基于 WordNet 词义消歧的系统融合

刘宇鹏 [1] 李 生 [1] 赵铁军 [1]

摘 要 最近混淆网络在融合多个机器翻译结果中展示很好的性能. 然而为了克服在不同的翻译系统中不同的词序, 假设对齐在混淆网络的构建上仍然是一个重要的问题. 但以往的对齐方法都没有考虑到语义信息. 本文为了更好地改进系统融合的性能, 提出了用词义消歧 (Word sense disambiguation, WSD) 来指导混淆网络中的对齐. 同时骨架翻译的选择也是通过计算句子间的相似度来获得的, 句子的相似性计算使用了二分图的最大匹配算法. 为了使得基于 WordNet 词义消歧方法融入到系统中, 本文将翻译错误率 (Translation error rate, TER) 算法进行了改进, 实验结果显示本方法的性能好于经典的 TER 算法的性能.

关键词 系统融合, 翻译错误率, 词义消歧, 混淆网络

DOI 10.3724/SP.J.1004.2010.01575

System Combination Based on WSD Using WordNet

LIU Yu-Peng[1] LI Sheng[1] ZHAO Tie-Jun[1]

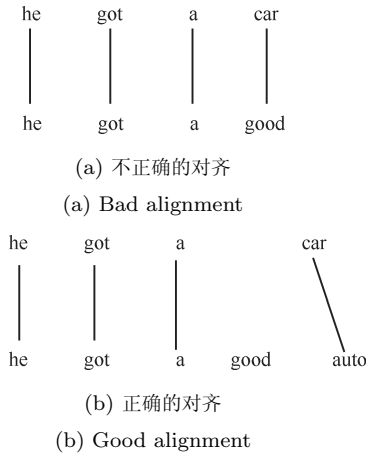
Abstract Recently confusion network decoding showed a better performance in combining outputs from multiple machine translation (MT) systems. However, overcoming different word orders presented in multiple MT systems during hypothesis alignment still remains to be the biggest challenge to confusion-network-based MT system combination. The previous alignment methods do not consider the information about semantics. In order to improve the system performance, we introduce word sense disambiguation (WSD) into confusion network alignment. Meanwhile, the selection of skeleton is taken through sentence similarity score, and the sentence similarity is computed by the largest bipartite graph matching algorithm. In order to combine WSD based on WordNet with our system, the experiments showed that the result using revised translation error rate (TER) algorithms is better than classic TER system combination.

Key words System combination, translation error rate (TER), word sense disambiguation (WSD), confusion network (CN)

系统融合技术通过多个机器翻译系统的输出结果来得到一个新的结果, 以提高机器翻译系统的性能. 系统融合技术可以认为是多个机器翻译结果的后处理过程, 与在机器翻译中一样, 对齐[1] 是系统融合的一个关键问题. 基于词的系统融合[2−6] 与基于句子[7−8] 和短语的系统[9] 融合技术相比, 在性能上得到了大幅的提高. 例如通过投票来进行句子级翻译候选的选择, 或是使用从源语言和目标语言得到的短语表进行调序. 对于词一级系统融合中混淆网络的构建含有四个步骤: 1) 骨架翻译的选择: 骨架翻译定义了翻译的词序; 2) 假设对齐: 在骨架翻译和每个假设翻译中建立对齐关系; 3) 混淆网络的构建: 基于假设对齐建立混淆网络; 4) 混淆网络解码: 从混淆网络中解码出最好的翻译. 在上述四个步骤中, 不同机器翻译系统结果的不同词序对于假设翻译的对齐是一个巨大的挑战. 基于翻译错误率 (Translation error rate, TER)[10] 对齐方法经常作为基线系统, 同时一系列其他的方法也被提出. 如间接隐马尔科夫模型 (Indirect hidden Markov model, IHMM)[2] 和基于反向转录文法 (Inversion transduction grammar, ITG)[3] 的系统融合. 通过对齐方法将骨架翻译和候选翻译对齐来构建混淆网络. 然而像 Rosti 指出的那样, 如果对齐存在错误, 将生成低效的混合网络. 对齐一直是生成高效的混淆网络的关键. 为了更好地提高对齐质量, 使词一级的系统融合获得更好性能, GIZA, TER, IHMM 以及 ITG 的方法[2−6] 被引入到混淆网络的对齐中. 虽然在文献 [4] 中出现了使用 WordNet 来进行同义词匹配, 但并没有考虑词义信息, 本文方法不仅考虑到了同义词信息, 而且还利用了语境信息. 例如假设我们有骨架翻译 “he got a car” 和一个候选翻译 “he get a good auto”, 对于 TER 的对齐方法, 很自然地生成如图 1 (a) 的对齐. 但是根据语言学知识, 我们知道这种对齐是不正确的. 而图 1 (b) 是一种正

收稿日期 2009-10-13 录用日期 2010-07-01
Manuscript received October 13, 2009; accepted July 1, 2010
国家高技术研究发展计划 (863 计划) (2006AA010108), 国家自然科学基金 (60736014), 微软亚洲研究院基金 (FY09-RES-THEME-158) 资助
Supported by National High Technology Research and Development Program of China (863 Program) (2006AA010108), National Natural Science Foundation of China (60736014), and Microsoft Research Asia IFP (FY09-RES-THEME-158)
1. 哈尔滨工业大学计算机科学与技术实验室机器翻译与智能实验室 哈尔滨 150001
1. Machine Intelligence and Translation Laboratory, School of Computer Science and Technology, Harbin University of Technology, Harbin 150001

确的对齐. 基于上面的原因, 我们提出把词义消歧 (Word sense disambiguation, WSD) 引入到混淆网络的构建中. 并对经典的 TER 算法进行了修改, 以获得更好的对齐. 本文主要是用词义消歧的方法来进行对齐, 把语义信息用 WordNet 引入到系统融合的框架下, 将词义相关的单词进行对齐来提高系统融合的性能.



(a) 不正确的对齐

(a) Bad alignment

(b) 正确的对齐

(b) Good alignment

图 1 混淆网络的两个对齐实例

Fig. 1 Two alignment examples of the confusion network

本文的结构安排如下: 第 1 节主要介绍本文所采用的词义消歧方法和选择骨架翻译的方法; 第 2 节介绍将词义消歧方法融入到 TER 算法中的方法和混淆网络解码; 第 3 节是基于 WordNet 词义消歧的实验结果和分析.

1 词义消歧方法及选择骨架翻译流程

WordNet[11] 最初在 1985 年由普林斯顿大学认知科学实验室实现, 它是在基于人类词汇记忆的心理语言学理论推动下产生的. 它是一部在线词典数据库系统, 采用了与传统词典不同的方式, 即按照词义而不是词形来组织词汇信息. 经过十几年的发展, 它将逐渐成为一种国际标准, 许多国家都在筹划和建立与英文 WorNet 兼容的本国语言 WorNet 系统, 如 Euro-WordNet. WordNet 有很多应用, 如词义标注、词义消歧、建立基于词义分类的统计模型、基于概念的文本检索、文本校对、知识推理、知识工程及概念建模等. WordNet 使用同义词集合 (Synset) 来代表词汇概念, 将英语的名词、动词、形容词和副词组织为 Synset, 并描述词汇矩阵模型, 即在词的形式和意义之间建立起映射关系. 每一个 Synset 表示一个基本的词汇概念, 并在这些概念之间建立了包括同义关系 (Synonymy)、反义关系 (Antonymy)、上下位关系 (Hypernymy and hyponymy)、部分关系 (Meronymy) 及整体关系等多种语义关系. WordNet 中的词由 Synset 组成, Synset 之间用关系指针指示它们的语义关系. 关系指针代表了一个 Synset 和另一个 Synset 之间的关系, 如: 同义、反义、上下位以及整体 – 部分关系. 以上构成了进行基于 WordNet 词语相似度计算的基础. 消歧是找出在给定语境下一个词的最合适意思. Lesk 算法[12] 是在句子语境下来消歧句子中的单词. 主要的目的是计算在两个解释中共享的词数. 重叠的单词越多, 语义就越相关. 为了进行词义消歧, 对每一个单词的解释由在短语中出现的其他单词的解释来做比较. 例如: 在执行短语 “pine cone” 的消歧过程中, 按照牛津高级自学字典: 单词 “pine” 有两个意思: “kind of evergreen tree with needle-shaped leaves waste away through sorrow or illness”; 单词 “cone” 有三个意思: “solid body which narrows to a point, something of this shape whether solid or hollow fruit of a certain evergreen tree”. 通过比较单词 “cone” 的三个解释意思中的每一个意思和单词 “pine” 的两个解释意思中的每一个意思, 我们发现单词 “evergreen tree” 出现在两个单词中的一个词义中. 当单词 “pine” 和 “cone” 一同使用时, 则将包含 “evergreen tree” 的两个词义选择为最合适的词义. 原来的 Lesk 算法不会利用语境, 这个贪心算法不会一直有效. 如果计算时间很长, 我们将通过如 Beam 的局部搜索来优化词义组合. 这种方法背后的操作是通过应用启发式的技巧来缩减搜索空间. 在搜索过程的每个阶段, Beam 搜索器将限制 k 个最有前景的候选, 而 k 是预先定义的数.

1.1 改进的 Lesk 算法

在词义消歧的过程中, 我们采用了改进的 Lesk 算法[12−13], 也就是在原来 Lesk 计算解释中的重叠词基础上, 加入各种重叠打分. 在本节中, 我们引入一个修改算法, 即访问词典中的词义层次结构来加入多种重叠打分以引入更多的相关信息: 这个改进不仅考虑了 Synset 集合的解释也考虑了相关词词义. 应用新的得分机制, 相对于原来 Lesk 算法的重叠词计数, 给出了更精确的得分. 为了给句长为 N 的句子中的每个单词进行消歧, 我们称每个被消歧的词为目标词. 算法描述如下:

步骤 1. 选择一个语境. 如果 N 很长, 为了优化计算时间, 在目标词的周围定义 K 个语境词, 即在目标词左侧的 $K/2$ 个词和在目标词右侧的 $K/2$ 个词. 例如如果 K 是 4, 则目标词的左侧是 2 个词, 目标词的右侧是 2 个词.

步骤 2. 对于在选定语境下的每一个词, 查找和列出两个词性 (名词和动词) 的所有意思.

步骤 3. 对于一个词的每个词义, 列出下列关系: 1) 由 WordNet 提供的解释, 包含实例项; 2) 由 Synset 通过上位关系连接到的解释, 如果一个词义有超过一个上位词, 每个上位词的解释被连接成单个解释串; 3) 由 Synset 通过下位关系连接到的解释; 4) 由 Synset 通过整体关系连接到的解释; 5) 由 Synset 通过局部关系连接到的解释.

步骤 4. 组合由步骤 3 提供的所有可能解释对, 通过搜索重叠来计算相关性. 当计算两个 Synset 集合 $s1$ 和 $s2$ 之间的关系时, hype-hype 意味着 $s1$ 的 hypernym 的解释可以和 $s2$ 的 hypernym 的解释作比较; hype-hypo 意味着 $s1$ 的 hypernym 的解释可以和 $s2$ 的 hyponym 的解释作比较. $\text{OverallScore}(s1,\ s2) = \sum_{\text{score}} \text{Score}\ (s1, s2)$, 其中 Score 是 $s1$ 和 $s2$ 在所有解释对上的比较函数. 在 "pine cone" 的例子中, pine 有 3 个意思, cone 有 6 个意思, 所以我们有 18 种可能的组合. 为了给重叠打分, 我们使用新的打分机制 (不同于 N 个独立词和 N 个连续词重叠, 把每个解释对待成一串词). 这是基于 ZipF 的定律, 单词的长度通常是与它们的使用情况成反比的. 长度越短的词使用的次数越多, 越长的词恰恰相反. 在两个串的重叠问题变成了求最大公共子串 (最大连续). 每一个重叠对于解释词义的组合贡献的得分为 N^2. 例如, "ABC" 有 9 个得分，而两个单一重叠有 5 个得分.

步骤 5. 一旦每个组合被打分, 我们选择有最高得分者作为目标词在特定语境下目标词的最合适的意思. 同时输出结果不仅给出了消歧后的意思, 而且给出了适当的词性.

1.2 基于路径长度的语义相似性计算方法

为了计算两个句子的相似度, 我们需要得到两个词义的语义相似度. 下面给出计算两个词义的语义相似度的方法. 在 WordNet 中每个词性被组织在一个分类中, 代表一个意思的每个节点是一系列同义词. 如果一个词含有超过一个意思, 它将出现在分类的不同地方的多个同义词集合. 在 Synset 之间的关系是语义关系, 在词义间的关系是词汇关系. 不同点是: 词汇关系是两个不同 Synset 集合中成员间的关系, 但语义关系是两个不同 Synset 集合的关系. 例如: 语义关系是上下位关系等, 而词汇关系是反义关系和推导关系. 例如: 名词 light 第十个意思 (light＃n＃10) 的反义词在 WordNet 中是名词 dark 的第一个意思 (dark＃n＃1). 这个 Synset 集合是 {light＃n＃10, lighting＃n＃1}. 显然 (light＃n＃10) 是 (dark＃n＃1) 的反义词, 但是 (lighting＃n＃1) 不是 (dark＃n＃1) 的反义词. 因此反义关系需要一个词汇关系, 而不是语义关系. 语义相似性是语义关系的一个特例, 我们仅考虑其是 IS-A 关系. 为了衡量两个 Synset 集合的语义相似性, 我们使用 hyponym/hypernym (或者是 IS-A 关系) 解决 noun-noun 和 verb-verb 的词性. 衡量两个 Synset 集合的语义相似性的简单方法是把分类看成一个无向图, 在 WordNet 中衡量它们的距离. Resnik 认为 "一个节点到另一个路径越短, 它们就越相似". 注意这个路径长度是由通过的点而不是通过的边来衡量. 在同一个 Synset 集合中两个成员的路径长度为 1. 图 2 展示了用路径长度相似度来计算上位分类的实例. 由图 2 可以看到, car 和 auto 的长度为 1, car 和 truck 的长度为 3, car 和 bicycle 的长度为 4, car 和 fork 的长度为 12. 两个 Synset 的共有父亲是 Sub-sumer, 最小公共祖先 (Least common sub-sumer, LCS) 也是两个 Synset 的 Sub-sumer. 换句话说, 两个 Synset 集合的 LCS 是两个 Synset 中具体的 Sub-sumer. 回到上面的例子: car, auto $\cdots$ 和 truck $\cdots$ 的 LCS 是 automotive, motor vehicle, 因为 automotive, motor vehicle比普通的 Sub-sumer wheeled vehicle 更具体. 路径长度给出了计算两个词义关系的简单方法.

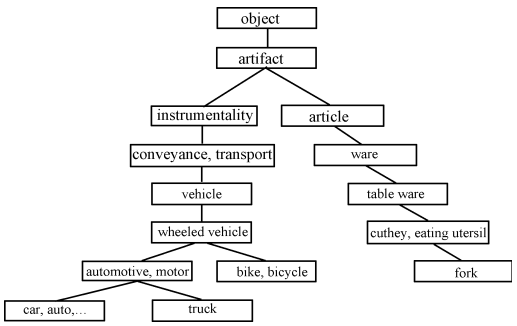


图 2 WordNet 中上位分类的实例

Fig. 2 An example of the hyponym taxonomy in WordNet

1) 来自于同一词性的两个 Synset 集合可能没有公共的 Sub-sumer. 因为我们没有把每一个词性分类的所有不同最高节点连接在一起. 在两个 Synset 集合之间的路径不是一直都能发现的. 但是如果唯一的根节点被使用, 则在两个 noun/verb Synset 集合之间将一直存在一条路径.

2) 注意在 WordNet 中是允许多继承的, 一些 Synset 集合属于多个分类, 所以在两个 Synset 集合中有两条路径时, 则选择最短的路径.

3) 词汇化: 当在 WordNet 中查找一个单词时, 词首先被词根化. 因此 "book" 和 "books" 的距离为 0, 因为它们是同一个词.

4) 这个方法仅仅比较有同样词性的两个词义. 这意味着我们不比较 noun 和 verb, 因为它们位于

不同的分类. 分别考虑 noun, verb, adjective, 由于忽略了词性标注器, 我们使用了 Lexicon 类. 当考虑一个词时, 我们检查它是否是名词, 如果是就认为是名词, 如果不是就接着检查动词 $\cdots$.

5) 对于两个 Synset 集合的语义相似性, 使用简单的衡量方法: $\mathrm{Sim}(s,t)=1/\mathrm{distance}(s,t)$, 距离是节点计数从 s 到 t 的最短路径长度, 用 SI 1 表示应用这个公式.

6) 以下公式由 Wu 和 Palmer[13] 提出:

$$\mathrm{Sim}\,(s,\,t)=\frac{2\times\mathrm{depth(LCS)}}{(\mathrm{depth}(s)+\mathrm{depth}(t))}$$

这个方法考虑了路径长度和 LCS 的深度, 其中 s 和 t 表示被比较的源和目标单词, $\mathrm{depth}(s)$ 表示 s 所在 Synset 分类中从根节点到节点 s 的距离, LCS 表示 s 和 t 的最小公共 Sub-submer, 用 SI 2 表示应用这个公式.

1.3 选择骨架翻译的流程

选择所有系统的最好翻译作为候选的骨架翻译, 计算任意候选骨架翻译和其他句子的句子相似度, 取平均, 把拥有最高数值的句子作为骨架翻译. 计算句子相似度的流程如下:

1) 断词.

2) 对每个单词进行还原词根.

3) 词义消歧.

4) 对于每一个词义对建立一个语义相关矩阵 $R[m,n]$, $R[i,j]$ 表示在 X 中位置 i 和在 Y 中位置 j 最相似词义的语义相关度. 因此 $R[i,j]$ 也是从 i 到 j 的弧上的权重. 如果一个词不存在字典中则使用编辑距离相似性, 输出一个编辑距离结果.

5) 把计算两个句子的相似度看成二分图的最大权重完全匹配, 其中 X 和 Y 是两个不相交的集合. 使用 Hungarian 算法来求最大加权的匹配.

6) 以上步骤的匹配结果形成了一个两个句子的相似度数值. 有很多策略来获得两个句子的总相似度数值. 在前面的论述中, 我们提出了两个简单的公式来计算词义的语义相似度. 对于每一个公式我们应用一个适当的策略来计算总得分:

a) 匹配平均: $2\times(\mathrm{Match}(X,Y))/(|X|+|Y|)$, 这里 $\mathrm{Match}(X,Y)$ 是 X 和 Y 匹配词的语义相似度和. 这个相似性是两个句子中所有匹配候选的相似度和除以词次的和. 一个重要之处是基于独立相似性数值, 以致整体相似性一直受其影响. 用 MS 1 表示应用这个公式.

b) DICE 系数: $2\times(X\cap Y)/(|X|+|Y|)$, 这个策略反映了匹配词数与总词数之比. 用 MS 2 表示应用这个公式. 因此, DICE 将一直返回比匹配更高的数值, 从而是更优化的. 在这个策略中需要选择超过给定值的匹配对 (所以要设定一个阈值), 用 MS 2 表示应用这个公式.

7) 找出与其他句子最相似句子作为骨架翻译; 计算两个句子总相似度的时间复杂度为 $\mathrm{O}(m\times n\times C)$, 其中 C 是计算两个词的语义相识度的时间复杂度.

2 TER 算法的改进和混淆网络解码

在系统融合中, 需要建立一个混淆网络来得到融合后的翻译结果. 基于混淆网络的系统融合的一般框架为:

1) 从 MT 系统中抽出 N-best 结果;

2) 挑选出一个骨架翻译;

3) 将所有的假设翻译与骨架翻译进行对齐, 以调整假设翻译的顺序;

4) 用重排序的翻译来建立混淆网络;

5) 使用句子级特征和词后验概率特征来对混淆网络进行解码;

6) 用开发集来优化参数, 最终在测试集上进行解码.

在这个框架下, 对混淆网络成功地解码依赖于两个重要的方面: 1) 骨架翻译的选择; 2) 其他假设翻译与骨架翻译的对齐. 为了选择更好的骨架翻译, 两个一般的方法是: 1) 用 TER 分数作为损失函数的最小贝叶斯风险 (Minimum Bayes risk, MBR) 来选择骨架翻译[8]; 2) 用选择每一个系统的最好假设翻译作为多个混淆网络的骨架翻译[1]. 本文中使用一种新的方式, 即采用基于语义的句子相似度来选择骨架翻译. 使用混淆网络主要的困难是把骨架翻译和不同假设翻译进行对齐, 因为不同的假设翻译有不同的词序. 对齐假设翻译的四个流行方法如下:

1) 基于 Levenshtein 编辑距离的多串匹配算法[14];

2) 基于启发式的对齐方法[15];

3) 用训练数据通过 GIZA++ 来进行对齐;

4) 使用 TER 算法来进行对齐.

这些方法在对齐的过程中都没有考虑词义. 由于基于 TER 对齐的方法好于其他对齐方法, 同时也是现在系统融合中的主流方法, 因此我们使用 WordNet 进行词根变换和建立语义关系来对该方法扩展.

2.1 TER 算法的改进

TER 算法主要是通过一系列的转移来计算两个句子的最小编辑距离. 翻译错误率 TER[9] 已经提出更加合理的评估标准. 算法描述如下.

输入. hypothesis H, reference R.

输出. TER 对齐.
步骤 1. 建立 “n-gram table”.
步骤 2. 计算最小编辑距离 min_edit_dis().
步骤 3. 当得到的 shift 集不为空时
  步骤 3.1. 计算最好的 shift 的情况
    1) 计算所有可能的移动 all_poss_shift ();
    2) 对每一种可能的 shift, 求最求小编辑距离;
    3) 令最小编辑距离变小程度最大的 shift 为最好的 shift.
  步骤 3.2. 传递 TER alignment 参数.

TER 的得分可以用下式进行计算:

$$TER(E, E_r) = \frac{INS + DEL + SUB + SHIFT}{N_r} \tag{1}$$

其中, N_r 是参考译文的长度. 与词错误率不同的是 TER 允许转移 (shift) 操作, 一系列词的转移计算成为一个编辑, 最小翻译编辑距离可以通过柱搜索来实现. 当有多个参考译文时, TER 的得分为最接近参考译文的编辑距离除以这个参考译文的长度. 多个参考翻译的最小编辑距离可以通过累加编辑距离和平均参考译文的长度来得到. 最好的 TER 得分为 0. 由于 Insertion 操作, TER 得分可能比 1 高. 最小编辑距离算法的时间复杂度为 $O(n^2)$, n 是单词的个数. 由于使用了柱搜索, 时间复杂度可以缩减到 $O(n)$, 这样可以有效地提高在长句子上的 TER 代码的效率. 为了把 WSD 的得分整合到 TER 算法中以提高系统的性能, 我们把替换的代价改成是 WSD 后的词义相似度的得分, 这样在搜索最小编辑距离时就考虑到了词义相似性.

2.2 混淆网络的解码

我们采用带有任意特征的 Log-linear 模型进行解码[16]. 为了使得解码的假设更加合乎语法结构, 我们引入了语言模型作为一个特征. 以下为解码的公式:

$$\log p(E_{j,n}|F_j) = \sum_{i=1}^{N_j-1} \log \left(\sum_{l=1}^{N_s-1} \lambda_l p(w|l,i) \right) + \upsilon L(E_{j,n}) + \mu N_{\text{null}}(E_{j,n}) + \xi N_{\text{words}}(E_{j,n}) \tag{2}$$

其中, υ 是语言模型权重, 而 $L(E_{j,n})$ 是语言模型 log 概率, $N_{\text{words}}(E_{j,n})$ 是在 $E_{j,n}$ 中的词数, 词的后验概率 $p(w|l,i)$ 是第 l 系统在节点 i 到节点 $i+1$ 的所有词进行归一化后得到的结果. 式 (2) 可以看成是句子级特征的 log-linear 求和. 第一个特征是词的 log 后验概率; 第二个特征是语言模型的 log 后验概率; 第三个特征是 NULL 的 log 后验概率; 最后一个是句子长度的 log 后验概率. 最后两个特征不是完全独立的, 但有助于实验结果.

3 基于 WordNet 词义消歧的实验结果和分析

在实验中选择的开发集和测试集语料分别是 NIST 06 和 NIST 08 的中英文部分, 分别用八个系统进行融合. 在开发集和测试集上最高的 BLEU 得分分别是: 32.60 和 27.75. 在开发集和测试集的每个源语言的翻译是从八个机器翻译系统中得到的. 对大小写不敏感的 BLUE-4[17] 用作评估标准. 我们选用的基线系统是 TER 算法 (加入了同义词和同根词匹配) 来进行系统融合. 加入了 WSD 后的 TER 学习在开发集上的结果见表 1. 在实验中使用的两种语义相似性计算方法, 分别是 SI 1 和 SI 2 (见第 1.2 节), 在选择骨架翻译时使用了两种计算句子相似度的方法: MS 1 (匹配平均) 和 MS 2 (DICE 系数). 加入了 WSD 后的 TER 学习在测试集上的结果见表 2.

表 1 在开发集上整合 WSD 的 TER 学习
Table 1 Integrating WSD into TER in development set

System	WSD TER
Baseline	37.7
SI 1+MS 1	37.8
SI 1+MS 2	38.02
SI 2+MS 1	38.11
SI 2+MS 2	38.21

表 2 在测试集上整合 WSD 的 TER 学习
Table 2 Integrating WSD into TER in test set

System	WSD TER
Baseline	30.6
SI 1+MS 1	30.71
SI 1+MS 2	30.82
SI 2+MS 1	31.93
SI 2+MS 2	31.13

我们发现, 整合了 WSD 的 TER 学习的效果要好于原来经典的 TER 学习. 在这些加入 WSD 的 TER 中, 最好的方法是使用 Wu 和 Palmer 提出的方法来计算相似度, 使用 DICE 系统来获得骨架翻译. 但是性能不是提高很多, 因为我们选择的八个翻译系统相似性较大, WSD 对性能的影响不是很大. 如果选用的系统相差性很大的话, 会得到更好的效果.

4 结论

本文在原有 TER 对齐建立混淆网络的基础上, 加入词义的相关知识, 提高了在 TER 算法中同义词的对齐. 对于传统的基于 WordNet 的消歧方法进行

了改进, 并把词义消歧的相关知识加入到句子相似度的计算当中, 从而确定混淆网络中的骨架翻译. 由于在传统的对齐模型中, 语义相关的词很难进行对齐, 同时也没有考虑到语境对于对齐的影响. 为了更好地使语境在对齐中起到重要的作用, 使用了改进的 Lesk 算法来进行词义消歧.

References

1 Och F J, Ney H. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 2003, **29**(1): 19−51

2 He X D, Yang M, Gao J F, Nguyen P, Moore R. Indirect-HMM based hypothesis alignment for combining outputs from machine translation systems. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Honolulu, USA: Association for Computational Linguistics, 2008. 98−107

3 Karakos D, Eisner J, Khudanpur S, Dreyer M. Machine translation system combination using ITG-based alignments. In: Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies. Columbus, USA: Association for Computational Linguistics, 2008. 81−84

4 Rosti Antti-Veikko I, Ayan N F, Xiang B, Matsoukas S, Schwartz R, Dorr B. Combining outputs from multiple machine translation systems. In: Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics. New York, USA: Association for Computational Linguistics, 2007. 228−235

5 Rosti Antti-Veikko I, Matsoukas S, Schwartz R. Improved word-level system combination for machine translation. In: Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics. Prague, Czech Republic: Association for Computational Linguistics, 2007. 312−319

6 Li C, He X D, Liu Y P, Xi N. Incremental HMM alignment for MT system combination. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP. Suntec, Singapore: Association for Computational Linguistics, 2009. 949−957

7 Akiba Y, Watanabe T, Sumita E. Using language and translation models to select the best among outputs from multiple MT systems. In: Proceedings of the 19th International Conference on Computational Linguistics. Taipei, China: Association for Computational Linguistics, 2002. 1−7

8 Wang B, Zhao T J, Yang M Y, Jiang H F, Li S. Stability vs. effectiveness: improved sentence-level combination of machine translation based on weighted MBR. In: Proceedings of the International Conference on Asian Language Processing. Singapore, Singapore: IEEE, 2009. 39−42

9 Huang F, Papineni K. Hierarchical system combination for machine translation. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. Prague, Czech Republic: Association for Computational Linguistics, 2007. 277−286

10 Snover M, Dorr B, Schwartz R, Micciulla L, Makhoul J. A study of translation edit rate with targeted human annotation. In: Proceedings of the 7th Conference of the Association for Machine Translation in the Americas. Cambridge, USA: Association for Computational Linguistics, 2006. 223−231

11 Christiane F. WordNet: an electronic lexical database [Online], available: http://WordNet.princeton.edu, March 12, 2009

12 Xu J, Zens R, Ney H. Sentence segmentation using IBM word alignment model 1. In: Proceedings of the 10th Annual Conference of the European Association for Machine Translation. Budapest, Hungary: Association for Computational Linguistics, 2005. 280−287

13 Wu Z, Palmer M. Verb semantics and lexical selection. In: Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics. Las Cruces, New Mexico, 1994. 133−138

14 Bangalore B, Bordel G, Riccardi G. Computing consensus translation from multiple machine translation systems. In: Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding. Washington D. C., USA: IEEE, 2001. 351−354

15 Jayaraman S, Lavin A. Multi-engine machine translation guided by explicit word matching. In: Proceedings of the Association for Computational Linguistics on Interactive Poster and Demonstration Sessions. Ann Arbor, USA: Association for Computational Linguistics, 2005. 101−104

16 Sim K C, Byrne W J, Gales M J F, Sahbi H, Woodland P C. Consensus network decoding for statistical machine translation system combination. In: Proceedings of the International Conference on Acoustics, Speech and Signal Processing. Honolulu, USA: IEEE, 2007. 105−108

17 Papineni K, Roukos S, Ward T, Zhu W J. BLEU: a method for automatic evaluation of machine translation. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics. Philadelphia, USA: Association for Computational Linguistics, 2002. 311−318

刘宇鹏 哈尔滨工业大学博士研究生. 主要研究方向为机器翻译和系统融合. 本文通信作者.
E-mail: ypliu@mtlab.hit.edu.cn
(**LIU Yu-Peng** Ph. D. candidate at the School of Computer Science and Technology, Harbin Institute of Technology. His research interest covers machine translation and system combination. Corresponding author of this paper.)

李 生 哈尔滨工业大学教授. 主要研究方向为自然语言处理和机器翻译.
E-mail: lisheng@mtlab.hit.edu.cn
(**LI Sheng** Professor at the School of Computer Science and Technology, Harbin Institute of Technology. His research interest covers natural language procession and machine translation.)

赵铁军 哈尔滨工业大学教授. 主要研究方向为自然语言处理和机器翻译.
E-mail: tjzhao@mtlab.hit.edu.cn
(**ZHAO Tie-Jun** Professor at the School of Computer Science and Technology, Harbin Institute of Technology. His research interest covers natural language procession and machine translation.)