# 摘要

# Abstract

# 目录

# 第1章 绪论

## 1.1 研究背景及意义

当前的社会是一个高度网络化的社会，随着近年来网络的发展，人们的社交有相当大的一部分逐渐转移到网络上。近年来，随着一些网络社交平台如国内的新浪微博、腾讯微博，国外如Twitter等的出现，大批的人群加入到网络社会这个虚拟的社会中，而网络社会里的信息也呈爆炸式的增长。这些社交媒体上包含了大量的有用的信息，我们从中可以发掘出有指导意义的内容。

新浪微博是一个可以把文字、图片等信息分享至自己的关注者的平台，它基于关注/取消关注来建立/解除好友关系。它的一大特点是发布的微博最多只能是140个字符，这种限制能够让用户把信息说明的更加清晰，因此，选择微博作为本文获取信息的来源。

要让计算机程序能够代替人力从微博中获取到有用的信息，就必须把微博文本转换成计算机能够识别的格式，并且相应的信息短能够用一些特定的标记符标记，这一过程通常被称作自然语言处理（Natural Language Processing，简称NLP）。

一些社会事件常常第一时间在微博平台上曝光出来，因此，微博上的信息具有及时性强的特点。然而，由于微博用户群体的庞大，虽然数据量会很大，但其稳定性却很难保证，因此，我们的微博数据来源主要是一些政府、相关部门，相关民间权威人士的微博，这样的数据更有真实性。

但微博这种类型的社交媒体属于新兴媒体，与传统媒体不同，它所携带的信息具有很多冗余，其中有很多都是无用信息。因此，如何从海量的信息中发掘出有用的信息点成了人们关注的焦点。

## 1.2 国内外研究现状

本小节将从自然语言处理和社交媒体事件发掘两个方面来阐述国内外的研究现状。

### 1.2.1 自然语言处理

自然语言处理是实现人与计算机之间直接通信的重要手段，自上个世纪90年代以来，国内外关于自然语言处理的研究日益增多，在特定的应用领域，一些商业化的应用已经开始出现，例如多语种数据库和专家系统的自然语言接口、各种机器翻译系统、全文检索系统等。

基于中文自然语言处理的难点目前主要在两个方面：一是当前的语言处理分析都限定在一个孤立的语句上，对于上下文环境的研究少之又少；二是由于自然语言的数据量庞大，无法完全存储到计算机中，当前的研究只能建立在一个有限的数据基础范围内。

### 1.2.2 社交媒体事件发掘

社交媒体事件发掘，就是从社交媒体平台上，海量用户发布的信息中，发掘出有价值的信息。随着网络的普及，社交媒体事件发掘的研究也越来越多，一些产品化的应用也开始出现，例如百度舆情等。

## 1.3 本文主要研究内容

本文主要研究如何从微博中发掘出有关的交通新闻类事件。主要包括：微博数据的抓取、过滤以及预处理部分、对微博预料的自然语言处理部分、事件挖掘的网站控制台部分以及事件挖掘的模型建立部分。

其中，微博数据的抓取研究了如何从微博平台抓取数据以及其标准化过程；对微博语料的自然语言处理部分研究了如何对微博语料进行自然语言处理；网站控制台部分研究了如何搭建一个可以查看并维护事件发掘系统的网站；模型建立部分研究了如何建立并完善一个微博事件主题模型。

本文所实现的事件发掘系统的整体结构及流程如诶图1.1所示。首先从微博获取数据经过相关过滤、标准化处理之后存入数据库，然后将数据库中的微博预料通过自然语言处理模块进行相关处理，同时，在网站上对事件进行人工标注相关元素，将人工标注后的结果和自然语言处理后的结果进行合并后转换成CRF训练输入标准格式的文件，通过CRF训练模块进行训练，最后得到事件发掘的结果。



图1.1 事件发掘系统流程图

经过本系统的训练处理，最后事件发掘的结果召回率为76.98%，精确度为83.62%，准确率为93.92%，综合F指数为80.17%。

## 1.4 本文主要结构

本文的主要结构如下：

1. 为绪论部分，论述本文研究的课题、项目背景、国内外研究现状以及本文所设计系统的结构、流程。
2. 论述了微博抓取及预处理模块的实现方案以及运行情况，讨论了相关方案的选择原因及性能。
3. 论述了自然语言处理也就是语言特征标注部分的实现过程及理论依据，描述了当前流行的一些自然语言处理方案的优缺点。
4. 论述了本系统网站控制台部分的技术选型以及实现过程，该网站控制台是系统操作的主要方式，本章还讨论了从网站完全控制系统运行的可能性。
5. 论述了条件随机场算法的定义以及核心思想，讨论了目前研究领域对CRF算法的应用以及CRF工具的选择。并论述了利用CRF工具完成事件发掘核心模块的实现方案。本章还给出了提高结果可靠性的方案及测试过程，展示了最优训练参数的选取过程，是本文的核心章节之一。
6. 描述了整个系统的运行情况，展示了事件发掘的结果。
7. 分析了系统运行结果，对系统的不足提出了改进方案。

# 微博数据获取及预处理

本章主要研究了微博数据的获取和预处理过程，微博数据获取是指从微博获得我们想要的微博文本，预处理是指通过一定的处理过程将上一步获取到的文本进行加工，使之符合下一步数据库入库标准。

## 2.1 数据需求简述及分析

本文所实现的事件发掘系统主要针对交通事件，由于需要一定量的交通事件微博作为原始语料来建立训练模型，所以这一步抓取的微博都应是交通主题的微博。为了达到这一个效果，微博抓取模块的数据来源被确定为一些有一定人气的交通方向的博主，例如“@武汉交通”、“@今日出行”等微博博主。

## 2.2 抓取方案选择

从微博获取数据的方式分为全手动和程序自动化获取两种方式，显然，采用编写程序抓取微博的方式的效率远高于手工方式。考虑到自己的技术能力，可选的方案有三种，分别是Python 爬虫框架Scrapy、自己编写爬虫程序和采用新浪微博API接口来获得数据，本节将分别分析这三个方案的优缺点并选择一个综合方案。

### 2.2.1 Python爬虫框架Scrapy

Python是一种面向对象的解释性计算机程序语言，其语言语法简洁明了，易于上手，在机器学习方面应用十分广泛。

Scrapy是采用Python语言编写的一个爬虫框架，该框架内部实现了几种常用的爬虫基类，通过简单的编写规则就可以从网站爬取信息，效率高，适合于爬取各种类型的信息型网站。

### 2.2.2 自写爬虫

由于编写爬虫可以使用很多种编程语言，例如Python、Java、Node.js等，这一类自写的爬虫相对Scrapy来说更易把握，只需要编写合适的正则表达式就可以从网页中提取信息，适合于中小型数据的爬取及简单网站的爬取。

### 2.2.3 新浪微博API

除了用爬虫获取数据的方式之外，新浪微博还为开发者提供了api来获取接口数据。只需要在新浪开放平台注册成为新浪开发者，就可以调用一部分接口来获取数据。以新浪微博api的“/status/user\_timeline”接口为例，它将使用JSON格式来返回关注的用户所发表的微博列表，这种格式的数据操作起来十分方便。但这种方式获取数据的缺陷在于新浪微博api对接口访问频次有较高的限制（不高于150次每小时），如果访问过于频繁，短时间内将不能再发起接口调用。

### 2.2.4 综合方案选择

综合三种方案，考虑到本文需要的数据量不大，网站简单，我们将以第三种方案为主，第二种方案为辅进行数据的抓取。即抓取数据时系统优先开启第三种爬取模式，如果遇到新浪微博的频次限制，则将爬取模式改为第二种。

## 2.3 微博的数据清洗及标准化

由于微博语言不像传统的新闻媒体报道一样十分严谨、严肃，它常常伴随着一些口语化的词语、表情、链接等无关信息，有的微博句式结构复杂，句子不止一条，这样的微博文本不能直接拿来作为标准语料，需要对其做一些预处理，这一过程成为微博的数据清洗及标准化。

我们需要过滤的无关字符主要包括：

1. 表情符号，如“[Dog]”；
2. 内嵌链接，如“http://t.cn/xxxx”；
3. 话题标签，如“#早安五月#”；
4. 人物关系标记，如“@新浪微博”;

过滤这些无关字符采用正则表达式匹配的方法。另外由于就目前的自然语言处理技术来说，所针对的对象都是单个句子，并不能对上下文环境做出分析，因此，我们还需要对主体句子进行提取，在过滤完无关字符后，我们选取最长的一个句子作为主体句子。

## 2.4 数据存储入库

在本论文所要实现的系统里，语料数据和网站后台数据是公用一个数据库的，出于格式的考虑可以选择非关系型数据库MongoDB，出于网站结构化数据的考虑，可以选择Mysql数据库。前者存储数据的类型是文档，后者是结构化数据，相比之下，前者操作更为方便，后者更适合作为网站后台，权衡之下，最终选择Mysql数据库作为本系统数据库。

将2.3节中经过标准化预处理的数据转换成需要的对象格式，存储到mysql数据库中，即完成了数据存储入库的过程。

# 语言特征标注

## 3.1 自然语言处理概述

前文中已经描述过，自然语言处理的目的是用来实现人机之间的通信。从中文语言的角度来说，所有的自然语言都是由汉字组成句子，再由句子组成段落、章节、文章，进而使得中文语言具有了表达意思的能力。自然语言处理从句子的层次出发，将章节、段落都拆分成句子，再对句子进行分词、词性标注、命名实体识别、依存句法分析等一系列处理，最终转换成计算机能够识别的语言。

本章接下来将论述上述过程的实现方案及各个过程的实现说明。

## 3.2 NLP工具选择

所谓的NLP工具，是指能够通过程序调用来实现自然语言处理过程的工具包。目前比较流行的工具包主要有：结巴分词、OPENNLP工具包、中科院分词系统ICTCLAS、哈工大-科大讯飞语言云LTP。下面将比较这些工具的优缺点并作出选择。

### 3.2.1 结巴分词

结巴分词是一个开源的python中文分词组件，具有分词、添加自定义分词词典、关键词提取、词性标注、并行分词、搜索引擎等关于分词和词性标注的功能。

使用结巴分词需要采用python程序开发语言，安装依赖库jieba，调用相关SDK接口，就可以完成分词和词性标注的功能。优点是分词速度快，分词速度默认模式下能够达到400Kb/s，缺点是只能实现分词和词性标注的功能需要结合其他语言处理套件来使用。

### 3.2.2 OPENNLP工具包

OPENNLP是Apache开源的一个基于JAVA程序开发语言的工具包，其支持windows和linux操作系统，能够实现分词、词性标注、命名实体识别的功能，优点是功能全面，缺点是JAVA代码调用比较复杂，对中文的支持不太友好。

### 3.2.3 哈工大-科大讯飞语言云LTP

LTP是由哈工大研发、科大讯飞商业化的一个国产语言云，能够完成一套NLP的分词、词性标注、命名实体识别、依存句法分析，语义依存分析的流水线功能，提供本地化SDK、python依赖库、HTTP/HTTPS语言云接口等多种调用方式，对中文的支持也十分友好。

### 3.2.4 处理工具选择

综上所述，最终本文选择采用LTP语言云的HTTP接口作为本文NLP工具包。接下来将会以这个工具包为例，分别描述NLP部分各个功能的实现过程。

## 3.3 LTP处理结果格式说明

我们采用Python程序开发语言访问LTP的REST处理接口，可以选择XML、JSON、CONLL、PLAIN这4种结果表示格式。其中较为常用的是CONLL格式和XML格式，下面分别对这两种格式加以说明。

### 3.3.1 XML格式说明

XML是一种常见的网络文本传输格式，其格式严格，识别方便。以句子“我是信息学院的学生。”为例，LTP返回的XML格式结果如图3.1所示。

节点标签分别为：xml4nlp、note、doc、para、sent、word、arg共七种节点标签，其中：

* xml4nlp为根节点，无任何属性值；
* note为标记节点，它具有的属性为：sent、word、pos、ne、parser、semparser、lstmsemparser、wsd、srl，分别表示分句、分词、词性标注、命名实体识别、依存句法分析、语义依存树分析、语义依存图分析、词义消除歧义、语义角色标注；这些属性的值可以是‘y’或‘n’，分别代表未处理和处理两种状态；
* doc为篇章节点，以段落为单位包含文本内容；
* para为段落节点，具有属性名id，其值从0开始递增；
* sent为句子节点，属性id为编号，属性cont为句子内容；
* word为分词节点，属性id是编号、cont是分词结果，可选属性包括pos、ne、parent、relate、semparent、semrelate；等，分别表示词性标注、命名实体识别等内容；
* arg为参数节点，包含相关参数；
* sem为语义依存节点；

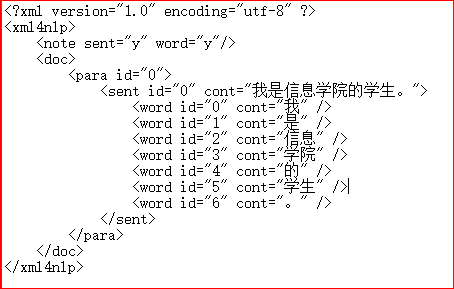


图 3.1 XML结果示意图

### 3.3.2 CONLL格式说明

CONLL格式是一种多列格式，每一行代表一条处理单元，每一列代表该处理单元的一个相关属性。由于这种格式容易转换成标准语料的格式，所以在NLP中使用的十分广泛。下面以句子“我是信息学院的学生。”为例，介绍LTP系统的CONLL格式的结果表示说明。该格式结果见图3.2；

可以看到该CONLL格式的结果共分为12列，每一行为一个分词单元，每个单元具有12个属性，这些属性分别为：

* 第一列为分词id，从0开始递增；
* 第二列为单词本身；
* 第三、四列为空，留作保留属性；
* 第五列为单词词性标注信息；
* 第六、七列表示依存句法关系；
* 第八、九列为空，保留列；
* 第十列为谓词；
* 第十一列及以后为语义角色标注信息；

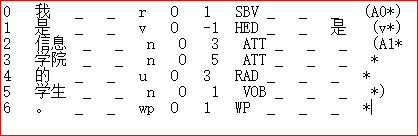


图 3.2 CONLL格式结果

在本位的系统中，这两种格式都将得到应用，XML格式用于在网站上导出人工标注文件时使用，应为XML更适合网络间传输。CONLL格式将在进行NLP过程时使用。

## 3.4 LTP各个处理过程说明

LTP提供的接口包含了一系列处理，具体包括中文分词、词性标注、命名实体识别、依存句法分析、语义依存分析等，下面分别加以举例说明。

1. 中文分词

中文分词（Word Segmentation，WS）指的是把中文句子切分成单词序列，单词是承载意思的基本单元，所以对句子正确的进行分词十分重要。例如，句子“我是中国人”经过分词得到的结果为“我/是/中国/人”，这样一来，句子就变成了五个独立的单词，每个单词都具有独立的含义，可进一步用于研究分析。

1. 词性标注

词性标注（Part of Speech tagging， POS）指的是在句子已经经过分词的情况下，对每个词给出词性的标注，是名词、动词、还是其他词性的词。词性标注是后续研究语义依存关系的基础，它能够显著提高后续处理的准确性。目前而言，词性标注主要是面向那些具有不同词性的词以及未收录的词的标注工作，这些词虽然有不同的词性或者词性未知，但在相同的上下文环境下，应该具有相同的词性。例如，“爱”这个单字词同时具有动词和名词两种词性，但在同一种上下文环境中，它的词性应该是相同的。

1. 命名实体识别

命名实体识别（Named Entity Recognition，NER）指的是识别出句子中具有特定含义的词，例如人名“张三”、地名“湖北”、组织机构名“教育部”等等。这种命名实体共分为三大类、七小类。三大类是指：时间、数字、实体类，七小类是指：机构、地点、人名、货币、时间、日期和百分数。

命名实体识别是被广泛应用在很多信息处理应用上的一个工具，它的结果直接会影响整个自然语言处理的结果的准确性。

1. 依存句法分析

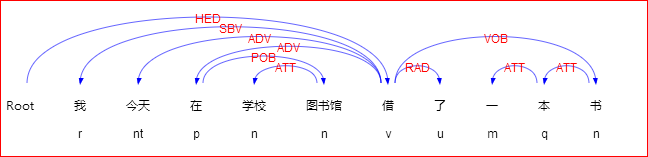
依存句法（Dependency Parsing，DP）分析是指通过分析语句单位之间存在的关系来寻找其语法结构，换而言之，就是找出句子里的“主谓宾”、“定状补”等语法语句成分，并对各个成分之间的关系加以分析。以句子“我今天在学校图书馆借了一本书”为例，进行依存句法分析的结果如图3.3所示。

图 3.3依存句法分析示意图

可以看到，该句子在分词、词性标注之后，再进行依存句法分析，能够看到主谓宾等语法成分都被标记了出来。HED是指整个句子的核心（“借”），SBV指明了主谓关系（我、借），ADV指明了句子里的状中结构，POB代表介宾关系，ATT代表定中关系，RAD代表又附加关系，VOB代表动宾关系（借、书）。

通过这种句法分析对句子成分加以标记，可以间接的来创造一种句子内的“上下文环境”，是计算机可以获取更多句子的信息加以分析。

## 3.8 人工事件标记

前文所说到的一系列处理、标注针对的对象是所有的中文语句通用的处理方法，然而，由于本系统是要对特定主题的事件进行建模，而上述过程并不能标记出一条句子是属于什么主题的事件类型，这就需要我们人工介入标注。以本文处理的交通类型的微博为例，我们对作为训练语料的数据进行人工主题标注，给微博文本的处理结果加上专属于交通主题类型的标记，这样建立出的模型就能够很好的识别交通类型的事件。

所谓标注，就是在经过LTP处理后的语料上，对符合相关主题的语料添加一列属性值，如果不是交通类型，该列值为“O”，如果属于交通类型的事件，则在LTP处理结果的基础上，该列的值可以根据实际情况被标记记录为who/whom/when/where/how/trigger这六个值之一。其中，when是指事件发生的时间，where是事件发生的地点、who是事件发生的主体人，whom是涉及到的客体，how是代表该词表明事件如何发生，trigger是指该词为这个主题事件的触发词。

在系统的后台网站上，可以对每一条微博进行标注，标注的界面如图3.4所示，标注好后可从网站导出相应的标记好的XML格式数据，其示例如图3.5所示。

图 3.4 在网站上标注微博界面

可以看到，在图3.5给出的结果示例里，一条交通主题的微博的成分被人工标记了trigger、whom、when等事件元素。这样的数据格式经过转换即可以作为CONLL格式数据的一个列属性，以便后续操作的进行。

图 3.5 网站导出的标注结果示例

通过后台网站上的标注功能，本文对八百条交通事件的微博进行了人工标注，把标注后的结果与LTP的处理结果相结合，即可获得训练与测试语料。本文接下来的两章将对网站和CRF训练进行详细的设计实现说明。

## 3.9 本章小结

本章介绍了语言特征标注模块的方案选择以及实现过程，也描述了如何进行人工事件标注，本章所介绍的语言特征标注模块是用来获取标准训练数据的模块，在本章的基础上，后文将使用本章的到的数据进行事件的展示、建模、训练和发掘测试。

# 后台管理网站搭建

## 4.1 网站需求分析与说明

## 4.2 网站技术栈确定

### 4.2.1 前端技术选型

### 4.2.1 后台技术选型

## 4.3 本章小结

# CRF算法与事件挖掘的实现

## 5.1 事件发掘的模型选取

## 5.2 条件随机场（CRF）

### 5.2.1 CRF定义

### 5.2.2 CRF优势

## 5.3 CRF工具

### 5.3.1 CRF工具选择

### 5.3.2 CRF++训练套件

## 5.4 CRF训练

### 5.4.1 训练数据的获取

### 5.4.2 训练模板的设计

### 5.4.3 训练结果测试

## 5.5 最优训练参数筛选

### 5.5.1 评测体系

### 5.5.2 训练参数选择

### 5.5.3 最优参数确定

## 5.6 本章小结

# 系统运行结果

## 6.1 系统整体结构

## 6.2 微博抓取模块结果

## 6.3 网站运行情况

## 6.4 语言特征标注结果

## 6.5 CRF训练结果测试

# 结论

## 7.1 本文工作总结

## 7.2 研究与展望

# 参考文献

# 附录

# 致谢