教育部高等学校大学计算机课程教学指导委员会

中国大学生计算机设计大赛

大数据/人工智能挑战类作品文档简要要求

作品编号：66821

作品名称：基于规则和深度学习的电子病历信息抽取系统

作　　者：颜烨、孙轶伟、王宇、谭帷、杜圣辉

版本编号：2.4

填写日期：2019.06.05

填写说明：

1、本文档适用于大数据挑战赛小类、人工智能挑战赛小类；
2、正文一律用五号宋体，一级标题为二号黑体，其他级别标题如有需要，可根据需要设置；
3、本文档为简要文档，不宜长篇大论简明扼要为上；
4、提交文档时，以 PDF 格式提交本文档；
5、本文档内容是正式参赛内容组成部分，务必真实填写。如不属实，将导致奖项等级降低甚至终止本作品参加比赛。

目　录

第一章　项目背景 3

第二章　系统关键问题研究 4

第三章　系统实现 9

第四章　测试结果 13

第五章　安装及使用 14

第六章　项目总结 14

参考文献 14

附录 15

第一章 项目背景

【填写说明：说明前人已有的相关工作，从多个维度分析已有工作与本作品的比较。在本章的最后，请用单独一节阐明本设计的创新点，并对作者自认为的测试结果进行100字内的简述】

第一节. 前人相关工作

本项目的工作属于自然语言处理中的信息抽取领域的命名实体识别方向，针对医疗中的肠癌电子病历数据。

现今信息抽取的研究主要方法有：基于规则和知识的方法；基于统计的方法，如n元模型、隐马尔科夫模型（HMM）、最大熵模型（ME）、条件随机场（CRF）等，其中公认比较好的CRF；基于深度学习的方法，如LSTM、BiLSTM；混合式的方法，如先利于规则剪枝，再用统计或深度学习方法，也有统计和深度学习结合的方法，如BiLSTM-CRF模型，是目前最前沿的方法。

同前人的工作相比，本作品采取了基于规则和深度学习的方法，使用了Bi-LSTM-CRF模型。由于肠癌电子病历数据规范性强，对于病情的描述医学上有专有词汇规定，因此取得了较好的效果，20个属性上的平均准确率94.23%。

第二节. 本作品工作创新

本作品的主要工作创新在于，将分治法的思想，融入进基于规则和知识的方法。具体而言，本作品先将样本切片成若干条分句，在底层通过初步的规则匹配到与属性相关的分句，再将对应分句传输到高层，在高层通过进一步详细的规则提取属性对应值，实现文本的自动抽取。

这样的好处主要有两点。首先，底层的识别分句结果为高层识别提供决策支持，将规模大的一个问题分治为规模小的两个问题，再逐个解决，有效的减轻了问题的难度，使得规则的设计更加简易。其次，底层的识别结果是直接可见的，可以分别对底层识别和高层识别的规则进行调整，二者不会相互影响，而端到端的方法难以看到中间过程，使调整模型存在难度，容易牵一发而动全身。

通过对大赛发放的1000个病例样本数据统计，我们发现除了题目中要求的14个属性之外，我们还发现另外的6个属性的出现率达到10%以上，并且这6个属性对于确定病人患病情况有这重要作用，因此我们决定增加对这6个属性的信息抽取，以此达到对电子病例信息的完善。因此我们统计病例样本中共计20个属性，其中新增的6个属性具体如下：“回肠切端是否累及”、“结肠切端是否累及”、“网膜是否累及”、“阑尾是否累及”、“肠旁淋巴结个数”和“参见报告”。

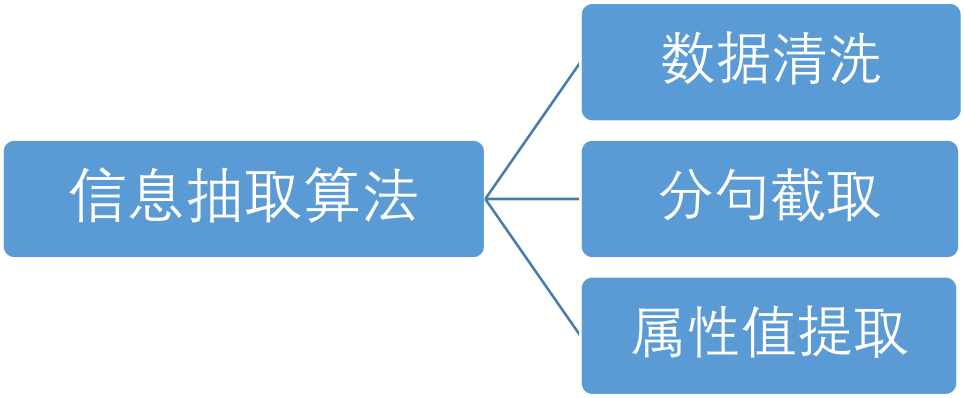
第二章 系统关键问题研究

【填写说明：介绍本设计所涉及的算法或自行提供的数据来源，作者必须在本章节中阐明算法的出处，以及作者对算法做出了哪些改进；如有必要，作者还必须阐明数据的来源、规模、处理过程等等】

本设计采用基于规则与深度学习结合的方法。基于规则的方法依赖规则库的构建，其中规则的提取来自赛题所给的 1000 例电子病历数据的观察总结规律；知识的构建来自于赛题数据以及肠癌领域的文献[3-5]；算法的思路来自论文[8-11]。

算法一：基于规则的信息抽取算法

病理和诊断知识的自动抽取算法主要由三个部分构成：数据清洗与预处理、截取属性对应分句、提取属性值。



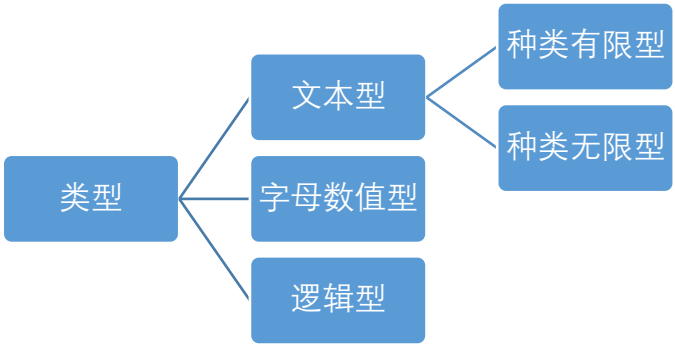
数据清洗与预处理主要包括：错别字的更改、乱码和异常符号的删除、分隔符的添加等。

截取属性对应分句和提取属性值的方法，需要维护规则库，再根据规则库构建提取方法，20 个属性不尽相同。

根据电子病历中常用的医学术语构建规则库（规则集合）。构建规则库的过程实际上也是命名实体识别过程。实体包括赛题中要求提取的 14 个属性与自行添加的 6 个属性。构建时必须对命名实体的构词、句法、语法特点进行分析，然后才可以在规则维护模块中定义规则，这是信息抽取工具最核心的模块。

1. 类型

指所要抽取的记录类型，按照电子病历常见记录类型分为 4 种，如下图所示。



文本型-种类有限型包括：病理、病理等级、性状；

文本型-种类无限型包括：标本、浸润；

字母数值型包括：肿瘤大小、分化等级、转移比例、肠旁淋巴结个数、参见报告；

逻辑型包括：上切端是否累及、下切端是否累及、基底切端是否累及、淋巴结是否转移、脉管侵犯、神经侵犯、回肠切端是否累及、结肠切端是否累及、网膜是否累及、阑尾是否累及。

2. 知识库建立

对于文本型-种类有限型的属性类型，可以通过事先归纳属性的取值范围，辅以其他规则帮助提取。

属性	取值范围
病理	乳头状腺癌，管状腺癌，黏液腺癌，粘液腺癌，梭形细胞癌，筛状粉刺型腺癌，髓样癌，微乳头状癌，印戒细胞癌，大细胞癌，小细胞癌，未分化癌，锯齿状腺癌，混合型腺神经内分泌癌，间质肿瘤，腺鳞癌，鳞状细胞癌，胰腺导管腺癌，透明细胞癌，黏液癌，错构瘤，淋巴瘤，绒毛状管状腺瘤，管状-绒毛状腺瘤，管状绒毛状腺瘤，癌疑，导管腺癌，细胞癌，管状腺瘤，神经内分泌癌，神经内分泌瘤，腺癌，腺瘤
性状	溃疡隆起型，弥漫溃疡型，弥漫浸润型，浅表隆起型,髓样型,局限溃疡型,IIa+IIc 型，溃疡浸润型，普通型，表浅平坦型，糜烂型，浅表溃疡型，肌壁间型，伴中度异型，内膜下型，壁间型，肠型，斑块型，Ⅱc 型，大细胞型，胆胰型，盘状型，浅表平坦型，外生型溃疡增殖型，髓质型，浅表凹陷型，表浅凹陷型，表浅隆起型，嗜酸细胞型，浸润至型，浆膜下型，不明确型，浸润型，增殖型，隆起型，平坦型，凹陷型，溃疡型，胶样型，菜花型，蕈伞型，中央型，周围型，弥漫型，表浅型
分化	中低分化，中-低分化，中-高分化，中高分化，高分化，中分化，低分化，g2，g1，g3，g4，G2，G1，G3，G4

3. 分句合并

将电子病历进行分句处理后，属性和其描述词可能会被分成两个句子，此时就需要对分句进行合并。如“上切端、下切端、基底切缘，均未见癌组织累及”会被分开，需要将其重新合并成一个句子“上切端、下切端、基底切缘均未见癌组织累及”。

4. 近义词

指向与关键词相类似的词或短语，在定义规则时需要考虑到这些形近词，以提高信息抽取的查全率。

关键词	形近词
上切端	上切缘,两侧切端,两切端,上、下切端,上、

	下切缘,两侧切缘,两切缘
下切端	下切缘,两侧切端,两切端,上、下切端,上、下切缘,两侧切缘,两切缘
累及	癌累及， 癌组织， 见肿瘤,肿瘤累及， 癌组织累及， 癌,侵犯，阴性，阳性
病理	组织学类型
分化等级	组织学分级
浸润	肿瘤扩散
侵犯	包绕、累及、内癌栓
×	*

5. 歧义词语

记录中可能有别的包含记录项的词语来混淆答案,抽取时可把这类歧义词语添加在歧义词语中进行排除。

关键词	歧义词
浸润	浸润性、浸润型、浸润至型、细胞浸润、浸润组织、浸润深度
网膜	网膜结节

6. 错别词

记录中可能存在由于数据录入失误导致的错别词或者不规范问题,抽取时先要对错别词进行更改。

关键词	错别词
切端	切断
未见	未及
增殖型	增值型
浸润型	浸润至型
III	Ⅲ
II	Ⅱ

7. 单位

有些记录是有单位的，特别是一些数值型记录，往往都带有固定单位，如“瘤体大小8.0×7.0×2.0cm”、“胰腺导管腺癌Ⅱ-Ⅲ级”,在从记录中抽取信息时，需把单位“cm”“级”等都定义进规则，以提高抽取命中率。

8. 可能答案

指对特定事物或记录的不同表达，但实质描述的事物或记录都是一样的。如判断“是否累及”记录时，与“是否累及”相关的可能答案还包括“未见肿瘤”“均未见癌”“见癌组织浸润”等。

9. 否定性答案的表述

医学术语中常包括很多否定性描述记录，在抽取时需对这类记录进行筛查，以提高查准率。如 “上切端、下切端及大网膜组织均未见癌累及”，虽然记录中出现“癌累及”，但前面含有了否定性词语，故抽取时需通过定义“否定性答案的表述”项进行排除。

算法二：基于统计学习方法：CRF 算法

CRF 主要用于序列标注问题，可以简单理解为是**给序列中的每一帧都进行分类**。在以往的分类算法（例如 CNN，RNN），都是在将序列进行编码之后，接一个全连接层用 softmax 激活，但是 softmax 激活函数在对于带有较强的上下文关联的输入层是便会产生缺陷，因此我们引入条件随机场模型（CRF）使得模型能够学习到输入层的上下文关联。具体对比如下：

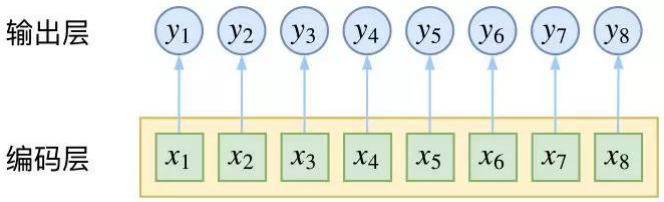


Fig1: 逐帧 softmax 并没有直接考虑输出的上下文关联

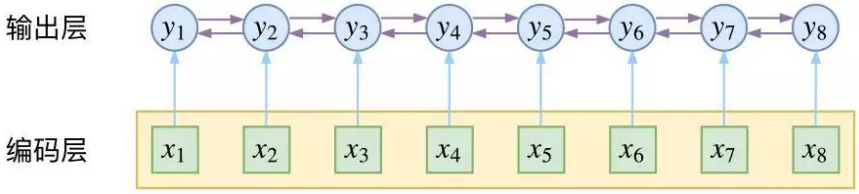


Fig2: CRF 在输出端显式地考虑了上下文关联

当然，引入输出的关联并不是 CRF 正精巧的地方，CRF 最优的优势在于它以路径为单位，考虑的是路径的概率。

模型概要：

假如一个输入有 n 帧，每一帧的标签有 k 中可能性，那么理论上就有 k^n 中不同的输入。我们可以将它用如下的网络图进行简单的可视化。在下图中，每个点代表一个标签的可能性，点之间的连线表示标签之间的关联，而每一种标注结果，都对应着图上的一条完整的路径。如下例子所示：

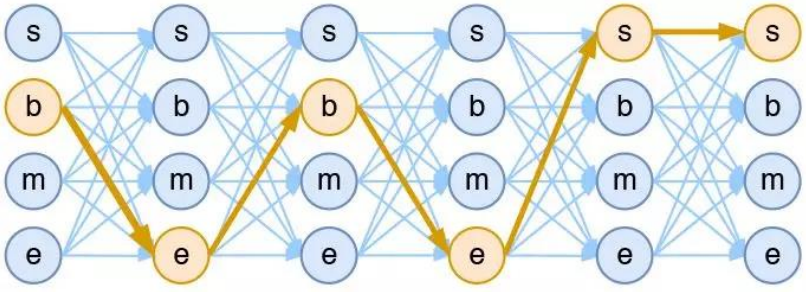


Fig3: 4tag 分词模型中输出网络图

特征函数：

$$score(l|s) = \sum_{j=1}^{m} \sum_{i=1}^{n} \lambda_j f_j(s, i, l_i, l_{i-1})$$

特征函数指数化和标准化：

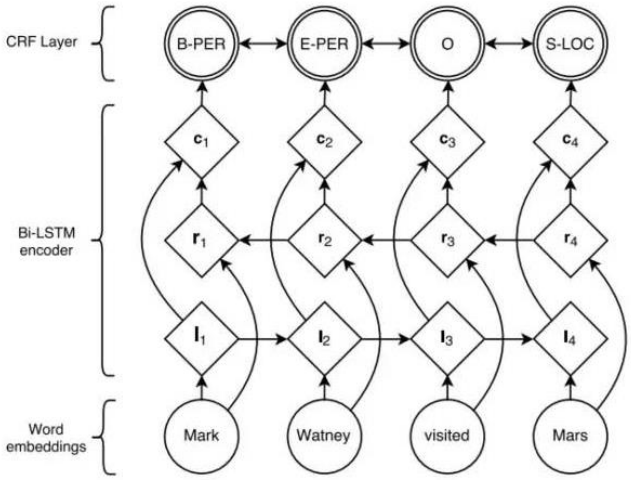
$$p(l|s) = \frac{exp[score(l|s)]}{\sum_{l'} exp[score(l'|s)]} = \frac{exp[\sum_{j=1}^{m}\sum_{i=1}^{n}\lambda_j f_j(s,i,l_i,l_{i-1})]}{\sum_{l'} exp[\sum_{j=1}^{m}\sum_{i=1}^{n}\lambda_j f_j(s,i,l'_i,l'_{i-1})]}$$

算法三：基于深度学习方法：Bi-LSTM-CRF 算法

从网络结构上来讲，Bi-LSTM-CRF 套用的还是 CRF 这个大框架，只不过把 LSTM 在每个 tt 时刻在第 i 个 tag 上的输出，看作是 CRF 特征函数里的“点函数”（只与当前位置有关的特征函数），然后“边函数”（与前后位置有关的特征函数）还是用 CRF 自带的。这样就将线性链 CRF 里原始的 w*f 这种形式的特征函数（线性）变成 LSTM 的输出 f_1（非线性），这就在原始 CRF 中引入了非线性，可以更好的拟合数据。

该框架优势总结起来就是：采用 CRF 获取全局最优的输出序列，对 LSTM 信息再次利用。

Bi-LSTM-CRF 算法体系架构图：



可以看到，Bi-LSTM-CRF 采用的双向 LSTM 架构，相比较于传统的 LSTM，Bi-LSTM 统考考虑过去的特征（通过向前过程提取）和未来的特征（通过向后过程提取）。

学习过程：

Bi-LSTM-CRf 架构对于输入的序列 X 对应的输出 tag 序列 y，定义如下分数，并且选择最大的分数作为最终输出的 tag 序列。

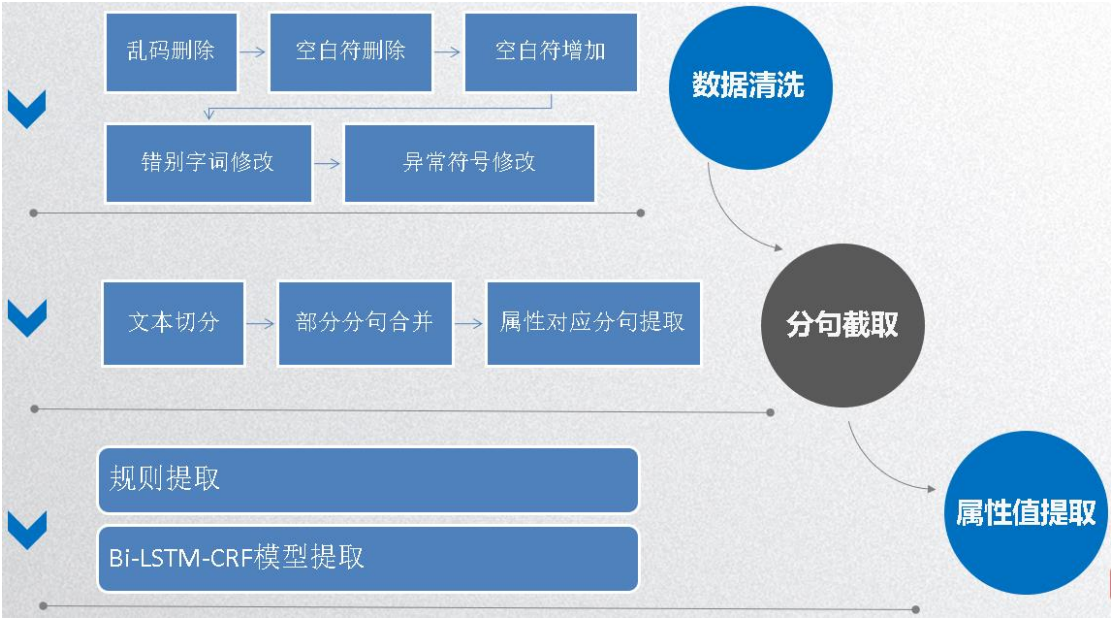
$$s(X,y) = \sum_{i=0}^{n} A_{y_i,y_{i+1}} + \sum_{i=0}^{n} P_{i,y_i}$$

第三章 系统实现

【填写说明：本章节组要阐明系统是如何实现的；对于有硬件参与的项目，这里还应当包括一切必要的，作者自行绘制的机械、电气、电路设计图纸。系统实现：系统具体的实现过程，其中遇到的困难，结局的方法等。本章节重点阐述作者自己的工作，非作者自己的工作简述即可。】

第一节. 信息抽取算法

一、算法思路



二、伪代码构建

建立好规则后，可以据此撰写伪代码和代码，属性对应分句截取和属性值提取的伪代码可见附录。

三、属性标注

在实验之前，由于大赛并没有给我们 1000 个病例对应的 14 个属性标签，因此我们必须要自己人工对 1000 个病例进行属性标注。我们具体的实验标注思路是首先将病例中有关属性出现的词语段截取出来；然后，我们采用 BIO 方法进行序列标注；最终我们进行人工将病例的文本无限型的标本和浸润标注出之后，用来测试我们的模型效果。

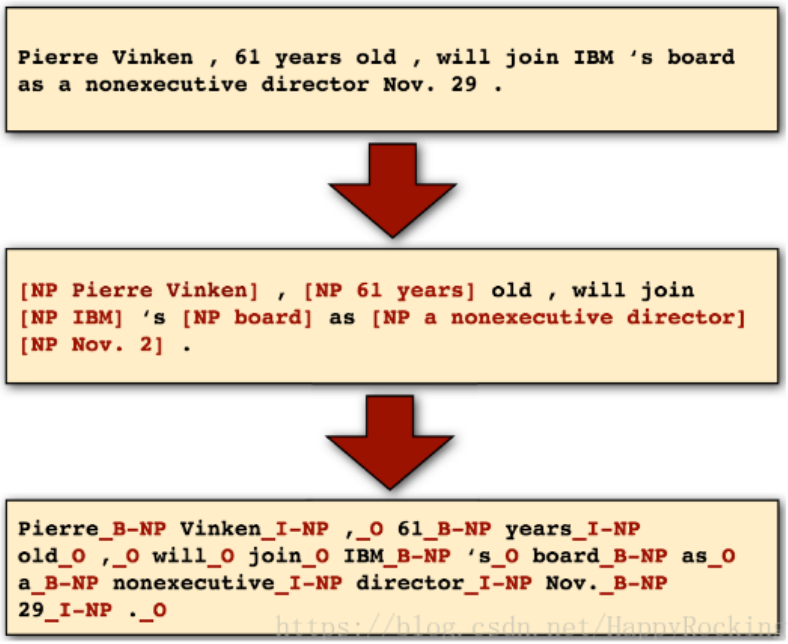
BIO 标注介绍：将每个元素标注为“B-X”、“I-X”或者“O”。其中，“B-X”表示此元素所在的片段属于 X 类型并且此元素在此片段的开头，“I-X”表示此元素所在的片段属于 X 类型并且此元素在此片段的中间位置，“O”表示不属于任何类型。比如，我们将 X 表示为名词短语（Noun Phrase，NP），则 BIO 的三个标记为：

（1）B-NP：名词短语的开头

（2）I-NP：名词短语的中间

（3）O：不是名词短语

因此可以将一段话划分为如下结果：



四、算法难点

规则制定需要大量的内部数据观察与外部信息获取。我们耗费大量时间来观察赛题所给数据的特点，从中捕捉医师书写病历的规律；同时，大量参考肠癌领域医学文献，以帮助我们更好地理解病历与属性值的关系。在规则库的建立上，针对各属性，我们分别建立了数据清洗与预处理、截取属性对应分句和提取属性值三个步骤的规则，包括知识库建立、分句合并、语义、歧义词语、错别词、单位、可能答案、否定性答案表述这些方面的规则。这样一来，避免了训练数据过少的缺点，算法的泛化性能有了一定提升。

Bi-LSTM-CRF 模型直接运用于本赛题存在一些问题。首先，本赛题需要提取 14 个属性值（后来我们增加到 20 个），数量大大超过了传统的做法，可能导致模型效果很差。其次，部分属性值的提取需要逻辑判断，而非简单的命名实体识别，如，判断切端是否累及。最后，命名实体识别以字为单位效果较好，对赛题所有数据的每个字都打上标签工作量太大。因此，我们采取了规则与深度学习结合的算法，我们对大部分的属性采用了规则直接提取值的方法，标本与浸润两属性运用了规则与深度学习结合的方法。

第二节. Web 应用系统

Web 应用系统以 Python 的 flask 为框架，sqlite 轻量型的关系型数据库架构作为存储形式来运行整个前端和后端的结构。主要实现功能为：原始数据的文件导入，处理过后的数据的展示与存储，以及数据的增删改查等基本功能，另外我们也添加了对于数据的可视化以及对于现有数据的导出功能。

一、前端框架

本 Web 应用主要以展示和修改数据为主，所以前端采用了非常轻量级的 Flask 框架，可以更加高效地运行，也方便添加新的功能与属性。

前端的页面设计主要存储"templates\records.html"中,采取了 bootstrap 模板的框架,使用已有的窗体结构 navigation 作为顶端的导航栏，左侧栏使用 button 连接到 modal 的渐隐窗口， 数据显示区使用 dataArea， 三者共同构成 index 页面的显示内容。

为了更好的实现效果，一些 JavaScript 的代码也需要产生作用，它们被存放在"static\js"文件夹中,用以生成动态的行为。

二、后台接口

为了将导入数据与语义处理系统算法相连接，我们都使用了 Python 来进行操作。我们首先将所有的文件数据都转化为 csv 格式的存档，然后后端的处理系统就依靠读取 csv 文件来进行操作，读取结束之后生成的数据再保存为 csv 格式并输出，这样就可以得到一个处理好的数据方便导入到数据库中。

三、数据库实现

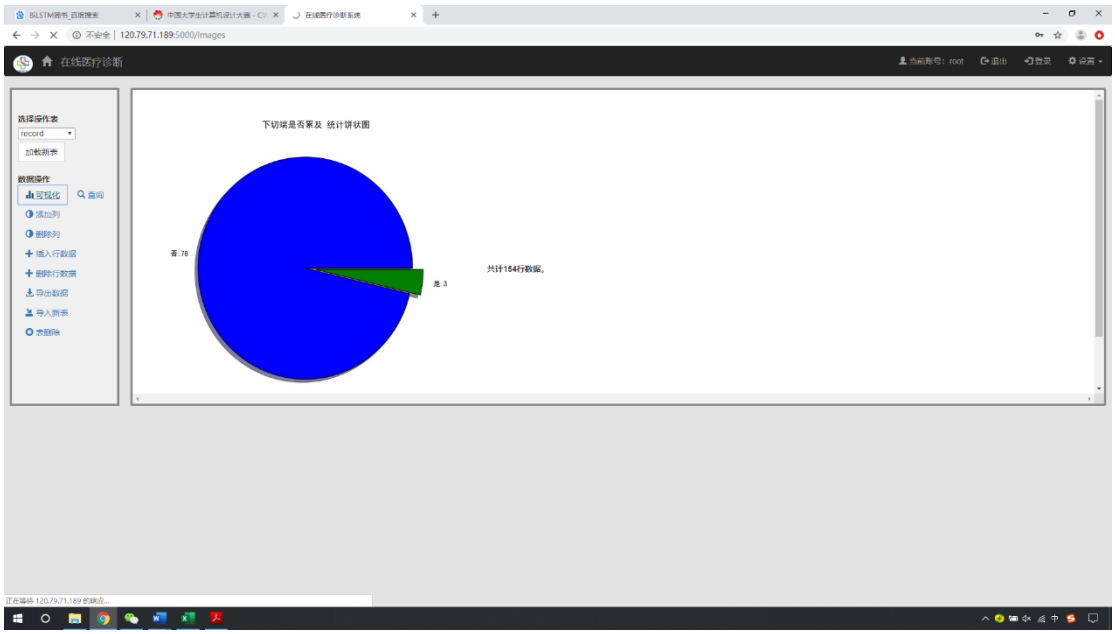
Sqlite 数据库是一个离线的本地关系型数据库，通过 python 的 sqlite3 包与其进行连接，将文件都存储在本地的"reportData.db"文件中，利用 sqlite.connect()函数进行连接。

Sqlite 可以使用基础的 sql 语句进行增删改查，比较麻烦的地方在于如何将网页前端的操作与数据库的执行语句相连接。这里我们选择使用 Flask 框架中的 session 和 request 两个交互存储变量来获取数据，并进行数据库的增删改查操作，将所有数据的修改都存储在数据库中。

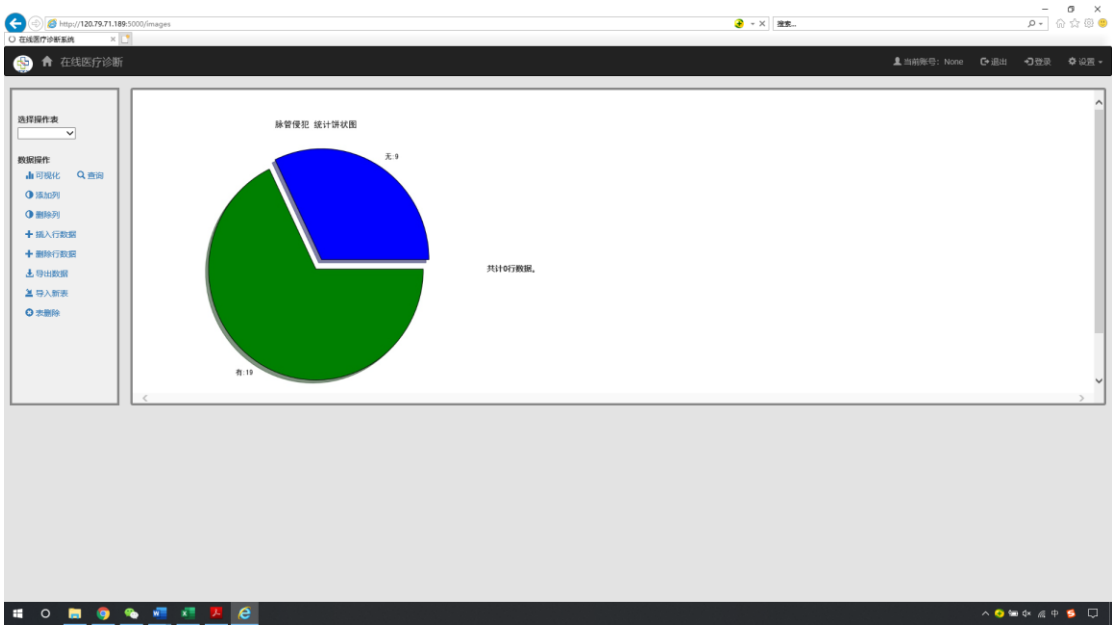
关于数据的导出，由于 sqlite 在 python 中没有直接导出的语句，我们先将用 pandas 读取 sql 数据库中的内容，再转化为 csv，使用 render_into_directory()函数发送到网页端。

四、数据可视化

这里我们采用了 python 的 matlibplot 包来绘图，主要是以饼状图的形式，展示不同属性的类型比例，对于暂时未得到的数据，不算在统计范围内。具体的数据可视化例子如下：

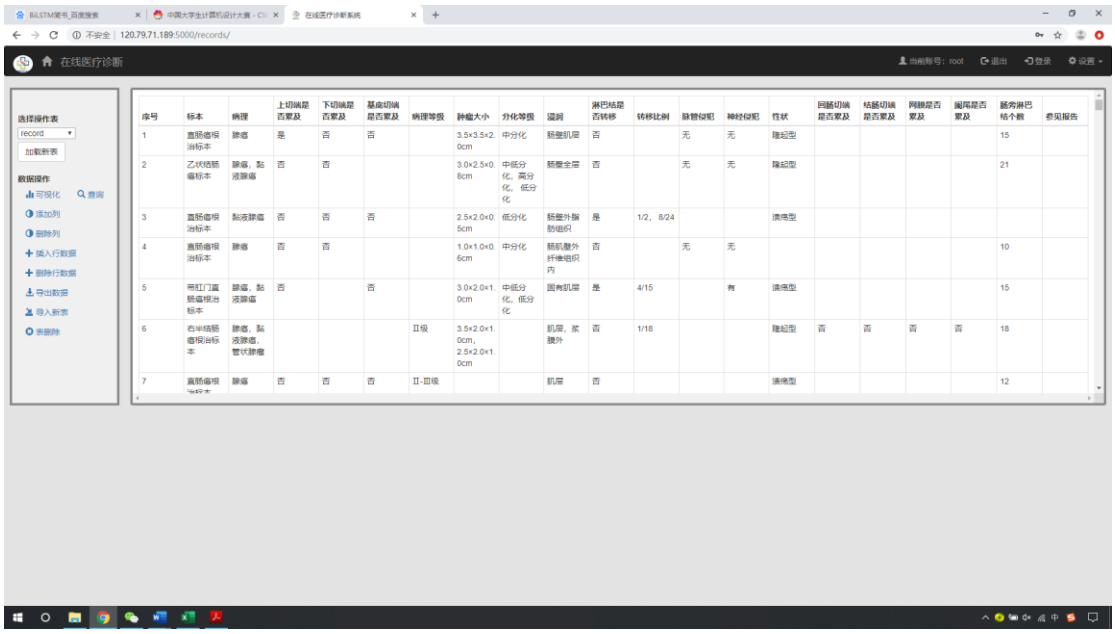


（下切端是否累及统计饼状图）



（脉管侵犯统计饼状图）

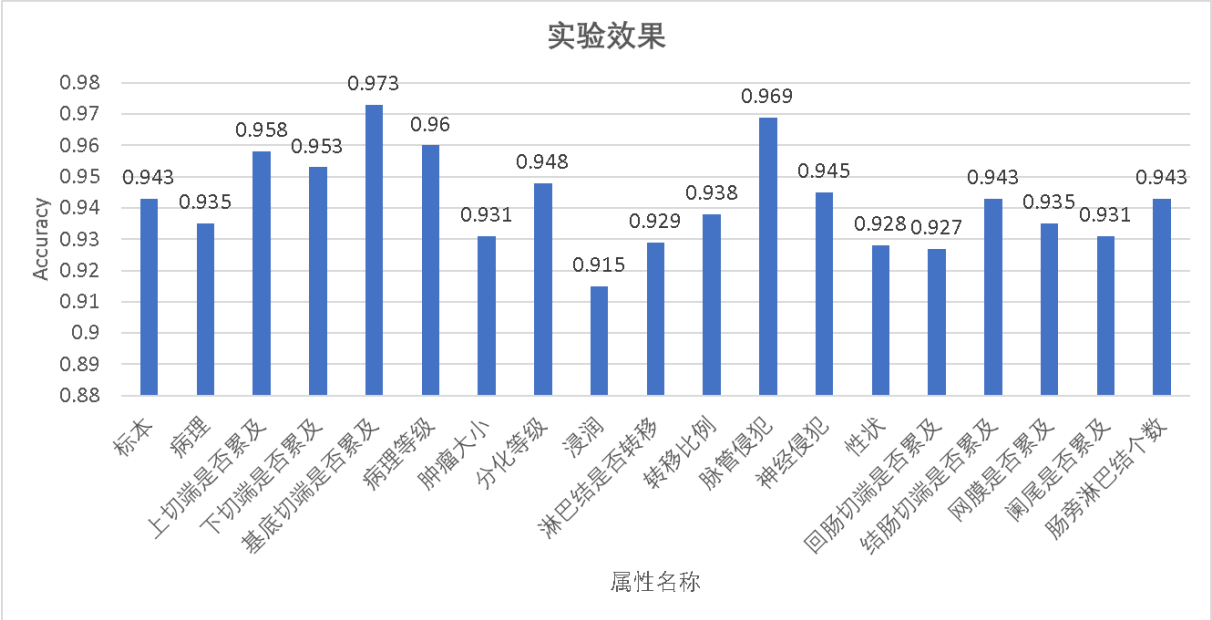
五、Web 界面预览：



第四章 测试结果

【填写说明：本章阐述作者自行进行测试的结果，描述内容包括测试过程、测试结果、修正过程等。】

通过人工标注和程序比对，我们得到了各属性的准确率。最终，平均准确率达 94.23%。



第五章 安装及使用

【填写说明：简要说明安装环境要求、安装过程、主要流程等。建议包含默认安装和典型使用流程。】

由于本项目需设计 web 应用系统，系统已上传至服务器，访问 http://120.79.71.189:5000 网站，登录时用默认的 root 用户即可正常使用。

如需在本地运行，请运行 WebDb_Linux/run.py，再进入本地网站 http://0.0.0.0:5000/（具体过程可参考演示视频）

所需环境为，Python3.6 下：tensorflow==1.8.0、jieba==0.39、Flask==0.12.2、matplotlib==2.1.2、chardet==3.0.4、pandas==0.23.4 、jieba==0.39、numpy==1.15.4 、

第六章 项目总结

【填写说明：作品制作开发过程中的一些感悟和后续升级等，如：项目协调、任务分解、面对困难、水平提升、升级演进、商业推广等诸方面。】

本项目建立了肠癌电子病历领域的信息提取算法以及相应的 web 应用系统。

在信息提取算法方面，该项目将规则与深度学习算法结合，取得了较好了效果。与使用单一规则相比，避免了规则过于生硬而泛化性能差的特点；与单一深度学习算法相比，避免了样本数据过少导致模型容易过拟合的问题。所用的 Bi-LSTM-CRF 模型比较先进，但仍可以考虑更先进的方法，如 BERT 算法、BERT-BLSTM-CRF 算法等。也可以整体全部采用算法，不使用规则，这样保证模型是端到端的，不仅能改进模型的性能，也能带来更好的开发速度和简洁性。

Web 应用系统方法，本项目采用了 flask 轻量级后端框架。如果实际应用中有并发性等进一步的要求，需要使用 django 等更高级的后端框架。

参考文献

[1]李保利,陈玉忠,俞士汶.信息抽取研究综述[J].计算机工程与应用,2003(10):1-5+66.

[2]张晓艳,王挺,陈火旺.命名实体识别研究[J].计算机科学,2005(04):44-48.

[3]杨军,郭睿,康安静,陈晓黎,苏宝山,黄小钟,靳耀锋,李宗芳.一种新的结直肠癌组织学分型、分级-评分方案[J].南方医科大学学报,2014,34(02):169-173.

[4]中华医学会外科学分会胃肠外科学组，中华医学会外科学分会结直肠外科学组，中国抗癌协会大肠癌专业委员会，et al. 中国结直肠癌肝转移诊断和综合治疗指南(2018 版)[J]. 中华消化外科杂志，2018，17(6):527-539.

[5]顾晋，汪建平，孙燕，et al. 中国结直肠癌诊疗规范（2017 年版）[J]. 中华临床医师杂志（电子版），2018，12(1): 3-23.

[6]苏韶生,杨勇,程敏婷,张淑娟.基于规则库的电子病历信息抽取研究[J].中国数字医学,2014,9(07):12-13+51.

[7]王贵齐,贺舜.ⅡC 型早期胃癌胃镜下诊断[J].内科理论与实践,2010,5(03):195-199.

[8] Zhiheng Huang, Wei Xu, Kai Yu. Bidirectional LSTM-CRF Models for Sequence

Tagging. ArXiv.org, 2015(08).
[9] Xuezhe Ma, Eduard Hovy. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. ArXiv.org, 2016(03).
[10] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, Chris Dyer. Neural Architectures for Named Entity Recognition. ArXiv.org, 2016(03).
[11] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, Philip H. S. Torr. Conditional Random Fields as Recurrent Neural Networks. IEEE ICCV, 2015(02).
[12] 中国中文信息学会。中文信息处理发展报告(2016)[R].北京.2016(12).

附录

变量解释

splits 表示分句列表 list

records 表示提取结果

标本属性

```
records=[]
records.append(splits[0])
for j in splits[1:]:
    if '标本”' in j:
        在 j 中匹配正则表达式”.*标本”
        if 匹配成功:
            specimen.append(匹配结果)
for i in range(records):
    if '标本' in records [i]:
        取'标本'前面的字
    else:
        寻找所有符合正则表达式 (?(.*?)) 的结果
        if 匹配成功:
            取匹配结果
    if records [i][:-2]!= '标本':
         records [i]= records [i]+ '标本'
return records
```

病理属性

```
# pathologies 表示已知病理类型
records=[]
for j in splits[1:]:
    for p in pathologies:
        if p in j :
            records.append(p)
return records
```

上切端是否累及属性

```
# 合并上切端属性和它的描述值
for i in range(len(splits)):
    if 遍历到最后一项:
        break
    if 上切端及其形近词 in splits[i] and '距' not in splits[i]:
        if 累及及其形近词 in splits[i]:
            continue
        for j in 当前分句的之后所有分句:
            if 累及及其形近词 in j:
                合并两个分句;
                跳出两层循环;
records = []
for j in splits[1:]:
    if 上切端及其形近词 in j and '距' not in j:
        if 累及及其形近词 in splits[i]:
            records.append(j)
result=[]
for i in records:
    if ('均未见') in i or('未见' in i):
        result.append(False)
    elif ('均见' in i) or('见' in i):
        result.append(True)
if result 不为空:
    if True in result:
        return '是'
    return '否'
return
```

下切端是否累及属性

```
# 与上切端是否累及属性的处理方式相似
```

基底切端是否累及属性

```
# 与上切端是否累及属性的处理方式相似
```

回肠切端是否累及属性

```
# 与上切端是否累及属性的处理方式相似
```

结肠切端是否累及属性

```
# 与上切端是否累及属性的处理方式相似
```

网膜切端是否累及属性

```
# 与上切端是否累及属性的处理方式相似
```

阑尾切端是否累及属性

```
# 与上切端是否累及属性的处理方式相似
```

病理等级属性

```
records=[]
```

```
for j in splits[1:]:
    if 匹配到正则表达式(Ⅰ|Ⅱ|Ⅲ|Ⅳ|II)([^\u4e00-\u9fa5]*?)级:
        records.append(匹配结果)
return records
```

肿瘤大小属性

```
records = []
for j in splits[1:]:
    if ('大小' in j) or ('直径' in j):
        records.append(j)
result = []
for i in range(len(records)):
    在 records [i]中匹配所有的正则表达式\d[^\u4e00-\u9fa5]*cm;
    在匹配结果中只保留 0123456789.cm×x-*;
    if '×'或'*'或'x' not in records[i] :
        records[i]='直径'+records[i]
return records
```

分化等级属性

```
# differentiations 表示已知分化等级类型
records = []
for j in splits[1:]:
    for d in differentiations:
        if d in j:
            records.append(d)
return records
```

浸润属性

```
records = []
for j in splits[1:]:
    if 浸润性 or 浸润型 or 浸润至型 or 细胞浸润 or 浸润组织 or 浸润深度 in j:
        continue
    if '浸润' in j:
        records.append(j)
for i in range(len(records)):
    str= records [i];
    删除无关字符;
    if '伴' in str:
        删除伴后面的字符;
    if 匹配到正则表达式均?可?见.*浸润:
        取匹配结果前的字;
    elif 匹配到浸润至:
        取匹配结果前的字;
    elif str[:4]=='肿瘤侵犯':
        continue:
        else:
            对 str 进行分词
```

```
        if 分词的第一个词是浸润
            取后面所有词;
        elif str 开头为局部浸润 or 肿瘤浸润 or 局灶浸润 or 癌组织浸润:
            取后面所有词;
        elif 匹配到正则表达式局限于.*内 or 受.*浸润 or(.*):
            取中间的词;
```

淋巴结是否转移属性

```
# 合并淋巴结属性和它的描述值
while flag:
    for i in range(len(splits)):
        if 遍历到最后:
            while 循环;
        if ('淋巴结' in splits[i] ) and ('转移' and 肿瘤' and '癌组织' not in splits[i]):
            if  ('转移' in splits[i+1]) or ('肿瘤' in splits[i+1]) or ('癌组织' in splits[i+1]):
                合并前后两个分句;
                break
records = []
for j in splits[1:]:
    if ('淋巴结' in j) and ('见' in j):
        records.append(j)
bool_list=[]
for i in records:
    if ('均未见' in i) or ('未见' in i):
        bool_list.append(False)
    elif '见' in i:
        bool_list.append(True)
if True in bool_list:
    return '是'
return '否'
```

转移比例属性

```
records = []
for j in splits[1:]:
    if '浸润深度' in j:
        continue
    if 匹配到正则表达式\d{1,2}\/\d{1,2}:
        records+=匹配结果
return records
```

脉管侵犯属性

```
records = []
for j in splits[1:]:
    if '脉管' in j:
        records.append(j)
```

```
result=[]
for i in range(len(records)):
    if '未见' in records [i]:
        result.append(False)
    elif '见'or '+' or '癌栓'or'瘤栓'or '存在'or '侵犯'or'侵及'in records [i]:
        result.append(True)
if result 为空:
    return
elif True in result:
    return '有'
elif False in result:
    return '无'
else:
    return result
```

神经侵犯属性

```
records = []
for j in splits[1:]:
    if '神经' in j:
        records.append(j)
result=[]
result=[]
for i in range(len(records)):
    if '未见' in records [i]:
        result.append(False)
    elif '见'or '+' or '累及'or'包绕'or '存在'or '侵犯'or'侵及'in records [i]:
        result.append(True)
if result 为空:
    return
elif True in result:
    return '有'
elif False in result:
    return '无'
else:
    return result
```

性状属性

```
# charalist 表示已知性状类型
records=[]
for j in splits[1:]:
    for c in charalist:
        if c in j:
            records.append(c)
    if 匹配失败:
        匹配正则表达式.{1,6}型或（.{1,6}型.*?）？
```

```
return records
```

肠旁淋巴结个数属性

```
records=[]
# 匹配 肠旁淋巴结
for j in splits[1:]:
    if 匹配到正则表达式肠旁淋巴结(.*\d)枚? :
        records.append(匹配结果)
for i in records:
    标准化处理;
    提取数字;
```

参见报告属性

```
records=[]
for j  in splits:
    if '报告' in j:
        records.append(j)
for i in records:
    标准化处理;
    多份报告拆分;
    异常字符删除;
```
