GIZA++

计算语言学 第七次作业

下载统计机器翻译工具包GIZA++（https://github.com/moses-smt/giza-pp）并学习如何使用。
以下面的中文-英文平行语料库（注意需要转成GIZA++规定的格式）作为输入，使用默认参数运行GIZA++，提交生成的**词语对齐结果文件**（后缀名是“*.A3.final”）。

我 喜欢 读书 。 lll i like reading .
我 也 喜欢 音乐 。 lll i also like music .
你 喜欢 读书 。 lll you like reading .
你 喜欢 音乐 吗 ？ lll do you like music ?
你 喜欢 足球 吗 ？ lll do you like football ?

1. 目标

1.1 下载并编译 GIZA++ 和 mkcls，获得所需的可执行文件。

1.2 利用平行双语语料库，通过 mkcls 构建 word classes。

1.3 利用平行双语语料库，通过 GIZA++ 进行 IBM Model 的训练。

2. 实验准备

2.1 GIZA++

GIZA++ 是由 Franz Och 开发的 GIZA 的一个扩展，并包含了许多其他的特点

2.2 mkcls

mkcls 是由 Franz Och 开发的一个训练词语聚类的工具。

2.3 平行双语语料库

汉英平行语料库，汉语语料需要切分，英语语料需要 tokenize

2.4 实验环境

Mac OSX 10.12.6 x64

Apple LLVM version 9.0.0 (clang-900.0.39.2)

3. 实验步骤

3.1 编译GIZA++

从github上clone GIZA++（https://github.com/moses-smt/giza-pp.git）因为GIZA++是在Linux下开发的，而且年代久远，不能被 gcc 4.3 或更高版本编译。需要对代码做一些微调，才能再Mac下编译通过。这些修改是要把tr1的所有调用都去掉。修改如下：

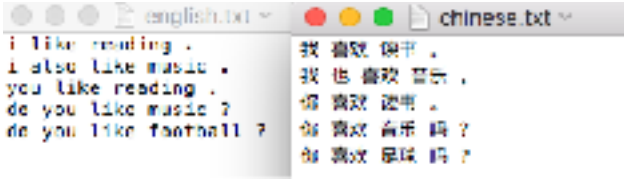
文件	操作
GIZA++-v2/mystl.h	修改#include <tr1/unordered_map>为 #include <unordered_map>
GIZA++-v2/mystl.h	删除using namespace std::tr1
mkcls-v2/myleda.h	修改#include <tr1/unordered_map> 为#include <unordered_map>
mkcls-v2/myleda.h	修改 std::tr1::unordered_map<A,B> 为std::unordered_map<A,B>
mkcls-v2/mystl.h	修改#include <tr1/unordered_map> 为#include <unordered_map>
mkcls-v2/mystl.h	删除namespace tr1 {……}
GIZA++-v2/Makefile	删除 -DBINARY_SEARCH_FOR_TTABLE

在giza-pp目录下通过make命令编译，得到可执行文件：

文件夹	文件
GIZA++-v2/	GIZA++
	plain2snt.out
	snt2cooc.out
	snt2plain.out
mkcls-v2/	mkcls

3.2 语料预处理

将提供的语料分成分好词的chinese.txt和english.txt，这是两个平行语料。文件中每句一行，需要有换行符。



将处理好的chinese.txt和english.txt，和编译GIZA++后生成的5个可执行文件都放入新文件夹。

3.3 构建GIZA++所需的文件

1) **运行命令**

./plain2snt.out chinese.txt english.txt

将普通文本转化为*GIZA++*的格式。生成如下两个文件：

文件	内容	示例
chinese.vcb	1. 单词编号 2. 汉语句子中的单词 3. 单词的出现次数	2 我 2 3 喜欢 5 4 读书 2
english.vcb	1. 单词编号 2. 英语句子中的单词 3. 单词的出现次数	2 i 2 3 like 5 4 reading 2
chinese_english.snt	1. 每个句子对出现的次数 2. 汉语句子中的单词编号 3. 英语句子中的token编号 注：0是保留给特殊的“空”token	1 8 3 11 9 10 9 8 3 11 10
english_chinese.snt	1. 每个句子对出现的次数 2. 英语句子中的token编号 3. 汉语句子中的单词编号 注：0是保留给特殊的“空”token	1 9 8 3 11 10 8 3 11 9 10

2) **运行命令**

./snt2cooc.out chinese.vcb english.vcb chinese_english.snt > chn_eng.cooc

./snt2cooc.out english.vcb chinese.vcb english_chinese.snt > eng_chn.cooc

获得共线文件chn_eng.cooc,eng_chn.cooc

3) **构建*GIZA++*所需的*mkcls*文件**

运行命令

./mkcls -pchinese.txt -Vchinese.vcb.classes opt

./mkcls -penglish.txt -Venglish.vcb.classes opt

参数设置：

参数	含义
–n	训练迭代次数，默认为1

参数	含义
–p	需要聚类的已分词文本
–V	输出信息
opt	优化运行

生成文件：

文件	内容	示例
chinese.vcb.classes	1. 按字母表序的单词 2. 单词词类	。 11 也 6
english.vcb.classes	1. 按字母表序的单词 2. 单词词类	also 7 do 5
chinese.vcb.classes.cat	1. 单词词类 2. 对应词类的一组单词	2:吗, 3:我,
english.vcb.classes.cat	1. 单词词类 2. 对应词类的一组单词	2:reading, 3:i,

4） 运行*GIZA++*

运行命令

./GIZA++ -S chinese.vcb -T english.vcb -C chinese_english.snt -CoocurrenceFile chn_eng.cooc -O c2e

./GIZA++ -S english.vcb -T chinese.vcb -C english_chinese.snt -CoocurrenceFile eng_chn.cooc -O e2c

运行参数：

参数	含义
–S	源端词汇表
–T	目标端词汇表
–C	源端到目标端的snt文件
–CoocurrenceFile	共现文件 .cooc
–O	输出文件名的前缀

生成文件：（以汉–英为例）

文件	内容	示例
Decoder.config	用作解码的配置文件，用于ISI Rewrite Decoder解码器	
trn.src.vcb，trn.trg.vcb	类似于chinese.vcb和english.vcb文件	
tst.src.vcb，tst.trg.vcb	由于没设置测试集，为空文件	
ti.final	从英文到中文的词语对齐 词语对齐通过token编号表示，并在每组数字后给出相应的对齐概率	11 11 1 10 10 1 9 9 1
actual.ti.final	从英文到中文的词语对齐 词语对齐通过实际 token 表示，并在每组 token 后给出相应的对齐概率 注意：如果需要生成该文件，需要删除GIZA++v2/Makefile中的–DBINARY_SEARCH_FOR_TTABLE选项，再重新编译即可，否则默认使用id的方式进行加载，不会生成该文件	football 足球 1 ？？ 1 do 吗 1 . 。 1 reading 读书 1 music 音乐 1 like 喜欢 1 i 我 1 you 你 1 also 也 1
A3.final	记录了在 IBM Model 3迭代训练后，每个句对的一个最佳对齐 (Viterbi Alignment)。 第一行是一个可用于作为对齐可视化工具的标题的一个标签，包含训练的语料库中的有关该句编号的信息在训练语料，句子长度和对齐概率。 第二行是目标语言，第三行是源语言，源语言中的每个记号之后是一个零或多个数字的集合，这些数字代表的与源语言记号连接的目标语言记号位置。	# Sentence pair (1) source length 4 target length 4 alignment score : 0.471324 i like reading . NULL ({ }) 我 ({ 1 }) 喜欢 ({ 2 }) 读书 ({ 3 }) 。 ({ 4 })
perp	在训练的最后生成，保存了训练过程中，多个模型每轮迭代的困惑度perlexity	

文件	内容	示例
a3.final	记录的是源端词位置和目标端词位置的概率关系，包含的形式如下的表 i j l m p (i / j, l, m) j = position of target sentence i = position of source sentence l = length of the source sentence m = length of the target sentence p(i | j, l, m) = is the probability that a source word in position i is moved to position j in a pair of sentences of length l and m	1 1 4 100 1 2 2 4 100 1 3 3 4 100 1 4 4 4 100 1 1 1 5 100 0.333333 4 1 5 100 0.666667 1 2 5 100 0.666667 2 2 5 100 0.333333 2 3 5 100 0.666667 3 3 5 100 0.333333 3 4 5 100 0.666667 4 4 5 100 0.333333 5 5 5 100 1
d3.final	反向的位变模型，类似于a3.final文件，只是 交换了i 和 j 的位置	
n3.final	繁衍率（fertility）文件，源语言token 的 fertility 分别为 0，1，…，n 时的概率表，形式如下： source__id p0 p1 p2 …. pn，p0 是fertility为0时的概率	
t3.final	IBM Model 3训练后的翻译概率表，形式如下： s_id t_id P(t_id/s_id) s_id：源语言token编号 t_id：目标语言token编号 P(t_id / s_id)：源语言token翻译为目标语言token的概率	
D4.final	IBM Model 4的distortion表	
d4.final	IBM model4 中的位变模型	
Gizacfg	包含训练当中所用的所用参数设置 训练可以精确复制	

4. 运行结果

中文对英文的字间映射概率存储在c2e.actual.ti.final文件中，英文对中文的字间映射概率存储在e2c.actual.ti.final文件中

英	中	英对中概率	中对英概率
football	足球	1	1
?	?	1	1

英	中	英对中概率	中对英概率
do	吗	1	1
.	。	1	1
reading	读书	1	1
music	音乐	1	1
like	喜欢	1	1
i	我	1	1
you	你	1	1
also	也	1	1

对应结果正确。

A3.final 记录了在 IBM Model 3迭代训练后，每个句对的一个最佳对齐 (Viterbi Alignment)。

第一行是一个可用于作为对齐可视化工具的标题的一个标签，包含训练的语料库中的有关该句编号的信息在训练语料，句子长度和对齐概率。

第二行是目标语言

第三行是源语言，源语言中的每个记号之后是一个零或多个数字的集合，这些数字代表的与源语言记号连接的目标语言记号位置。

c2e.A3.final（中文到英文的最佳对齐）

对齐概率	目标句子	源语言句子
0.471324	i like reading .	NULL ({ }) 我 ({ 1 }) 喜欢 ({ 2 }) 读书 ({ 3 }) 。 ({ 4 })
0.375791	i also like music .	NULL ({ }) 我 ({ 1 }) 也 ({ 2 }) 喜欢 ({ 3 }) 音乐 ({ 4 }) 。 ({ 5 })
0.167943	you like reading .	NULL ({ }) 你 ({ 1 }) 喜欢 ({ 2 }) 读书 ({ 3 }) 。 ({ 4 })
0.253913	do you like music ?	NULL ({ }) 你 ({ 2 }) 喜欢 ({ 3 }) 音乐 ({ 4 }) 吗 ({ 1 }) ? ({ 5 })
0.253912	do you like football ?	NULL ({ }) 你 ({ 2 }) 喜欢 ({ 3 }) 足球 ({ 4 }) 吗 ({ 1 }) ? ({ 5 })

e2c.A3.final （英文到中文的最佳对齐）

对齐概率	目标句子	源语言句子
0.471324	我 喜欢 读书 。	NULL ({ }) i ({ 1 }) like ({ 2 }) reading ({ 3 }) . ({ 4 })
0.375791	我 也 喜欢 音乐 。	NULL ({ }) i ({ 1 }) also ({ 2 }) like ({ 3 }) music ({ 4 }) . ({ 5 })

对齐概率	目标句子	源语言句子
0.471324	你 喜欢 读书 。	NULL ({ }) you ({ 1 }) like ({ 2 }) reading ({ 3 }) . ({ 4 })
0.375791	你 喜欢 音乐 吗 ？	NULL ({ }) do ({ 4 }) you ({ 1 }) like ({ 2 }) music ({ 3 }) ? ({ 5 })
0.37579	你 喜欢 足球 吗 ？	NULL ({ }) do ({ 4 }) you ({ 1 }) like ({ 2 }) football ({ 3 }) ? ({ 5 })

训练语料是句子级别对齐的，词对齐工具用来从句子对齐中学习到词对齐，对齐这个概念可以理解成为两个词的翻译对应关系。

5. 实验结论

词语对齐采用开源工具完成，比如现在用的最多的GIZA++，但是在平行语料数据量大的情况下，可能其完成整个词语对齐的过程耗时较长。现在也有MGIZA++，它是GIZA++的一个多线程版本，并对GIZA++的内存使用等方面做了优化，可以同时在linux下和windows下编译。

GIZA++是GIZA（SMT工具包EGYPT的一个组成部分）的扩展，扩展部分主要由Franz Josef Och开发。GIZA++主要算法是IBM model、HMM等。

提供给GIZA++的平行句对中的中文部分都被切分成了相应的短语，而英文的大小写、格式、相应的空格也都需要加上了。词语对齐的目标是得到中英文词或短语的对齐信息，便于翻译系统做解码时寻找相应的phrase。