MERT

2016年11月1日 星期二　　下午7:49

一、简介

MERT（minimal error rate training），是一种参数优化的算法。可以说是坐标下降（Coordinate Descent Optimization）的一种，也是逐个参数优化，某一参数优化的时候，把其他参数视为常量。
https://en.wikipedia.org/wiki/Coordinate_descent

用于机器翻译中，机器翻译的log-linear中weight参数很多。

二、变量定义：

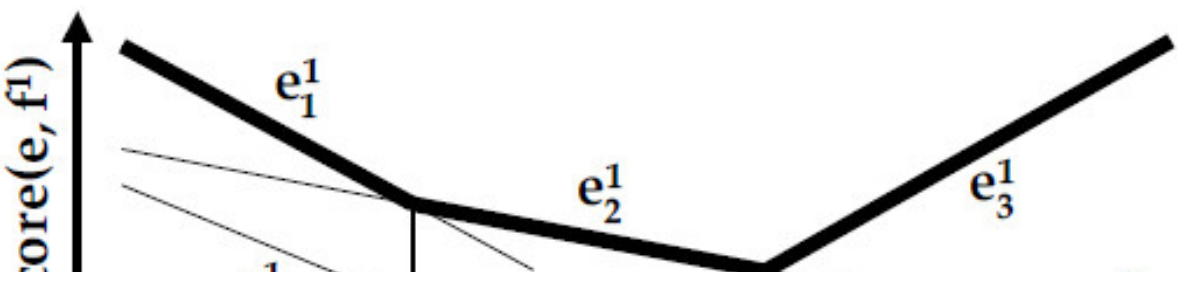
一个模型有M个参数λ_i，D个句子d_j，每个句子有很多个候选翻译结果e_jk。在给定的评估方式下，在不同的每种翻译结果，在不同的参数下有一个对应一个打分S_ijk。

三、算法思想

按照顺序依次调M个参数，（调完一轮再迭代下一轮）

1. 先按住其他的M-1个参数，这些参数认为是常量，只调第i个参数λ_i，只有这个参数目前是变量。

在其他都固定的情况下，打分S_ijk根据λ_i的不同而不同，这是一个一次函数，可以画出来，横轴是参数λ_i，纵轴是S_ijk。 把一个句子d_j的所有候选e_jk都画到一个坐标系中，是这样，每条线（一次函数）都表示一个候选。



2. 我们要找打分S_ijk的最大值。

有些候选结果S很低，我们只关心最高的那一条线，它代表相同的参数λ_i的情况下，要挑最好的候选结果。

这是一个分段函数，不同段代表在模型参数λ_i取值不同的情况下，得分最大的翻译结果是不一样的。分段函数的断点代表翻译结果变化的那一刻。

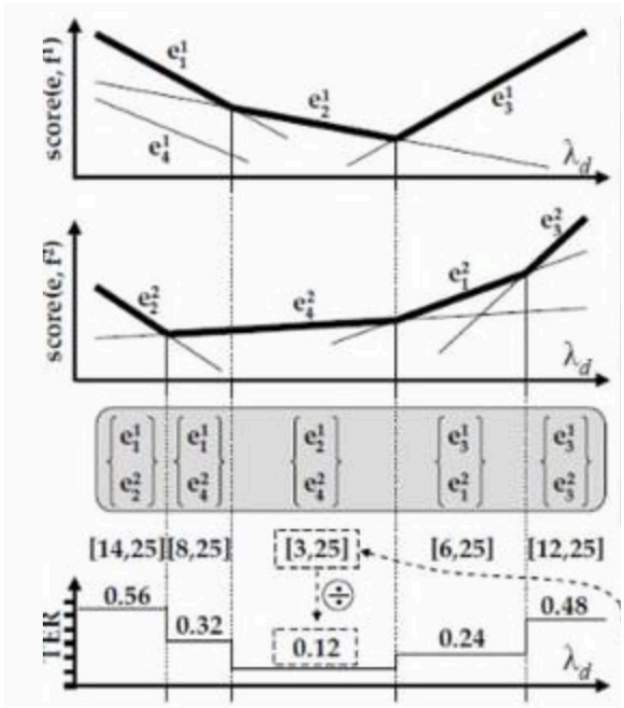
我们要找打分S_ijk最大的，得分最大的就是上图的粗线。

3. 找到整个数据集上最优的λ_i

我们找到了S_ijk与参数λ_i的映射关系，现在为了找到参数λ_i的最优解，需要关注整个数据集上的打分S。

这一步其实不难，就是把每一个样本的上述分段函数叠加，合成一个大的分段函数。

如下图



大的分段函数也是一个线性的分段函数，很容易找到最大值点。

那么λ_i这个参数就调好了。

4.接下来继续循环，调下一个参数λ_i+1。

5.等一轮所有的参数λ都调完了，再开始下一轮迭代。

四、算法过程

1、伪代码

Algorithm 1 Z-MERT: Optimization of weight vector Λ to minimize error, using for line optimization (line 17) the efficient method of Och (2003).

Input: Initial weight vector $\Lambda^0 = \{\Lambda^0[1], ..., \Lambda^0[M]\}$; `numInter`, the number of initial points per iteration; and `N`, the size of the candidate list generated each iteration.

Return: Final weight vector $\Lambda^* = \{\Lambda^*[1], ..., \Lambda^*[M]\}$.

1. Initialize $\Lambda \leftarrow \Lambda^0$
2. Initialize `currError` $\leftarrow +\infty$
3. Intialize the cumulative candidate set for each sentence to the empty set.
4. **loop**
5. Using Λ, produce an `N`-best candidate list for each sentence, and merge it with the cumulative candidate set for that sentence.
6. **if** no candidate set grew **then** Return Λ // MERT convergence; we are done.
7.
8. Initialize $\Lambda_1 \leftarrow \Lambda$
9. **for** ($j = 2$ to `numInter`), initialize $\Lambda_j \leftarrow$ random weight vector
10.
11. Initialize $j_{best} \leftarrow 0$
12. **for** ($j = 1$ to `numInter`) **do**
13. Initialize $\texttt{currError}_j \leftarrow \text{error}(\Lambda_j)$ based on cumulative candidate sets
14. **repeat**
15. Initialize $m_{best} \leftarrow 0$
16. **for** ($m = 1$ to M) **do**
17. Set (λ,`err`) = value returned by efficient investigation of the m^{th} dimension and the error at that value (i.e. using Och's method)
18. **if** ($\texttt{err} < \texttt{currError}_j$) **then**
19. $m_{best} \leftarrow m$
20. $\lambda_{best} \leftarrow \lambda$
21. $\texttt{currError}_j \leftarrow \texttt{err}$
22. **end if**
23. **end for**
24. **if** ($m_{best} \neq 0$) **then**
25. Change $\Lambda_j[m_{best}]$ to λ_{best}
26. **end if**
27. **until** ($m_{best} == 0$)
28. **if** ($\texttt{currError}_j < \texttt{currError}$) **then**
29. $\texttt{currError} \leftarrow \texttt{currError}_j$
30. $j_{best} \leftarrow j$
31. $\Lambda \leftarrow \Lambda_j$
32. **end if**
33. **end for**
34. **if** ($j_{best} == 0$) **then** Return Λ // Could not improve any further; we are done.
35. **end loop**

伪代码第17行是核心，他做的是三、1~3步讲的，具体怎么做，伪

代码也没说。其他行都是废话。

五、优缺点

六、参考资料

1. Discriminative Training and Maximum Entropy Models for Statistical Machine Translation

2. Minimum Error Rate Training in Statistical Machine Translation

3. Z-MERT: A Fully Configurable Open Source Tool for Minimum Error Rate Training of Machine Translation Systems

https://en.wikipedia.org/wiki/Coordinate_decent

中文博客：http://www.tuicool.com/articles/vEJRzi