基于百科知识的中文自动问题生成技术的研究与系统实现

摘　　要

传统的互联网搜索引擎以关键词为媒介处理用户的信息需求。一般而言，用户得到的信息反馈是海量网页的超链接。为了缓解用户对这些信息反馈进行再筛选的压力，对检索到网页进行排序的方法的研究得到了很大关注。然而，这种以关键词为输入，以排序后的网页超链接为输出的信息检索方式并没有从本质上改善人类信息获取的效果。最新研究表明，人类的信息需求往往是以自然语言问题的形式进行表达，而期望得到的是对于所提问题的精确回答。因此，未来信息获取的方式将会面向问答系统的方向发展。本文将从对现今问答系统的构建方法的研究入手，发现其分析问题的不足，提出一套以句法分析为基础，以候选知识为答案资源自动生成对应问题的算法。同时为加强自然语言问题生成的灵活度，提高问答系统的性能，该文进一步深入探讨相似问答对的搜索模式，研究出新颖的联想式问题标签生成方法，帮助用户快速查找与信息需求相关的问答对。实验表明，本文提出的一系列算法均能够有效提升现行问答系统的性能。同时，从工程角度，以 Java Web 开发技术为基础，有机结合上述算法，搭建了名为 AQ (Answer-Question) 的基于百科知识的问答平台。

关键词　问题生成　问答系统　标签

Chinese Natural Question Generation Scheme Study and System Implementation Based on Wiki Knowledge

ABSTRACT

Traditional web search engine uses key phrases as the media to deal with users' information needs. In general, the feedback to user is massive web page hyperlinks. To alleviate the burden of re-screening the feedback and retrieve more relevant information, page ranking method has drawn much attention during the past decade. However, the information retrieval methods, such keywords as input and sorted web page hyperlink as output, cannot essentially improve the effects of human access to information. Recent research shows that human beings often express their information needs in the form of natural language questions and expect to get precise answers to the questions, which leads the rising development of question answering system. This paper will first survey state-of-the-art approach on question answering system and then find out its defects. Then in order to remedy the defects, we propose a Chinese natural question generation scheme supported by syntactic analysis based on wiki knowledge. At the same time, to strengthen the flexibility of the scheme and improve the performance of question answering systems, this article further discuss the method on similar question searching, and come up with a novel associative tag generation model to help users quickly find relevant question-answer pairs so as to satisfy their information needs. The experiments demonstrate the proposed algorithm can effectively improve the performance of state-of-the-art question answering system. What's more, from the perspective of engineering, we have developed a Java web based wiki question answering platform named as AQ, which implements the proposed scheme.

KEY WORDS question generation question answering system tag

目　　录

第一章　引言

本章将首先概述研究的背景情况，然后就中文自动问题生成的研究动机予以阐述，进而提出本文所要论述的研究内容，即中文自动问题生成算法的研究，用于强化基于上述算法的问答系统性能的问题标签推荐算法的研究，以及原型系统的设计与实现。

1.1　研究背景

近二十来，网络的飞速发展使得互联网信息爆炸性地增长。用户共同创造的海量电子化文本信息在带给我们无限成就感的同时，也不可避免地引来了信息泛洪的问题，使得用户无法有效地获取其需要的信息。为了帮助用户快速满足其信息需求，二十世纪 90 年代，出现在互联网上的搜索引擎[1]应用，如 Google, Yahoo 等，得到了无数用户的青睐。搜索引擎以其简洁的交互方式，帮助用户以短小的关键词的形式获取内容相关的网页链接，同时借助如 PageRank[2]等高效排序算法推荐出最可能符合用户信息需求的一系列网页超链接。

然而，用户在频繁地使用传统搜索引擎的过程中，也发现其存在许多需要改进的地方。其中最为突出的便是用户与搜索引擎在信息获取模式的不契合性，即用户的信息需求往往以问题的形式提出[3]，而搜索引擎的检索模式却要求用户将问题形式的信息需求用关键词来表述，而检索的成功与否则要依赖于用户关键词的质量，这样给相当一部分用户带来了不便。因此用户迫切需要一种新的信息获取机制，能够处理其自然语言表述的问题需求，这种刚性需求也进而激发了研究人员对于问答系统的研究兴趣。

1.2　研究动机

问答系统是能够接受用户以自然语言形式描述的提问，并能从大量的异构数据中查找或推断出用户问题答案的信息检索系统[4]。一般问答系统的架构如图 1-1 所示：

从图 1-1 可以看出，问答系统依赖于提问处理模块、检索模块以及答案抽取模块的叠加，同时意味着最终答案的精确度等于上述三个模块各自处理性能的乘积。但是由于用户提问的语言的多样性和非标准性造成对其句法分析的困难。因此本文试图另辟蹊径，从语言规整的百科文档入手，以句子中的知识点为答案，逆向自动生成针对于这些预定答案的规整问题。同时为了弥补用户提问句法不规则所造成的难以理解的缺陷，进一步探讨如何生成问题中的标签提供语义相似问答对满足用户的信息需求。

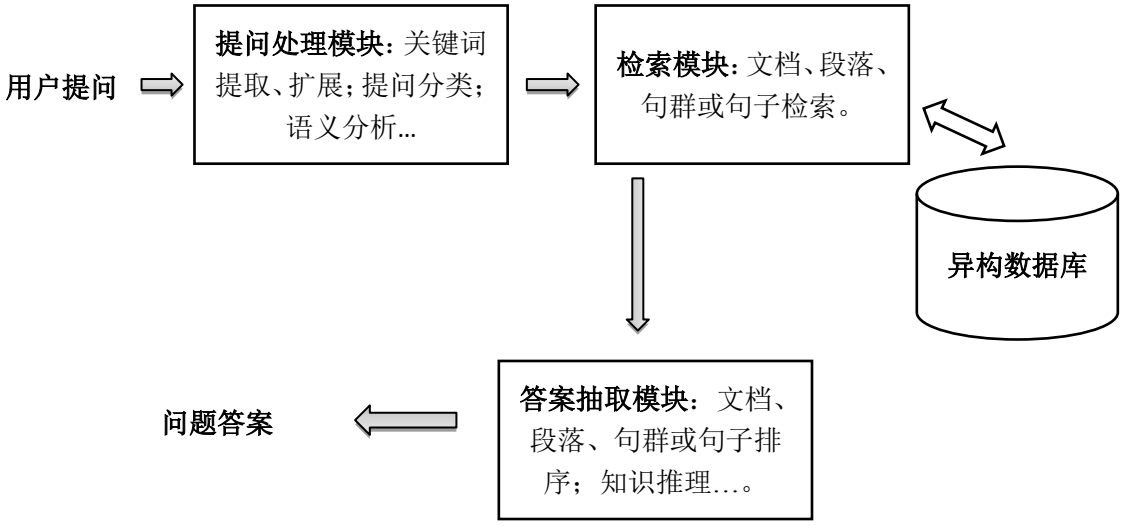


图 1-1 问答系统一般架构

1.3 研究目标

本文的研究共探讨如下三方面内容：

- 基于百科知识的中文自动问题生成技术的研究
- 在自动生成的问答对的基础上，为问题生成语义标签，提高问题检索的灵活性
- 在上述资源的基础上，构建 Java Web 为实现手段的问答系统

第二章　相关研究

本章将对自动问题生成研究领域以及标签推荐研究领域的相关研究进行归纳总结，进而分析现有研究方法的利弊，同时对本文的研究工作具有对比和借鉴意义。

2.1 自动问题生成算法相关研究

自动问题生成（Question Generation）是一个新兴的研究领域，这项研究的出发点在于，希望可以通过自动生成大量具有良好语言特征的问题来帮助人们完善自身对信息需求的提问[5]。从 2008 年起，每年都会召开一次国际研讨会[6]方便世界各地的研究人员交流最新的研究成果，迄今为止已成功举办 4 届，涉及问答系统[7]、对话生成[8]、智能教学[9]等多个国际和地区的研究人员（德国、印度、加拿大、美国、英国等）。并且在 2010 年的研讨会上，首次举行了自动问题生成的评测。评测主要根据产生问题的语料规模分为基于句子的问题生成和基于段落的问题生成[10,11,12]。总体而言，目前的研究大多集中在基于句子的问题生成上[7,10,11,12]，仅有宾夕法尼亚大学参与了段落级别的问题生成评测[13]。基于句子的问题生成的主流方法是，在对句子进行句法分析和命名实体识别后，根据预先制定的规则[7, 14]调整语序生成对应的问题。尽管卡耐基梅隆大学 LTI 中心于 2009 年提出可以通过过量产生问题并利用统计的方法对若干候选问题进行打分从而增加问题的多样性，但是其中问题产生的核心仍然是利用规则的方法[15,16]。

2.1.1 输入与输出

自动问题生成技术其实在 2008 年前就有一些学者从事着相关工作[17,18]。然而，真正将自动问题生成（QG）作为一项研究议题并且提出每年组织一次专门的会议对这个议题进行讨论，还要归功于 Paul Piwek 等人在 2008 年发表的论文[19]。文中除对自动问题生成技术的光明前景做了一些分析外，还分别从自动问题生成的输入、输出以及二者之间的关系三个方面对这一技术做了前瞻性的分析。

- 输入

自动问题生成的输入形式不仅仅局限于语言或者语音中的句子，当然不一定是陈述句，其他形式的如祈使句甚至是疑问句均可。Paul 等人认为图像、图表、手势或者是这几种形式的组合都可以作为自动问题生成的输入。笔者认为这恰恰和搜索引擎的技术革命是同步的。目前的信息检索向着多媒体搜索和开放领域问答方向发展，而自动问题生成技术刚好可以作为过渡时期的自然语言问答搜索的一部分[3]，同时其输入还与搜索引擎的底层处理的数据多样性相吻合[20]。

- 输出

与输入对应，输出的形式也同样包括语言、语音、图像等一系列的形式的问题。虽然 Paul 等人没有具体说明这些输入与输出形式是否是一一对应的，但是笔者认为输入与

输出形式多对多的映射才是我们研究处理的终极目标。比如，我们从文本语言中产生出的问题形式可以语言文本化的问题，也可以是语音问题，甚至是图片、手势等等。

- 输入与输出的关系

1. 输出问题补完输入查询关键词

这种关系最广泛是应用在搜索引擎中[21]，特别是在搜索引擎逐渐向着问答系统方向革命的过程中，在保证既不会立刻转变用户的输入习惯，又可以使得搜索引擎向着精确问答过渡。于是，对用户输入查询的关键词进行合理的自然语言问题补完就成为了一种新的用户接口模式，见图 2-1。

图 2-1 搜索引擎问题补完

2. 输出问题是对输入答案的提问

与问题相对应的是答案，这是再自然不过的关系。而以答案作为输入，问题作为输出就类似教师出考题的过程。因此，这种关系模式最早用在智能教学领域，为了减轻教师的工作量，同时也提高学生的学习效率和掌握知识的能力。然而笔者认为这种模式同样适用于目前问答系统的构建，自动问题生成技术作为一个能够自动生成大量问答对的模块不仅可以快速定位用户部分潜在问题，同时也可以帮助用户补完其表述不清的信息需求，本文将对这种模式做深入探讨。

3. 输入问题与条件，输出推理问题

这种模式多见于复杂的原因与推理领域。常常根据上文推理出与上文表意一致的问题。

输入: 如果约翰的车在车库中，那么他一定在家。他在家吗？

输出: 约翰的车在车库中吗？

2.1.2 技术方法

- 基于手工规则的方法

卡内基梅隆大学 LISTEN 项目组的研究员，根据领域特点，采取手工编制适合智能教学方面的问题句子规则，然后根据学生的反馈向这些提问规则中填充词语来生成问题[22]。规则样例如下：

What did <character><verb>?

Why/How did <character><verb><complement>?

Why was/were <character><past-participle>?

- 基于日志模板的方法

百度研究人员利用海量用户使用百度搜索以及百度知道的日志进行问题模板提取[21]，流程见图 2-2。百度的研究人员在这些日志信息中找出所有如下描述的日志行为模板：用户在搜索引擎中输入关键词之后直接到百度知道中提问了一个问题。于是研究人员假定出现这种行为的原因多是用户利用关键词没有在搜索引擎中找到适合其问题的答案，因此借助社区问答平台求助。而由于用户提问的关键词与后续问题有人工补完的关系，所以我们可以通过替换掉这些出现在问题中的关键词来获取问题模板，从而帮助后续的用户补完搜索中的问题。

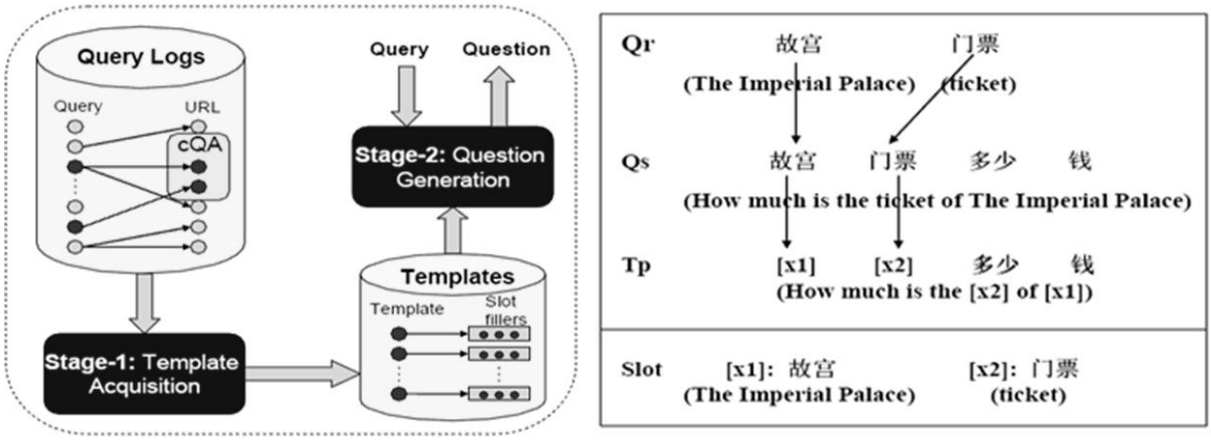


图 2-2 日志模板提取方法图[21]

- 基于句法和关键词的方法

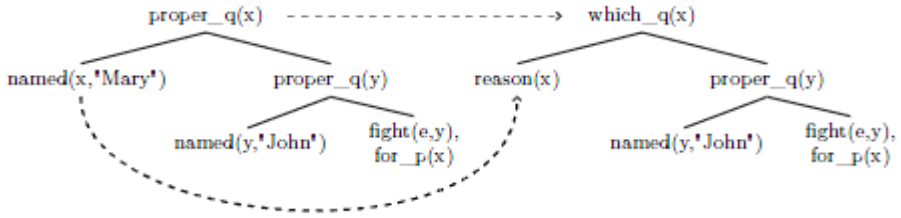
随着问题产生需求的逐渐扩大，研究人员希望可以直接从文档中提取句子甚至是段落中的问题。一些研究人员认为要对文档中的句子和段落区别对待[7,23]，句子级别的问题生成通过预先制定好的句法模板与提问成分之间的映射将候选句子一一选入固定的问题集中，然后根据不同的问题类型调整语序达到问题生成的目的。而对于段落级别的问题，则是通过筛选关键词，并根据关键词的语义类型采用固定的模板生成定义型的问题[7]。

- 基于句法依赖关系模板方法

还有学者希望可以将问题产生技术应用到问答式的搜索引擎中，作为搜索引擎向问答系统的平滑过渡技术一部分[3]。但是由于在搜索引擎所处理海量开放文本下很难通过人工制定模板来产生足够的问题，因此这些研究人员综合利用句法依赖分析和命名实体识别技术来从海量的文本中自动抽取需要的候选问题模板。方法是首先对文档进行分句处理，然后对每个句子进行命名实体识别，根据实体类型映射到不同的提问形式，接着自动调整语序生成一套依赖句法模板对应问题模板的集合。

- 基于语义分析的方法

有些学者认为，仅仅依靠句法分析这种浅层语言分析还不足以达到自动生成问题的质量标准，特别是如果仅靠句法模板有时产生的问题往往不可读。因此，研究人员期望通过找到句子中成分间存在的语义关系来对应确定问题的类型以及形式。目前，有学者提出以动词为核心的主宾关系块作为产生问题的基础，同时分别分析成分的语义类型作为辅助确定问题的类型。图 2-3 为一个原因型问题的例子。



(d) "John fights for Mary." → "Why does John fight?"

图 2-3 基于语义的问题生成

通过对上述关于问题生成方面的研究的分析，本文总结自动问题生成技术的现状如表 2-1 所示。

表 2-1：自动问题生成技术现状

	规则	**日志模板**	**句法分析**	**依存分析**	**语义关系**
应用领域	智能教学	搜索引擎	问答系统（不明确）	问答搜索引擎（革新）	未知
实现基础	手工语句规则	用户搜索日志	句法树中被提问节点的句法查找路径	整体句法依赖树模板	以动词问中心的关系三元组
问题类型	不受限	不受限	命名实体类提问	命名实体类提问	命名实体类型
总结	手工制定，领域受限	无法推广，仅局限在搜索引擎日志分析	依赖句法分析和命名实体识别	依赖句法依存关系	问题非常简单

2.1.3 评价指标

QGSTEC 2010 是迄今第一次针对问题自动生成的评测会议，会议分为两大任务，分别是从单句中生成规定的目标问题以及从段落层面上自动生成6类不同提问范围的问题。

Task A QG From Paragraph	**Input:** Abraham Lincoln(February 12, 1809-April 15, 1865), the 16th President of the United States, successfully led his country through its greatest internal crisis, the American Civil War, preserving the Union…
	Questions: Who is Abraham Lincoln?(General); What major measures did President Lincoln introduce?(Medium); How did President Lincoln die?(Medium); When was Abraham Lincoln elected president?(Specific); When was President Lincoln assassinated?(Specific); What party did Abraham Lincoln belong to?(Specific)
Task B QG From Sentence	<instance id="3"> … <text>The poet Rudyard Kipling lost his only son in the trenches in 1915</text> <question type="when"> ***When did Rudyard Kipling lose his son?***</question> <question type="how many">***How many sons did Rudyard Kipling have?***</question> </instance>

图 2-4：QGSTEC 2010 评测语料样例

图 2-4 分别展示了 QGSTEC 2010 的自动问题生成评测语料的两类任务（A 和 B）的评测样例。

对于任务 A，要求参评系统从上述一段文字中生成 1 个以整段话作为答案的总结性问题，2 个以多句话作为答案的区域性问题以及 3 个以其中单句作为答案的问题。

对于任务 B，要求参评系统读入指定的句子以及要求生成的问题类型，然后对应输出规定类型下的问题。

如图 2-5 所示，评测的时候，引入 5 类指标通过多位人工判定的方式对这些系统所生成的问题进行打分，这 5 类指标包括：相关度，问题类型，提问正确性以及流利程度，明确度，多样性。打分由 1-4 代表评价逐级下降。

category / score	Relevance	Question type	Syntactic correctness and fluency	Ambiguity	Variety
1(Best)	√		√		
2		√			
3				√	
4(Worst)					√

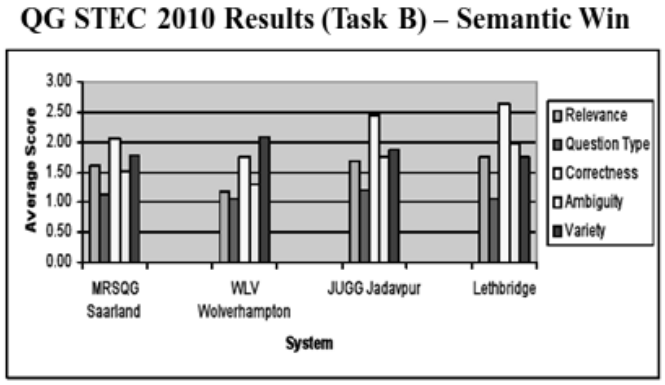


Figure 1: Results for QG from Sentences (without penalty for missing questions)

图 2-5：人工测评表样例以及 QGSTEC 2010 评测结果

2.2 问题标签生成算法相关研究

社会标签 (social tags)，也被称作大众分类法 (folksonomy)[23]，是一种新型的网络资源索引方法，允许用户用自造的文字标签来标注资源（微博、博客、电影、图书、音乐、照片等等），而对标签的选词范围和个数上没有强制要求。由这一方法所构建的系统（社会化标签网站应用），是 Web 2.0 时代以来，采用了“众包”[24]这一思想的新型网络应用模式。在这类以社会标签模式构建的网络应用中，用户可以自由上传资源，并且可以对他们感兴趣的网络资源标注文字化的标签。为了降低用户手动标注的负担，同时也为了提高用户参与标注的兴趣，标签推荐系统，作为可以向用户自动提供合适标签的功能，近几年来得到了来自研究界和工业界的广泛关注[25,26]。图 2-6 展示了新浪爱问中某问答中的问题标签，这种标签不仅便于对问题的进行分类，同时也以关键词的形式表述了提问的核心内容。

图 2-6：新浪爱问标签标注的问答对（红色图框）

2.2.1 技术方法

标签推荐算法主要分为基于协同过滤的[27,28]和基于内容的推荐方式[29]。基于协同过滤的推荐方式需要根据用户的标注历史或者资源本身被标注的情况推荐相似对象拥有的标签。显然这类方式存在很大的缺陷，研究界经常称这种缺陷为“冷启动”，即对于缺乏标注历史信息的资源，比如冷门资源，新资源等，协同过滤的推荐方式很难找到与他们相似的其他资源，因此，这些缺乏信息的资源就很难被推荐给用户。

为了弥补“冷启动”这一缺陷，基于内容的推荐方式着眼于资源的描述本身，因此不管是热门资源，冷门资源甚至是新资源，只要拥有关于其本身的描述，就能够通过相应的算法从这些描述中提取潜在的标签。一种很直接的想法就是把基于内容的推荐方法视作分类问题，将标签视为类别，根据已有标签的资源描述构建分类器，进而预测一个未标注内容的标签。常用的算法如 KNN[30]，SVM[31]等。然而分类算法对训练数据，特别是用于分类的标签的准确度的依赖很强。社会标签的实质就是发掘用户的自由和个性，因此标注出来的标签也是五花八门，各有其特色，如果将这些数据用于分类，结果自然不理想。于是 X. Si[32,33]等对话题模型 LDA 进行调整，提出 TAG-LDA 模型，同时对标签产生的原因进行分析建模，试图削减个性化标签所带来的噪音影响，这类方法虽然比较有效，但是削弱了个性化标签的推荐效果，并且 X. Si 等的观点认为每个标签的生成原因都只源自于资源内容中的一个词。事实上，用户决定一个词语是否可以被认作某一

资源的标签通常需要依靠对资源全文的理解，即常常帮助用户决定一个词是否能够作为标签的内容词汇不只一个。

所以 Z. Liu 等[34] 2011 年发表的方法采用统计机器翻译的词对齐模型，在大训练集下学习出内容中的词汇触发出潜在标签词汇的概率，对于给定的某个资源描述，通过对触发出的标签作简单的加权求和处理就可以获得比较好的推荐效果，原因在于这种方法没有忽视个性化标签的作用，只要是存在潜在的触发关系的词汇对都予以考虑。但是这种最新的研究成果也同样也暴露出问题，Z. Liu 的简单加权求和的方法很容易放大从训练数据中习得的噪音。

2.2.2 评价指标

一般引入精确度、召回率、F1 值等指标来评价标签推荐算法的性能[34]。对于一个给定的待推荐标签的资源集合R，如果其人工标注的标签集合为T_O，使用算法自动推荐的标签集合为T_R。那么正确推荐标签的集合可以被表示为$\mathrm{T}_R \cap \mathrm{T}_O$。因此精确度、召回率、F1 值可以分别表征为式 2-1，

$$\mathrm{p} = \frac{|\mathrm{T}_R \cap \mathrm{T}_O|}{|\mathrm{T}_R|}, \mathrm{r} = \frac{|\mathrm{T}_R \cap \mathrm{T}_O|}{|\mathrm{T}_O|}, F1 = \frac{2\,p\,r}{(p+r)} \qquad \text{式 } 2-1$$

第三章　中文自动问题生成方法

本章将介绍中文自动问题生成的方法。按照方法的研究实现流程，共分为文本预处理（3.1 节）、“答案-问题”模板提取（3.2 节）、问题自动生成（3.3 节）。

3.1 文本预处理

要处理句子级别的自动问题生成，针对于百科文档，首先需要对文档中的句子进行预处理。处理内容包括对通篇文档进行分句、指代消解以及对于可以产生多种类型问题的句子进行合理的切分和补完，使得无论是对于“答案-问题”模板的提取还是中文问题的自动生成都有帮助意义。

1. 文档分句

我们使用哈尔滨工业大学的 LTP[35]对百科文档进行简要的分句。这种分句方法主要依靠识别句末具有截断意义的标点来实现。

2. 指代消解

由于句子受到上下文的影响，常常为了语言简练和流畅使用代词代指上文的主语，然而我们的句子级别生成问题时与上下文无关的，因此需要对以代词作为主语的句子进行指代消解。

3. 句子的切分和补完

经过对语料的观察，我们发现一部分句子常常包含多种潜在的问题模式，或者包含多个相同类型问题提问的可能，比如：“**猕猴桃除含有猕猴桃碱、蛋白水解酶、单宁果胶和糖类等有机物，以及钙、钾、硒、锌、锗等微量元素和人体所需 17 种氨基酸外，还含有丰富的维生素、葡萄酸、果糖、柠檬酸、苹果酸、脂肪”。**这句话包含了 2 个列举型的问题和一个事实型问题，分别是：

Q(List):猕猴桃含有**哪些**有机物？（问题 1）

Q(List):猕猴桃含有**哪些**微量元素？（问题 2）

Q(Fact):猕猴桃含有人体所需的**多少**种氨基酸？（问题 3）

而产生这些问题的前提是，我们需要对例句切分并补完为以下三个短句：

A:**猕猴桃除含有猕猴桃碱、蛋白水解酶、单宁果胶和糖类等有机物。（答案 1）**

A:**猕猴桃含有钙、钾、硒、锌、锗等微量元素。（答案 2）**

A:**猕猴桃含有人体所需 17 种氨基酸。（答案 3）**

因此，我们无论是在抽取模板还是生成被测语料前都需要对句子进行切分和补完。

3.2“答案-问题”模板提取

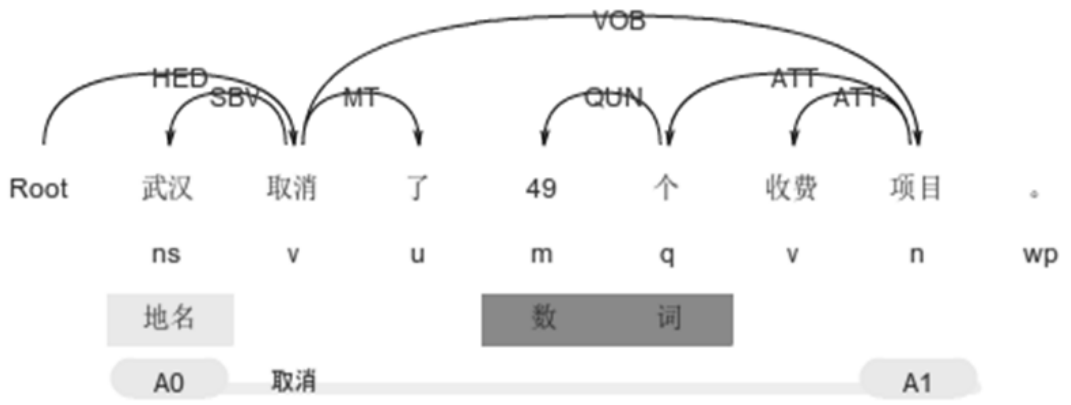
因为这是对中文自动问题生成的首次探究，因此我们需要根据中文自然语言处理的特点从浅层语言表示到深层语言表示等多种模板角度进行分析。我们为了探究深层语言分析是否提高对中文自动问题生成的性能，计划提出如下“答案-问题”模板：

- 分词+词性标注模板
- 分词+词性标注+依赖关系模板
- 句法依赖树前序遍历模板
- 引入领域词典句法依赖树前序遍历模板

以如下问答对为例，使用 HIT-LTP 对其中的答案进行词性标注和句法依赖分析，对问题进行分词处理，分别叙述上述 4 个模版的提取流程，详见图 3-1。

问题：武汉取消了多少个收费项目？

答案：武汉取消了 49 个收费项目。



问题:武汉 取消 了 多少 个 收费 项目？

图 3-1 问答对的分词、词性标注、句法依赖分析样例

- 分词+词性标注模板

图 3-1 中的答案句“武汉取消了 49 个收费项目”对应的分词以及词性标注序列为：{武汉:ns}{取消:v}{了:u}{49:m}{个:q}{收费:v}{项目:n}。其中每个词汇后面的字母代表其词性，例如：武汉为名词中的地名，因此根据中文词性标注规范，其被标注为 ns。这样得到如图 3-2 上半部所示的处理后的问答对。在提取问答模板的时候，消除问答对中相同的词汇，只保留对应的词性标注，如图 3-2 下半部所示。

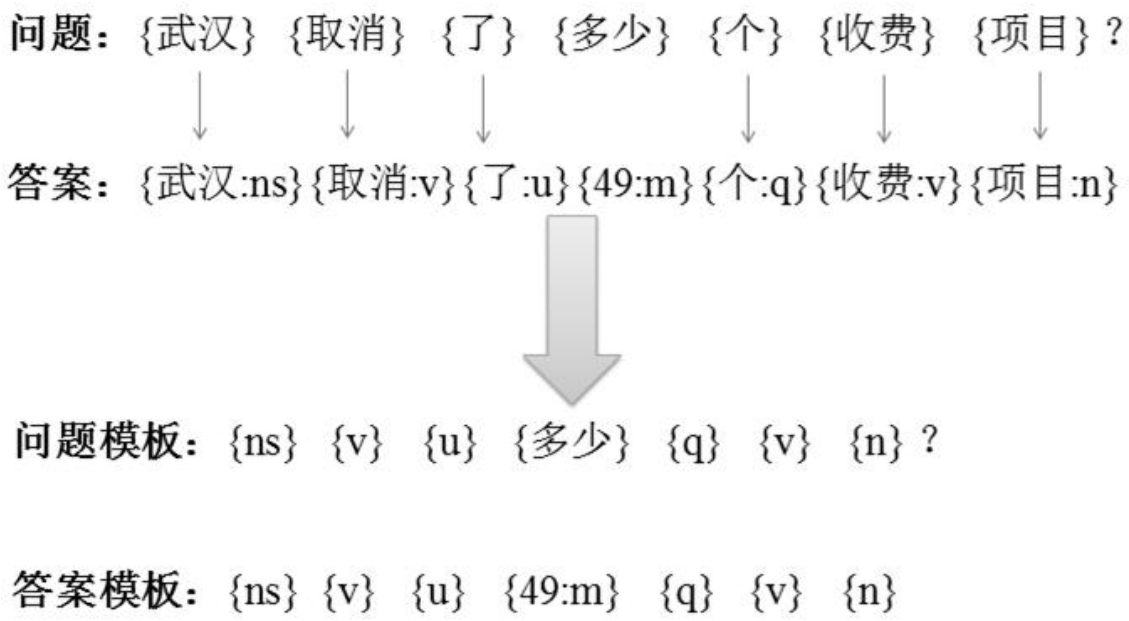


图 3-2 分词+词性标注问答模板样例

- 分词+词性标注+依赖关系模板

图 3-3 中的答案句“武汉取消了 49 个收费项目”对应的分词、词性标注以及句法依赖序列为：{武汉:ns:SBV}{取消:v:HED}{了:u:MT}{49:m:QUN}{个:q:ATT}{收费:v:ATT}{项目:n:VOB}。其中每个词汇后面的字母分别代表其词性和依赖关系，例如：武汉为名词中的地名，因此根据中文词性标注规范，其被标注为 ns；同时其与父节点{取消:v:HED}的关系为 SBV，因此进一步标注为{武汉:ns:SBV}。这样得到如图 3-3 上半部所示的处理后的问答对。在提取问答模板的时候，消除问答对中相同的词汇，只保留对应的词性标注和句法依赖关系，如图 3-3 下半部所示。

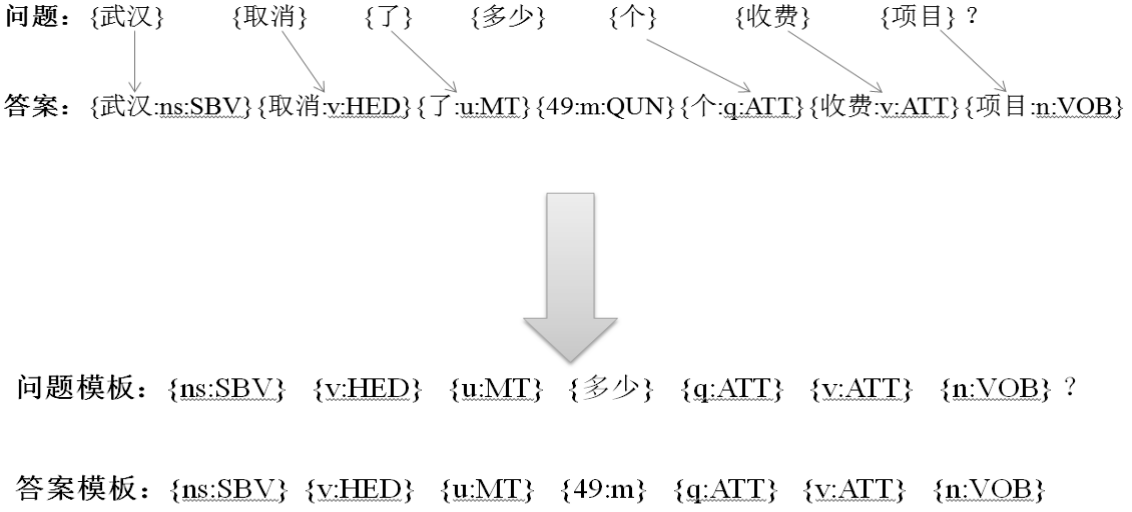


图 3-3 分词+词性标注+句法依赖问答模板样例

- 句法依赖树前序遍历模板

首先需要对问答对中的答案进行句法分析（图 3-1），这样得到如图 3-4 中所示的句法依赖树结构。

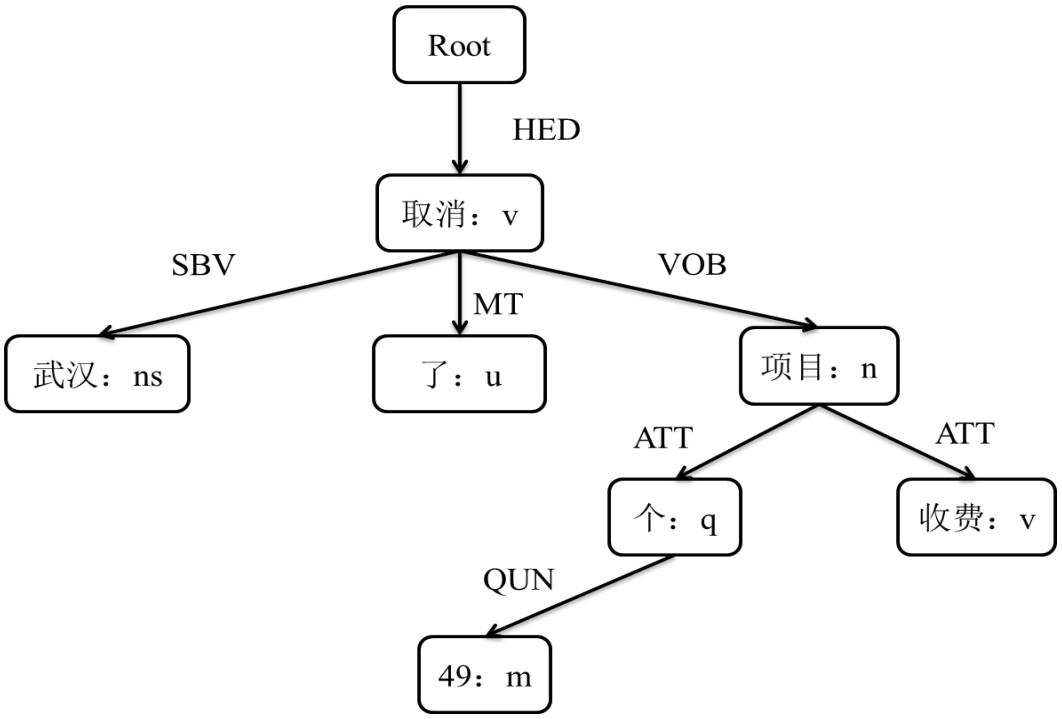


图 3-4 句法依赖树样例

对图 3-4 中的树进行前序遍历，@{1:取消:v}代表词汇节点，#{DOWN:HED}代表树的边。遍历从节点@{ROOT}开始，下行(DOWN)经过名为 HED 的依赖关系的边(整体表征为#{DOWN:HED})到达节点@{1:取消:v}，依次类推，直到回到@{ROOT}节点结束，得到图 3-5 中的结果。同时对问题进行分词。然后将分析后得到的词汇进行一一匹配，找出其中相同的部分，将词从模板中剔除掉，得到图 3-5 下半部分所示的句法依赖问答模板。

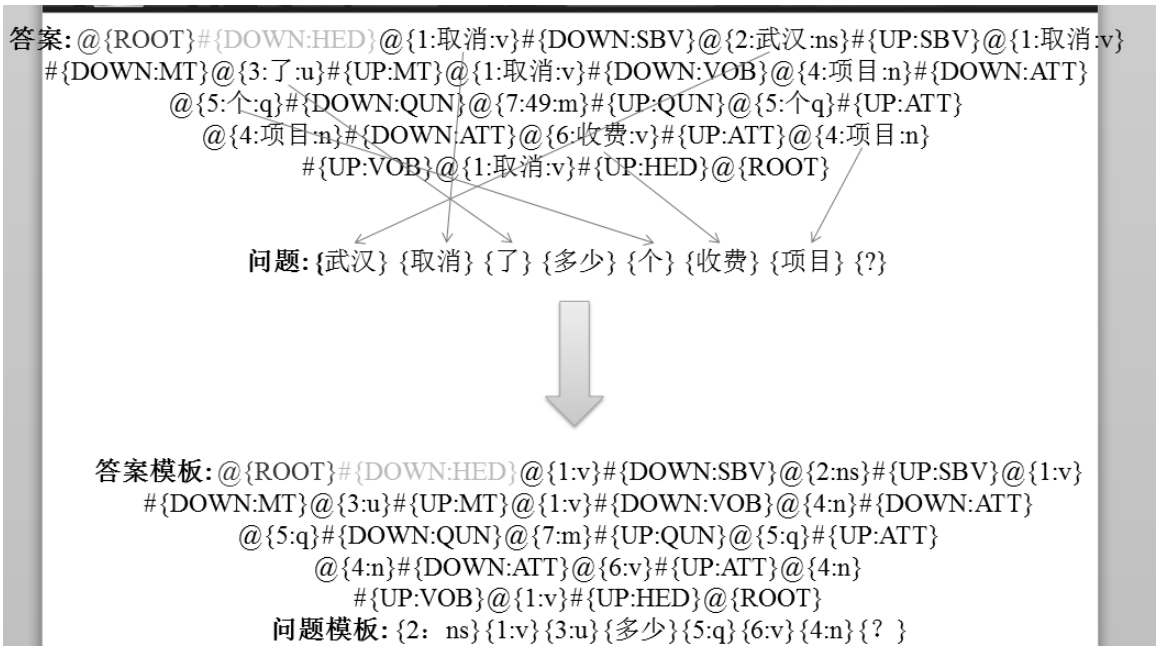


图 3-5 句法依赖问答模板生成

- 引入领域词典的句法依赖树前序遍历模板

本模板的生成方式与图 3-5 中相似，只是由于引入领域词典，提高分词的精度，因此本部分不再赘述。

3.3 问题自动生成

中文问题的表达方式千差万别，在现有的网络问答平台（新浪爱问、百度知道）向用户展示了人们曾经提出的近百万条各式各样的问题。尽管如此，我们仍然可以从中找到这些问题的共性加以分类。目前，网络平台为了便于检索，采用与提问中心词相关的语义类别进行划分。然而，问题的类别不是一成不变的。Graesser 提出问题分类的几点原则：问题的目的、问题寻求的信息类型、答案的来源、答案的文本量以及问题涉及的认知过程。

鉴于以上原则，结合我们从百科文档生成问题这一任务的需要，我们通过对新浪爱问和百度知道上面用户提出的问题进行人工分析和统计，根据产生问题源的文本量进行分类（见表 3-1），用于指导后续的中文自动问题产生技术方法的选择。

表 3-1：问题分类

问题源文本量级别	问题类型	问题特点描述	百科问题例句
简单单句级别	是非型	用户期望得到是或非的回答。	奇异果是猕猴桃吗? 猕猴桃营养价值高吗?
	事实型	用户期望得到如人名，地点，组织，时间等命名实体的简短答案。	猕猴桃是什么时候传入中国的? 猕猴桃最先出现在哪里?
	定义型	用户期望得到对于某个概念的定义性描述。	猕猴桃是什么?
	引用型	网络问答的特殊种类，用户期望得到的参考网址。	请给我猕猴桃健康吃法相关的网址?
复杂单句级别	列举型	用户将会得到一组答案，这些答案通常针对问题存在语义并列关系。	猕猴桃的营养元素都有哪些?

复杂单句或者段落级别	原因型	用户想要探究某个现象的原因，答案往往是一个复杂句子或是一段文本。	猕猴桃的英文为什么叫kiwi fruit?
段落级别	解决方案型	用户期望得到某个一组解决方案，通常是步骤或者是方法等。	怎么种猕猴桃? 猕猴桃怎么吃?

我们的研究将从句子级别入手，对是非型、事实型、定义型、列举型和部分原因型问题进行分析，得出实用的中文问题生成方法。图 3-6 以是非型和定义型两种目标问题生成为例，依托百科知识，展示了中文自动问题生成的方法流程。

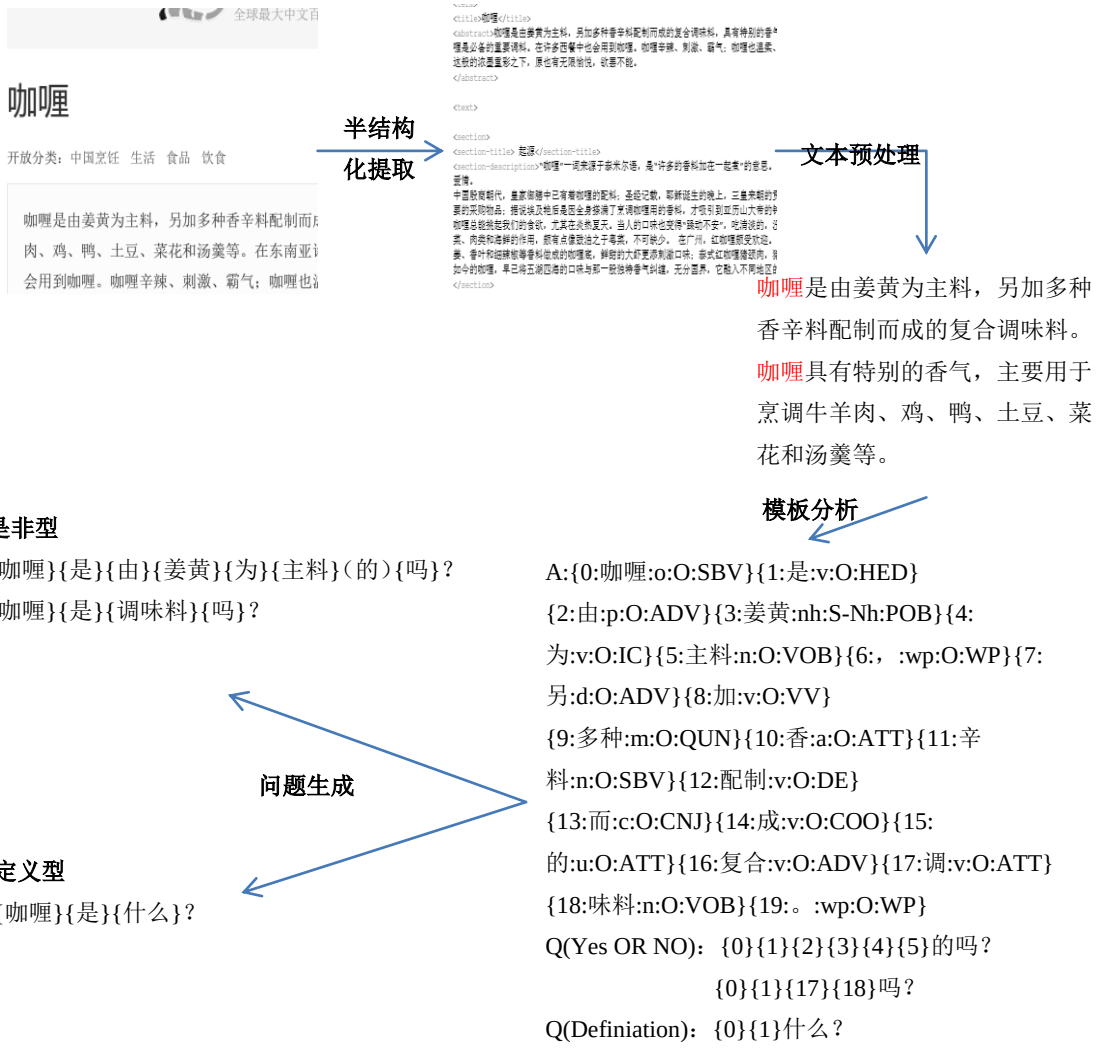


图 3-6：中文问题生成方法流程

3.4 小结

尽管本章提出了一套针对于百科知识（答案）逆向自动生成问题的算法流程，然而，从算法的描述和分析中，不难发现，本文提出的中文自动问题生成算法存在着一个明显的缺陷，即模板匹配所带来的问答对应僵化问题，即答案必须在完全符合模板的前提下，才能生成对应问题。这种缺陷的直接原因在于中文句法分析器的性能瓶颈和复杂的中文表达。因此，本文将在下一章中试图采用问题标签推荐算法对系统所产生的问题进行分析，进而在问题标签的维度帮助用户查找语义相关的问答对，从而提高问答对的利用率，缓解中文自动问题生成算法所带来的问答对稀疏僵化的问题。

第四章　联想式问题标签生成强化算法

本章介绍的联想式问题标签生成算法是以大规模已标注资源中的问题和对应标签集合作为训练输入，进行标签语义相关性学习，得到大批量词汇语义关联概率作为训练输出；以学习到的词汇语义关联概率作为基础，针对每一个作为测试输入的待推荐标签的问题，对其问题描述中的实意词汇构建社会标签联想词网；在所获得的词网内，采用新颖的标签排序推荐算法获得每个候选标签的得分，进而得分高的标签作为输出。这样能够缓解模版匹配所带来的问答对产生过少，影响问答系统性能的问题。算法流程框架见图 4-1。

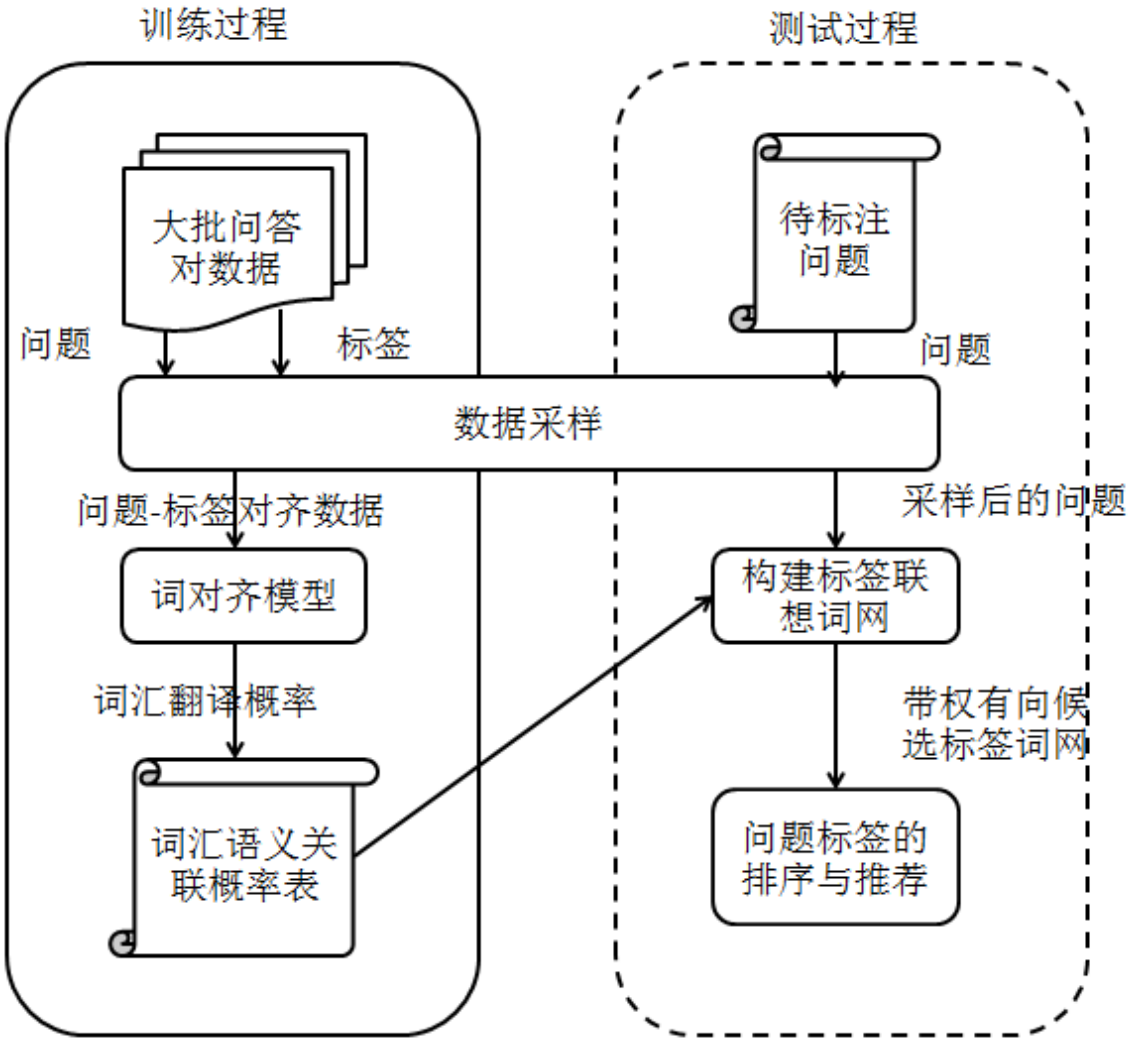


图 4-1：基于语义联想的问题标签推荐算法流程框架

4.1 词汇语义关联学习

框架提出采用统计机器翻译中的词对齐模型（IBM Model_1）学习社会标签语义的相关性。词对齐模型原本用于学习和刻画描述相同客观事物的两个不同语言平行文本间的词汇对位关系。而在同种语言环境下，这种关系演变成揭示词与词的深层语义联系[16]的钥匙。比起使用类似 WordNet，HowNet 等语义词典，基于统计的词对齐模型能够根据训练语料灵活学习，快速更新；同时继承了其独立于语言环境使用的特性。

为了习得社会标签词汇语义的相关性，框架首先准备大规模包含“标签-问题”对的平行文本用于词对齐模型的训练，表 4-1 选取了其中一部分中文样例。这些原始的数据需要做进一步的采样处理之后才能够用于词对齐模型的训练，原因如下：

- 不是所有出现在问题部分的词语都具有实意。因此需要通过分词、词性标注的过程将标注为名词词汇以及在对应标签集合中出现的词汇筛选出来。由此对于给定资源$r \in R$，标签集合可以进一步形式化表示为$a_r = \{(t_i, ct_i)\}_{i=1}^{M_r}$，筛选后的问题词汇集$w_r = \{(w_i, cw_i)\}_{i=1}^{N_r}$，即每个标签或者词汇都是由其本身以及其出现的频率的二元组进行表征，进而分别构建其所在集合。

- Och 和 Ney 认为如果平行语料中两个句子的长度差距比较大的话，会导致词对齐模型训练性能大大降低，因此我们需要在w_r和a_r的基础上进一步进行采样获得最终用于词对齐训练的数据。我们引入参数$\delta = N_r / M_r$来表征平行语料的词长比。

通过调整词对齐模型，使其适合于我们的算法整体框架，形式化表述为式 4-1：

$$\Pr(\mathrm{T}_r \mid \mathrm{W}_r) = \sum_{A} \Pr(T_r, A | W_r) \qquad \text{式 } 4-1$$

其中$\mathrm{W}_r = \{w_i\}_{i=1}^{N_r}$采样后的问题词汇集合，$T_r = \{t_i\}_{i=1}^{M_r}$采样后的标签集合。上述集合的条件概率$\Pr(\mathrm{T}_r \mid \mathrm{W}_r)$通过隐变量$A = \{a_i\}_{i=1}^{M_r}$来表征。比如$a_j = i$代表标签集合$\mathrm{T}_r$中第j位置上的标签$t_j$与位于$\mathrm{W}_r$中第i位置上的词汇$w_i$对齐。

根据词对齐模型进一步扩展式 4-1，得到式 4-2 和式 4-3：

$$\Pr(T_r, A | \mathrm{W}_r) = \frac{\varepsilon}{(N_r + 1)^{M_r}} \prod_{j=1}^{M_r} p(t_j | w_{a_j}) \qquad \text{式 } 4-2$$

$$\Pr(T_r|\mathrm{W}_r)=\frac{\varepsilon}{(N_r+1)^{M_r}}\sum_{a_1=0}^{N_r}\cdots\sum_{a_{M_r}=0}^{N_r}\prod_{j=1}^{M_r}p(t_j|w_{a_j}) \qquad \text{式 } 4-3$$

$p(t_j|w_{a_j})$被称作给定词汇w_j翻译出标签t_{a_j}翻的概率，也就是本框架在训练学习词汇语义关系的基本元素。

$$\sum_t p(t|w)=1 \qquad \text{式 } 4-4$$

式 4-3 的目标就是在式 4-4 的约束下，让$\Pr(T_r|\mathrm{W}_r)$的值最大化，通常使用 EM 算法[9]进而获得无监督的收敛效果。于是可以得到一系列的语义相关概率$p(t|w)$。

表 4-1：原始“标签-问题”平行文本对

新浪爱问 商品经济类	**问题：** 2012 伦敦奥运会赞助商都有哪些?
	标签： 伦敦奥运会，商都，赞助
	答案： 有很多，这几个比较大牌：松下、宝马、阿迪达斯、可口可乐、酷乐仕维他命获得、麦当劳、宏碁集团、北电网络、劳埃德银行、英国电网、英国航空公司、英国石油集团公司、英国电信……
新浪爱问 电脑互联网类	**问题：** XP 中 CHM 文件出现乱码怎么办?
	标签： 乱码，文件
	答案： jjhd.reg 内容 REGEDIT4 [HKEY_LOCALMACHINE\SOFTWARE\Microsoft\HTMLHelp]…

4.2 联想词网的构建

利用 4.1 节中得到的一系列标签语义相关概率$p(t|w)$，给定某个待标注标签的资源，框架首先从其问题描述中提取实意词汇，然后以这些词汇构建种子结点集合V_s，通过查找语义相关概率表找出所有与V_s中词汇语义相关的标签词汇，构成联想集合V_a。V_s与V_a共同构成标签候选集V。结点之间的有向联想关系通过语义相关概率来表征$\mathrm{w}(\mathrm{e}_{ij})=p(v_j|v_i)$。由此社会标签联想词网可以被形式化表征为一个带权有向图，见式 4-5，

$$\begin{cases} \mathrm{G} = (\mathrm{V}, \mathrm{E}) \\ V = \mathrm{V}_s \cup \mathrm{V}_a \\ E = \{\mathrm{e}_{ij}\} \\ \mathrm{e}_{ij} = \{(v_i, v_j), v_i, v_j \in V\} \\ \mathrm{w}(\mathrm{e}_{ij}) = p(v_j|v_i) \end{cases} \quad \text{式 } 4-5$$

很多常用词汇具有非常多的语义意项，能够联想触发出很多词汇。这些联想词汇往往只有一少部分具有相对较高的联想触发概率，如果全部引入，不仅增加了算法的复杂度，而且对整体性能的提升不明显，因此算法引入θ作为联想网络结点的最大出度，即选择联想触发概率最高的前θ个词汇加入到网络中。

4.3 问题标签推荐

传统的 TextRank 算法[20]简单认为文档中在位置上紧邻的两个词互为推荐，他们之间存在语义相关关系，而这种关联程度也没有合理的量化评估方法。因此只提出如式 4-6 算法模型，

$$\mathrm{Score}(v_i) = (1-d) + d * \sum_{v_j \in In(v_i)} \frac{1}{|Out(v_j)|} \mathrm{Score}(v_j) \quad \text{式 } 4-6$$

即每一个结点的评分取决于与之相连的结点分配给其的推荐分数。这种算法存在一定的弊端。首先，简单地认为两个词互为推荐本身没有合适的理论作为假设支持；其次将词汇语义关联程度全部视作相同也不符合常理。因此为了弥补这两点弊端，同时也为了适应该算法框架，我们提出如式 4-7 的社会标签排序算法(TagRank)，

$$\mathrm{Score}(v_i) = (1-d) + d * \sum_{v_j \in In(v_i)} \frac{\mathrm{w}(\mathrm{e}_{ji})}{\sum_{v_k \in Out(v_j)} \mathrm{w}(\mathrm{e}_{jk})} \mathrm{Score}(v_j) \quad \text{式 } 4-7$$

对于每一个结点v_j，$\frac{\mathrm{w}(\mathrm{e}_{ji})}{\sum_{v_k \in Out(v_j)} \mathrm{w}(\mathrm{e}_{jk})}$代表由候选标签$v_j$联想触发出$v_i$的能力，与$v_j$本身的得分相乘便是其对$v_i$的推荐程度。将所有推荐$v_i$的结点($v_j \in In(v_i)$)得分加和，并且通过一个系数$d$来调整这种推荐的权重，就得到了一次迭代后$v_i$的得分。计算开始时要确定所有结点的初值，经过几轮迭代直到收敛到所有结点迭代前后的差值小于阀值δ。最终，根据结点的得分，选择得分最高的前M个作为推荐标签。

第五章　AQ 系统设计实现

为便于描述，本章首先给出名为 AQ-BETA 的针对百科知识的问答系统的整体设计实现架构图（图 5-1）。系统命名为 AQ 的原因在于该系统从现有的百科知识（Answer）中逆向生成对应问题（Question），利用生成的问题以及对应的答案为基础构建系统的基础数据。从图 5-1 中，可以看出，AQ-BETA 系统整体以数据库为中心，整体分为离线处理部分和在线处理部分。考虑到 Web 应用的时效性强的特点，系统将复杂的算法成分移至离线部分进行处理，保证用户在使用百科问答搜索时能够快速获得需要的答案。

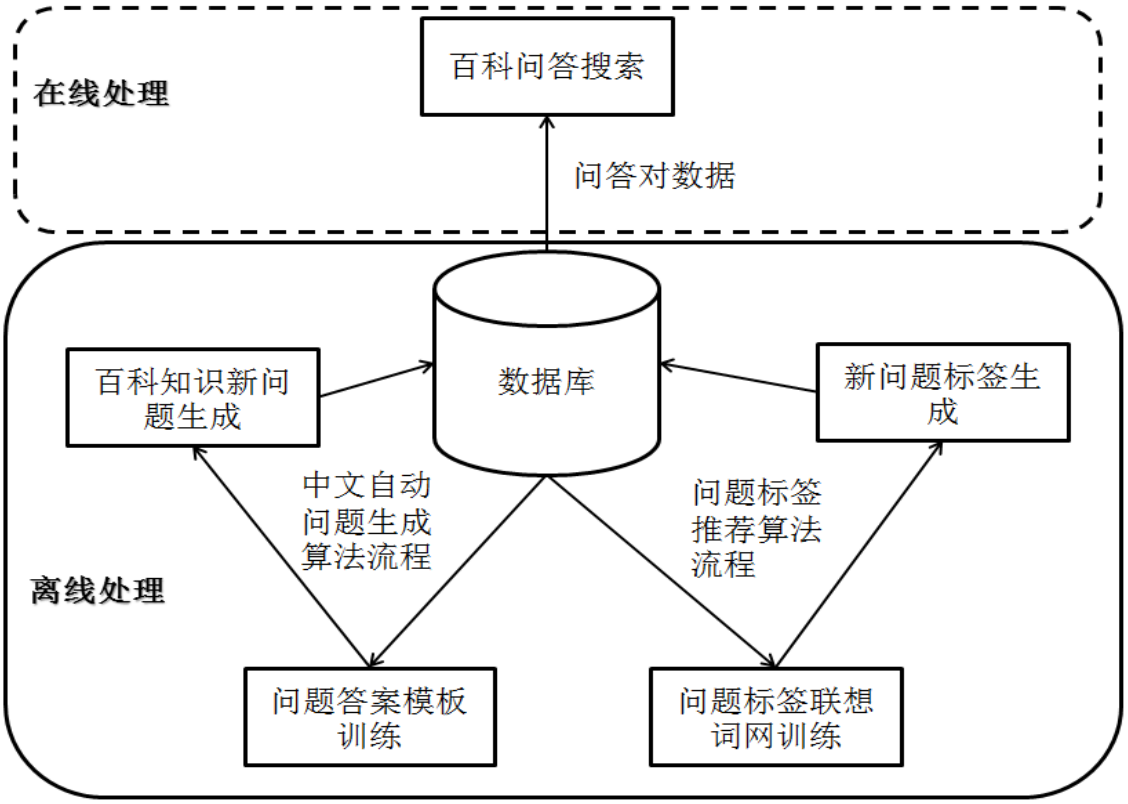


图 5-1：AQ-BETA 百科知识问答平台框架

5.1 数据库设计

表 5-1 所所显示的是数据库中表名为 qna 的数据表，主要用于记录问答对相关的数据。数据内容包括问题、对应的答案、问题标签、问题分类等等。

表 5-1：数据库 aqbeta-表 qna

字段名称	数据类型	长度	允许空值	字段描述
Question	Longtext	默认	不允许	问题

Answer	Longtext	默认	允许	答案
Label	Varchar	255	允许	问题标签
Category	Varchar	255	允许	问题类型

表 5-2 所显示的为数据库 aqbeta 中，记录问题和问题模板对应关系表 question。以 QuestionID 为主键，将问题类型的序号和句法依赖标注的序号一一对应起来。

表 5-2：数据库 aqbeta-表 question

字段名称	数据类型	长度	允许空值	字段描述
QuestionID（主键）	Bigint	20	不允许	问题序号
QuestionModelID	Bigint	20	允许	问题模型序号
QuestionShape	Text	默认	允许	问题句法依赖标注

表 5-3 所显示的为数据库 aqbeta 中记录问题模板的表 questionmodel。以 QuestionModelID 为主键，记录问题类型和问题的四种模板的对应关系数据。QuestionModelShape1 至 QuestionModelShape4 依次对应问题模板中的分词+词性标注模板、分词+词性标注+句法依赖模板、前序遍历句法依赖树模板以及引入领域词典的前序遍历句法依赖树模板。

表 5-3：数据库 aqbeta-表 questionmodel

字段名称	数据类型	长度	允许空值	字段描述
QuestionModelID（主键）	Bigint	20	不允许	问题模板序号
QestionType	Varchar	255	允许	问题类型序号
QuestionModelShape1	Text	默认	允许	问题句法依赖标注模版 1
QuestionModelShape2	Text	默认	允许	问题句法依赖标注模板 2
QuestionModelShape3	Text	默认	允许	问题句法依赖标注模板 3
QuestionModelShape4	Text	默认	允许	问题句法依赖标注模板 4

表 5-4 所显示的为数据库 aqbeta 中，记录答案和答案模板对应关系表 sentence。以 SentenceID 为主键，将答案的序号和句法依赖标注的序号一一对应起来。

表 5-4：数据库 aqbeta-表 sentence

字段名称	数据类型	长度	允许空值	字段描述
SentenceID（主键）	Bigint	20	不允许	答案序号
SentenceModelID	Bigint	20	允许	答案模型序号
SentenceShape	Text	默认	允许	答案句法依赖标注

表 5-5 所显示的为数据库 aqbeta 中记录问题模板的表 sentencemodel。以 SentenceModelID 为主键，记录答案的四种模板的对应关系数据。SentenceModelShape1 至 SentenceModelShape4 依次对应答案模板中的分词+词性标注模板、分词+词性标注+句法依赖模板、前序遍历句法依赖树模板以及引入领域词典的前序遍历句法依赖树模板。

表 5-5：数据库 aqbeta-表 sentencemodel

字段名称	数据类型	长度	允许空值	字段描述
SentenceModelID（主键）	Bigint	20	不允许	答案模板序号
SentenceModelShape1	Text	默认	允许	答案句法依赖标注模版 1
SentenceModelShape2	Text	默认	允许	答案句法依赖标注模板 2
SentenceModelShape3	Text	默认	允许	答案句法依赖标注模板 3
SentenceModelShape4	Text	默认	允许	答案句法依赖标注模板 4

表 5-6 所显示的为数据库 aqbeta 中，记录问答模板对应关系表 qsregister。以 QsID 为主键，将问题模板的序号和答案模板的序号一一对应起来。

表 5-6：数据库 aqbeta-表 qsregister

字段名称	数据类型	长度	允许空值	字段描述
QsID（主键）	Bigint	20	不允许	问答模板关联
QuestionModelID	Bigint	20	允许	问题模型序号
SentenceModelID	Bigint	20	允许	答案模型序号

5.2 离线部分

离线部分需要做如下准备：问题答案模板训练、针对百科知识的新问题的生成、问题标签联想词网训练、新问题标签生成。图 5-2 展示中文问题生成子系统的模块设计。图 5-3 显示问题标签推荐子系统的模块设计。

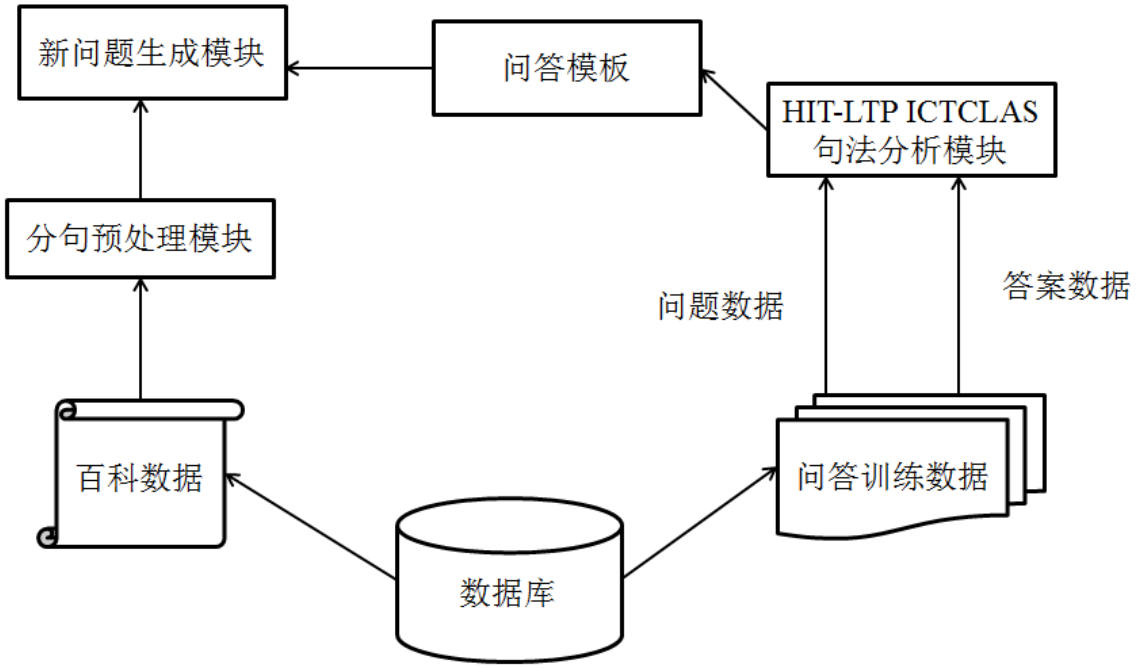


图 5-2：中文问题自动生成子系统模块设计

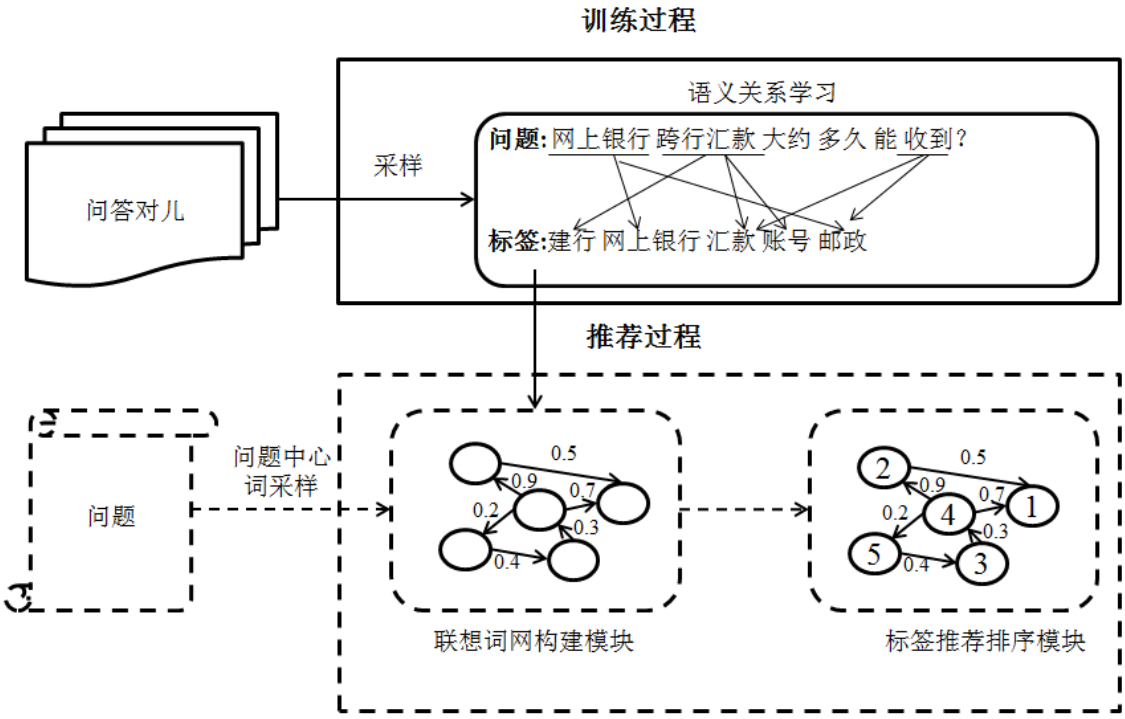


图 5-3：问题标签推荐子系统模块设计

5.2.1 问题答案模板训练

通过现有人工编写的问答对（30 万新浪爱问数据），提取问答模板。如图 5-4 的 XML 文件问一组模板问答对。

```
<?xml version="1.0" encoding="gb2312" ?>
<items>
<item>
<answer>李开复是中国著名企业家，出生于台湾。</answer>
<question>李开复是哪国企业家？</question>
<question>李开复出生在哪？</question>
</item>
</items>
```

图 5-4：问答对 XML 文件

对应生成的四种问题模板（以问题“李开复是哪国企业家”为例）分别为：

模板 1：{0:nh}{1:是}{-1:哪}{-1:国}{4:企业家}{-1:？}

模板 2：{0:nh:SBV}{1:是:HED}{-1:哪:ATT}{-1:国:ATT}{4:企业家:VOB}{-1:？:WP}

模板 3：@{ROOT}#{DOWN:HED}@{1:是}#{DOWN:SBV}@{0:nh}#{UP:SBV}@{1:是}#{DOWN:VOB}@{4:企业家}#{DOWN:ATT}@{-1:国}#{DOWN:ATT}@{-1:哪}#{UP:ATT}@{-1:国}#{UP:ATT}@{4:企业家}#{UP:VOB}@{1:是}#{UP:HED}@{ROOT}

模板 4：@{ROOT}#{DOWN:HED}@{1:是}#{DOWN:SBV}@{0:nh}#{UP:SBV}@{1:是}#{DOWN:VOB}@{4:企业家}#{DOWN:ATT}@{-1:国}#{DOWN:ATT}@{-1:哪}#{UP:ATT}@{-1:国}#{UP:ATT}@{4:企业家}#{UP:VOB}@{1:是}#{UP:HED}@{ROOT}

5.2.2 针对百科知识的新问题的生成

继续以上例的模板进行研究，对于图 5-5 中的一个百科知识句，我们可以生成的问题为：**“李湘是哪国主持人？”**

```
<?xml version="1.0" encoding="gb2312" ?>
<items>
<item>
<answer>李湘是中国著名主持人，生活在湖南。</answer>
</item>
</items>
```

图 5-5：知识句 XML 样例

5.2.3 问题标签联想词网训练

该方法使用 GIZA++在 Ubuntu 10.4 平台下对 30 万对新浪爱问问答对进行词对齐训练，以下面的一系列**问题-标签**对为例，

<question>差旅费中的每日补助缴个税吗？</question>
<tags><tag>差旅费</tag><tag>补助</tag><tag>差旅</tag><tag>个税</tag><tag>报销单</tag></tags>

<question>担保费属于财务费还是管理费用?</question>
<tags><tag> 担 保 费 </tag><tag> 财 务 费 </tag><tag> 管 理 费 用 </tag><tag> 财 务 费 用 </tag><tag>担保公司</tag></tags>

首先需要对问题进行分词处理，需要输入的两组对齐文件的命名和内容分别为（问题和标签在两个文件中的行数是一一对应的）：

Questions 文件
差旅费每日补助缴个税
担保费属于财务费管理费用
…
Tags 文件
差旅费补助差旅个税报销单
担保费财务费管理费用财务费用担保公司
…

在 GIZA++中输入如下命令：

```
./plain2snt.out questions tags
./snt2cooc.out questions.vcb tags.vcb questions _tags.snt >questions_tags.cooc
./snt2cooc.out tags.vcb questions.vcb tags_questions.snt > tags_questions.cooc
./mkcls –pquestions –Vquestions.vcb.classes opt
```

./mkcls –ptags –Vtags.vcb.classes opt

./GIZA++ -S questions.vcb –T tags.vcb –C questions_tags.snt –CoocurrenceFile questions_tags.cooc –O q2t

./GIZA++ -S tags.vcb –T questions.vcb –C tags_questions.snt –CoocurrenceFile tags_questions.cooc –O t2q

最终输出的 t2q.actual.ti.final 文件包含了所有训练语料中问题词汇联想到标签词汇的语义相关概率，每一行以空格间隔分别代表：问题词汇、标签词汇、由问题词汇联想到标签词汇的概率,如表 5-7：英国伦敦 0.0333749；这组数据代表计算机获取到“英国”联想出“伦敦”的概率为 0.0333749。

表 5-7：联想词表样例

种子结点（问题中词汇）	联想触发标签	语义相关概率
英国	英国	0.629699
英国	英国文学	0.164421
英国	外国文学	0.0468611
英国	伦敦	0.0333749
英国	庄园	0.0108718
证券	投资	0.249354
证券	金融	0.218331
…	…	…

5.2.4 新问题标签生成

由上节得到的联想词网，通过式 4-6，计算得出排序后的语义标签。图 5-6 例举了一个新问题的所有潜在语义标签。

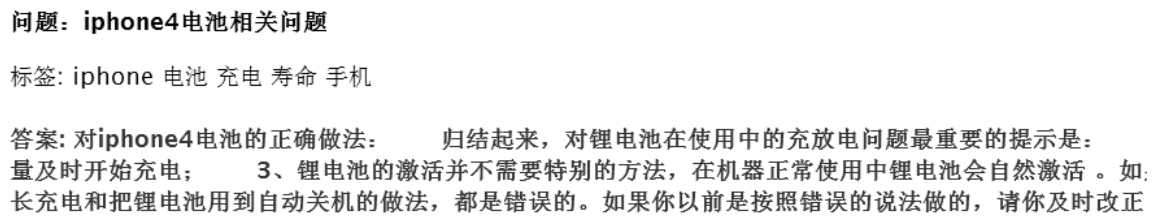


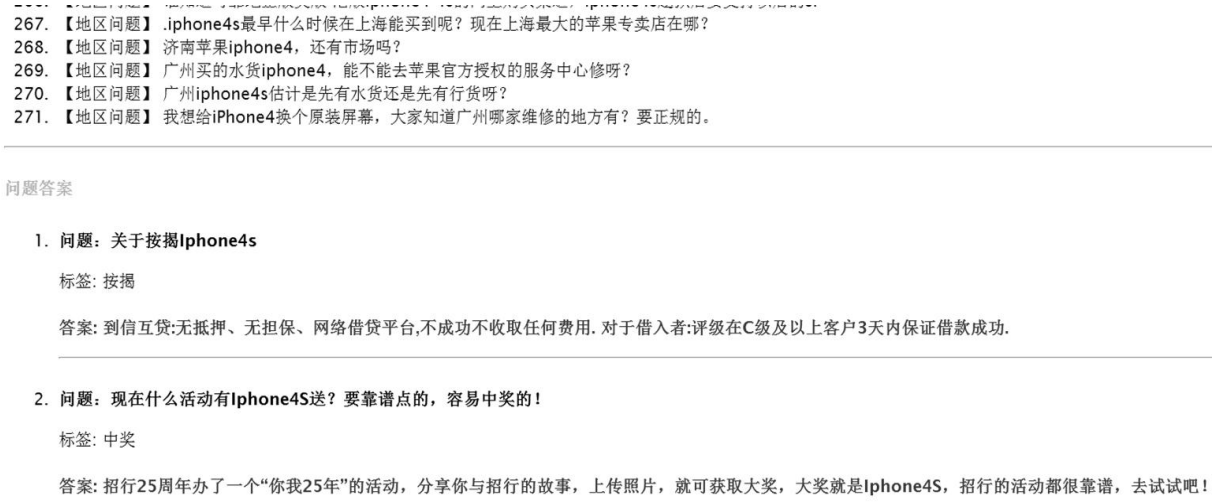
图 5-6：问题生成语义标签样例

5.3 在线部分

考虑到用户访问速度的问题，尽可能地将核心算法转移至离线部分完成，在线部分设计一个良好的问答搜索的接口，颠覆传统百科词条的搜索，如图 5-7 所示。

图 5-7AQ-BETA WEB 问答搜索接口

图 5-7 展示用户在 AQ-BETA 系统中搜索 iphone4 相关的问答知识的样例。图 5-8 为搜索到的问答包括问题、问题标签、以及对应的答案。


267. 【地区问题】 .iphone4s最早什么时候在上海能买到呢？现在上海最大的苹果专卖店在哪？
268. 【地区问题】 济南苹果iphone4，还有市场吗？
269. 【地区问题】 广州买的水货iphone4，能不能去苹果官方授权的服务中心修呀？
270. 【地区问题】 广州iphone4s估计是先有水货还是先有行货呀？
271. 【地区问题】 我想给iPhone4换个原装屏幕，大家知道广州哪家维修的地方有？要正规的。

问题答案

1. **问题：关于按揭Iphone4s**

   标签: 按揭

   答案: 到信互贷:无抵押、无担保、网络借贷平台,不成功不收取任何费用. 对于借入者:评级在C级及以上客户3天内保证借款成功.

2. **问题：现在什么活动有Iphone4S送？要靠谱点的，容易中奖的！**

   标签: 中奖

   答案: 招行25周年办了一个“你我25年”的活动，分享你与招行的故事，上传照片，就可获取大奖，大奖就是Iphone4S，招行的活动都很靠谱，去试试吧！

图 5-8AQ-BETA WEB 问答搜索结果样例

第六章　实验论证

6.1 数据集

我们从新浪爱问中收集到 30 万条问答数据作为问答模板的训练集合以及问题语义标签的学习数据。XML 格式数据样例如下：

<?xml version="1.0" encoding="gb2312" ?>

<item>

<question>iphone4s 电信版套餐是怎样的？好买么？**</question>**

<labels>

<label>好买**</label>**

<label>套餐**</label>**

<label>电信**</label>**

</labels>

<description>有木有人知道**</description>**

<answer>现在预订电信 iphone4S 可以直接去拉手网独家预订，4900 元预存话费送一台 4S，电信直接烧号的，三年内返还 4900 的自由话费，每月无最低消费，每月返还实际使用话费的 40%（从自由话费中扣除），也就是说一月你用 100 元话费，电信返还你 40，知道 4900 的话费扣完为止（4900 元在三年内有效），电信版本 4S 不支持其他运营商的卡（移动联通的就别想了），也就是说三年结束后还是要用电信的号（如有高人破解的话另当别论），不支持合约版！

</answer>

</item>

6.2 实验测试

6.2.1 测试方法

我们在 15 类 30 万问答对中各随机选取 2000 条，共 3 万条问答对作为测试集，其余 27 万作为训练数据。共有 5 名志愿者（致谢部分提及）人工对 3 万条测试集中的答案进行人工生成问题以及标签标注。

两项测试均可使用同一客观指标，即准确率，召回率，以及 F 值。在问题生成实验环境下，准确率为人工生成的问题集合为Q_O，系统生成的问题集合为Q_R，问题生成的各项评价指标准确率p，召回率r以及 F 值分别表示为式 6-1 和式 6-2：

$$p = \frac{|Q_R \cap Q_O|}{|Q_R|}, r = \frac{|Q_R \cap Q_O|}{|Q_O|}, F = \frac{2pr}{(p+r)} \qquad \text{式 } 6-1$$

同理，问题标签的准确率p，召回率r以及 F 值

$$\mathrm{p}=\frac{|\mathrm{T}_R\ \cap\ \mathrm{T}_O|}{|\mathrm{T}_R|},\mathrm{r}=\frac{|\mathrm{T}_R\ \cap\ \mathrm{T}_O|}{|\mathrm{T}_O|},F=\frac{2\,p\,r}{(p+r)} \qquad \text{式 } 6-2$$

同时，问题生成领域另有一套人工评价指标，相关工作中有所提及。在评测的时候，引入 5 类指标通过多位人工判定的方式对这些系统所生成的问题进行打分，这 5 类指标包括：相关度，问题类型，提问正确性以及流利程度，明确度，多样性。打分由 1-4 代表评价逐级下降。

6.2.2 问题生成性能测试

由于本文的重点，中文问题自动生成是对一个新兴领域的探讨，因此很难找到中文相关的方法与本文提及的方法进行性能上的比较，因此，该文的测试结果可作为为这一领域的基础客观参考性能。表 6-1 显示了 AQ-BETA 系统在中文问题生成方面的客观性能。

表 6-1：问题生成客观性能测试结果

准确率	召回率	F 值
0.924	0.135	0.236

尽管中文自动问题生成属于新兴领域，学术界对英文问题生成研究方法有一套评价指标（见章节二）。本文将这套评价指标用于中文自动问题生成方法的评价中，邀请 5 名志愿者对自动生成的问题依据如表 6-2 进行打分，分数 1-4 分不等。4 分为最高，1 分为最低（与英文评价方式略有不同，依照中国人的评价习惯，4 分的评价度更高）最终结果由 5 人的平均值决定。

表 6-2：问题生成人工评价结果

	相关度	类型	正确性	流利度	明确度	多样性
1 号志愿者	4	4	4	2	2	1
2 号志愿者	3	4	4	2	2	1
3 号志愿者	4	3	4	3	1	1
4 号志愿者	3	4	3	1	3	2
5 号志愿者	3	4	3	2	3	1
平均值	3.4	3.8	3.6	2	2.2	1.2

6.2.3 标签生成性能测试

文本的问题标签生成算法属于基于内容的标签推荐方法。基于内容的标签推荐算法主要分为四类：基于分类器的方法[36]，基于信息检索的方法[37]，基于隐话题模型（LDA）的方法[32,33]以及基于统计机器翻译（SMT）[34,38]的方法。

基于分类器的方法将每个标签视作分类标记，多种分类器如 KNN,SVM,NB,神经网

络等均被用于标签推荐上。但是由于多数问题资源是由人工进行标注的，所以其准确性十分受局限，而分类器的性能很大程度上取决于分类器训练数据的质量，因此分类器不适用于基于内容的标签推荐方法。

基于信息检索的方法利用用户或者资源问题的背景内容，采用向量空间模型或者 BM25 模型进行相似度的比较，用于比较相似性的元素的权值常常使用 TFIDF 值进行计算。如此一来，相似资源的标签便可以被推荐到新资源问题的标签集合中。

LDA 模型经常用于提取隐含的语义词汇，在问题标签推荐的环境下，适用于隐含语义标签的抽取，对于冷门资源问题有好的效果。但是这种方法的局限性在于，对于新提问的问题，LDA 模型不适用。

使用统计机器翻译的词对齐模型能够找出词汇之间的语义关系，进而不论是冷门资源问题还是新问题，只要问题中有主题词汇，其相关的语义标签便可以被触发，进而被推荐。

为了验证我们的语义标签推荐方法 STR (Semantic Tag Recommendation)在性能上的优势，在相同数据集下，我们将 STR 与现今比较流行的基于内容的标签推荐 WTM，TextRank， TFIDF 方法进行比较。表 6-3,6-4,6-5 分别代表推荐标签数量对应系统精确度、召回率以及 F 值的结果：

表 6-3：STR,WTM,TextRank,TFIDF：精确度-推荐标签数量比较

	STR	WTM	TextRank	TFIDF
1	0.713	0.443	0.218	0.190
2	0.721	0.468	0.213	0.175
3	0.528	0.351	0.154	0.120
4	0.471	0.321	0.136	0.103
5	0.424	0.296	0.121	0.091
6	0.385	0.274	0.110	0.082
7	0.354	0.256	0.100	0.075
8	0.328	0.240	0.092	0.07
9	0.305	0.227	0.086	0.065
10	0.286	0.215	0.081	0.061

表 6-4：STR,WTM,TextRank,TFIDF：召回率-推荐标签数量比较

	STR	WTM	TextRank	TFIDF
1	0.089	0.055	0.027	0.023
2	0.181	0.117	0.053	0.043
3	0.198	0.131	0.058	0.045
4	0.235	0.161	0.068	0.051
5	0.265	0.185	0.076	0.057

6	0.289	0.206	0.082	0.062
7	0.310	0.224	0.088	0.066
8	0.328	0.241	0.092	0.070
9	0.344	0.255	0.097	0.074
10	0.358	0.269	0.101	0.077

表 6-5：STR,WTM,TextRank,TFIDF：F 值-推荐标签数量比较

	STR	WTM	TextRank	TFIDF
1	0.158	0.098	0.048	0.042
2	0.288	0.187	0.085	0.070
3	0.288	0.191	0.084	0.066
4	0.314	0.214	0.090	0.070
5	0.326	0.228	0.093	0.071
6	0.331	0.236	0.094	0.071
7	0.331	0.239	0.094	0.071
8	0.328	0.241	0.092	0.069
9	0.323	0.241	0.091	0.068
10	0.318	0.239	0.089	0.068

图 6-1 中，从左上至右下，各个点代表推荐标签数量以 1-10 递增。显然，本文论述的 STR 方法在召回率和准确率方面总体优于其他主流的基于内容的标签推荐方法。

图 6-2 利用 F1 值的评价指标对 4 中方法进行综合测评，结果得出，STR 方法在推荐不同数量的标签环境下依然保持优势性能。

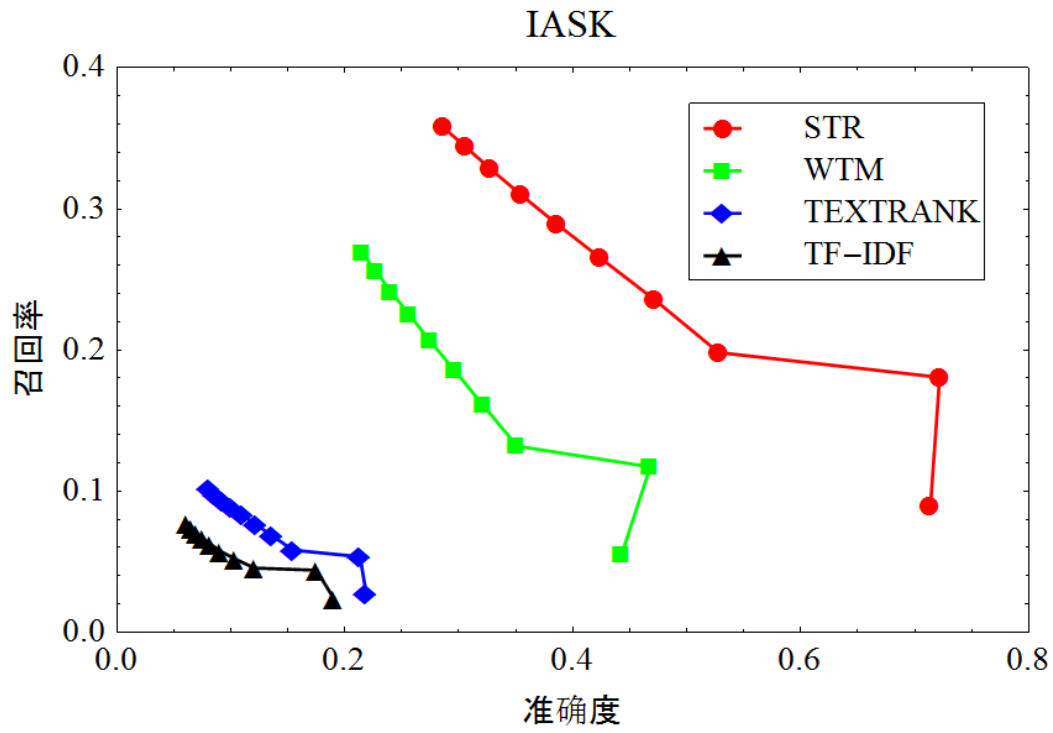


图 6-1：STR,WTM,TextRank,TFIDF：精确度-召回率比较曲线

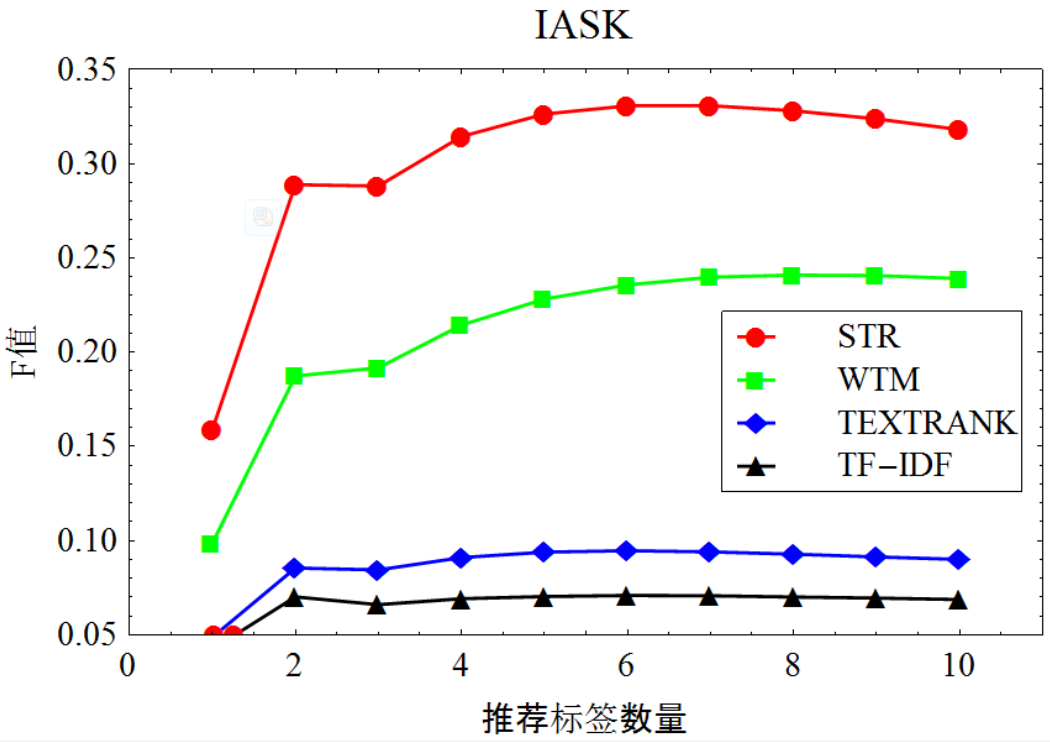


图 6-2：STR,WTM,TextRank,TFIDF：F 值-推荐标签数量比较曲线

实验证明，STR 方法在总体性能上优于现今主流的基于内容的标签推荐方法。

第七章　研究贡献

本文通过对现有一般问答系统的优缺点进行分析，探究一种让计算机由百科知识（答案）自动生成问题的方法，同时在该方法的基础上，为了强化问答搜索效果，进一步提出一种针对问题生成语义标签的方法。

实验证明，该文提出的中文问题生成方法的精确度为 0.924，召回率为 0.135，F 值为 0.236；人工评测生成问题的各项指标平均结果分别为：问答相关度 3.2，问题类型正确度 3.8：问题提问正确性 3.6 问题流利度 2 问题明确度 2.2 问题多样性 1.2。作为对中文问题自动生成这一新兴领域的探索，该文的方法结果可以供后续研究参考。

为强化问答搜索结果所提出的语义问题标签推荐方法（STR）与现今主流的基于内容的多种标签推荐方法（WTM, TEXTRANK, TF-IDF）在相同数据集下的测试结果表明，该文提出的问题标签方法在性能上具有明显的优势。

结合上述提出的两类方法构建了新一代基于百科知识的问答平台 AQ-BETA，其功能特征在于从百科知识中提出问题，生成问答对。在生成问题的基础上，为强化语义搜索效果，生成语义标签，提高用户体验。

第八章　总结和展望

该文提出的中文问题生成方法以及问题搜索强化算法不仅对新兴研究领域有着科学的探索，同时也提高了现有研究领域的方法性能。

尽管如此，仍然有许多相关问题值得探索和研究，这里仅例举以下几项用于后续的讨论和交流：

- 该文提出的中文问题生成方法比较受答案语法质量的局限。这个局限是造成召回率下降的主要原因。因此后续如何对更加自由化的自然语言提出问题是一项有意义的研究方向。
- 该文提出的语义标签推荐方法是从浅层语义研究向深层语义研究的一大迈进，该方法可以对短文本语义相似、句子中心意义相似等研究提供有益参考。

参考文献

[1] Candy Schwartz. Web search engines.Journal of the American Society for Information Science.Volume 49, Isuee 11. 1998.pages 973-982.

[2] Larry Page, Sergey Brin, R. Motwanietc. The PageRank Citation Ranking: Bringing Order to the Web. Technical Report. Stanford InfoLab. 1999.

[3] Alexander Kotov, Chengxiang Zhai. Toward Natural Question-Guided Search.World Wide Web Conference. 2010.pages 541-550.

[4] 宗成庆. 统计自然语言处理. 第 3 版. 清华大学出版社. 2011.

[5] Vasile Rus and C.Graesser.The Question Generation Shared Task and Evaluation Challenge Workshop Report, The University of Memphis, 2009.

[6] Boyer, Kristy Elizabeth and Piwek, Paul eds. Proceedings of QG 2010: The Third Workshop on Question Generation. Pittsburgh: questiongeneration.org. 2010.

[7] Saidalavi Kalady, Ajeesh Elikkottil etc.Natural Language Question Generation UsingSyntax and Keywords. Proceedings of QG 2010: The Third Workshop on Question Generation. 2010.

[8] Paul Piwek and Svetlanan Stoyanchev.Question Generation in the CODA Project, Proceedings of QG 2010: The Third Workshop on Question Generation. 2010.

[9] Jeremiah Sullins, Art Gaesser etc.The Effects of Cognitive Disequilibrium on Question Generation While Interacting with AutoTutor. Proceedings of QG 2010: The Third Workshop on Question Generation. 2010.

[10] Rodney D. Nielsen. Question Generation: Proposed Challenge Task and Their Evaluation, In V. Rus and A. Graesser, editors, Proceedings of the Workshop on the Question Generation Shared Task and Evalution Challenge, Arlington, Virginia, September 2008.

[11] R. Vasile, W. Brendan,P. Paul etc.The First Question Generation Shared Task Evaluation Challenge. Inproceedings of the Sixth International Natural Language Generation Conference (INLG 2010), 7-9 July 2010, Trim Castle, Ireland

[12] Vasile Rus, Brendan Wyse etc. Question Generation Shared Task and Evaluation Challenge – Status Report, In Proceedings of the 13th European Workshop on Natural Language Generation (ENLG), September 2011, Nancy, France.

[13] Prashanth Mannem, Rashmi Prasad and Aravind Joshi.Question Generation from Paragraphs at Upenn: QGSTEC System Description. Proceedings of QG 2010: The Third Workshop on Question Generation. 2010.

[14] Michael Heilman, Noah A.Smith. Question Generation via Overgeneratiing Transformations and Ranking.CMU-LTI. 2009.

[15]Michael Heilman, Noah A.Smith. Good Question! Statistical Ranking for Question Generation.NAACL 2010 Los Angeles, California, June 2010.

[16] Michael Heilman.Automatic Factual Question Generation from Text.Ph.D.

dissertation.CMU-LTI.2011.

[17]V.Rus, Z. Cai, and A.C. Graesser. 2007. Experiments on Generating Questions About Facts.In Proceedings of CICLing 2007.volume 4394 of LNCS, pages 444-455, Berlin. Springer Verlag.

[18] P. Piwek, H. hernault, H. Prendinger, and M. Ishizuka. 2007. T2D: Generating Dialogues between Virtual Agents Automatically from Text. In Proceedings of IVA07, LNAI 4722, pages 1616-174, Berlin. Springer Verlag.

[19] P. Piwek, Helmut Prendinger et al. Generating Questions: An Inclusive Characterization and a Dialogue-based Application. Online Proceedings of Workshop on the Question Generation Share Task and Evaluation Challenge.NSF, Arlington, VA. September 25-26 2008.

[20] Ricardo A. Baeza-Yates, Andrei Z.Broder, Yoelle S. Mararek. 2011. The New Frontier of Web Search Technology, Search Computing, LNCS 6585, pages 3-9, Berlin, Springer Verlag.

[21] Shiqi Zhao, Haifeng Wang, Ting Liu et al. Automatically Generating Questions from Queries for Community-based Question Answering, IJCNLP ’11

[22]Husam Ali, Yllias Chali, and Sadid A. Hasan. Automation of Question Generation From Sentences. Proceedings of QG 2010: The Third Workshop on Question Generation.

[23] X. Si. Content-based recommendation and analysis of social tags.Ph.D. thesis.Tsinghua University.2010.

[24] 杰夫 豪 (著), 牛文静(译).众包：群体力量驱动商业未来,中信出版社.2011.

[25] C. Musto, F. Narducci, M. de Gemmis, P. Lops, and G. Semeraro. STaR: a social tag recommender system. In Proceeding of ECML/PKDD 2009 Discovery Challenge Workshop. 2009.pages 215-227.

[26] F. Ricci, L. Rokach, B. Shapira and P. B. Kantor.Recommender Systems Handbook.Springer Press. 2011.

[27]D. Eck, P. Lamere, T. Bertin-Mahieux, and S. Green.Automatic generation of social tags for music recommendation.In Proceedings of NIPS.2007. pages 385-392.

[28] S. Fujimura, KO Fujimura, and H. Okuda. Blogosonomy: Autotagging any text using bloggers’ knowledge. In Proceedings of WI. 2007.pages 205-212.

[29]I. Cantador, A. Bellog ń, D. Vallet. Content-based recommendation in social tagging systems.In Proceedings of ACM RecSys. 2010.pages 237-240.

[30] C.D. Manning, P. Raghavan, and H. Schtze.Introducation to information retrieval.Cambridge University Press. 2008. New York, USA.

[31] H. Cao, M. Xie, L. Xue, C. Liu, F. Teng, and Y. Huang.Social tag predication base on supervised ranking model. In Proceeding of ECML/PKDD 2009 Discovery Challenge Workshop, 2009.pages 35-48.

[32] X. Si and M. Sun. Tag-LDA for scalable real-time tag recommendation. Journal of Computational Information Systems. 2009. pages 23-31.

[33] X. Si, Z. Liu and M. Sun. Modeling social annotations via latent reason identification. IEEE Intelligent Systems.Issue 99. 2010. Pages: 42-49.

[34] Z. Liu, X. Chen and M. Sun.A word trigger method for social tag suggestion. In Proceedings of EMNLP. 2011. Pages: 1577-1588.

[35] Wanxiang Che, Zhenghua Li, Ting Liu. LTP: A Chinese Language Technology Platform. In Proceedings of the Coling 2010:Demonstrations. 2010.08 Beijing, China. Pages 13-16,.

[36] J. Illig, A. Hotho, R. Jaschke and Gerd Stumme.A comparison of content-based tag recommendations in folksonomy systems. 2009. In KPP’07: Postproceedings of the International Conference on Kenowledge Processing in Practice.

[37] I. Cantador,A. Bellog ń, D. Vallet. 2010. Content-based recommendation in social tagging systems.In Proceedings of ACM RecSys, 237-240.

[38] M. Fan, Y. Xiao, Q. Zhou. 2012. Bringing the Associative Ability to Social Tag Recommendation. In Proceedings of ACL 2012 Workshop: TEXTGRAPH-7

致　　谢

感谢我的导师吴国仕教授、李晶老师对本人的精心指导和悉心培养，同时致谢清华大学周强研究员对本论文提出的良多建议。感谢我的父亲范垂仁和母亲刘丽，在你们哺育和关心下，我能够专心奋斗。感谢实验室同学们的热情帮助，特别感谢国际学院 08 级的罗艺，软件学院 10 级肖英男、姚旭、朱晨超等人在这篇论文的写就过程中给予的支持。感谢她为我的大学生活画上圆满的句号。感谢所有曾经帮助、指导过我的同学和朋友。特此致谢。

附　　录

系统核心代码张贴：

ICTCLAS 5.0JNI 接口

```
package ICTCLAS.I3S.AC;
importjava.io.*;
publicclass ICTCLAS50
{
    //public enum eCodeType
    //{
    //      CODE_TYPE_UNKNOWN,//type unknown
    //      CODE_TYPE_ASCII,//ASCII
    //      CODE_TYPE_GB,//GB2312,GBK,GB10380
    //      CODE_TYPE_UTF8,//UTF-8
    //      CODE_TYPE_BIG5//BIG5
    //}

    publicnativeboolean ICTCLAS_Init(byte[] sPath);
    publicnativeboolean ICTCLAS_Exit();
    publicnativeint ICTCLAS_ImportUserDictFile(byte[] sPath,int eCodeType);
    publicnativeint ICTCLAS_SaveTheUsrDic();
    publicnativeint ICTCLAS_SetPOSmap(int nPOSmap);
    publicnativeboolean ICTCLAS_FileProcess(byte[] sSrcFilename, int eCodeType, int
bPOSTagged,byte[] sDestFilename);
    publicnativebyte[] ICTCLAS_ParagraphProcess(byte[] sSrc, int eCodeType, int
bPOSTagged);
    publicnativebyte[] nativeProcAPara(byte[] sSrc, int eCodeType, int bPOStagged);
    /* Use static intializer */
    static
    {
        System.loadLibrary("ICTCLAS50");
    }
    }
```

HIT-LTP 接口：

```
package sentence_parser;
```

```
import java.io.IOException;
import java.util.ArrayList;

import org.jdom.JDOMException;

import word_seg.ICTCLAS_Seg;

import edu.hit.ir.ltpService.LTML;
import edu.hit.ir.ltpService.LTPOption;
import edu.hit.ir.ltpService.LTPService;
import edu.hit.ir.ltpService.Word;

public class LTP_Parser {
    LTPService ls;
    public LTP_Parser()
    {
        ls = new LTPService("fanmiao1120@gmail.com:20110713");
    }

    public ArrayList<Word> sentenceParser(String sentence)
    {
        ICTCLAS_Seg ictclas_seg = new ICTCLAS_Seg();
        return this.sentence_parser(ictclas_seg.ParagraphProcess(sentence));

    }

//  public ArrayList<Word>sentenceParserWithoutICT(String sentence){
//      try {
//          LTML ltml = ls.analyze(LTPOption.ALL, sentence);
//          ArrayList<Word> wordList = ltml.getWords(0);
//          return wordList;
//      } catch (JDOMException e) {
//          // TODO Auto-generated catch block
//          e.printStackTrace();
//      } catch (IOException e) {
//          // TODO Auto-generated catch block
```

```
//              e.printStackTrace();
//          }
//          return null;
//
//      }
        public ArrayList<Word> sentence_parser(ArrayList<String> wordArrayList)
        {

        try {
                LTML ltml = new LTML();
                ArrayList<Word> mergeList = new ArrayList<Word>();
                for(int i = 0; i < wordArrayList.size(); i++)
                {
                    Word mergeWord = new Word();
                    mergeWord.setWS(wordArrayList.get(i));
                    mergeList.add(mergeWord);

                }
                ltml.addSentence(mergeList, 0);

                ltml.setOver();
            System.out.println("\nmerge and get parser results.");
            LTML ltmlOut = ls.analyze(LTPOption.ALL, ltml);

                ArrayList<Word> wordList = ltmlOut.getWords(0);
                for (int j = 0; j < wordList.size(); ++j) {
                    System.out.print("\t" + wordList.get(j).getID());
                    System.out.print("\t" + wordList.get(j).getWS());
                    System.out.print("\t" + wordList.get(j).getPOS());
                    System.out.print("\t" + wordList.get(j).getParserParent()
                            + "\t" + wordList.get(j).getParserRelation());
                    System.out.print("\t" + wordList.get(j).getNE());
                    System.out.print("\t" + wordList.get(j).getWSDExplanation());
                    System.out.println();
                }
                return wordList;
        } catch (Exception e) {
```

```
            e.printStackTrace();
        }
        return null;
    }

    }
```

问答模板训练：

```
package todb;

import java.util.ArrayList;

import mysqlJDBC.PatternJDBC;

import jdbcModel.SentenceNPattern;

import tmpXML.XMLParser;
import a2qsmatch.MatchParse;
import answerModel.Answer2Questions;
/**
 * 用于处理训练逻辑*/
public class Train {
    /**
     * url 变量为需要处理的问题-答案 xml
     * */
    public static void getXMLParse(String url) {
        XMLParser parser = new XMLParser(url, 2);
        ArrayList<Answer2Questions> answer2QuestionsList = parser.parseXML2();
        for (Answer2Questions answer2Questions : answer2QuestionsList) {
            System.out.println("AAA" + answer2Questions.getAnswer() + "AAA");
            for (String s : answer2Questions.getQuestions()) {
                System.out.println("QQQ" + s + "QQQ");
            }
        }
        for (Answer2Questions answer2Questions : answer2QuestionsList) {
            generatePatternsIntoDB(answer2Questions);
        }
```

```
        }
        /**将一个答案及其对应问题的四种模型生成并写入数据库*/
        public static void generatePatternsIntoDB(Answer2Questions answer2Questions){
            MatchParse matchParse = new MatchParse();
            PatternJDBC patternJDBC = new PatternJDBC();
            matchParse.setAnswer2Questions(answer2Questions);

            matchParse.setMode(1);
            SentenceNPattern sentenceNPattern1 = matchParse.getSingleAnswerParse();
            matchParse.setMode(2);
            SentenceNPattern sentenceNPattern2 = matchParse.getSingleAnswerParse();
            matchParse.setMode(3);
            SentenceNPattern sentenceNPattern3 = matchParse.getSingleAnswerParse();
            matchParse.setMode(4);
            SentenceNPattern sentenceNPattern4 = matchParse.getSingleAnswerParse();

            //这里添加数据库写入逻辑

    patternJDBC.setSentenceNPatternsToDB(sentenceNPattern1,sentenceNPattern2,sentence
NPattern3,sentenceNPattern4);
        }


        public static void main(String[] args) {
            String url = "\\resources\\likaifu.xml";
            Train.getXMLParse(url);
        }


    }
```

新问题的生成：

```
package todb;

import java.util.ArrayList;

import tmpXML.XMLParser;
```

```
import jdbcModel.SentenceNPattern;

import mysqlJDBC.PatternJDBC;

import a2qsmatch.MatchParse;
import a2qsmatch.QuestionGenerater;
import answerModel.Answer2Questions;

public class Generation {
    private QuestionGenerater questionGenerater;
    private PatternJDBC pJDBC;
    public Generation(){
        questionGenerater = new QuestionGenerater(1);
        pJDBC = new PatternJDBC();
    }
    public void getQuestionsNtoDB(String sentence){
        ArrayList<String> questions =this.questionGenerater.getQuestions(sentence);
        pJDBC.insertSentenceNQuestions(sentence, questions);
        System.out.println("写入完成");
    }
    /**
     * 读入格式 2 的 xml 文件从中生成问题写入数据库
     * */
    public    void sentencesFromXML(String url) {
        MatchParse matchParse = new MatchParse(1);
        XMLParser parser = new XMLParser(url, 2);
        ArrayList<Answer2Questions> answer2QuestionsList = parser.parseXML2();
        for(Answer2Questions answer2Questions : answer2QuestionsList){
            this.getQuestionsNtoDB(answer2Questions.getAnswer());
        }

    }
    /**
     * 单句产生问题*/
    public static void main2(String[] args){
        Generation generation = new Generation();
```

```
        String sentence = "苹果，为荔枝的一个品种，别名西瓜。";
        generation.getQuestionsNtoDB(sentence);
    }
    /**
     * 格式 2 的 xml 中的单句*/
    public static void main(String[] args){
        Generation generation = new Generation();
        String url = "\\resources\\lixiang.xml";
        generation.sentencesFromXML(url);
    }


}
```

问题标签推荐排序算法：

```
package tagRecommendation;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.StringReader;
import java.util.HashMap;
import java.util.Scanner;
import java.util.StringTokenizer;
import java.util.Vector;

import Analyze.Analyzeit;

public class tagRecommendator {
    final String charset = "utf-8";

    //Recommendation
    private int Tag_choose = 50;
    //factor
    private double y_factor = 0.5;
    //out_degree
    private int chooseNum = 30;
```

```
public HashMap<String, Vector<Node>> Weight = new HashMap<String, Vector<Node>>();

Analyzeit analyzeit;

public tagRecommendator() throws Exception {
    Init();
    analyzeit = new Analyzeit();
    analyzeit.Init();
    // TODO Auto-generated constructor stub
}


public void ReadWeight() throws FileNotFoundException{
    int importnum = 0;
    String s,s1,s2;
    StringTokenizer ss;
    int n;
    double k;

    Scanner cin = new Scanner(new FileInputStream("Weight_Rr.txt"),charset);

    while (cin.hasNext()){
        ++importnum;
        Vector<Node> ret = new Vector<Node>();
        s = cin.nextLine();
        s2 = cin.nextLine();
        n = Integer.parseInt(s2);

        for (int i=1;i<=n;++i){
            s2 = cin.nextLine();
            ss = new StringTokenizer(s2);
            s1 = ss.nextToken();
            k = Double.parseDouble(ss.nextToken());

            ret.add(new Node(s1,k));
```

```
                }
                Weight.put(s, ret);
            }

            System.out.println(importnum + " Records imported.");
        }

        public void Init() throws Exception{
//          ReadRS reader = new ReadRS(tmpPath,y_factor,chooseNum);
            ReadWeight();
        }

        public Vector<String> Work(String s) throws Exception{
            s = analyzeit.getWords(s, Analyzeit.mark_noun);
            Graph g = new Graph();
            Scanner sin = new Scanner(new StringReader(s));
            Vector<String> V = new Vector<String>();
            while (sin.hasNext()) V.add(sin.next());

            double k = 1.0/(double)V.size();
            for (int j=0;j<V.size();++j){
                String s1 = V.get(j);
                Vector<Node> tmp = Weight.get(s1);
                g.getNode(s1,k);

                if (tmp==null) continue;
                for (int i=0;i<tmp.size();++i)
                    g.add(s1, tmp.get(i).s, tmp.get(i).key);
            }

            g.RunTextRank();
            g.GetResult(Tag_choose);

            Vector<Node> ans = g.GetResult(Tag_choose);
            V.clear();
            for (int i=0;i<ans.size();++i) V.add(ans.get(i).s);
            return V;
```

```
        }
    }
```

相关资源下载：

GIZA++: http://code.google.com/p/giza-pp/

HIT-LTP:http://ir.hit.edu.cn/ltp/

ICTCLAS:http://hi.baidu.com/drkevinzhang/blog/category/ictclas%B7%D6%B4%CA