跨语言情感分析 实验报告

- 实验任务和数据集

本实验在 NLPCC2013 跨语言情感分类任务上进行[1]。本任务为对三个领域的评论文档 music、book、DVD 标注情感倾向（正向、负向）。三类数据的规模如表 1，所有标准数据中，正向和负向的标注量都是相同的。

类型 / 领域	中文标注数据	英文标注数据	中文未标注数据	中文测试数据
book	40	4000	47071	4000
music	40	4000	29677	4000
DVD	40	4000	17814	4000

表 1. 实验数据规模

本实验使用谷歌翻译对所有数据进行了翻译[2]。对对所有文档，将标题与内容连接起来抽取词频特征。在判定时时，统一使用 sklearn[3]提供的 LinearSVR 工具。

- 实验结果

Baseline：只使用标注数据作为训练数据，提取词频特征，对测试数据进行预测，LinearSVR 采用默认参数。结果如表 2。combine_add 为对中英文的结果简单相加，combine_concat 为将中英文特征向量简单相连。ALL 列表示把三类数据混在一起的实验结果。

	Book	Music	DVD	Average	All
Cn	73.33%	67.98%	71.18%	70.83%	72.15%
En	72.93%	66.25%	69.00%	69.39%	71.33%
combine_add	76.95%	**68.18%**	**73.10%**	**72.74%**	**74.94%**
combine_concat	**77.35%**	67.28%	72.90%	72.51%	73.41%

表 2. 几组简单设置下的实验结果

[1] http://tcci.ccf.org.cn/conference/2013/pages/page04_sam.html

[2] 参考：http://www.cnblogs.com/wnzhong/p/6666911.html

[3] http://scikit-learn.org

由表 2 中可以看出，翻译对于跨语言的情感分类是有效的。对两种语言的结果的简单加和与将特征进行串联都可以让效果略有提升。将三类数据混在一起的效果比三类数据分开的效果更好。

由于三类混在一起的设置与相加组合的方法效果相对最好，于是在该设置下对一些 tf-idf 预处理特征的方法进行尝试，设置如下，结果如表 3 所示。

$$tf_1 = \min(f, 1), \quad tf_2 = f, \quad tf_3 = \frac{f}{\sum_d f}, \quad tf_4 = 1 + \log(f), \quad tf_5 = 0.5 + 0.5 * (\frac{f}{\max_d f})$$

$$idf_1 = 1, \quad idf_2 = \log\left(\frac{\mathrm{N}}{n}\right), \quad idf_3 = \log\left(\frac{N}{n+1}\right), \quad idf_4 = \log\left(\frac{N}{n} - 1\right)$$

	En	Cn	combine_add
tf_1idf_1	71.23%	72.29%	75.65%
tf_2idf_1	71.33%	72.15%	74.94%
tf_3idf_1	68.34%	71.77%	73.15%
tf_4idf_1	71.93%	72.22%	75.81%
tf_5idf_1	**72.73%**	**74.25%**	**77.18%**
tf_5idf_2	69.02%	73.27%	74.96%
tf_5idf_3	68.90%	73.23%	74.88%
tf_5idf_4	67.31%	73.15%	74.71%

表 3. Tf-idf 处理效果

对效果最好的配置：combine_add+ tf_5idf_1调整 LinearSVR 正则化系数-C，结果如图 4 所示。

-C	En	Cn	combine_add
1.0	72.73%	74.25%	77.18%
0.5	73.23%	74.93%	77.83%
0.2	74.41%	76.53%	78.83%
0.1	74.98%	77.29%	79.48%
0.08	**75.09%**	77.41%	79.58%
0.05	74.98%	**77.98%**	**79.66%**
0.03	74.99%	77.79%	79.40%
0.02	75.04%	77.68%	79.25%

表 4. LinearSVR 调整正则化系数的结果

对效果最好的配置：combine_add+ tf_5idf_1 –C=0.05 绘制准确率召回率曲线，如图 1 所示。从图 1 中可以看出，top0.5%左右的准确率实际上是偏低的，top1%~top5%这部分的结果的准确率是很高的，

基本上是在 95%以上的，之后准确率就开始不断下降了。所以我们在后面进行 co-training 时，更倾向于利用在未标注数据集中，打分在 0.5%~5%之间的这部分数据。

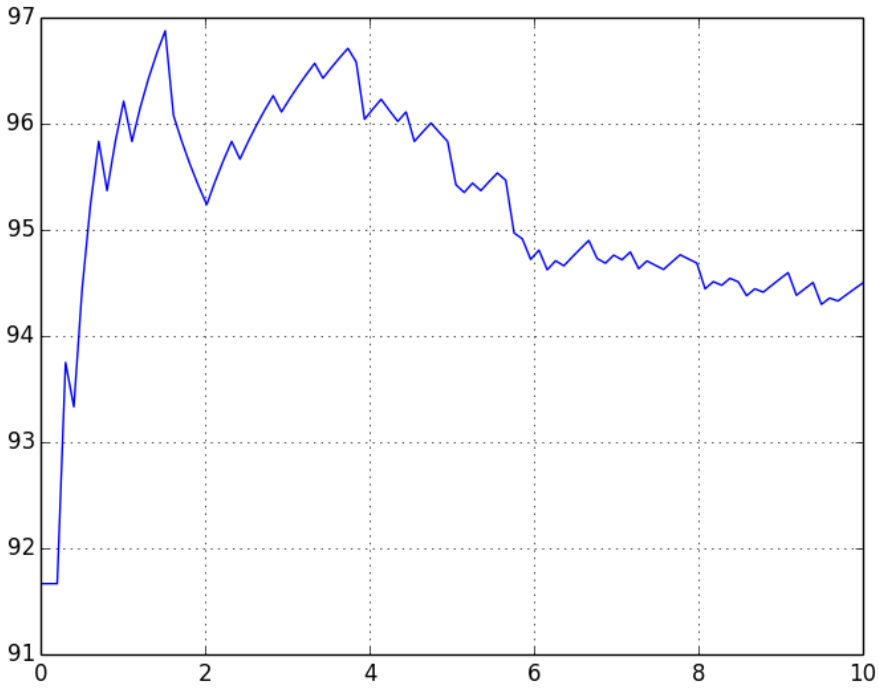

图 1. combine_add+ tf_5idf_1 –C=0.05 的准确率召回率曲线

接下来进行 co-training。每次根据 combine_add+ tf_5idf_1的结果选择未标注集 top n 的数据进行标注并加入训练集。由于上面的观察，我们舍弃 top a 的数据，co-training 的结果如表 5 所示。其中，iter 表示达到该效果时的迭代轮数，0 轮为不进行迭代即不使用 co-training 时的效果。

top n	ignore a	En	Cn	combine_add	iter
25	0	75.72%	78.02%	**79.88%**	35
50	0	75.95%	77.59%	79.80%	19
100	0	75.92%	77.53%	79.75%	10
200	0	74.97%	77.97%	79.67%	0
25	150	76.00%	**78.22%**	**79.88%**	40
50	150	76.05%	78.10%	79.87%	23
100	150	75.66%	78.05%	79.73%	8
200	150	75.88%	78.09%	79.77%	4
25	300	76.03%	78.13%	79.84%	35
50	300	**76.06%**	78.17%	79.83%	20
100	300	75.86%	78.11%	79.72%	8
200	300	76.02%	78.32%	79.83%	5

25	600	75.97%	77.96%	79.67%	38
50	600	74.98%	77.97%	79.67%	0
100	600	74.98%	77.97%	79.67%	0
100	1000	75.60%	77.93%	79.77%	2
100	2000	74.97%	77.97%	79.67%	0
100	4000	74.96%	77.97%	79.66%	0

表 5 co-training 效果

由上表中可以看出，ignore a 的作用看起来并没有从图 1 中预期得那么明显（全部未标注数据约 9w 多组，600≈0.6%， 300≈0.3%，150≈0.15%），一般在每次更新较少的数据时，co-training 效果比较好。但是效果达到最好时普遍没有使用很多的未标注数据（最多 1000 of 9w）。相对而言 co-training 对单语言分类器提升效果更大一点（En: 74.98→76.06, Cn: 77.98→78.22）但对复合后的效果提升较小（79.66→79.88）。这可能是由于 co-training 的过程减小了两个模型之间的差异性，导致的集成效果减弱。

图 2 中显示的是 top n=25，ignore a=150 时效果随迭代轮数的变化。由此可以看到，的确还是在 co-training 开始时效果有一个比较明显的下降的。而且单一语言上的效果提升相对与整体的效果提升而言要更明显一些。

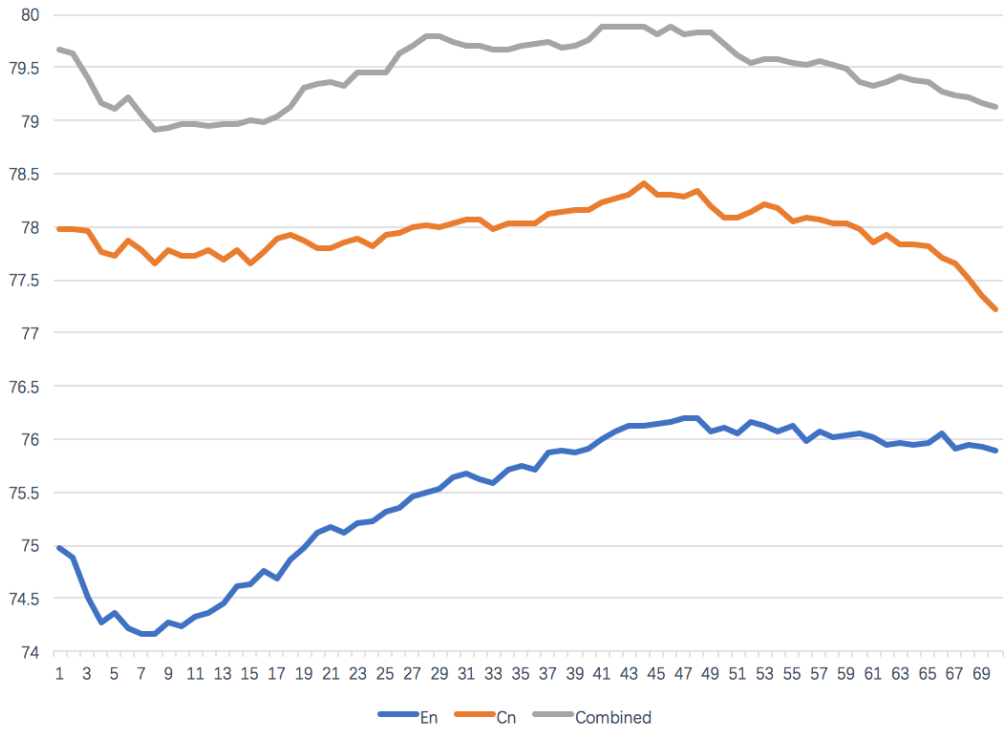

图 2 迭代轮数对 co-training(25-150)效果的影响

考虑到 co-training 方法可能会降低两个模型的差异性，从而降低集成后模型的效果。于是分别对两种语言的模型进行自学习，效果如表 6 所示。在忽略高分的未标注数据时，两种语言的模型的高分数据都会被考虑到（即 ignore a=150 时，两个排名的前 150 名都被忽略），所以这里的 ignore a 相对于表 5 而言比较小。

top n	ignore a	En	Cn	combine_add	iter
25	0	75.31%	77.90%	79.85%	84
50	0	75.23%	78.20%	79.84%	27
100	0	74.99%	77.97%	79.67%	0
200	0	75.30%	78.22%	79.76%	7
25	75	75.54%	77.61%	79.69%	74
50	75	74.97%	77.97%	79.67%	0
100	75	75.52%	78.14%	79.69%	13
200	75	74.97%	77.97%	79.66%	0
25	150	75.64%	78.66%	80.24%	74
50	150	75.82%	**78.69%**	**80.34%**	35
100	150	**75.84%**	78.41%	80.25%	16
200	150	75.74%	78.12%	80.15%	10
25	300	75.70%	78.11%	80.05%	61
50	300	75.63%	77.93%	80.00%	31
100	300	75.60%	78.09%	79.87%	15
100	600	75.43%	78.23%	79.81%	14
100	1000	75.54%	77.92%	79.77%	13

表 6 self-training(50-150)效果的影响

对比表 6 和表 5 可以看出，使用 self-training 而不是 co-training 对于提升单一分类器准确率上并没有太大优势，但的确可以增加两种语言分类器之间的差异性从而提升集成后的效果。并且使用 self-training 通常可以在取得最好效果时利用更多的未标注数据（1750~2*1750 Vs. 1000）。

图 3 中显示的是 top n=50，ignore a=150 时效果随迭代轮数的变化。Self-training 在训练初期准确率的波动更为明显。

最终的结果汇总见表 7。可以看到，利用了未标注数据的效果显著优于没有利用未标注数据的效果。最好准确率达到 **80.34%**，最好的单纯基于中文特征的分类器的准确率达到了 78.69%。

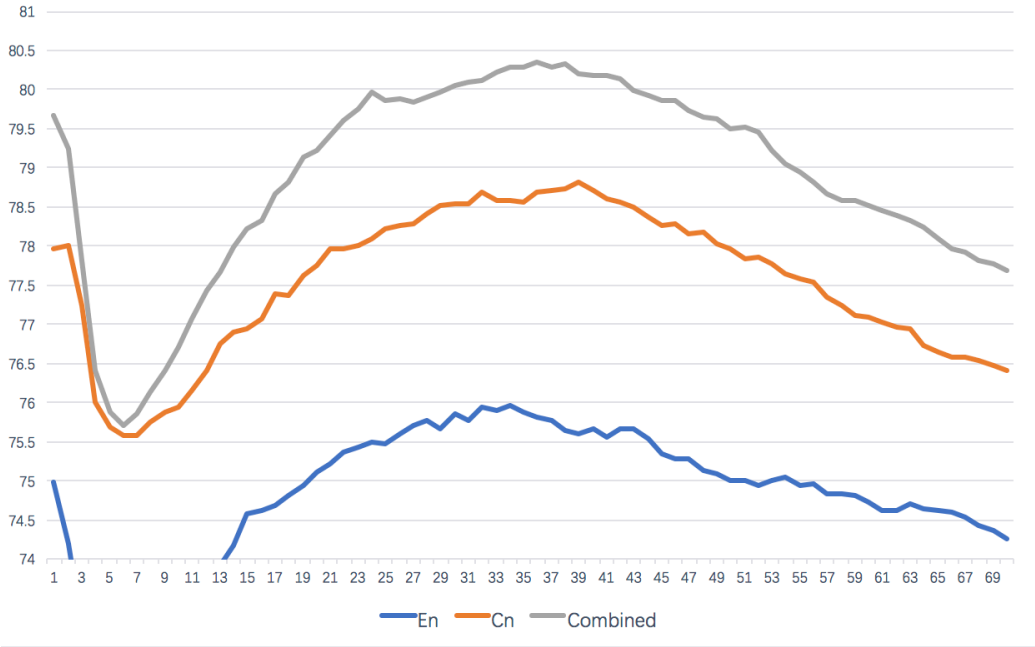

图 3 迭代轮数对 self-training 效果的影响

	Book	Music	DVD	Average
simple_Cn	77.75%	77.62%	78.53%	77.97%
simple_En	75.08%	74.58%	75.30%	74.98%
simple_combine	79.88%	79.25%	79.90%	79.67%
co-Training_Cn	78.83%	77.50%	78.35%	78.22%
co-Training_En	76.33%	**75.10%**	**76.58%**	**76.00%**
co-Training_combine	80.33%	79.08%	80.25%	79.88%
self-Training_Cn	**79.08%**	**77.97%**	**79.03%**	**78.69%**
self-Training_En	**76.50%**	74.60%	76.35%	75.82%
self-Training_combine	**80.67%**	**79.67%**	**80.67%**	**80.34%**

表 7 结果汇总

GitHub 地址：

https://github.com/Erutan-pku/NLPCC2013_Cross-Lingual_Sentiment_Classification