# 第九章 文本生成的应用

## 9.1 基于规则方法的天气预报自动生成

自然语言生成（Natural Language Generation，NLG）是研究使计算机具有人一样的表达和写作的功能，自然语言生成任务的输入形式各异，例如数据到文本（data-to-text generation）和文本到文本（text-to-text generation）都是自然语言生成的任务。摘要生成就是典型的文本到文本的生成任务，数据到文本的生成任务以结构化的数据为输入，由赫瑞瓦特大学启动的E2E Challenge就是数据到文本生成的例子。

自然语言生成在方法上基本可以分成基于规则的方法和基于神经网络的方法， Reiter\cite{t4}将基于规则的NLG归纳为三步的流水线架构，其中包括：Content Determination、Sentence Planner、Surface Realizer，Content Determination负责内容的规划，通常包括内容的选择和结构的规划，在内容确定后Sentence Planner进行句子结构的规划，最后在内容和句子结构规划完成后，通过Surface Realizer生成最后的文本。

而后随着深度学习方法的不断发展，神经网络的方法也被用于自然语言处理领域，得到了一定的突破。基于神经网络的方法采用了端到端的思路，学习数据的表示和映射关系来生成文本。

传统的NLG任务可以分为多个子任务，包括：内容确定（Content Determination）、文本结构（Text Structuring）、句子聚合（Sentence Aggregation）、词汇化（Lexicalisation）、指代表达式生成（Referring Expression Generation）、语言实现（Linguistic Realisation）等。

* 内容确定，即确定文本中要传达哪些信息，如果数据中包含的信息比文本要传达的信息多，就需要舍弃一些信息。例如我们在生成球赛的报道时，并不需要把所有的得分信息都包含进来，这时候就需要进行筛选。
* 文本结构即确定在文本中呈现这些信息的顺序，合理的展示所有信息，基于规则的NLG系统中主要是通过定义模板来定义文本的结构。
* 句子聚合是合并一些信息，把可以在一个句子里表达的信息合并。
* 词汇化是将前三步已经确定的句子内容转换成自然语言，找到正确的词汇或短语来表达所有信息。
* 指代表达式生成是将之前已经提到的实体用代词、专有名称或者文字描述来指代它。
* 语言实现是在最后将所有的词汇和短语都组合起来，使用基于手工构建的模板、基于语法或者基于统计的方法生产正确的句子。

在这里，我们用公开的天气预报API获取有关天气实况、天气预报和生活指数的数据，搭建一个基于规则的天气预报生成系统。我们需要定义一个模板，根据获取到的天气数据的值对模板进行适当的选择。如果当前天气较冷，我们需要提醒用户注意保暖不要着凉；如果现在紫外线很强，我们需要提醒用户注意防晒。最终的模板如下：

现在{loc}的天气是{weather}，气温{temperature}℃。[天气{dressing}，注意保暖不要着凉。|{flu}感冒，请注意保暖。|紫外线{uv}，要注意防晒哦。]未来两天{loc}的天气变化情况为：{date}，日间天气{day}，晚间天气{night}，最高气温{high}℃，最低气温{low}摄氏度，风速{wind\_speed}km/h，降水量为{r}mm，相对湿度为{humidity}%。【截止到{last\_update}】

## 9.2 基于深度学习的聊天机器人开发

聊天机器人是对话系统的一种，可以使用自然语言与人对话和聊天。根据对话的需求不同，对话系统可以分为任务型、问答型和聊天型。任务型对话系统可以通过与用户对话来给用户提供帮助或者完成用户下达的指令，例如各类移动端提供的语音助手、微软Cortana等。而问答型问答系统可以回答用户的问题，例如网站上的问答机器人等。聊天机器人则大多是以娱乐为目的，没有特定的任务目标，仿照人类对话的特点，让人和机器尽可能的多聊下去，去消耗时间。这一类聊天机器人也有很多，例如微软小冰等。

自然语言是十分复杂的，在我们的日常对话中，存在非常多的歧义或多义，要让程序与人类进行自然的对话是十分困难的。早期的对话系统是基于规则实现的，方法简单也取得了不错的效果，但难以适应较为复杂的对话任务。随着深度学习的发展，端到端的模型被应用到了聊天机器人任务中，模型能从对话语料库中自动学习对话知识，摆脱了对于规则的依赖。

**基于规则的方法**

最早的聊天机器人ELIZA就是完全基于规则的系统，它模拟了罗杰式心理医生的对话方式，把用户作为中心，聆听用户的烦恼。ELIZA采用了pattern/transform的模式，定义了一系列与关键词相关联的规则，如果用户的输入符合这条规则，就按照规则的定义把输入转换后输出。在实际使用中，虽然ELIZA只是使用简单的规则对输入进行转换，不会进行其他补充，但是用户认为它可以理解用户所讲的内容，让用户对它敞开心扉。

例如，ELIZA按照如下规则\cite{t5}，判断句子“You hate me”符合以下规则，将其转换为“What makes you think I hate you”。

Pattern：(\* YOU \*\* ME)

->

Transform：(WHAT MAKES YOU THINK I \*\* YOU)

在ELIZA的规则中，不同关键词有着不同的优先级，关键词越少见，它的权重越高，当用户的输入中包含多个关键词时，ELIZA会选择权重更高的关键词对应的规则。同时ELIZA使用了memory trick，当检测到“my”这个词时，ELIZA认为这是和用户有关的信息，把这句话相关的结果存进一个先进先出的memory queue中。如果用户输入中没有匹配的关键词，ELIZA会回答一个无关紧要的句子或者从memory queue中推出一句话作为回答。

以下是ELIZA的规则中与memory trick相关的内容：

(MEMORY MY

(\* MY \*\* = LETS DISCUSS FURTHER WHY YOUR \*\*)

(\* MY \*\* = EARLIER YOU SAID YOUR \*\*)

(\* MY \*\* = DOES THAT HAVE ANYTHING TO DO WITH THE FACT THAT YOUR \*\*)

在ELIZA出现之后，很多聊天机器人都采用了ELIZA的方法，例如1971年用来研究精神分裂症的PARRY，它在ELIZA的基础上加上了自己的情感状态模型，加入了恐惧和愤怒的变量，某些话题会让PARRY的恐惧或愤怒增大，那么PARRY会生成有敌意的句子。PARRY也是第一个通过了图灵测试的聊天系统。

**基于语料库的方法**

基于语料库的方法则是从大量的人与人对话数据中学习对话特点，基于语料库的方法都是需要大量数据的，这些数据可以来自电影电视剧的对白，可以从推特、微博、豆瓣等论坛上爬取，或者由人工生成。目前聊天机器人系统的公开数据集有很多，例如UDC、Switchboard corpus、CALLHOME、 CALLFRIEND、Cornell Movie-Dialogs、心理咨询问答语料库、Topical-Chat、EMPATHETICDIALOGUES。并且在聊天机器人投入使用后，也会产生大量的数据，可以供聊天机器人系统训练使用。

生成对话的方式可以是基于检索的或者生成式的。基于检索的回答方式把用户最近的输入作为查询输入query，使用信息检索的方法，从query-response语料库中检索到与用户输入匹配度最高的query，把它对应的responses作为最佳答案输出。但是仅仅把用户最近的输入作为查询，缺少了上下文信息，不具备记忆能力，很难保证聊天一致性，因此可以把之前的对话都作为查询或者使用更复杂的神经网络架构。

而生成式的方式则可以看作是另一种形式的机器翻译，将结构化的数据翻译为对应的文本。生成式的方法可以加入更多的上下文信息，例如把之前所有的对话，包括用户所有的输入和系统生成的回答。例如\cite{t3}使用了动态记忆网络，输入不再被认为是不相关的，通过情景记忆模块结合输入、前一时刻的记忆和问题迭代生成记忆，并且生成一个答案的向量表示，最终根据答案向量以及问题向量来生成回答。但是seq2seq方法倾向于生成一般性的回答，如果使用包含更多信息的文本作为数据集，那么聊天机器人就会更为博学。例如微软小冰收集了公开演讲和新闻数据，系统可以使用这一部分信息来回答用户的问题。

**混合方法**

也可以将基于规则和基于语料库的方法结合起来，例如使用多个对话生成模型，每个模型都有一个优先级，最后使用排序算法选择最优的回答输出。

## 9.3 自动问答系统的搭建

随着互联网的发展，互联网中的信息量飞速增长，越来越多的用户利用互联网进行学习工作。用户向搜索引擎提出问题，在搜索引擎返回的一系列网页中寻找答案，而自动问答系统是信息检索系统的另一种形式，与搜索引擎不同，自动问答系统可以直接用简洁、准确的文字返回与用户问题有关的答案。

**问答系统的分类**

根据依赖的知识领域，问答系统可分为面向限定域的问答系统和面向开放域的问答系统。顾名思义，面向限定域的问答系统主要使用限定领域的语料库，只限定于回答某个领域的问题，例如最高人民法院上线的“法信（智答版）”[[1]](#footnote-1)，能向法律职业群体提供专业的智能问答服务。而开放领域的问答系统则没有领域限制，更多用来回答事实类问题，也就是可以用简单事实来回答的问题。

根据系统处理的数据格式，问答系统可分为基于结构化数据的问答系统、基于自由文本数据的问答系统、基于问题答案对数据的问答系统。结构化数据主要是指结构化数据库，系统将问题转化成query语言对数据库进行检索得出答案，自由文本数据则是预先建立的大规模真实文本语料库，系统需要对文档进行理解从中挑选出问题的答案，而问题答案对主要是指特定应用场景中的常见问题（FAQ）和社区论坛中用户的问题与回答（CAQ）。

而根据实现方法的不同，自动问答系统可以分为基于信息检索的问答系统和基于知识的问答系统。基于信息检索的问答系统是通过信息检索的方式，在文档库中找到与用户问题相关的文档，再通过阅读理解的方法，在文档中抽取出问题的答案。而基于知识的问答系统则直接建立了问题的语义表示，例如将自然语言文本转换为逻辑表示，再用问题的语义表示去检索数据库得出答案。

**基于信息检索的方法**

基于信息检索的自动问答系统依赖于大量网页或科学论文文本，它的基本思路是，建立大量的本地文档库，用户输入问题$q$后，根据问题在文档库中进行检索，选择$n$个与用户问题最相关的文档，并通过答案抽取系统，在文档中选择出最符合用户问题的回答$a$。

我们首先考虑文档库检索模块，这一部分属于信息检索问题，在实际应用中，也可以通过调用搜索引擎来实现。对于文档库中的每一个文档$d$，系统都需要按照一定的标准计算出与问题$q$的相似度，并选择其中相似度最高的前$n$个文档。传统方法就是基于词频统计（例如TF-IDF、BM25）来计算$q$与每篇文档$d$的相似度。使用TF-IDF对词语进行编码，得到的是稀疏的向量。

在对文档库进行预处理后，就形成了一个词汇表，对词汇表中的每个词都计算了TF、IDF和TF-IDF值并进行保存。那么对问题$q$中的每个词$t$，计算$t$与文档$d$的相似度分数，并进行相加：

\begin{equation}

\operatorname{score}(q, d)=\sum\_{t \in q} \frac{t f-i d f(t, d)}{|d|}

\end{equation}

按照TF-IDF的计算方法，对于每一个问题，都要遍历一遍文档库来寻找哪些文档中含有问题中的词语，效率显然是比较低的。因此，为了提高效率，检索系统可以提前生成一个倒排索引文件。在倒排索引文件中，包含了文档库中的所有词语以及哪些文档包含这些词语，在计算符合度分数时，只需要根据词语在倒排索引文件中查找出相应的文档，就可以快速地完成相似度分数的计算。

但是TF-IDF方法也有一定的缺陷，我们考虑$t f-i d f(t, d)$的计算方法如下：

\begin{equation}

t f-i d f(t, d)=t f\_{t, d} \* i d f=t f\_{t, d} \* \log \frac{N}{d f\_{t}}

\end{equation}

其中$t$为问题$q$中的词，$d$为文档，$N$为文档库所含有的词语个数，$d f\_{t}$为有词语$t$出现的文档数目。TF-IDF方法的基本思想就是文档中出现该词语的次数越多，它与这个词语的相关性就越高，所以理论上TF-IDF值可能是无限大的。但实际上当词频增大到一定值后，词频的增加就不会对文档的相关性有显著的影响。另外，某个词在文档中出现的频率高可能是因为文档很长，并不意味着这篇文档与这个词语有着很强的相关性。因此BM25增加了可调节的常数参数$k$和$b$，其中参数$k$限制了$t f\_{t, d}$值的增长速度，参数$b$对文档长度进行归一化。

\begin{equation}

B M 25(t, d)=\frac{t f\_{t, d}}{k\left(1-b+b\left(\frac{d\_{l}}{a v g d\_{l}}\right)\right)+t f\_{t, d}} \* \log \frac{N}{d f\_{t}}

\end{equation}

其中$d\_{l}$为文档$d$的长度，avgd $\_{l}$为文档库中文档的平均长度，$b$在$[0,1]$区间内取值，如果$b=0$，那么BM25就不会进行文档长度归一化，如果$k=0$，那么BM25退化为TF-IDF。

但是TF-IDF、BM25这类基于词频的方法有一个缺陷，如果问题$q$有词语在文档库中没有出现，那么这个词不会对检索有所帮助，用户需要尽量使用文档库中含有的词语进行检索，才有可能得到预期的结果，这个问题被称作信息检索中的词汇不匹配问题(vocabulary mismatch problem)。为了解决这个问题，$q$和$d$的编码方式必须能处理一词多义和一义多词的情况，例如使用隐性语义索引（LSI）方法来进行检索，将词和文档映射到潜在语义空间，能一定程度上提高检索的精确度。还可以使用神经网络的方法来进行训练，得到$q$和$d$的嵌入表示，比如使用两个BERT模型，分别对问题$q$和文档$d$进行编码，最后计算两个向量的点积，就得到了$q$和文档$d$的相似度分数。

\begin{equation}

\operatorname{score}(q, d)=\cos (q, d)=\frac{q}{|q|} \cdot \frac{d}{|d|}

\end{equation}

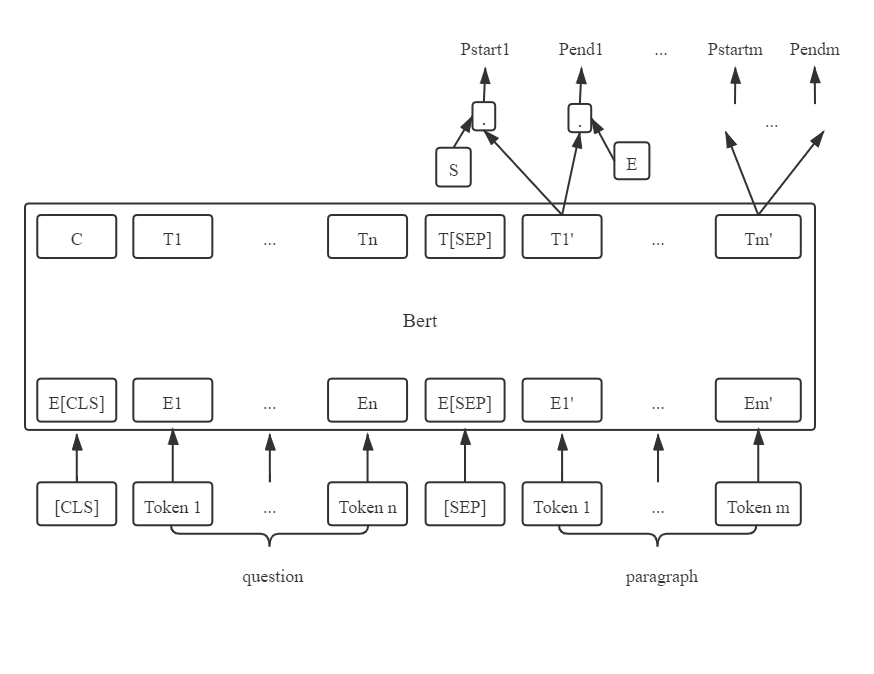
但使用BERT模型还需要注意文档$d$可能超出了模型编码器能处理的范围，还需要使用其他方法来进行处理，例如把文档拆分成几个片段，或者在BERT中引入滑动窗口来读取文档数据。

**阅读理解模型**

在完成了文档库的检索后，就需要在检索出来的文档中抽取出最符合问题的片段。这一步通常是用机器阅读理解模型，模型可以从文档中抽取一段子序列作为答案，也可以根据文档生成一段答案。在这里，我们讨论抽取式的阅读理解模型，模型将问题和文档作为输入，从文档中抽取能回答问题的连续子序列，把该子序列作为正确答案$a$并返回。

对于问题$q\left(q\_{1}, q\_{2}, \ldots, q\_{n}\right)$，以及段落$p\left(p\_{1}, p\_{2}, \ldots, p\_{m}\right)$，模型的目标是要最大化概率$P(a \mid q, p)$。如果答案子序列$a$在段落$p$中对应的下标为$a\_{s}, a\_{s+1}, \ldots, a\_{e}$，即$a=p\_{a\_{s}} p\_{a\_{s+1}} \ldots p\_{a\_{e}}$，那么模型的目标可以简化为最大化概率$P(a \mid q, p)=P\_{\text {start }}\left(a\_{s} \mid q, p\right) P\_{\text {end }}\left(a\_{e} \mid q, p\right)$。

我们以BERT模型为例\cite{t2}，在BERT的fine-tuning阶段，用$[C L S]$表示输入样本的开始，添加在输入样本的前面，用$[S E P]$把$q$和$p$序列分隔开，将$[C L S] q\_{1} q\_{2} \ldots q\_{n}[S E P] p\_{1} p\_{2} \ldots p\_{m}$输入模型编码器，得到$p\_{i}$的embedding表示。接下来，对于每一个$p\_{i}$，都需要计算它作为连续子序列的起始和结束的概率，即$P\_{\text {start }}(i)$和$P\_{\text {end }}(i)$。因此，模型中加入了$S$ (span-start embedding)和$E$ (span-end embedding)向量，通过计算$S$向量和$p\_{i}$向量的点积，并通过softmax运算得到概率$P\_{\text {start }}(i)$，$P\_{\text {end }}(i)$的计算同理。最终，找到一组下标$(i, j), j \geq i$，它的概率最大，那么这一段子序列就作为答案输出。如果在文档中得不到问题的答案，模型就输出$[C L S]$。



同样在训练模型时，段落$p$可能超出了模型编码器能处理的范围，尤其是在处理百度百科或维基百科这一类长文本的时候。因此，我们可以规定一个窗口大小，滑动这个窗口来遍历整个长文本，从中找到最佳答案。

**基于知识图谱的方法**

正如之前提到的，基于知识的自动问答系统的基本思想是将问题从自然语言形式转换成它的语义表达，再在结构化的语义知识库中寻找答案。这一类方法把大量的文档库数据聚合成了更为简洁的结构化知识库，减少了查询时的难度，对查询的速度也有了一定的提升。

知识库通常是用RDF（Resource Description Frame资源描述框架）描述的，其中的数据采用三元组的形式，实体作为节点，关系作为边把实体和实体或者实体和属性联系起来，构成实体-关系-实体或者实体-属性-属性值的三元组。知识库的构建可以有很多方法，可以从结构化的数据直接映射为知识库，也可以利用信息抽取技术，从非结构化的文本中进行命名实体识别、关系抽取和事件抽取等来构建知识数据库，在这里我们就不过多介绍了。关于基于知识的自动问答系统目前有很多公开的数据集，例如FREE917、SimpleQuestions、FreebaseQA、DBpedia、WebQuestions、WebQuestionsSP、ComplexWebQuestions、PQ等等。

对知识库进行查询首先需要将问题转换成相应格式的查询语言，例如，SPARQL、λ-DCS等。那么，怎么把问题转化为它的语义表达呢？基于知识的问答系统的工作方式可以分为基于语义解析、基于答案排序的两种。

**基于语义解析的方法**

语义解析算法可以将问题转换为逻辑形式，例如谓词演算的形式，再把逻辑形式转换为特定的SQL查询语言来查询知识库。语义分析方法可以分为基于文法的语义解析方法和基于神经网络的语义解析方法。

基于文法的语义解析方法依赖于有语义表示的标注数据，从标注数据中抽取语义分析规则的集合。然后，采用基于动态规划的语义解析算法，例如CYK算法和Shift-Reduce算法，将用户输入的问题转换为对应的语义表示。

基于神经网络的语义解析算法则采用端到端的结构，完成语义表示的转换。语义解析算法可以是全监督的也可以是弱监督的，传统的全监督算法需要把问题对应的逻辑形式作为问题的标签，输入到Seq2Seq网络中，来实现问题到逻辑形式的映射。而弱监督的算法需要把问题对应的答案作为问题的弱标签，逻辑形式被建模为隐性变量。例如在\cite{t1}中，在通过构建的词汇表进行了命名实体识别和关系识别后，为了正确的将他们联系起来，作者使用桥接自底向上地构建了所有可能的语法树，然后根据训练的分类器来找出最佳的语法树。

**基于答案排序的方法**

基于语义解析的方法依赖于带语义标注的问题集合，对于数据的标注量、标注程度都有一定的要求，而基于答案排序的方法则把该任务看成信息抽取和信息检索的任务，减少了标注数据的时间成本。

给定问题query，首先使用信息抽取的方法把非结构化的文本中的信息提取出来并转化为结构化的数据，它的子任务包括命名实体识别、指代消解、关系抽取、事件抽取等。在识别出问题的主题词等特征后，根据这些特征在知识图谱中抽取与主题词相关的子图，把子图中的节点或关系视作候选答案。例如\cite{t6}将知识库视为 ‘topic’ 互相连接的集合，提取了问题的疑问词、焦点词、动词和主题作为特征，来确定候选答案。

对答案进行打分和排序可以使用基于特征或基于向量的方法。基于特征的答案排序方法首先要对问题进行分析，得到问题的结构化表达（例如语法依存树），保留树中的关键信息，删去其他信息，就得到了问题的特征。最后将问题的特征和候选答案输入分类器，得到最优答案。而基于向量建模的方法则是将候选答案和问题都转换为向量，计算向量的点积得到相似度分数，得分最高的候选答案作为最终答案输出。

## 参考文献

[1] Berant J, Chou A, Frostig R, et al. Semantic parsing on Freebase from question-answer pairs[C]//Proceedings of the 2013 Con-ference on Empirical Methods in Natural Language Processing. [S.l.:s.n.], 2013: 1533

@inproceedings{t1,

title = "Semantic Parsing on {F}reebase from Question-Answer Pairs",

author = "Berant, Jonathan and

Chou, Andrew and

Frostig, Roy and

Liang, Percy",

booktitle = "Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing",

month = oct,

year = "2013",

pages = "1533--1544",

}

[2] Devlin J , Chang M W , Lee K , et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding[J]. 2018.

@inproceedings{t2,

title = "{BERT}: Pre-training of Deep Bidirectional Transformers for Language Understanding",

author = "Devlin, Jacob and

Chang, Ming-Wei and

Lee, Kenton and

Toutanova, Kristina",

booktitle = "Proceedings of the 2019 Conference of the North {A}merican Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)",

month = jun,

year = "2019",

pages = "4171--4186",

}

[3] KUMAR A, IRSOY O, SU J, et al. Ask me anything: Dynamicmemory networks for natural language processing[J]. CoRR, 2015,abs/1506.07

@article{t3,

author = {Ankit Kumar and

Ozan Irsoy and

Jonathan Su and

James Bradbury and

Robert English and

Brian Pierce and

Peter Ondruska and

Ishaan Gulrajani and

Richard Socher},

title = {Ask Me Anything: Dynamic Memory Networks for Natural Language Processing},

journal = {CoRR},

year = {2015},

}

[4] Reiter E. An architecture for data-to-text systems[C]//Proceedingsof the Eleventh European Workshop on Natural Language Generation(ENLG 07). [S.l.: s.n.], 200

@inproceedings{t4,

title = "An Architecture for Data-to-Text Systems",

author = "Reiter, Ehud",

booktitle = "Proceedings of the Eleventh {E}uropean Workshop on Natural Language Generation ({ENLG} 07)",

month = jun,

year = "2007",

pages = "97--104",

}

[5] Weizenbaum, Joseph. ELIZA—A Computer Program For the Study of Natural Language Communication Between Man And Machine[J]. Communications of the Acm, 1983, 9(1):36-45.

@article{t5,

author = {Weizenbaum, Joseph},

title = {ELIZA—a Computer Program for the Study of Natural Language Communication between Man and Machine},

year = {1966},

issue\_date = {Jan. 1966},

journal = {Commun. ACM},

month = jan,

pages = {36–45},

numpages = {10}

}

[6] Yao X, Van DURME B. Information extraction over structured data: Question answering with Freebase[C]//Proceedings of the 52nd AnnualMeeting of the Association for Computational Linguistics (Volume 1:Long Papers). [S.l.: s.n.], 201

@inproceedings{t6,

title = "Information Extraction over Structured Data: Question Answering with {F}reebase",

author = "Yao, Xuchen and

Van Durme, Benjamin",

booktitle = "Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)",

month = jun,

year = "2014",

pages = "956--966",

}

1. http://www.lawiask.com/ [↑](#footnote-ref-1)