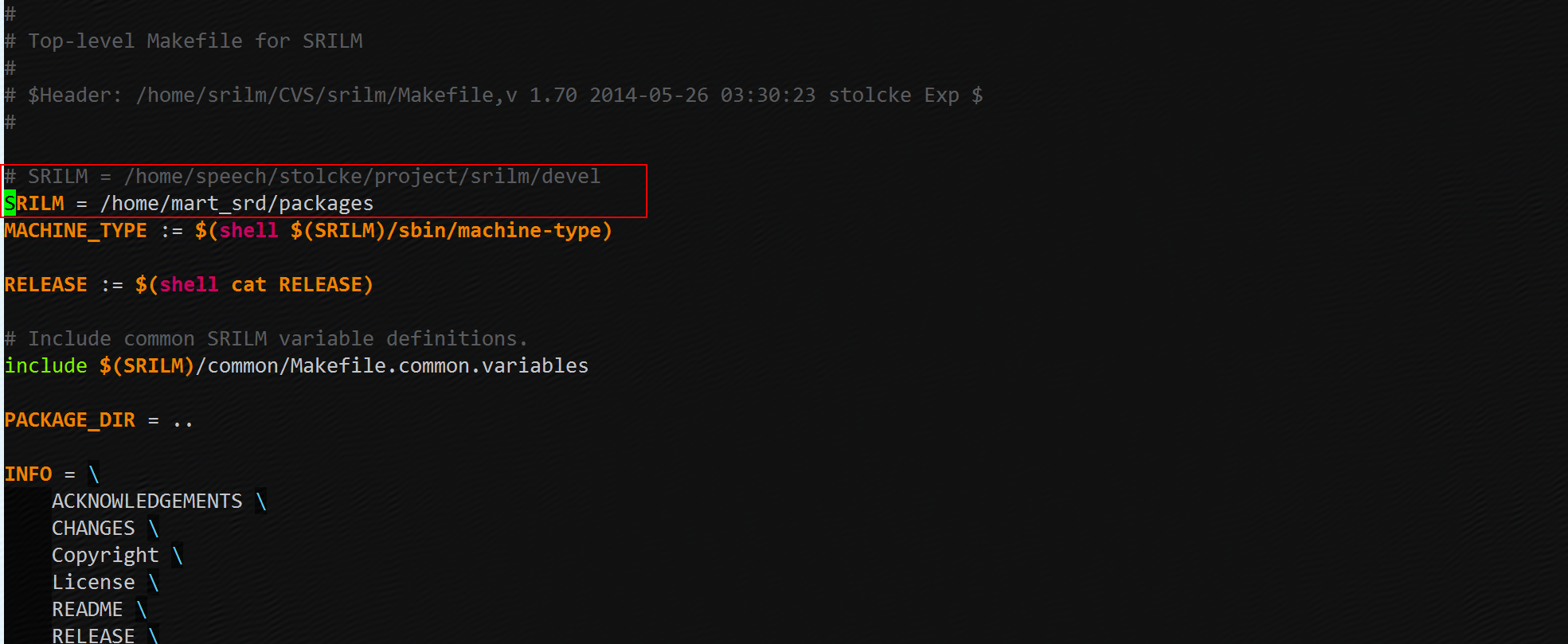
1. 下载安装，srilm，官方只提供源码，因此需要自己下载编译。

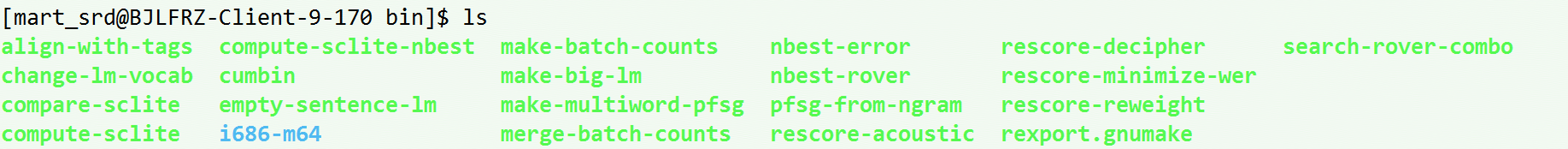
下载地址： <http://www.speech.sri.com/projects/srilm/download.html>

然后解压，修改根目录下的Makefile, 将SRILM变量的值改为源码目录的绝对路劲，如下图：



详情参看根目录下的INSTALL文件。

然后根目录下执行make，等编译完成后在根目录下会生成bin目录，srilm命令都在该目录下。



Ngram相关的命令都在i686-m64目录下，make-big-lm、merge-batch-counts等都是一系列辅助脚本，用来处理大文本的情况。具体情况参见：<http://www.52ml.net/4981.html>

“

**利用srilm构建大语言模型**

原理上，语言模型模型越大，机器翻译质量越好，但是当语言模型的训练集非常大时，例如GB级别的时候，受限于时间和机器的内存等因素，传统的ngram-count训练方式无法满足实际需要，因此srilm的 FAQ 中提到了训练语言模型的方法，基本思想就是将大文件拆分成多个小文件，然后再将多个小文件的count合并，完成最终的语言模型训练。

其基本方法：

1. 把大文件分割成小文件，放在一个目录下，然后生成一个文件名列表文件，如filelist ，一般使用按行分割的形式，split -l 100 test.txt out

使用split将一个大文件分成最多26\*26（使用字母后缀，这是默认的行为）或者是100（使用数字后缀，需要-d参数）个文件，可以将文件按行拆分（使用-l num参数）或者是按大小拆分（使用-b size参数），还可以给出文件的前缀（或者使用默认的x）。在进行拆分的时候将文件会将每num行放到一个文件中，文件按字母序产生，对于语言模型的使用来说，一般使用按行分割的形式 split -l 100 big\_file split\_file

2. 使用 make-batch-counts分别统计各个分割文件中的词频，make-batch-counts filelist 5 cat counts -order 5，其中filelist为需要统计的小文件名列表，5表示每5个小文件用于一次ngram-count训练，cat lmcount 表示输出到counts，后续则是提交给ngram-count的参数

3. 使用merge-batch-counts将所有的小count文件合并成一个大的count文件，merge-batch-counts [ -l N ] counts [ filename-list ]，将counts目录下的所有文件合并成一个文件，如果有些文件不用参与合并，可以在最后添加一个filename-list，只有在filename-list里面出现的文件才会被用于合并；-l N参数之处，一次同时合并N个文件

4. 使用make-big-lm生成语言模型，参数类似于ngram-count

”

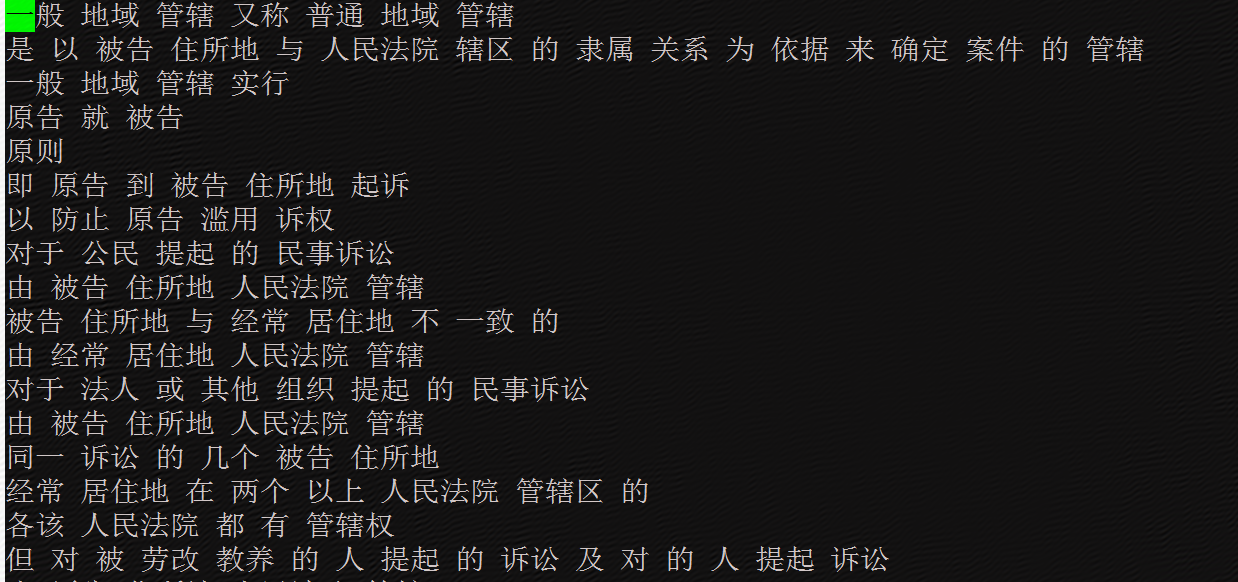
各命令的官方手册在：<http://www.speech.sri.com/projects/srilm/manpages/>

（非常重要，一定要仔细阅读！）

1. 使用Ngram-count统计词频。基本用法：

./ngram-count -no-sos -no-eos -text daren\_article10K.txt -lm daren\_article10K.lm -write daren\_article10K.count

其中 daren\_article10K.txt：输入文件，对于中文要先分词后加空格，如下图：



（我们只保留中文字符，其他英文、数字、符号等都过滤了）

daren\_article10K.lm：生成的模型文件

daren\_article10K.count：生成的词频统计文件

1. 测试困惑度：

ngram -ppl test.txt -lm sku\_description1K.lm -debug 2

test.txt是待测试的样本文件，同样也是分词后的文本。

关于困惑度，参见<https://en.wikipedia.org/wiki/Perplexity>

<http://blog.csdn.net/jiaqiang_ruan/article/details/77989459?locationNum=2&fps=1>

简单说来就是衡量一个概率分布或概率模型对样本预测的有多好，值越低代表预测的越好。