

# 浙江工业大学



## 《文本分析与挖掘》

**2023/2024(1)**

### 期末综合实验报告

报告题目：基于 LSTM 的命名实体识别

学生班级：大数据分析 2101

学生姓名：温家伟

学生学号：202103151422

### 摘要

**命名实体识别** (Named Entity Recognition, 简称 NER) 是自然语言处理领域中的一项关键任务, 旨在从文本中识别和分类出具有特定意义的实体名称, 如人名、地名、组织机构名等。NER 技术可以帮助机器理解文本中的重要信息, 并提供对这些信息的结构化表示。

**智能问答**是一种通过自动化的方法从大量文本中提取问题和答案, 并且, 以简洁、易读的形式呈现给用户的技术。它结合了自然语言处理、信息检索和知识图谱等技术, 可以帮助人们快速地获取所需信息, 提高信息获取的效率。

**LSTM+CRF** 是一种常用的深度学习模型, 用于命名实体识别。该模型结合了长短期记忆网络和条件随机场两种技术, 能够有效地捕捉句子中的上下文信息以及实体之间的关系。LSTM 是一种递归神经网络的变体, 通过引入门控机制来解决传统 RNN 在处理长序列时的梯度消失和梯度爆炸问题。LSTM 能够对输入序列进行建模, 并利用记忆单元来保留和更新历史信息, 从而更好地理解句子的语义。CRF 是一种序列标注方法, 通过对整个序列进行联合建模, 可以考虑相邻标签之间的依赖关系。在 NER 任务中, CRF 模型可以将 LSTM 输出的特征序列作为输入, 通过定义转移概率来对标签序列进行建模, 从而得到最优的标签序列。

在阿里天池的测试平台上, 我提交了任务的对应文件, 如图 1 所示, 发现平台打分 3.236, 其中 CMeEE-V2-F1-score 可以达到 60 以上, 取得了还算不错的效果。

接着, 我根据命名实体识别得到的实体名词为关键词, 用 python 爬取了百度百科的相关数据, 以此作为这个医疗智能问答系统的知识库, 在 unit 平台上做一系列设置后, 训练并部署模型, 调用这个 unit 的 api 接口, 自己打包了动态库, 实现了智能问答系统中问答部分的 api 调用。

在系统的前端部分, 为了实现用户与系统之间的交互, 我使用 **httpplib**。该网络库用于处理 HTTP 请求和响应, 使我能够与后端服务器进行通信。同时, 我还利用了常用的 Web 开发技术, 如 jQuery、HTML 和 CSS, 来构建一个简单而直观的前端界面。通过这些技术, 我可以创建用户友好的输入框、按钮和其他交互元素, 使用户能够方便地输入医疗相关问题。当用户输入医疗相关的问题后, 系统会将该问题发送给后端进行处理。后端调用百度 unit 的 api 接口, 对问题进行分析和理解, 并生成相应的回答。

**关键词:** 命名实体识别   LSTM   CRF   智能问答   unit   httpplib

## 目录

<b>1</b>	<b>问题分析</b>	<b>3</b>
1.1	背景描述 . . . . .	3
1.2	需求分析以及存在的挑战 . . . . .	3
1.2.1	需求分析 . . . . .	3
1.2.2	存在的挑战 . . . . .	4
1.3	已有相关方法和技术调研 . . . . .	4
1.4	挖掘目标与任务 . . . . .	5
1.4.1	任务描述 . . . . .	5
1.4.2	数据的格式 . . . . .	6
1.4.3	评价准则 . . . . .	7
1.4.4	数据集详情 . . . . .	7
1.4.5	天池平台提交结果 . . . . .	8
<b>2</b>	<b>系统构架</b>	<b>8</b>
2.1	总体解决方案 . . . . .	8
2.2	核心算法 . . . . .	8
2.2.1	Bi-directional LSTM . . . . .	9
2.2.2	条件随机场 (CRF) 作为标记模型 . . . . .	9
2.2.3	基于部首的 LSTM 模型 . . . . .	11
2.2.4	LSTM 的变体 . . . . .	11
2.3	系统应用 . . . . .	12
2.3.1	爬取百度百科数据 . . . . .	12
2.3.2	调用百度 api 构建问答系统 . . . . .	14
2.3.3	搭建前端界面 . . . . .	15
<b>3</b>	<b>功能展示</b>	<b>19</b>
3.1	命名实体识别 . . . . .	19
3.2	前端知识库智能问答 . . . . .	19
<b>4</b>	<b>总结</b>	<b>20</b>

# 1 问题分析

## 1.1 背景描述

近年来，多项下游任务随着预训练语言模型和大模型技术的不断发展已取得了最先进的结果。其中，一个重要趋势是出现了多任务评测基准，这些基准旨在评估语言模型的泛化性能，并提供公平开放的评测基准，吸引了众多 NLP 研究者的关注，进一步推动了大模型技术的应用和发展。

中文 NER 与分词相关。命名实体边界也是字词边界。执行实现中文 NER 的一种直观方式，是在应用单词序列注释之前执行分词，即先对输入进行分词，然后再应用词序列进行标注。

医学领域的自然语言文献包含了大量的医学专业知识和术语。将实体识别技术与医学领域结合起来，利用机器读取医学文本，可以显著提高临床科研的效率和质量，并为下游子任务提供服务。让计算机能够准确地从大量医学文本中提取关键信息是实现机器“读懂”医学数据的关键，而命名实体识别和关系抽取等自然语言处理技术则涉及到实现这一目标的必要工具。医学领域的非结构化文本由中文自然语言句子或句子集合组成。实体抽取是从非结构化的医学文本中找出医学实体（如疾病、症状）的过程。

医学命名实体识别任务是对电子病历中的医学实体进行自动识别和分类，对于下游任务例如信息检索、知识图谱等有着十分重要的作用。往往需要通过海量高质量标注数据来帮助实体识别模型的训练，但是大规模标注数据的获取存在诸多困难。因此，越来越多的科研人员开始关注如何通过文本信息自身的相关性来提高实体识别的准确性。

## 1.2 需求分析以及存在的挑战

### 1.2.1 需求分析

- **功能需求 1：**构建一个实体识别系统，能够从医学文本中准确地提取出医学实体，如疾病、症状等。
- **功能需求 2：**实现关系抽取功能，能够识别医学文本中的实体之间的关系，如疾病与症状之间的关系。
- **非功能需求：**确保系统能够处理大规模的医学文本数据，并提供稳定可靠的结果。

### 1.2.2 存在的挑战

- **技术挑战：**医学领域的文本包含大量的专业知识和术语，需要进行领域特定的自然语言处理和机器学习算法的开发，以提高实体识别和关系抽取的准确性。
- **资源限制：**医学文本数据规模庞大，需要充足的计算资源来处理这些数据和训练模型。
- **时间限制：**后端模型提取命名实体的速度不能过慢，否则会影响到前端系统的交互。

### 1.3 已有相关方法和技术调研

近年来，作为信息提取的一项基本任务，命名实体识别 NER 一直受到研究人员的关注<sup>1</sup>。该任务本质是序列标注问题，其中实体的边界和类别标签作为两个维度进行联合预测。目前，英文 NER 的最高水准是使用长短期记忆网络—条件随机场 LSTM-CRF 模型实现的<sup>2</sup>，其中字符信息被整合到词表征中。

中文 NER 与分词相关。命名实体边界也是字词边界。执行实现中文 NER 的一种直观方式，是在应用单词序列注释之前执行分词，即先对输入进行分词，然后再应用词序列进行标注<sup>3</sup>。然而，在分割 NER 的流程中，由于 NE 是未登录词（out-of-vocabulary, OOV）的主要来源，且分割错误的实体边界会导致整个 NER 错误，因此存在误差传播的潜在问题。这个问题在开放领域将可能非常严重。因为跨领域分词仍然是一个未解决的难题<sup>4</sup>。

有研究表明，在中文 NER 中，基于字符的表现方法要显著优于基于词汇的表示方法。

在本研究中，利用点阵长短期记忆网络 LSTM 来表征句子中的单词，然后将潜在的单词信息整合到基于字符的 LSTM-CRF 中。由于在网格中存在

---

<sup>1</sup>Ma X, Hovy E. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF[J].arXiv preprint arXiv, 2016.

<sup>2</sup>Liu L, Shang J, Xu F F, et al. Empower Sequence Labeling with Task-Aware Neural Language Model[J]. Thirty-Second AAAI Conference on Artificial Intelligence, 2017.

<sup>3</sup>HUANG Z, XU W, YU K. Bidirectional LSTM-CRF models for sequence tagging[J]. arXiv: 1508.01991v1.

<sup>4</sup>Yajie Miao, Mohammad Gowayyed, Florian Metze. EESSEN: End-to-End Speech Recognition using Deep RNN Models and WFST-based Decoding[J]. 2015 IEEE Workshop on. IEEE, 2015: 167-174.

指数数量的单词到字符路径，从而使用 LSTM 结构来自动控制从句子开头到结尾的信息流。在对预先准备的训练数据进行训练之后，我发现 LSTM 可以从上下文中自动学习，找到更多有用的单词来提升 NER 的性能和准确度。基于字符与基于词的 NER 方法相比，本文提出的模型具有使用显式部首信息而不是字符序列标注的优点，并且未发生明显的分词错误。

## 1.4 挖掘目标与任务

CMeEE-V2 相较于其原始版本 CMeEE，主要的更新在于修正了原始数据中的一些标注错误，从而提高了语料的质量。此外，实体的下标位置也从原来的左闭右闭方式调整为了左闭右开的标记方式。

### 1.4.1 任务描述

这个任务是针对中文医学文本的命名实体识别，也就是说，给定一个 schema 和一个句子 sentence，任务的目标是从一组纯医学文本文档中识别和提取出与医学临床相关的实体，并将这些实体分类到预定义的类别中。医学文本命名实体被分为九大类，包括：

表 1: 九类医疗实体类型

疾病	(dis)
临床表现	(sym)
药物	(dru)
医疗设备	(equ)
医疗程序	(pro)
身体	(bod)
医学检验项目	(ite)
微生物类	(mic)
科室	(dep)

对于命名实体的标注，我们遵循以下基本规则：

- 在“临床表现”实体类别中，我们允许实体的嵌套，即这类实体内部可以包含其他八类实体；

- 对于除“临床表现”实体以外的其他医学实体，我们采用“最大单位标注法”，也就是说，如果一个实体内部包含其他实体，我们只标注最大的实体，而不进行嵌套标注。
- 为了确保医学实体的意义清晰和完整，我们允许九类实体内部包含必要的标点符号，实体可以是一个词、短语或句子。

#### 1.4.2 数据的格式

我们提供的输入数据是 json 格式，包含两个字段，如下所示：

- "text": 这是需要识别的医学文本句子。
- "entities": 这是已经标注的医学实体列表，每个实体都包含实体的起始位置、结束位置和实体的类型。

```
{  
  "text": "(3) 非持续性，不考虑为无休止性的阵发性室性心动过速（即一次监视数小时或任何一小时记录的心电图条带几乎均可出现），心室功能正常。",  
  "entities": [  
    {  
      "start_idx": 3,  
      "end_idx": 26,  
      "type": "dis",  
      "entity": "非持续性，不考虑为无休止性的阵发性室性心动过速"  
    },  
    {  
      "start_idx": 44,  
      "end_idx": 47,  
      "type": "pro",  
      "entity": "心电图"  
    },  
    {
```

```
        "start_idx": 57,
        "end_idx": 59,
        "type": "bod",
        "entity": "心室"
    }
]
}
```

### 1.4.3 评价准则

我将使用严格的 Micro-F1 作为主要的评价标准，这意味着只有当预测的实体的起始和结束下标以及实体类型都完全匹配时，才会被视为预测正确。

### 1.4.4 数据集详情

这个评测任务提供了训练集、验证集和测试集。

- 训练集：包含 15,000 条数据
- 验证集：包含 5,000 条数据
- 测试集：包含 3,000 条数据

数据集的名称是:CMEE-V2 (Chinese Medical Entity Extraction dataset)

您可以从名为 CMEE-V2.zip 的压缩文件中下载数据集，该文件包含以下文件：

- CMEE-V2\_train.json: 训练集数据
- CMEE-V2\_dev.json: 验证集数据
- CMEE-V2\_test.json: 测试集数据
- example\_pred.json: 提交结果的示例

在提交测试集的预测结果时，您需要为每条记录填充 "entities" 字段，这是一个列表。每个识别出的实体都必须包含 "start\_idx", "end\_idx", "type" 三个字段。



### 1.4.5 天池平台提交结果

如图 1 所示,我最终的分数为 3.236, CMeEE-V2-F1 为 61.4886, 其中 CMeEE-V2-P 为 62.1209, CMeEE-V2-R 为 60.8690。

序号	提交时间	模型	状态	Score <small>↓</small>	CMeEE-V2-F1 <small>↓</small>	CMeEE-V2-P	CMeEE-V2-R	CMeEE-V2-F1 <small>↓</small>	操作
1	2023-12-31 12:39:44	阿伟的神奇模型4.0	已完成	3.236	61.4886	62.1209	60.8690	0.0	编辑

图 1: 平台提交分数图

## 2 系统构架

### 2.1 总体解决方案

本次实验的总体系统架构图如下

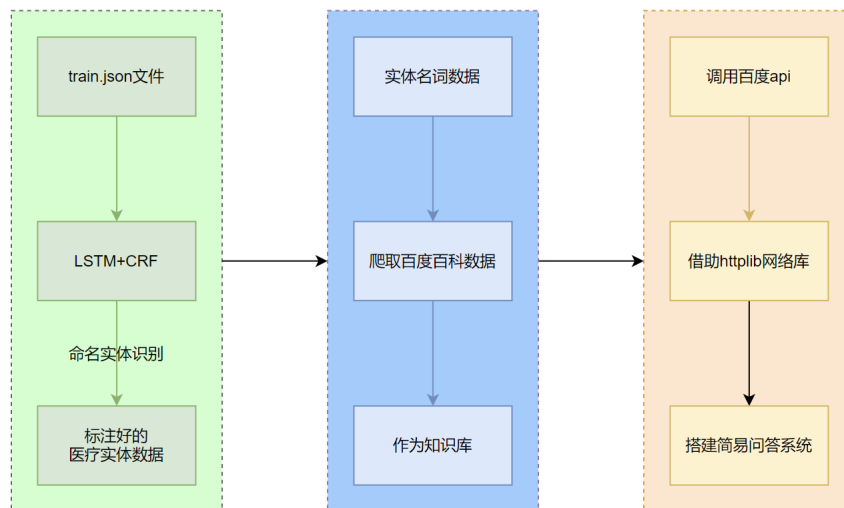


图 2: 系统总体架构图

### 2.2 核心算法

我们将进行 LSTM 和 CRF 的简要描述,并介绍了一种混合标记架构。该架构与 Collobert 等人 (2011) 和 Huang 等人 (2015) 提出的架构类似。

### 2.2.1 Bi-directional LSTM

递归神经网络 (RNN) 是一种专门处理序列数据的神经网络类型。它接收一个向量序列  $(x_1, x_2, \dots, x_n)$  作为输入, 并输出一个向量序列  $(h_1, h_2, \dots, h_n)$ , 该向量序列表示了输入序列每个步骤的信息。虽然理论上 RNN 能够学习长期依赖关系, 但实际上, 它们往往无法实现这一点, 而且更倾向于关注最近的输入 (Bengio 等人, 1994)。长短期记忆网络 (LSTM) 通过引入记忆单元来解决这个问题, 并已经证明能够捕获长期依赖关系。它们通过几个门来控制输入到记忆单元的分配, 以及从先前状态中遗忘的信息的比例 (Hochreiter 和 Schmidhuber, 1997)。我们采用以下实现:

$$\begin{aligned}
 i_t &= \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) && \text{(input gate)} \\
 f_t &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) && \text{(forget gate)} \\
 c_t &= (1 - i_t) \odot c_{t-1} + i_t \odot \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) && \text{(cell state)} \\
 o_t &= \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) && \text{(output gate)} \\
 h_t &= o_t \odot \tanh(c_t) && \text{(output)}
 \end{aligned}$$

其中,  $\sigma$  是逐元素的 Sigmoid 函数,  $\odot$  表示逐元素的乘法。

对于一个包含  $n$  个词的句子  $(x_1, x_2, \dots, x_n)$ , 其中每个词都是一个  $d$  维向量, LSTM 计算了左上下文的表示  $\vec{h}_t$ 。同样, 生成右上下文的表示  $\overleftarrow{h}_t$  也应该提供有价值的信息。这可以通过使用一个反向读取相同序列的第二个 LSTM 来实现。我们将前者称为前向 LSTM, 后者称为后向 LSTM。这种前向和后向 LSTM 的组合被称为双向 LSTM (Graves 和 Schmidhuber, 2005)。

这种模型的单词表示是通过连接其左右上下文表示得到的, 即  $h_t = [\vec{h}_t; \overleftarrow{h}_t]$ 。这些表示实际上包含了上下文中的单词表示, 这对于许多标记任务来说是非常有用的。

### 2.2.2 条件随机场 (CRF) 作为标记模型

采用  $h_t$  作为特征, 对每个输出  $y_t$  进行独立的标记决策是一种简单且有效的标记模型 (参见 Ling 等人, 2015b 的工作)。尽管这种模型在处理如词性标注等简单问题时已经取得了成功, 但在输出标签之间存在强依赖关系的情况下, 其独立分类决策的能力就会受到限制。

命名实体识别（NER）就是一个例子，因为表征可解释的标记序列的“语法”会施加一些硬约束，这些约束是无法通过独立假设来建模的。

例如，当上下文中出现关键提示词如：「检查」、「提示」、「查体」、「试验」、「测定」、「听诊」、「观察」、「诊断」、「监护」等，其对应的检查项目应标注为医学检验项目 **ite**。

然而，如果标注实体后面紧跟着「检查」、「测定」、「诊断」等词，这表明是医生为了诊断或治疗而采取的一系列操作或过程，因此，应将医学检验项目和这些词语作为一个整体，标注为医疗程序 **pro**。

因此，我们选择使用条件随机场（CRF）来联合建模标记决策（Lafferty 等人，2001）。对于一个输入句子

$$X = (x_1, x_2, \dots, x_n) \quad (1)$$

我们将  $P$  视为双向 LSTM 网络输出的得分矩阵。 $P$  的大小为  $n \times k$ ，其中  $k$  是不同标记的数量， $P_{i,j}$  对应于第  $i$  个单词的第  $j$  个标记的得分。对于一个预测的标签序列

$$y = (y_1, y_2, \dots, y_n) \quad (2)$$

我们定义其得分函数为

$$s(X, y) = \sum_{i=0}^n (A_{y_i, y_{i+1}} + P_{i, y_i}) \quad (3)$$

这里， $A_{y_i, y_{i+1}}$  代表第  $i$  个词和第  $i+1$  个词之间的转移得分， $P_{i, y_i}$  代表第  $i$  个词的第  $y_i$  个标记的得分。

$A$  是一个转移得分矩阵， $A_{i,j}$  代表从标签  $i$  到标签  $j$  的转移得分。 $y_0$  和  $y_n$  是句子的起始和结束标签，我们将其添加到可能的标签集合中。因此， $A$  是一个大小为  $k+2$  的方阵。对所有可能的标签序列（包括不符合 IOB 格式的序列）进行 softmax 运算，得到序列  $y$  的概率：

$$p(y|X) = \frac{e^{s(X, y)}}{\sum_{\tilde{y} \in Y_X} e^{s(X, \tilde{y})}} \quad (4)$$

在训练过程中，我们的目标是最大化正确标签序列的对数概率：

$$\log(p(y|X)) = s(X, y) - \log \left( \sum_{\tilde{y} \in Y_X} e^{s(X, \tilde{y})} \right) \quad (5)$$

$$= s(X, y) - \text{logadd}_{\tilde{y} \in Y_X} s(X, \tilde{y}) \quad (6)$$

这里， $Y_X$  表示句子  $X$  的所有可能标签序列（包括不符合 IOB 格式的序列）。从上述公式可以看出，我们鼓励网络生成一个有效的输出标签序列。在解码过程中，我们预测得分最大的输出序列：

$$y^* = \arg \max_{\tilde{y} \in Y_X} s(X, \tilde{y}) \quad (7)$$

由于我们只对输出之间的 bigram 交互进行建模，因此公式 6 中的求和以及公式 7 中的最大后验概率序列  $y^*$  可以使用动态规划来计算。

### 2.2.3 基于部首的 LSTM 模型

汉字由更小的部首构成，这些部首是构建汉字含义的基本元素。这些部首是汉字的内在特征，为我们提供了富有语义的额外信息。

例如，汉字「病」、「症」和「痛」都与疾病有关，因为它们都包含部首「疒」。这种部首的语义信息对于在向量空间中具有相似部首的汉字聚集在一起非常有帮助。这激发了我们对汉字部首的关注。

在现代汉语中，一个汉字通常由多个部首组成。在 MSRA 数据集中，包括训练集和测试集，有 75.6% 的汉字包含多个部首。我们从在线的新华字典中获取汉字的部首信息。在简化汉字中，汉字的部首可能已经改变了其原始形状。例如，汉字「腿」的第一个部首是「月」，它是传统部首「肉」的简化形式，而「朝」的部首也是「月」，表示月亮。这可能会引起混淆。

为了处理这些变体，我们用其传统形式的部首来替换最重要的简化部首，以恢复其原始含义。汉字的简化部首和传统部首都可以在在线的新华字典中找到。对于只有一个部首的汉字，我们只使用它本身作为其部首部分。在进行这种替换之后，我们获取所有组成部首，以构建每个汉字的部首列表。由于一个汉字的每个部首在其内部具有独特的位置，我们将一个汉字的部首视为按照书写顺序的序列。我们使用部首级别的双向 LSTM 来捕捉部首信息。

### 2.2.4 LSTM 的变体

我们在医疗数据集上比较了不同的 LSTM 变体，以找到更好的 LSTM 变体。最初版本的 LSTM 单元包括 cell、input 和 output gate 来解决梯度消失/爆炸问题。因此，在大多数变体中，我们保留了输入和输出门。以下是我们提到的 LSTM 的派生版本：

1. 无 peepholes, 无 forget gate, 只耦合输入门 (NP, NFG, CIG)
2. peepholes, 无 forget gate, 只耦合输入门 (P, NFG, CIG)
3. 无 peepholes, forget gate, 只耦合 forget gate (NP, FG, CFG)
4. 无 peepholes, forget gate, 耦合输入和 forget gate (NP, FG, CIFG)
5. 无 peepholes, forget gate(1), 耦合输入和 forget gate (NP, FG(1), CIFG)
6. 门控循环单元 (GRU)

如果我们使用 CIG 来更新 cell state, 则输入和 cell state 只有一个门, 因此 forget gate 将被省略。这等效于设置  $f_t = 1$  而不是单独使用 forget gate。GRU 是 LSTM 的一种变体, 没有单独的存储单元, 并且每次都会暴露整个状态。其中第五条的含义是 forget gate 的偏置初始化为 1 而不是 0。我们根据测试结果最后选择了第一条。

## 2.3 系统应用

### 2.3.1 爬取百度百科数据

在命名实体识别后, 我们得到了带标签的许多实体名词。我们首先用这些实体名词作为关键词来爬取百度百科的数据, 然后再进行数据清洗, 作为下一步知识库的输入。具体流程如下图所示:

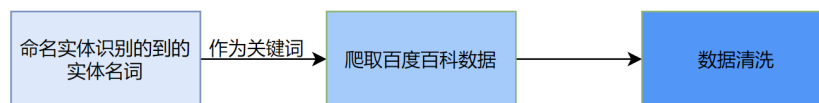


图 3: 爬取百度百科数据

本实验中所用到的 requests 和 urllib.parse 库是 Python 中常用的爬虫技术。requests 是一个简单易用的 HTTP 库, 它允许我发送 HTTP/1.1 请求, 同时支持自动解码响应内容。而 urllib.parse 库则提供了 URL 处理方法, 包括实现 URL 编码、解码等。

在爬虫过程中，我首先需要获取网页内容，这里使用 requests 库的 get 方法来发送 HTTP 请求，并以关键词为参数构建 URL 链接。在发送请求时，我还需要设置 headers 信息，以模拟浏览器的请求。在本实验中，我设置了 User-Agent，模拟了使用 Chrome 浏览器发起的请求。

获取网页内容后，我还需要对其进行解析和处理。这时候，我可以使用 BeautifulSoup 库，通过解析 HTML 文档，提取出需要的信息。不过，在本实验中，我并没有使用 BeautifulSoup 库，而是直接将结果保存到本地文件中。

在使用 requests 库发送 HTTP 请求时，我需要将关键词进行 URL 编码，以避免出现链接不完整或乱码等问题。在 Python 中，可以使用 urllib.parse 库的 quote 方法来对字符串进行 URL 编码。

在本实验中，我对关键词进行了 URL 编码，并在构建 URL 链接时将其作为参数传递给 requests 库的 get 方法。这样就可以保证构建的链接正确，同时也能够避免因为关键词中包含特殊字符而导致的请求失败。

在爬虫过程中，我通常需要将获取到的数据进行处理和存储。在本实验中，我将获取到的网页内容保存到本地文件中。这里使用了 Python 内置的 open 函数，以及 with 语句，来打开文件并写入内容。

在打开文件时，我需要指定文件路径、文件名以及操作模式。其中，文件路径和文件名组成完整的文件路径，操作模式则指定文件的读写方式。在本实验中，我使用了 'w' 模式，表示以写入模式打开文件。这样，在文件不存在时，会自动创建新的文件；在文件已存在时，会清空文件内容，并写入新的数据。

在进行爬虫开发时，我需要遵守相关法律法规，尊重网站所有者的权益，避免恶意爬虫和数据侵犯等不良行为。同时，在发起 HTTP 请求时，我还需要注意一些网络安全方面的问题。

比如，在本实验中，我在发送 HTTP 请求时，设置了 User-Agent 信息，模拟了使用 Chrome 浏览器发起的请求。这样可以避免因为请求头信息不完整或错误而被服务器拒绝请求。另外，如果需要进行登录认证、验证码识别等操作，我还需要使用更加复杂的爬虫技术，比如使用 selenium 库来模拟用户交互操作等。

### 2.3.2 调用百度 api 构建问答系统

对话系统是计算机试图通过文本或语音交互的方式来模拟人类对话的计算机程序,通过这个程序能完成人类与机器的对话。对话系统是人机交互技术最核心的领域之一,它是人与机器之间进行双向信息交换以满足人的特定任务需求的计算机软硬件系统<sup>5</sup>。

对话系统的研究历史悠久,最早可以追溯到 1950 年图灵提出的图灵测试。图灵测试是以人机对话的形式进行的,是最早的人机对话模式。1964 年,ELIZA 对话系统诞生<sup>6</sup>,它可以分析输入语句,并根据其分析创建回复。20 世纪 90 年代,微软推出了 Office 的办公助手 Clippy,被认为是最早推向商用的人机对话系统<sup>7</sup>。进入 21 世纪,理查德·华勒斯利用启发式的会话规则并嵌入 AIML 创造了 ALICE<sup>[6-7]</sup>,对话质量获得大幅提高。2008 年,奇迹工场推出 Cleverbot,它与其他对话系统不同,它的回复是直接从人类的输入中学习<sup>[8]</sup>。2014 年,对话系统得到了进一步发展,涌现出以微软小冰为代表的聊天机器人和百度 UNIT 这种智能对话系统开发平台。目前,微软、谷歌和苹果等科技公司开始研制对话系统,这使得对话系统迅速成为 21 世纪科技发展的重点。

百度 UNIT 是百度推出的理解与交互技术,用户能通过 UNIT 平台搭建满足自己需求的对话机器人。UNIT 平台能够提供丰富的可定制化服务,可进行对话定制、问答定制、引导定制等。UNIT 平台包括大规模预置知识,能够进行精准的训练数据推荐,有效降低了数据富集成本。同时,UNIT 还提供多种接口,可以轻松调用 ChatBot<sup>8,9</sup>。

单轮对话模式可以理解为简单的一问一答模式,在这个模式中系统能清楚理解用户意图,直接反馈正确答案。单轮对话模式的开发流程如下图所示。

---

<sup>5</sup>俞凯,陈露,陈博,等. 任务型人机对话系统中的认知技术:概念、进展及其未来 [J]. 计算机学报,2020,38(12):2333-2348.

<sup>6</sup>Weizenbaum J. ELIZA-A Computer Program for the Study of Natural Language Communication between Man and Machines[J]. Communications of the ACM, 1983, 26(1):23-28.

<sup>7</sup>陈健鹏, 马建辉, 王怡君. 基于多轮交互的人机对话系统综述 [J]. 南京信息工程大学学报, 2019, 11(3):256-268

<sup>8</sup>方爱国, 张博锋, 郭晓燕, 等. 基于学习过程的智能导学系统的研究与实现 [J]. 计算机应用研究, 2005, 22(8):140-142.

<sup>9</sup>李艳红, 樊同科. 基于 Agent 技术的智能导学系统设计 [J]. 电子设计工程, 2016, 24(7):26-28.

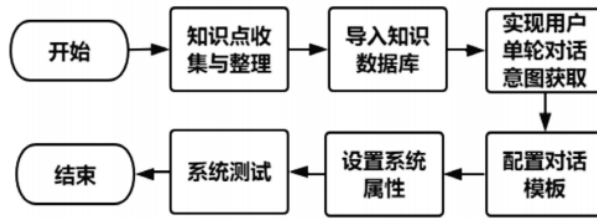


图 4: 单轮对话模式的开发流程

首先，将爬取到的百度百科数据进行数据预处理，然后，把预处理得到的数据作为知识库导入并训练。其次，在 UNIT 平台新建问答意图，并将相对应的答案回复设置完成，配置相应的对话模板。最后，设置系统的各个属性，进行系统测试，顺利完成后即完成开发。

我的服务概览

可用研发环境数 (个) 0 [购买](#) [详情](#) 可用生产环境数 (个) 0 [购买](#) [详情](#)

模型列表 [?](#)

[训练并部署训练时信息](#)

版本	描述	训练时间	训练进度 <a href="#">?</a>	训练环境	生产环境 <a href="#">?</a>	操作
v2		2023-12-29 22:54:11	● 训练完成	运行中 <a href="#">详情</a>	未部署 <a href="#">部署</a>	<a href="#">删除</a>
v1		2023-12-29 21:24:49	● 训练完成	部署当前版本	未部署 <a href="#">部署</a>	<a href="#">删除</a>

图 5: 训练并部署

### 2.3.3 搭建前端界面

由于所学技术有限，本系统所使用的前端技术比较简单，基于 httpplib 库、html、css 和 JQuery 库搭建的简易问答系统展示界面。整个系统界面如下图所示：



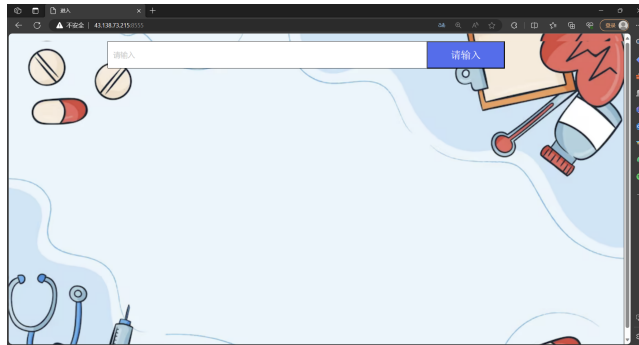


图 6: 系统界面图

我们可以看到在界面上方有一个输入框，用于用户的交互，在用户输入医疗相关的问题，点击右边的“请输入”按钮，系统会在下方给出问题的答案。

cpp-httplib 是个开源的库，是一个 c++ 封装的 http 库，使用这个库可以在 linux、windows 平台下完成 http 客户端、http 服务端的搭建，这是一个多线程“阻塞” HTTP 库。使用起来非常方便，只需要包含头文件 `httplib.h` 即可。他与业务进程的关系如下图所示：

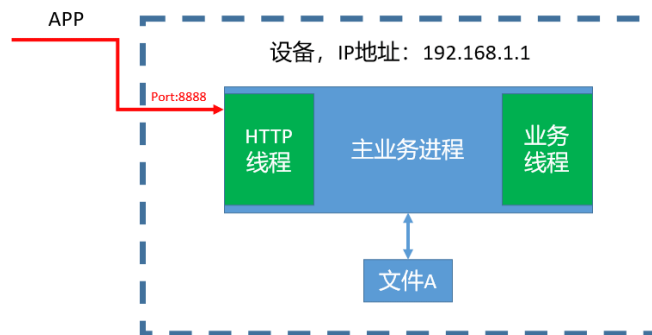


图 7: httplib 业务逻辑

jQuery 是一个 JavaScript 库，它简化了 JavaScript 编码的过程，并提供了许多强大的功能和工具。而 AJAX (Asynchronous JavaScript and XML) 则是一种用于在后台与服务器进行异步通信的技术。

在使用 jQuery 和 AJAX 进行开发时，通常会使用 jQuery 的 AJAX 方

法来发送 HTTP 请求并获取服务器响应。这个方法可以接收一个包含配置选项的对象作为参数，其中包括请求的 URL、请求类型（GET 或 POST）、数据（可选）、成功回调函数（可选）等。

当通过 AJAX 发送请求时，jQuery 会在后台使用 XMLHttpRequest 对象或者新的 Fetch API 来处理实际的 HTTP 请求。它可以向服务器发送 GET 或 POST 请求，并且可以附带数据。服务器将根据请求的类型和数据进行处理，并返回相应的结果。一旦服务器响应返回，jQuery 将触发成功回调函数，并将服务器响应作为参数传递给该函数。

通过使用 jQuery 的 AJAX 方法，我们可以在不刷新整个页面的情况下，与服务器进行数据交换。这使得网页可以在不中断用户操作的情况下更新部分内容，提高了用户体验。

总结起来，jQuery+AJAX 的原理就是通过 jQuery 库中的 AJAX 方法发送 HTTP 请求，与服务器进行异步通信，并在服务器响应返回后执行相应的回调函数来处理响应结果。这种方式可以在不刷新整个页面的情况下，实现动态更新和交互。

为了考虑之后的复用方便性，我把调用百度 unit 的 api 封装了动态库并打包，这里首先简单地介绍一下动静态库：

- 静态库（.a）：程序在编译链接的时候把库的代码链接到可执行文件中。程序运行的时候将不再需要静态库。
- 动态库（.so）：程序在运行的时候才去链接动态库的代码，多个程序共享使用库的代码。

并且，一个与动态库链接的可执行文件仅仅包含它用到的函数入口地址的一个表，而不是外部函数所在目标文件的整个机器码。在可执行文件开始运行以前，外部函数的机器码由操作系统从磁盘上的该动态库中复制到内存中，这个过程称为动态链接（dynamic linking）。动态库可以在多个程序间共享，所以动态链接使得可执行文件更小，节省了磁盘空间。操作系统采用虚拟内存机制允许物理内存中的一份动态库被要用到该库的所有进程共用，节省了内存和磁盘空间。

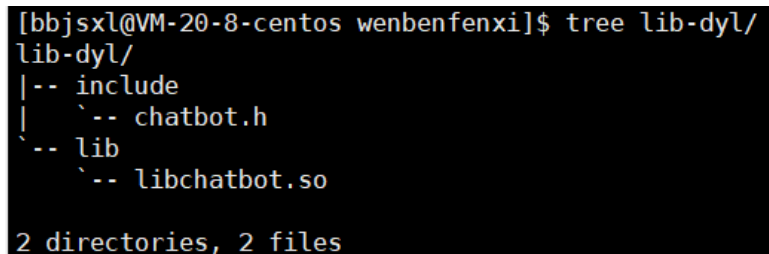
其实 C++ 程序链接的本质就是把所有的.o 链接形成一个可执行程序，这时，如果.o 文件太多了，别人使用起来就会很不方便，所以需要动态库。

我打包动态库的 makefile 文件如下：

```
1 libchatbot.so: chatbot.o
```

```
2      g++ -shared -std=c++11 -lpthread -I./include -L./  
      lib  
3      -ljsoncpp -L/usr/local/lib -I/usr/include/jsoncpp  
      -lcurl  
4      -o libchatbot.so chatbot.o  
5  
6      chatbot.o: chatbot.cc  
7      g++ -fPIC -c chatbot.cc -o chatbot.o  
8  
9      .PHONY: clean  
10     clean:  
11         rm -rf *.o *.so  
12  
13     .PHONY: dyl  
14     dyl:  
15         mkdir -p lib-dyl/lib  
16         mkdir -p lib-dyl/include  
17         cp *.so lib-dyl/lib  
18         cp *.h lib-dyl/include
```

打包好后我们可以看到动态库的目录结构如下图所示：



```
[bbjsxl@VM-20-8-centos wenbenfenxi]$ tree lib-dyl/  
lib-dyl/  
|-- include  
|   |-- chatbot.h  
|-- lib  
|   |-- libchatbot.so  
  
2 directories, 2 files
```

图 8: 动态库目录结构

至此，前端搭建工作基本完成。

### 3 功能展示

#### 3.1 命名实体识别

如下图所示，模型训练了 17 个小时后终于结束。运行模型输入句子，模型会预测出实体名词：

```
2023-12-30 15:25:24 epoch 11, step 300, loss: 27.18, global_step: 4990
2023-12-30 15:25:24 epoch 11, step 469, loss: 14.43, global_step: 5159
=====validation / test=====
processed 275417 tokens with 26282 phrases; found: 24127 phrases; correct: 15015.
accuracy: 81.94%; precision: 62.23%; recall: 57.13%; FB1: 59.57
bod: precision: 61.53%; recall: 58.39%; FB1: 59.92 6407
dep: precision: 75.53%; recall: 82.56%; FB1: 78.89 94
dis: precision: 72.31%; recall: 80.16%; FB1: 76.04 7007
dru: precision: 69.00%; recall: 69.57%; FB1: 69.28 1574
equ: precision: 46.67%; recall: 35.44%; FB1: 40.29 120
ite: precision: 39.87%; recall: 37.76%; FB1: 38.79 1106
mic: precision: 74.88%; recall: 72.46%; FB1: 73.65 601
pro: precision: 64.70%; recall: 55.62%; FB1: 59.82 2799
sym: precision: 47.32%; recall: 32.88%; FB1: 38.80 4419
2023-12-30 15:52:19 epoch 12, step 1, loss: 8.841, global_step: 5160
```

图 9: 模型训练

```
Dep: []
Please input your sentence:
肺部感染
['B-dis', 'I-dis', 'I-dis', 'I-dis']
['B-dis', 'I-dis', 'I-dis', 'I-dis']
['肺', '部', '感', '染']
Pro: []
Dis: ['肺部感染']
Dru: []
Bod: []
Sym: []
Mic: []
Equ: []
Ite: []
Dep: []
Please input your sentence:
六、新冠肺炎疫苗接种的发展热源自1964年五国疫苗临床试验以来，随着医学技术的发展
```

图 10: 命名实体识别

#### 3.2 前端知识库智能问答

如图 8-9 所示我们输入问题“头痛发热应该怎么办”和“感冒了要注意什么”，点击提交按钮后，系统会在输入框的下方给出问题的回答。



图 11: 头痛发热

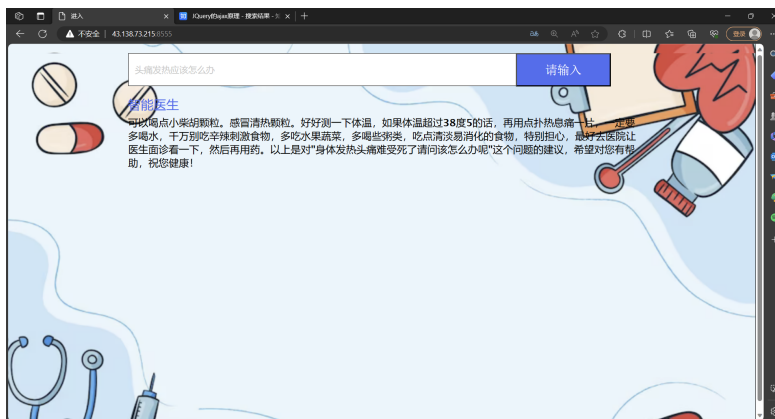


图 12: 感冒

## 4 总结

本论文提出了一种基于命名实体识别（NER）和百度 unit 的 api 构建的智能医疗问答系统。该系统旨在通过自动化的方法从大量文本中提取问题和答案，并提供简洁、易读的回答给用户，以提高医疗信息获取的效率。

首先，论文介绍了命名实体识别（NER）技术的重要性和应用场景。NER 技术可以识别和分类出具有特定意义的实体名称，如人名、地名、组织机构名等。在医疗领域，NER 技术可以帮助机器理解文本中的重要信息，并提

供对这些信息的结构化表示。

其次，论文详细介绍了 LSTM+CRF 模型在命名实体识别中的应用。LSTM+CRF 模型结合了长短期记忆网络（LSTM）和条件随机场（CRF）两种技术，能够有效地捕捉句子中的上下文信息以及实体之间的关系。LSTM 通过引入门控机制解决传统 RNN 在处理长序列时的梯度问题，CRF 则通过联合建模考虑相邻标签之间的依赖关系。在医疗领域，LSTM+CRF 模型可以对实体进行准确的识别和分类。

然后，论文描述了如何利用爬虫技术从百度百科获取医疗相关实体的知识，并作为系统的知识库。通过将命名实体识别得到的实体名词作为关键词，爬取百度百科的相关数据，可以丰富系统的知识，并提供更准确和全面的回答。

接着，论文介绍了系统的架构和实现细节。系统使用阿里云天池的测试平台进行模型训练和评估，利用百度 unit 平台进行智能问答模型的部署和调用。同时，前端部分使用 httpplib 和常用的 Web 开发技术，如 jQuery、HTML 和 CSS，实现用户与系统的交互界面。

最后，论文对系统进行了实验评估，并总结了实验结果。在阿里天池测试平台上，系统的得分达到了 3.236，其中 CMeEE-V2-F1-score 可以达到 60% 以上，取得了还算不错的效果。实验结果表明，基于命名实体识别和百度 unit 的 api 构建的智能医疗问答系统可以准确识别和回答医疗相关问题，提高了医疗信息获取的效率。

综上所述，本论文提出了一种基于命名实体识别和百度 unit 的 api 构建的智能医疗问答系统，并通过实验证明了系统的有效性和可行性。该系统在医疗领域具有广泛的应用前景，可以帮助人们快速获取医疗相关信息，提高医疗服务的质量和效率。