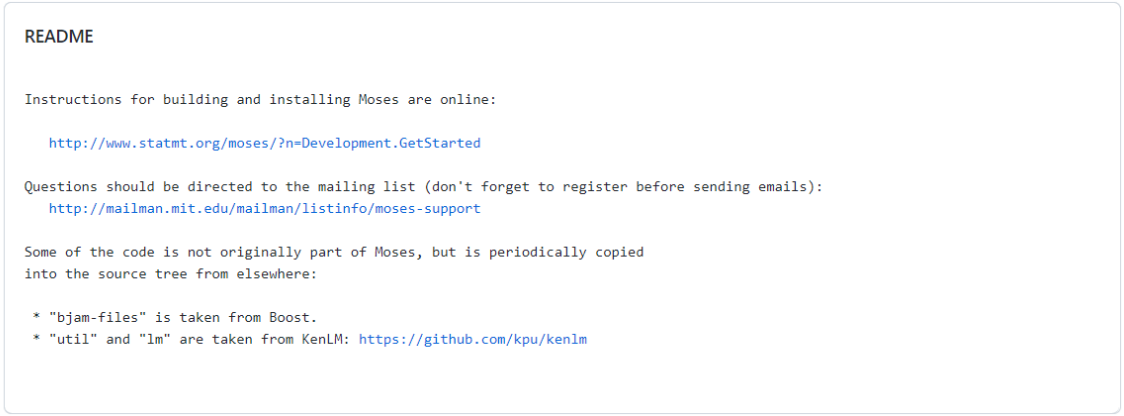
主要功能

1. 分词

- `jieba.cut` 方法接受四个输入参数: 需要分词的字符串；cut_all 参数用来控制是否采用全模式；HMM 参数用来控制是否使用 HMM 模型；use_paddle 参数用来控制是否使用paddle模式下的分词模式，paddle模式采用延迟加载方式，通过enable_paddle接口安装paddlepaddle-tiny，并且import相关代码；
- `jieba.cut_for_search` 方法接受两个参数：需要分词的字符串；是否使用 HMM 模型。该方法适合用于搜索引擎构建倒排索引的分词，粒度比较细
- 待分词的字符串可以是 unicode 或 UTF-8 字符串、GBK 字符串。注意：不建议直接输入 GBK 字符串，可能无法预料地错误解码成 UTF-8
- `jieba.cut` 以及 `jieba.cut_for_search` 返回的结构都是一个可迭代的 generator，可以使用 for 循环来获得分词后得到的每一个词语(unicode)，或者用
- `jieba.lcut` 以及 `jieba.lcut_for_search` 直接返回 list
- `jieba.Tokenizer(dictionary=DEFAULT_DICT)` 新建自定义分词器，可用于同时使用不同词典。`jieba.dt` 为默认分词器，所有全局分词相关函数都是该分词器的映射。

分词：英文分词-Moses

https://github.com/NLP-lecture/part-1



README

Instructions for building and installing Moses are online:

http://www.statmt.org/moses/?n=Development.GetStarted

Questions should be directed to the mailing list (don't forget to register before sending emails):
http://mailman.mit.edu/mailman/listinfo/moses-support

Some of the code is not originally part of Moses, but is periodically copied
into the source tree from elsewhere:

* "bjam-files" is taken from Boost.
* "util" and "lm" are taken from KenLM: https://github.com/kpu/kenlm

https://github.com/moses-smt/mosesdecoder

1、从GitHub上面下载压缩包

2、解压

3、perl ./mosesdecoder-master/scripts/tokenizer/tokenizer.perl -l en -no-escape -threads 6 < train.en > train.en.out

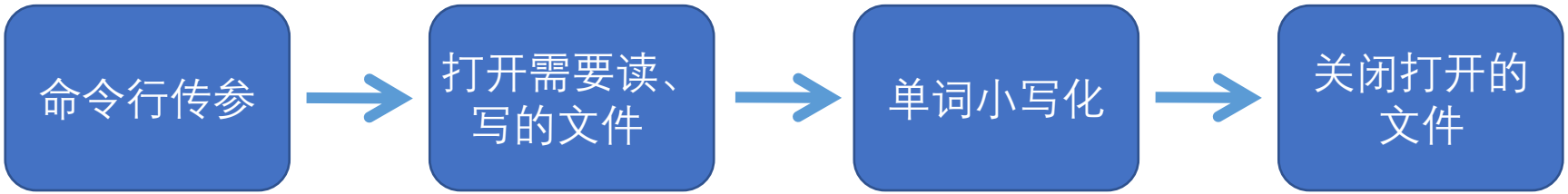
清洗：单词全部小写化

最简单，但是在单词特征提取、机器翻译等任务上都能用上

https://github.com/NLP-lecture/part-1

```
python lowercase.py test_data/lowercase.en test_data/lowercase.en.out
```

line.lower()

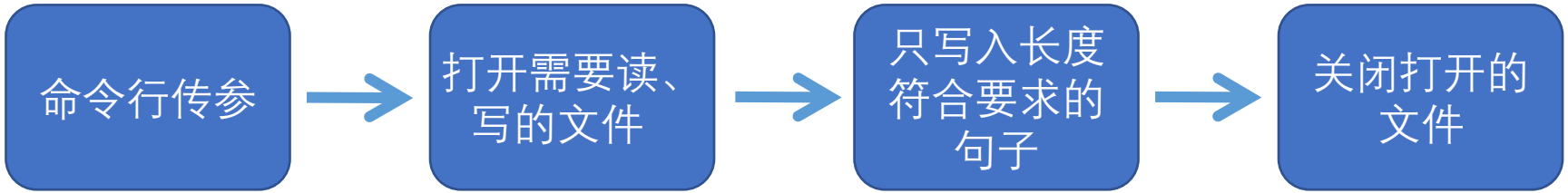


清洗：删除长句子

https://github.com/NLP-lecture/part-1

有的时候数据中可能有非常长的一句话（多句话在一行里，单词数量>120个），这不利于机器翻译等任务的训练，所以需要去掉。程序同样 狠 简单

python remove_long_sentence.py test_data/long_sentence.en test_data/long_sentence.zh test_data/long_sentence.en.out test_data/long_sentence.zh.out

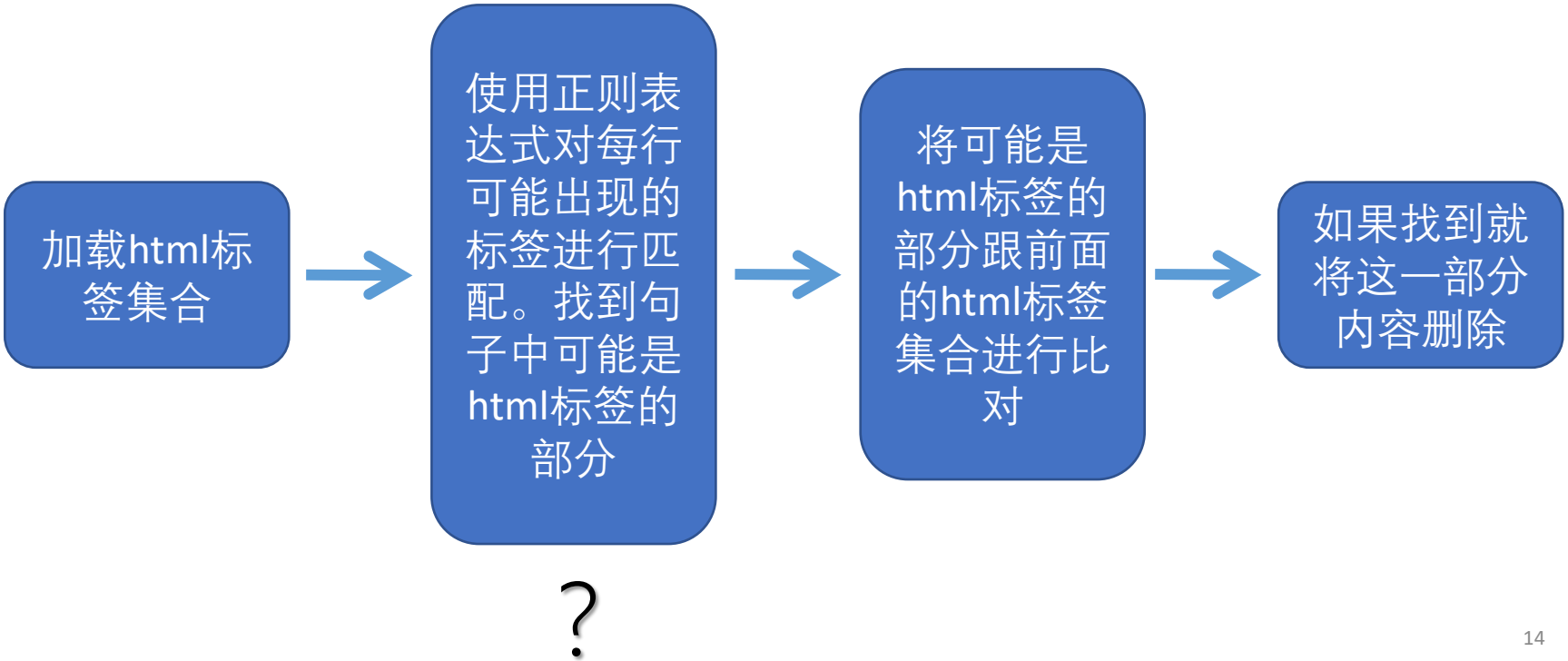


清洗：html标签

从网页爬取的数据可能带有html标签，我们应该如何去除？

https://github.com/NLP-lecture/part-1

python remove_html.py test_data/html.zh test_data/html.zh.out test_data/html.zh.err



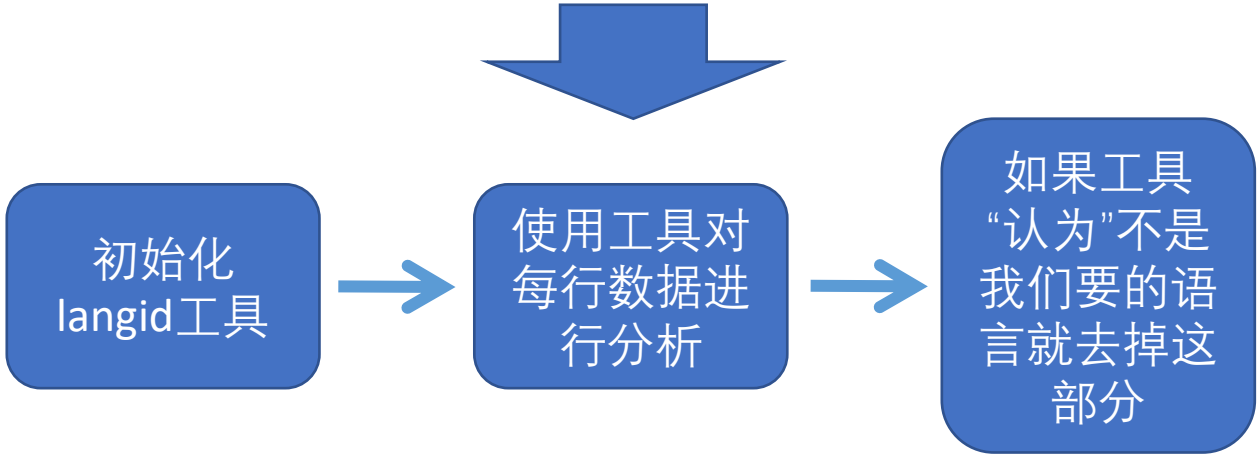
清洗：语言检测

如果数据里面存在一些我们用不到的语言，比如机器翻译英文to中文，但是数据里面出现了跟英语相近的德语、跟中文相近的日语，我们应该如何清洗？

cture/part-1

python language_detect.py test_data/language.en test_data/language.zh test_data/language.en.out test_data/language.zh.out test_data/language.err en zh

如果我们能找到一个识别语言的工具的话……



清洗：语言检测

https://github.com/NLP-lecture/part-1

`langid.py` readme

Introduction

`langid.py` is a standalone Language Identification (LangID) tool.

The design principles are as follows:

1. Fast
2. Pre-trained over a large number of languages (currently 97)
3. Not sensitive to domain-specific features (e.g. HTML/XML markup)
4. Single .py file with minimal dependencies
5. Deployable as a web service

All that is required to run `langid.py` is >= Python 2.7 and numpy. The main script `langid/langid.py` is cross-compatible with both Python2 and Python3, but the accompanying training tools are still Python2-only.

`langid.py` is WSGI-compliant. `langid.py` will use `fapws3` as a web server if available, and default to `wsgiref.simple_server` otherwise.

`langid.py` comes pre-trained on 97 languages (ISO 639-1 codes given):

> af, am, an, ar, as, az, be, bg, bn, br, bs, ca, cs, cy, da, de, dz, el, en, eo, es, et, eu, fa, fi, fo, fr, ga, gl, gu, he, hi, hr, ht, hu, hy, id, is, it, ja, jv, ka, kk, km, kn, ko, ku, ky, la, lb, lo, lt, lv, mg, mk, ml, mn, mr, ms, mt, nb, ne, nl, nn, no, oc, or, pa, pl, ps, pt, qu, ro, ru, rw, se, si, sk, sl, sq, sr, sv, sw, ta, te, th, tl, tr, ug, uk, ur, vi, vo, wa, xh, zh, zu

https://github.com/saffsd/langid.py

清洗：语言检测

https://github.com/NLP-lecture/part-1

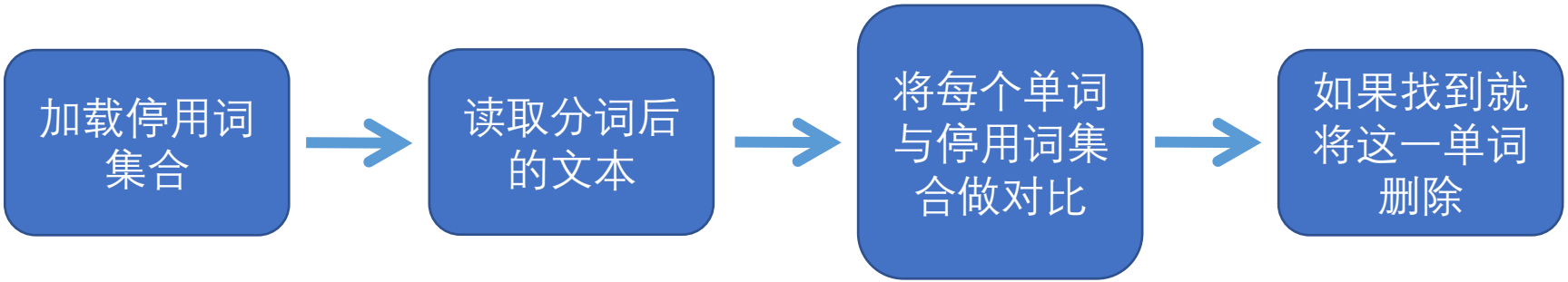
The simplest way to use `langid.py` is as a command-line tool, and you can invoke using `python langid.py`. If you installed `langid.py` as a Python module (e.g. via `pip install langid`), you can invoke `langid` instead of `python langid.py -n` (the two are equivalent). This will cause a prompt to display. Enter text to identify, and hit enter:

```
(Lecture) [redacted]]$ pip install langid
Looking in indexes: https://mirrors.bfsu.edu.cn/pypi/web/simple
Collecting langid
  Downloading https://mirrors.bfsu.edu.cn/pypi/web/packages/ea/4c/0fb7d900d3b0b9c8703be316fbddffecdab23c64e1b46c7a83561d78bd43/langid-1.1.6.tar.gz (1.9 MB)
     |[redacted]| 1.9 MB 5.6 MB/s
Collecting numpy
  Downloading https://mirrors.bfsu.edu.cn/pypi/web/packages/14/32/d3fa649ad7ec0b82737b92fefd3c4dd376b0bb23730715124569f38f3a08/numpy-1.19.5-cp36-cp36m-manylinux2010_x86_64.whl (14.8 MB)
     |[redacted]| 14.8 MB 70.0 MB/s
Building wheels for collected packages: langid
  Building wheel for langid (setup.py) ... done
  Created wheel for langid: filename=langid-1.1.6-py3-none-any.whl size=1941187 sha256=56d98a5dc00592f9c00f9ac556c6602abb034d8021c2fbf1e576547d1d19349c
  Stored in directory: /[redacted]/.cache/pip/wheels/c6/c7/11/a018a3c27df3620f02f75bb1aaf514cf8141f4dd42a7e5825f
Successfully built langid
Installing collected packages: numpy, langid
Successfully installed langid-1.1.6 numpy-1.19.5
```

清洗：删除停用词

https://github.com/NLP-lecture/part-1

停用词：一些对句义几乎没有影响的单词
中文停用词：一下、一些、万一、上下……
主要是单词特征提取、信息检索等任务需要用到，这样会直接去掉这些高频但是无意义词，可以让单词特征提取更加准确、信息检索更快速等
python remove_stop_words.py test_data/stop_words.zh test_data/stop_words.zh.out test_data/stop_words.zh.err



清洗：特殊字符

https://github.com/NLP-lecture/part-1

这一步是需要通过观察数据集，找到问题进行清洗。数据集不同，需要清洗的字符也不同

主要用到 replace()、正则表达式等

remove_special_symbol.py（只作参考，不演示）

还有很多清洗方式……

https://github.com/NLP-lecture/part-1

无用标签（html）、**过长句子**、**乱码**、全大写、**语言检测**、半角变全角、长度比过滤、**停用词过滤**、**英文全小写**

休息一会儿……

https://github.com/NLP-lecture/part-1