[1 绪论 3](#_Toc4399)

[课题背景及目的 3](#_Toc10013)

[现实意义 3](#_Toc27413)

[理论意义 3](#_Toc12952)

[1.2 国内外研究状况 3](#_Toc29626)

[论文构成及研究内容 4](#_Toc13182)

[推荐算法的设计与实现 5](#_Toc19495)

[2.1 背景知识 5](#_Toc5633)

[2.1.0 命名实体识别： 5](#_Toc27105)

[2.1.2 LSTM/CRF 命名实体识别 6](#_Toc21291)

[2.1.3 自编码器的原理和应用 7](#_Toc15614)

[2.1.4 8](#_Toc20770)

[2.2 顶层设计： 8](#_Toc6093)

[2.2.0 数据预处理 8](#_Toc4787)

[2.2.1 数据的格式 8](#_Toc27835)

[2.2.2 数据的初步处理 10](#_Toc6179)

[2.3 特征提取 11](#_Toc23644)

[2.3.1 基于正则表达式的格式化信息提取 11](#_Toc9106)

[2.3.2 关键词提取 11](#_Toc7359)

[2.3.3 自编码机提取one-hot编码的关键词特征 14](#_Toc25977)

[2.4 预测模型的设计 15](#_Toc19582)

[2.4.1 模型的输入输出 15](#_Toc31235)

[2.4.2 模型架构 15](#_Toc5925)

[2.4.3 模型的训练 16](#_Toc9818)

[3、推荐算法的评估 16](#_Toc32223)

[3.1 技术关键词的结果分析 16](#_Toc32022)

[3.1.1 评价指标 16](#_Toc28303)

[3.1.2 关键词提取结果 17](#_Toc2932)

[3.1.3 自编码器的实现与结果分析： 19](#_Toc25798)

[3.2 推荐算法在测试数据集上的结果 20](#_Toc32002)

[3.2.1 评价指标 20](#_Toc27430)

[4、结论与展望 22](#_Toc20958)

# 1 绪论

## 课题背景及目的

### 现实意义

目前，越来越多不同特点的应聘者和越来越细化的岗位之间，存在巨大的信息不对称。如何能够高效、准确的将合适的人推荐到合适的岗位，成为各大公司关注的重点[1]。特别是本项目关注的技术岗位的智能推荐，由于技术岗位的分化日趋复杂，人们的专业背景比较之前更加多样，各个新兴公司对于员工的要求更加多元，催生了一大批招聘网站和猎聘公司。[5] 这些网站的推送和公司的筛选，花费了大量的人力资源。根据统计：报告显示，截至2014年，网络招聘市场份额达到33.6亿元[16]如果能够设计一套更加智能的简历自动推送算法，将能够极大提高人们的工作效率，产生积极的社会影响。

### 理论意义

简历的智能推送算法，涉及到自然语言处理中的许多重要课题。简历和岗位描述，都是文本。正确的进行推送，需要挖掘文本背后的语义。因此，项目实现过程中，可能会涉及到：名称实体识别（Named Entity Recognition， 下文缩写为NER）, 推荐系统（Recommendation system, 下文缩写为 RS）等自然语言处理的重要课题。这里会遇到语义复杂性的问题，因此，会涉及歧义消解，属性抽取等基础理论的研究与应用来解决这些困难。

特别地，简历的文本信息中存在大量实体歧义的问题需要解决。同一个技术，同一个公司和岗位，可能在简历中出现不同的描述方式。这些歧义的消解，对于后面的推荐效果影响很大：如果实体个数多，并且存在大量未合并的冗余，就会导致后面推荐的部分，存在更加严重的“长尾”现象。这一问题，对于推荐系统的有效性，构成巨大挑战。为了应对这一挑战，本课题将会用到知识图谱的相关理论，解决信息提取时候样本稀疏的问题。

最后，该算法可以为许多类似的问题提供解决的思路。比如法律卷宗的智能检索等。

## 1.2 国内外研究状况

目前，有不少关于简历智能推荐的研究和应用 [2][3][4]。其中，比较普遍的模式为：搭建一个投放、展示简历的平台。有的课题在其上设计了简单的推荐算法。刘、于、杜，基于ASP实现了一个毕业生就业招聘网站。该网站，减少了用人单位的招聘和毕业生的应聘之间不必要的限制,促进了双方更广泛的交流,提高了双向选择的效率和成功率[4]。罗仕鉴 、陈杭渝，同样设计了同一个基于网站的平台，展示学生简历信息[3]。然而这些工作只提供了一个更快、更便捷的平台。仍然需要资深的工作人员筛选简历。在应聘者越来越多，应聘岗位越来越复杂的当下，这将耗费大量的人力资源。陈晓、王建民，提出并实现了一种高效的基于自动用户需求建模的算法，采用遗传算法对以往简历信息进行学习，建立该职位对于简历的需求模型,达到自动向用户个性化推荐简历的目。[2]然而，对于本项目针对的技术岗位招聘，由于岗位需求的多样性和简历特点的多元化，以及标注数据的匮乏，这种启发式的算法不适用。 该系统，采用了概率分类模型，分别从简历和工作描述中提取了关键实体，并对他们进行匹配。同样，对于本课题针对的复杂岗位招聘，该方法将遇到“长尾效应”等困难。

## 论文构成及研究内容

本课题针对的问题是，技术岗位简历的智能推送。利用猎聘公司员工工作过程中留下的标记以及相应简历和工作描述的数据，设计简历智能推送算法。这一推荐过程，等价于预测工作描述对于简历的偏好。在预测的时候，需要进行信息的筛选和过滤，提取从文本中提取有用的信息。这与推荐系统的定义一致。因此，本课题可以参考关于推荐系统的设计方法。

由于简历和工作描述主要以非结构化的文本构成，因此，主要的工作重点分为两部分：文本特征的提取和特征的相似度比较。

对于本项目，文本信息特征的提取十分困难。由于技术岗位和相关技术数量庞大、种类繁多，因此直接提取的特征稀疏，难以匹配。本课题的期望利用岗位技能知识图，结构化这些特征，从而解决这一问题。

得到提取的特征后，进行简历推荐的过程可以参考推荐系统的若干算法（Recommendation System）。目前，推荐系统主要分成两种类型[10][11]：Content-based System (下面缩写为：CBS)和Collaborative filtering systems （下面缩写为： CFS）。由于每个工作描述对应的人和总人数相比差距很大，CFS 对应的矩阵过于稀疏，不容易训练。并且，通常新的工作描述没有任何简历与之对应，因此，CRS方法不适用。因此，基本的思路是采用CBS的方式。

关于职位技能“知识图”，这部分在网上没有找到直接可用的资料。因此，这个部分需要自己实现。首先提取简历和工作描述中的关键词。之后，获得对于关键词的数学表示（向量表示）。最后，通过提取短文本，并训练变换模型的方式，获得工作简历和工作描述的向量表示，通过训练一个分类器，预测工作简历和工作描述的匹配打分。根据打分，做简历的推荐。

# 推荐算法的设计与实现

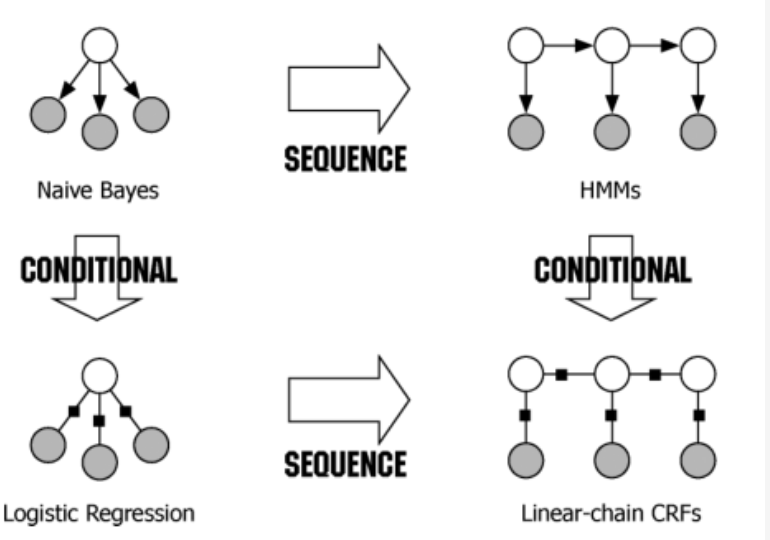
## 2.1 背景知识

### 2.1.0 命名实体识别：

命名实体识别是一项具有挑战性的任务，传统上需要以特征工程和词典的形式获得大量知识才能实现高性能。

命名实体识别是自然语言处理（NLP）领域的一项重要任务。传统的方法，比如应用CRF，SVM或感知器模型，依赖手工提取的特征（Ratinov和Roth，2009【19】【20】; Passos等【21】，2014; Luo【22】等，2015实现了对于自然语言中命名实体的识别任务。 然而，Collobert等人 【23，24，25】提出了一个有效的神经网络，这个网络模型只需要很少的特征工程，而是从中学习重要的功能字嵌入训练大量未标记的文本 。因为，最近十分成功的，在无监督的词嵌入学习中取得进展，大量的数据（Collobert和Weston，2008; Mikolov等，2013）和神经网络训练算法允许深层架构（Rumelhart等，1986）【26】。

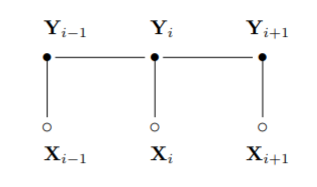
2.1.1 CRF命名实体识别

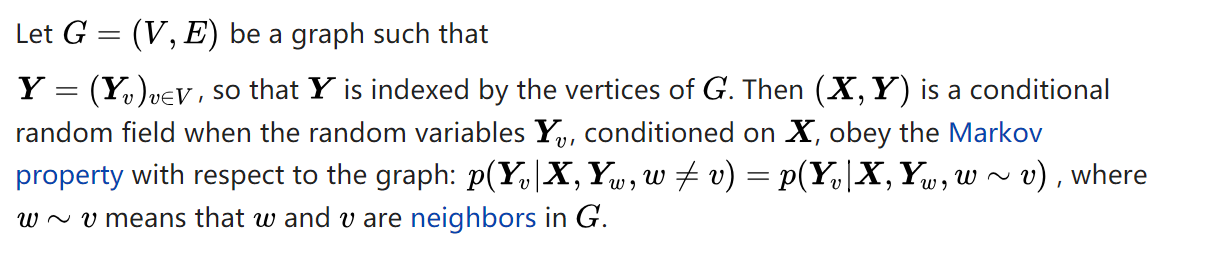


条件随机场（CRF）是一类特殊的概率图模型，常用于模式识别和机器学习，并用于结构化预测。条件随机场最初由Lafferty【16】提出。论文中，他指出：条件随机场与常用的隐马尔可夫模型相比，存在很多优点---包括模型不依赖强独立假设，因为从模型对于隐状态条件独立。另外，条件随机场还有效避免了，基于有向图模型的经常收到的状态出现不均的问题。状态出现不均这个问题，在很多的任务中都有出现，限制了大量模型基于有向图模型的应用。比如：在自然语言的预测标签的问题中，由于很多时候，标签存在长尾效应。长尾效应最初由 克里斯·安德森（Chris Anderson）在2004年首次提出，指的是尽管有些独自占比重很小，个性化很强的成分，但是他们加起来，却在整体中，占有很大的比重。比如，在网易音乐的平台上，很多人有非常个性化的偏好，这导致了许多歌曲，只有非常少的人喜欢，但是这些歌曲加起来，却吸引了超越那些非常流行的歌曲的听众。在这个命名实体识别的任务中，大量的实体在很少的句子中出现。但是这些实体，却有很大的价值：他们占了所有实体的很大的比重。传统的基于有向图的模型，比如隐马尔科夫模型，他依赖对于隐状态转移概率的建模。如果对应的标签出现的频率很低，会导致出现转移概率“假零”的问题。然而，条件随机场，是对所有隐状态的条件下，观测的条件概率建模，可以非常有效的避免这个问题。

条件随机场（CFR）模型介绍：

CRF是一类基于无向图的判别模型：概率图中，有两类节点：观测节点和隐含节点。观测节点描述的是可观测的信息，隐含节点描述的是隐含的状态。在具体应用中的例子如下：比如在命名实体识别中，观测为输入的自然语言，隐含的状态是语言对应的标签，是或者不是某一个实体。再比如语音处理的任务中，观测对应输入的声音信号，隐状态对应声音信号背后的语言符号【18】。条件随机场利用概率图对于观测值和隐藏状态建模。条件随机场的具体数学定义为：

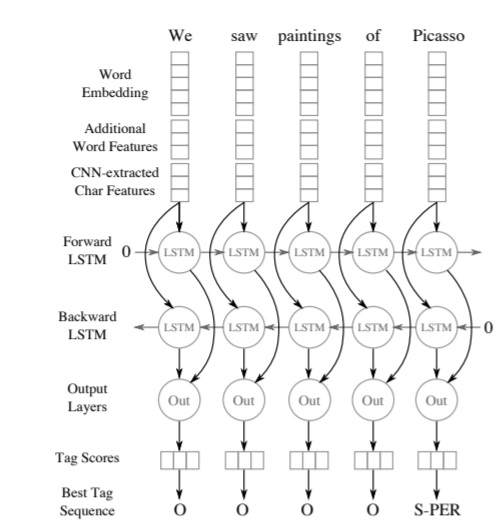


 {\displaystyle G=(V,E)}

优化的目标函数为：

### 2.1.2 LSTM/CRF 命名实体识别

然而，上面介绍的条件随机场模型，输入的特征是固定的。尽管条件随机场模型本身，允许灵活的定义。但是，这些特征都是预先定义的，“静态”的特征。但是自然语言有很大的灵活性：很多词语的含义依赖复杂的上下文，这种静态的特征，并不能很好的表述词语在语句中的含义。另外，由于概率图模型本身的限制，条件随机场不能很好的刻画和捕捉这种“上下文”复杂的信息，导致对于一般自然语言的命名实体识别问题很难处理。针对这个问题，【19】提出了一种新型的神经网络架构，它使用混合双向LSTM和CNN架构自动检测字和字符级特征，从而消除了对大多数特征工程的需求。 文章还提出了一种在神经网络中编码部分词典匹配的新方法，并将其与现有方法进行比较。 广泛的评估表明，只有标记文本和公开可用的词嵌入，他们的系统在CoNLL-2003数据集上具有竞争力，并且超过了以前报道的OntoNotes 5.0数据集上的先进性能状态2.13 F1指数。文章采用如下的模型架构：

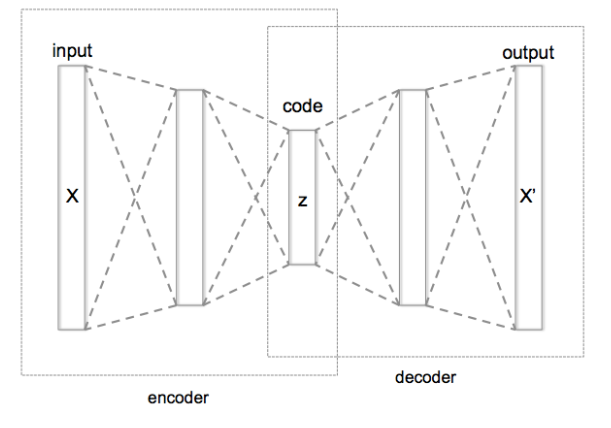


由于LSTM可以在实际训练过程中，动态的调节对于输入的“编码”，产生跟上下文相关的，有丰富语义的表示能够有效地提取文字的特征。因此，bi-LSTM成为一种对于传统随机向量场的改进。

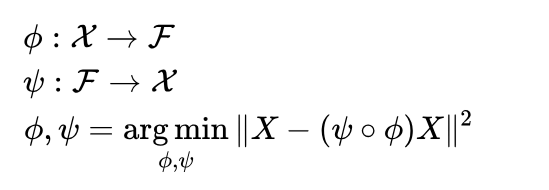
### 2.1.3 自编码器的原理和应用

自编码器是一种人工神经网络，用于高效编码的无监督学习。自动编码器最早由Rumelhart在1986 年提出 【28】，用来处理复杂、高维数据的降维问题 。自动编码器的目的是学习一组数据的表示（编码），通常用于降维。 最近，自编码器的概念已经越来越广泛地用于学习数据生成模型。 2010年的一些最强大的人工智能涉及在深度学习网络中堆叠稀疏自动编码器[27]。

自编码器的模型架构如下图：



自动编码器学习将来自输入层的数据压缩成短代码，然后将该代码解压缩为与原始数据非常匹配的东西。这迫使自动编码器参与降维，例如通过学习如何忽略噪声。自编码器有编码器和解码器两个部分组成：



其中。。。

### 2.1.4

## 2.2 顶层设计：

这个部分介绍智能推荐算法的顶层设计。主要分两部分：信息特征的抽取和分类模型的设计。对于第一部分，信息抽取，本文将其划归为两个子任务：结构化数据的抽取和非结构化的信息抽取。对于结构化的部分，通过对于求职简历中结构化字段，进行信息挖掘实现。包括做基本的数据清洗，以及通过正则表达式提取关键信息。对于非结果画的文本数据，本文再用层级信息提取的方式，一步步得到富含语义的向量表示：首先在少量带标注的数据上，训练关键词提取的模型，之后，利用这个模型，抽取无标注的求职简历中的关键词信息，得到关键词列表。之后对提取到的关键词进行简单的统计和分析，以及初步的信息筛选，完善和维护关键词列表。最后，通过自编码器无监督的学习非结构文档在关键词特征上的向量表示。并通过可视化，进行了进一步的分析。

对于第二部分，分类模型的设计，搭建了一个二分类的模型。以工作描述和求职简历的向量表示作为输入，匹配的简历结果作为标签，训练模型。前一部分实现的结果化数据，则作为以后推荐平台用来筛选使用的数据，不参与这个二分类模型的预测。

### 2.2.0 数据预处理

### 2.2.1 数据的格式

数据包含两部分：工作简历的数据和工作描述的数据。其中，工作描述为一段中文文本，描述岗位的相关要求：包括技能的要求以及对求职者经历的要求等。下面是一段具体的工作描述：

|  |
| --- |
| Web前端及UI设计 |
| 职位描述：  职责描述：  1、负责UI设计，网站前端开发；  2、基于HTML5.0标准进行页面制作，编写可复用的用户界面组件；  3、负责WEB端交互产品的前端系统设计、开发与实现；  任职要求：  4、熟悉互联网设计流程，熟练掌握PS／AI／Sketch等设计工具，了解WEB端的设计规范；  5、精通JavaScript、HTML5、CSS语言及相关技术，熟悉跨终端、跨浏览器的开发模式和平台特性；  6、精通Ajax、DOM、XML、JSON等相关技术，能使用原生JS编程，精通面向对象，使用过jquery、ext、Dojo、Prototype等js框架的至少一种；  7、具有前端UI框架 bootstrop、easyUI开发经验者优先；  8.、参与过大型java项目的开发者优先；  9、对用户体验、交互操作流程、及用户需求有深入理解。 |

可以发现，工作描述的格式相对规则，语言也比较规范。在工作描述里面，包含了大量的关于技术的关键词。以及对于求职者工作经历的细节要求：比如要求“参与过大型java项目的开发者优先”。

工作简历为一份文档，样例参见附录一。在MongoDB中，包含的字段为：['id', 'birthday', 'skill', 'height', 'politics\_status', 'crawled\_at', 'location', '\_id', 'education', 'gender', 'tel', 'train', 'resume\_id', 'email', 'crawled\_time', 'intention', 'age', 'image\_url', 'work\_experience', 'degree', 'household', 'award', 'marital\_status', 'work', 'project', 'url', 'update\_time', 'self\_evaluation'}工作简历（由于不同来源的简历字段上存在细节的差异，因此，这里只展示了51job这一个来源的数据字段）。这些字段可以分成两类：一种是结构化的字段，比如性别、年龄、工作年限等。另一种是非结构化的字段，比如工作经历的描述，对参加过项目的描述。样例见下表。

|  |
| --- |
| 结构化数据样例： |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | **最近工作** （2年 2个月 ） | | | 职　位： | 高级软件工程师 | | 公　司： | ucloud云计算 | | 行　业： | 互联网/电子商务 | | |  |  | | --- | --- | | **最高学历/学位** | | | 专　业： | 系统科学与工程 | | 学　校： | 华中科技大学 | | 学历/学位： | 硕士 | | |

|  |
| --- |
| 项目描述样例： |
| |  |  | | --- | --- | | 所属公司： | ucloud云计算 | |
| |  |  | | --- | --- | | 项目描述： | 新增云postgresql服务。  使得用户可以通过控制台或是RESTful接口快速获得托管的postgresql服务。  能够给公有云用户提供日常维护、监控、告警服务。 | |
| |  |  | | --- | --- | | 责任描述： | 主导功能设计、开发、测试、线上发布   postgresql数据库调研，功能分析、裁剪。平衡postgresql功能的多样性和现有框架局限性之间的矛盾  完成高可用方案调研、设计、开发  完成分布式方案调研、设计、开发  提供用户日常操作维护功能接入能力  运行环境及数据库服务关键监控指标的分析、制定及实现 | |

|  |
| --- |
| 工作经历样例： |
| |  |  | | --- | --- | | 2015/1-至今 | ucloud云计算 (2年 2个月 ) | |  | 互联网/电子商务|500-1000人|民营公司 | | 关系存储开发部 | **高级软件工程师** | | |  |  | | --- | --- | | 工作描述： | 进入公司以来主导开发mysql\postgresql云数据库产品。面向开发者提供易获  得、易扩容的普通、高可用数据库产品，同时带领DBA团队向用户提供产品  相关的线上线下维护、咨询服务。  目前云postgresql产品是继阿里云之后，国内第二款同类产品，产品相关性能领先于阿里云.  同时负责客户需求采集及分析、竞品分析、产品设计，以及部门代码版本管理及灰度发布。  主要面向Linux服务器后台C/C++开发。涉及分布式、docker 、zookeeper、protobuffer 等开发平台。 精通C/C++编程，对面向对象的编程思想、STL、设计模式等有很好的理解和掌握。熟练使用shell \python\node.js编程。  熟悉Linux系统应用编程、网络应用编程，对TCP/IP协议等有较好的了解和掌握。  熟悉Linux开发、编辑、编译、调试环境。 | | | |

通过对比，可以发现：在个人的求职简历当中，特别是非结构化的字段里面，语言的用词比较工作描述而言，更加随意，属于更加日常的自然语言。在这些更加日常的文本中，存在大量歧义的词汇：比如对于技术技能的描述，存在着很多歧义和同义的表达：比如对于关键技术词汇的缩写---numpy 缩写为np。另外，还存在大量的拼写错误，比如把python 错误的贫血成为pytohn。正是这些多元的日常语言中的表达方式，导致关键词提取时候，标签的长尾效应：即大量的标签出现频率很低，但是却非常重要，不能轻易的忽略。因为，如果粗暴的忽略了这些低频出现的标签，会导致很多关键信息的丢失。特别的，如果一个人习惯于在简历中使用与众不同的简写和缩写，那么他的求职简历当中的关键词提取就表现很差。最后的结果对于这样的求职简历是不公平的。从实际的角度看，往往是领域从业经历丰富的人，习惯于使用各种专业的简称和缩写。这样的分类系统，就容易忽略这样的潜在优秀人才。

### 2.2.2 数据的初步处理

文本当中的原始数据，经过初步的数据统计后发现，存在着非常多的“非法字符”，这些字符对于后面进一步的处理不利，因此需要专门清除。由于这些“非法字符”主要是编码错误导致的乱码，可以通过python的字符串处理和过滤简单的清除。具体的流程如下：

* 1. 全文本切分为句子
     1. 利用标点符号："。|；|？|’|‘|'|\\n"将短文本切分为句子
  2. 句子切分为短句
     1. 利用标点：",|、|（|）|{|}|【|】|\\n|\(|\)|\[|\]"，将句子切分为短句。
  3. 短句分词
     1. 利用jieba.cut将短句分词
  4. 筛选和清理停词
     1. 从网上下载中文停词

（http://blog.csdn.net/shijiebei2009/article/details/39696571）

* + 1. 清理停词，标点等符号
    2. 去除长度小于2的单词
    3. 去除少于两个单词的句子

除了清除“非法字符”之外，还需要为不同格式的工作简历，完成读入的接口。由于工作简历来自“51job”，“猎聘网站”，“智联招聘”等不同资源，具体的数据字段存在差异。并且，文件的格式也存在差异。其中，有.doc格式的文档，也有.html格式的数据，还有在Mongodb上面，分字段保存好的数据。因此，需要为每种格式的数据，分别设计对应的读入模块。这里在处理.doc文档的时候，由于没有方便的python接口，建立的信息都是存储在word文档的表格结构当中，并且这些网格结构都是嵌套的。（具体的情况参见附件一的样例。）针对这个困难，我利用python广度优先的便利这些嵌套的表格，利用递归的算法，实现数据的提取。

## 2.3 特征提取

### 2.3.1 基于正则表达式的格式化信息提取

对于简历中，结构化的信息，需要进一步的处理。比如在“工作年限”字段，很多人的回答是文字字符串的形式。比如“工作四年”。但是，后期做筛选的时候，需要经行数值上的比较。因此，这里利用正则表达式，匹配数字文本，进一步还原为数值形式。

这里仅仅以“工作年限”字段为例子。实际的求职简历当中，还包含着例如“期望薪金”等更加复杂的字段。比如，“期望薪金”字段，很难找到具体的规则来写正则表达式，因此，这个部分只是部分的完成。未来在这个“短文本”的信息抽取领域，还有更多的挑战有待解决。

### 2.3.2 关键词提取

#### 2.3.1.1 数据说明

训练提取关键词的模型，是利用6000多条带标准的句子的训练数据来做的。训练原始数据的格式是：{x,y}其中x为一个句子，y为对应的标签。y是与x等长的零一向量，对应x中每个“单词”的标准信息。

这里对原始数据进行了进一步的处理：利用python的jieba分词工具，进一步扩展原始数据的词性标签。得到的结果为：每一个“单词”，最后具体的形式为一个二元组（单词，词性）。每一个二元组，对应一个之前的“零一标签”。这个扩充“单词语义”的部分，需要重新对原始数据进行分词。但重新分次的结果，可能跟之前的对不上。造成标签的不匹配问题。这里采用空格分隔的方式，重新拼接之前分好的单词。这样，程序会自动对这些空格的区域分开。但问题是，存在把之前分好的一个单词，拆开的情况。这里，经过统计，发现这种情况出现的非常少，因此，本文中直接将比对不上的单词删去，比对上的单词，保留之前的标签。使用这样的方法，扩充单词从字符串到元组。

#### 2.3.1.2 条件随机场模型提取关键词

这里，采用了两种版本的条件随机场模型，进行关键词的提取。模型的实现，是依赖于python提供的一接口：pycrfsuite提供的实现方式。

第一种条件随机场模型是直接依赖pycrfsuite实现，没有自己设计模型进一步的特征表示，而是直接依靠隐含状态在图模型中的连接关系，有数据驱动的方式，直接学习相关的特征，训练关键词提取的模型。后文把这种模型的名字定义为：crf\_baseline。

另外一种条件随机场模型，是自己人工进一步提取和扩充每个单词的信息。这里，我显示的引入了前后两个单词作为中间单词的特征。具体的形式如下：

一个句子可以表达为 S,w1,w2..wt-1,wt,wt+1,...wn,E

在这样的设定下，每个单词wi有前面和后面各一个单词（这里允许出现S和E）每个单词的原始格式是一个二元组（字符，词性）。这里扩城的是字符部分。将字符，扩充为一个特征列表。其中包含前后一个单词的信息。因此，调整后，数据的形式为（特征列表，词性）。特征列表的形式为[feature\_f,feature\_p,feature\_n], feature\_f, feature\_p, feature\_n分别对钱一个单词，当前单词以及后一个单词的特征。对于第i个单词字符串：wstr\_i,提取他的特征总结在下面的表格中:

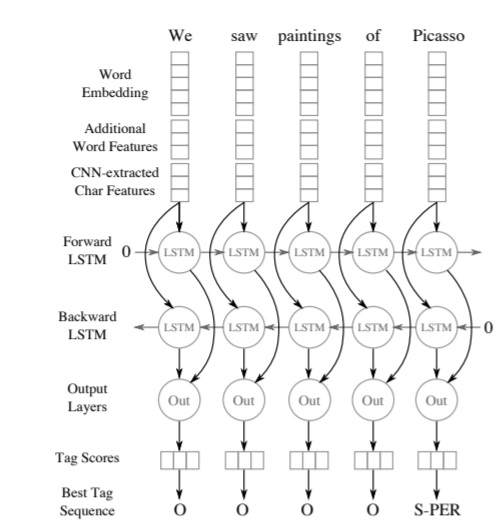
|  |  |
| --- | --- |
| 特征名称 | 特征含义 |
| lower | 字符串的小写形式。因为存在很多英文单词，特别是技术关键词，在工作描述和求职简历当中，有多元的表达方式。其中一种最常见的，就是大小写的不同。比如python和Python，就是两种应该合并的词汇。这里提取lower这个特征，显示的处理这种特殊的“多元表达” |
| issuper | 一个bool类型的特征，判断字符串是不是完全是大写。一般而言，一些关键词会因为大小写与一般出现的单词区分开。特别是对于关键词的提取，很多技术关键词全程很长，因此，存在将他首字母大写，再拼接起来的多元表达方式。比如，Visual Studio，在很多的简历当中，出现的形式是VS。对于这种情况，这个特征能够有效地捕捉多元的表达。 |
| 标注词性 | 这里显示的把词性添加到单词的特征当中，强调词性特征对于模型准确预测的重要性。 |

#### 2.3.1.3 bi-LSTM-CRF

对于之前部分介绍传统CRF的的不足，这里利用文章【19】提出用来做命名实体抽取任务的模型：bi-LSTM-CRF，视图解决之前手动提取特征的一些不足。包括：特征是静态的，以及对于特征信息提取的不全面。网络架构与原来文章的相同，代码是在github上面的公开代码基础上，进行的修改和调整。

模型的整体架构如下图：

由双向连接的LSTM和CRF构成。LSTM为文本特征动态提取的工具，用来抽取文本当中的语义信息。动态的生成句子中词汇的语言信息。之所以这里采用的是双向链接的LSTM模型，是因为文本数据，不仅仅只有下文的信息对于理解当前单词有影响，上文的信息也对于理解单词含义有影响。因此，双向链接的LSTM对于文本，能有更好的提取效果。



其中，最大的改动有两个地方。一个是，之前做命名实体识别工作，该模型是一个多分类的模型，对应很多不同实体的标签。但这里做的是关键词的抽取，不区别关键词的具体小分类，因此只是一个二分类问题。

另外一个改动，是关于输入单词特征的改动。由于工作描述和求职简历的文本比较特殊，专业性比较强。另外，求职简历当中，个人化的描述存在很多非正规的表达。这导致了不能直接使用论文中已有的工具，将单词直接转化为向量表示。原始的代码支持两种不同的转换方式：一种是one-hot编码，另一种是自己添加的word embedding字典。

One-hot 编码，是先统计一共出现了多少单词，经过筛选和预处理之后，形成词典的列表。每个单词对应一个列表中的id，利用这个id完成单词到向量的转换。例如：此表中有N个单词，单词w出现在第i个位置。那么w的one-hot编码：vec\_oh[w]为一个N长的向量，第i个位置是1，其他位置都是0.

word embedding的编码，是利用别人预先训练好的词向量，作为单词的数学表达。一个单词，如果没有在此表中找到对应的单词，就会根据预先的设的，赋值一个特定的向量。

经过分析发现，这两种方式都不能够直接套用。因为大量的单词在求职简历中有多元的表达，得到的词表有严重的长尾效应。对于one-hot编码，会出现大量的单词不再列表之中。而对于word embedding,大量没有比对上的单词，都会被赋予相同的数值。因此，这两种方式都不可取。

因此，这里的解决的方案是：在求职简历和工作描述中，自己构建词表，生成对应的one-hot编码，作为单词的向量表示，输入模型经行训练。

另外，值得特别说明的是，迷行采用的损失函数为：交叉熵损失函数。通过梯度下降的方法，经行训练。

模型具体的参数，参见下表：

|  |  |  |
| --- | --- | --- |
| 参数名称 | 参数赋值 | 参数含义 |
| batch size | 64 | 随机梯度下降过程中，每个批的样本个数 |
| epoch | 40 | 最多训练多少轮 |
| hidden\_dim | 300 | LSTM隐状态的维度 |
| lr | 0.001 | 学习速率 |
| embedding\_dim | 300 | LSTM内部词向量word embedding的维度 |

### 2.3.3 自编码机提取one-hot编码的关键词特征

由于获得的关键词词表很大，并且里面的信息冗余，因此需要经行降维和进一步的信息抽取工作。之前步骤中，模型预测的搭配的关键词词表，维度很高，并且长尾效应严重：大量出现统一关键技术实体的不同关键词汇的多元表达。这些多元的表达可以通过简单的清洗工作，合并一些：例如，可以通过全部转换为小写字符的方式，去除Python和python这样的多元表达。

尽管如此，得到的关键词词表，依旧有很大的冗余和很高的维度。因此，我采用了自编码器来进行进一步的数据降维和处理工作。

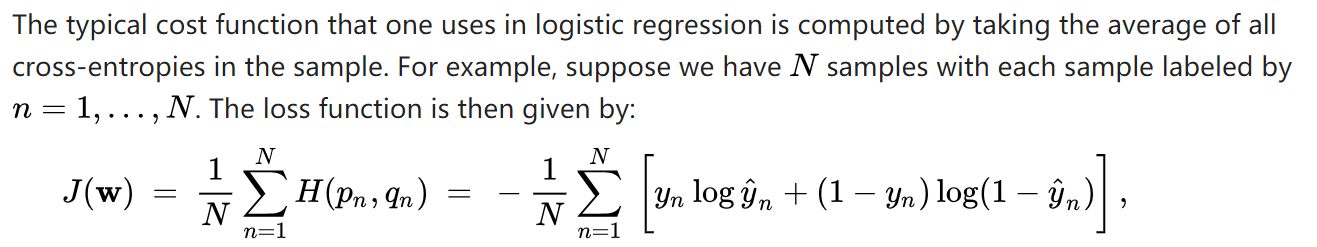
首先，自编码器的模型架构为5个隐藏层的结构，具体的参数见下表。

|  |  |  |
| --- | --- | --- |
| 参数名称 | 参数赋值 | 参数含义 |
| encoder\_1 | 500 | 编码器的第一层的维度,激活函数为线性函数 |
| encoder\_2 | 250 | 编码器的第二层的维度,激活函数为线性函数 |
| encoder\_3 | 64 | 编码器的第三层的维度,激活函数为线性函数 |
| decoder\_1 | 250 | 解码器的第一层的维度,激活函数为线性函数 |
| decoder\_2 | 500 | 解码器的第二层的维度,激活函数为线性函数 |
| output | 2058 | 输出层的维度，激活函数为sigmoid |

模型的具体输入和输出是未标注简历对应关键词的向量。具体的构建方式如下：经过之前关键词模型的提取，一共在已经收集到的简历中，挖掘出K个不同的关键词，将这K个不同的关键词构成列表。第i个关键词key\_i对应的向量表达是vec[key\_i],其中第i个位置为1，其他位置为0。对于一个简历的文档，提取出的关键词如果为key\_1....key\_m（这里提取的，都是互不相同的关键词，需要过滤掉反复出现的关键词）。这些关键词对于的向量表示分别为vec[key\_i]，i = 1,2...m。那么，该段文本的向量表示为：vec[doc] = vec[key\_i]求和。

将vec[doc]作为输入，经过编码器和解码器之后，标准输出为原始的输入向量：vec[doc]。

自编码器的损失函数为类别交叉熵损失函数。具体的数据定义如下：



## 2.4 预测模型的设计

### 2.4.1 模型的输入输出

预测模型的输入是工作要求（JD）和求职简历(JL)的向量。

生成向量的流程为：首先提取工作要求和求职简历当中的文本信息。进行初步的数据清洗。文本清洗包括去除无效字符，统一小写，分词，去除无用标点和符号。最后得到清晰完整的原始文本信息。

提取完文本信息后，在之前步骤得到的关键词表中，找到对应的关键词，并生成对应的文本的关键词向量。在把得到的关键词向量，输入自编码器中，得到关键词向量的编码表示。之后，将这个表示作为输入训练模型。标签为：工作介绍和求职简历是否匹配。如果匹配，则标记为1，没有匹配，则标记为0。

### 2.4.2 模型架构

模型的架构为多层感知机，具体的架构信息参见下面的表格：

|  |  |  |
| --- | --- | --- |
| 参数名称 | 参数赋值 | 参数含义 |
| hidden\_1 | 200 | 多层感知机的第一层的维度,激活函数为“relu” |
| hidden\_2 | 50 | 多层感知机的第二层的维度,激活函数为“relu” |
| output | 1 | 输出层的维度，激活函数为sigmoid |

### 2.4.3 模型的训练

模型训练集大小为372，其中172个匹配的带标注阳性数据，和200个自己生成的阴性数据。阴性数据的生成，是通过把不匹配的工作描述和求职简历进行拼接得到。为了避免阳性样本点的比例过小，将生成阴性样本的个数限制在200。另外，之前自编码器输出是一个64维度的向量，对应工作描述或者求职简历的抽象向量表示。而这里的输出是工作描述和求职简历的配对，也就是这两个向量的拼接结果。得到的向量长度是128维度的。

# 3、推荐算法的评估

## 3.1 技术关键词的结果分析

### 3.1.1 评价指标

根据信息抽取任务的一般评价标准，这里引入三个指标来评价模型关键词提取的性能。他么分别是准确率（prec）,召回率（recall）和F1值。他们分别的定义参见下表：

在定义这三个评价标注之前，先引入四个符号：

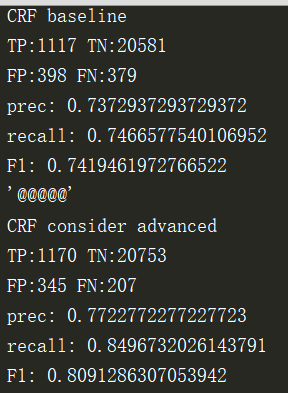
|  |  |
| --- | --- |
| 符号 | 含义 |
| TP | 真阳性个数 |
| FP | 假阳性个数 |
| TN | 真阴性个数 |
| FN | 假阴性个数 |

之后，依据上面定义的四个符号，进一步定义评价模型的三个指标：

|  |  |  |
| --- | --- | --- |
| 符号 | 数学表达 | 含义 |
| prec | TP/(TP+FP) | 准确率：判断为阳性的样本中，真阳性站的比例 |
| recall | TP/(TP+FN) | 召回率：判断对的阳性样本个数，占全体阳性样本的比例 |
| F1 | 2TP / (2TP + FP + FN) | F1值：一个综合评价模型的指标。综合了prec和recall,给出一个模型整体的评估。 |

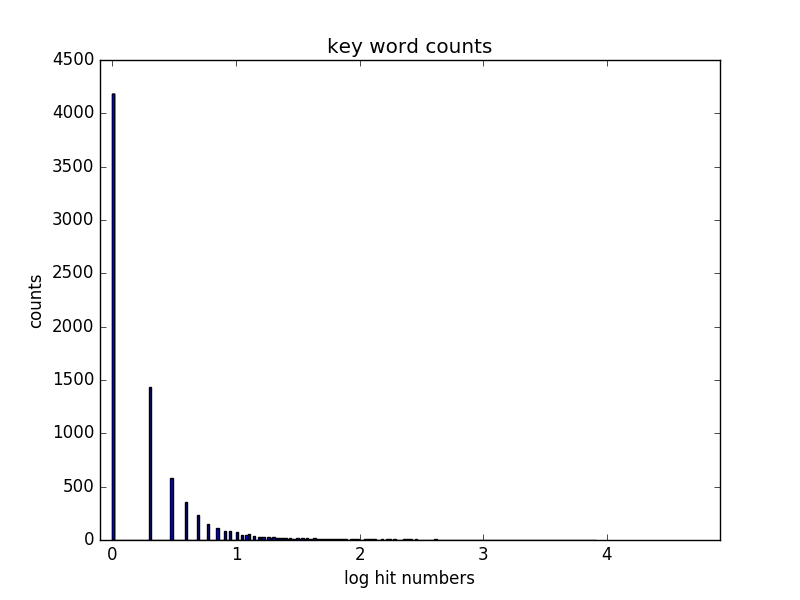
### 3.1.2 关键词提取结果

|  |  |  |  |
| --- | --- | --- | --- |
| **指标** | **CRF\_baseline** | **CRF\_advance** | **biLSTM-CRF** |
| 准确率 | 0.737 | 0.772 | 0.712 |
| 召回率 | 0.746 | 0.850 | 0.859 |
| F值 | 0.742 | 0.809 | 0.783 |



从原始的结果可以看出：改进后的条件随机场模型效果最好：综合起来的F值更高。但是，bi-LSTM-CRF没有实现预期的效果。经过具体的分析，数据量。最后筛选出来的模型，尽管loss比其他的模型小，但是依旧是一个比较大的数值。并且，模型最后的F值却并不理想。这提示了模型目标函数的选取不合适或者模型没有得到充分的训练。这将是下一步改进的方向。

在带标注的数据6000多句文本上面训练完模型之后，在51job的5636个求职简历数据集合中，提取了关键词。一个得到原始的关键词8251个，平均每个关键词出现23次。下图为关键词出现次数的统计图。横坐标是出现次数的对数，纵坐标是区间内关键词的个数：

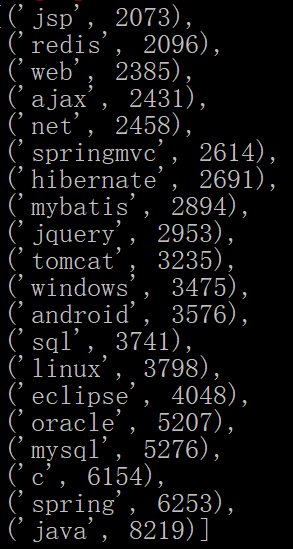


从图中可以发现，有非常严重的长尾效应：将近90%+的单词，出现的次数小于5次。只出现一次的关键词，占到了整体的50%。由于提取到的关键词中，存在很多潜在的假阳性，并且很多的关键词的错误拼写被提取出来了：比如观察到的有python错误拼写成pytohon。这些都不利于后面进一步的处理。因此，这里人为的过滤掉出现频率小于3次的关键词。过滤之后，剩下了2058个关键词。下图为出现频率最高的前几个关键词的截图：从图中可以看出：出现最多的关键词是“java”,一共出现了8219次。排名前几的，大多是关于编程语言的关键词。这些技能性质的关键词，在简历中反复出现。注意到，“jsp”的出现频率也很高，达到了2063次。这个其实是Java script的缩写形式。可以看出，关键此表中，依旧存在很大量的冗余现象。存在很多关键词，虽然他们的形式不一样，但背后表示的实体其实是同一个。这些需要进一步的处理和筛选。在本文中，采取的方法是利用自编码器，来压缩特征的长度。

extract from 5636 JL data from 51job



关于keyword 的统计：



一共找到8251个关键词，平均一个关键词hit 23次

困难：错误拼写：



remove all the word which hit times<=3, left 2058 key words

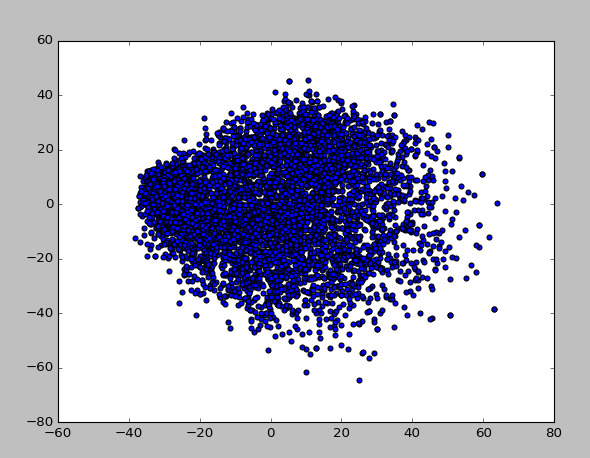
auto encoder之后，保留最大的两个主成分：

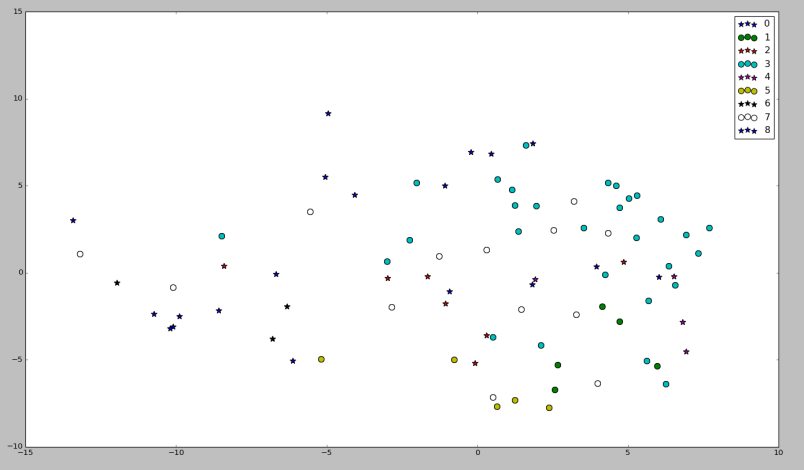
### 3.1.3 自编码器的实现与结果分析：

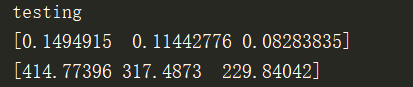
根据之前的网络结果，设计并实现了自编码器。输入是原始的向量，经过编码器之后，得到了64维的文段关键词的隐含表示。

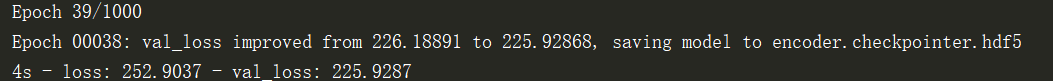
结果见下面的图：为了进一步分析和可视化，对得到的64维向量使用主成分分析，提取两个“主成分”，对向量在这主成分的投影图做可视化的散点图。下图对应的是比配过的数据的散点图。每一个类别，对应一个特定的工作表述，表明这个简历适用于这份工作描述，也就是说，他们之间匹配的程度很高。因此，同一个类别下的简历，在技术关键词层面，应该是相似的。期望在图上，会聚在一起。

实际的结果，符合这的预期：相同类别的，出现一定的“聚类的”情况。这表明模型通过关键词提取和自编码器，确实有得到更加抽象的向量表征。并且这些向量表征，能匹配上对应的实际内涵。这些结果都表明，这一系列方法对于关键词的提取是有效的。









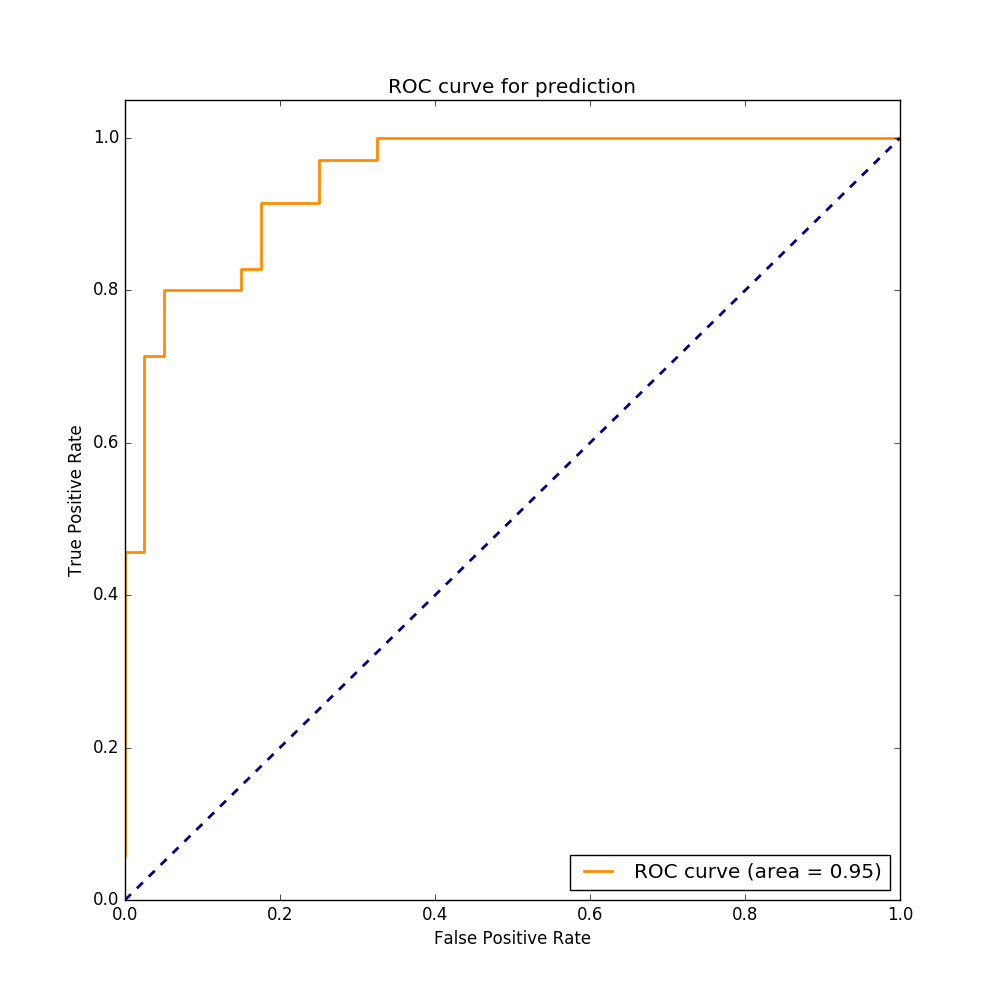
## 3.2 推荐算法在测试数据集上的结果

### 3.2.1 评价指标

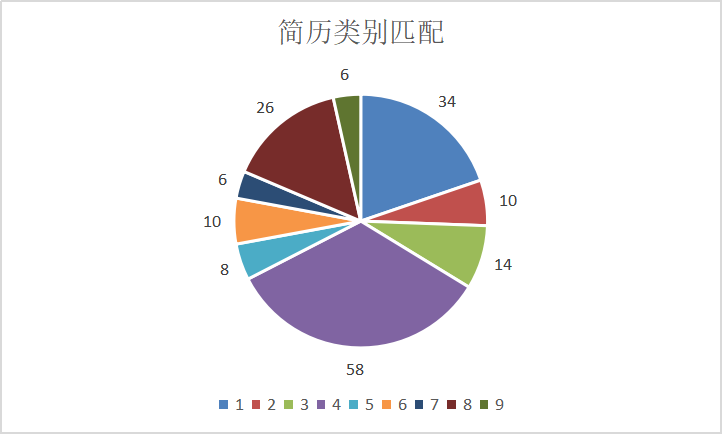
该算法最后等价于一个二分类问题，输入求职简历和工作描述的“向量表示”（上文提到的“encoder”的输出结果），输出是一个0到1之间的实数y\_pred。含义为：模型预测称该求职简历，匹配对应工作描述的“概率”。因此，这里的评价方式，是画ROC曲线和计算曲线的积分面积，即AUC值，来评估模型的分类效果。对于ROC曲线而言，如果是随机的打分，曲线为第一象限的对角线。曲线越向上，AUC的数值越大（在[0,1]区间上的积分制越大），表明模型的分类效果越好

3.2.2 结果与分析

下图为模型的ROC曲线，计算得到的AUC为0.95。这个结果表明，模型的分类效果比较理想，达到了预期的效果。



然而，这个目前的结果仍旧存在问题：主要是带标注的数据不足，以及数据的丰富程度不够。这导致木气的结果无法直接用于实际使用。下图统计了带标注的数据，以及每个工作描述对应的求职简历的个数。



从图中可以看出，数据集一共只有9个工作描述，一共172封求职简历。并且，这些求职简历对应的工作描述的分布非常不均匀：最多的一个工作描述，有58封求职简历，而最少的工作描述，只有6封简历。尽管在关键词部分，分析了每个工作描述对应的求职简历，提取得到的关键词向量，在经过自编码器之后，有一定“聚类”的效果，但是实际中，潜在的工作描述分类更加细致：比如同样是网页方面的工作，有不同岗位更细节的需求。

# 4、结论与展望

本论文的任务是，设计简历智能推荐的算法。算法的输入是一份工作描述和一份求职简历，输出是他们匹配程度的打分。本文的工作主要是将已有的自然语言处理的方法，应用到这个问题场景，解决实际问题。主要的工作分成两大部分：特征的提取和分类与预测模型的搭建。在特征提取的任务中，对于结构化的信息，提取了关键字段的信息，进一步利用正则表达式，基于规则的提取和处理了结构化数据中的文本信息。在对于非结构化的文本信息的处理中，重点挖掘了建立文本正的关键词信息。首先，在少量带标注的数据上训练关键词抽取的模型。这一步，本文尝试了三种不同的模型，包括条件随机场的baseline模型和人工提取特征后的两种，再加上Bi-LSTM-CRF模型。在三个指标：准确率、召回率和F值的综合考量和对比下，挑选了人工抽取特征的条件随机场模型。在获得了训练好的关键词提取模型后，利用模型提取了“51job”简历数据库中，5000多封求职简历的关键信息。对提取的关键词进行了统计上的分析和初步的清洗。分析发现，抽取得到的关键词存在“长尾效应”：即大量出现频率很低的词汇，占了很大的比重。这主要是由于同一个技术实体，对应非常多元化的表达方式：比如缩写和错误拼写等问题。在数据处理中，观察到Java,jsp,java script这样的多元表达，以及类似于python和pytohon这样的错误拼写。这些问题，并没有能够通过条件随机场模型中，手动设计的参数完全解决。因此，利用自己设计的自编码器，无监督的提取了，以文档为基础单位的，关键词的抽象向量表达。通过主成分分析进行降维之后，可视化了分类后的模型。结果表明，自编码器确实学到了有意义的向量表征。表现在结果是，体现的是匹配到同一类工作描述的求职简历，最后得到的关键词向量表示聚在一起。

在完成特征抽取任务之后，进一步实现了一个二分类模型，输入为给定的工作描述和求职简历，输出为他们相似程度的打分。模型的设计采用多层感知机，将之前特征提取部分得到的，求职简历和工作描述的向量表示，进行拼接后作为输入。利用172个匹配好的求职简历和工作描述作为阳性数据，随机匹配不同类别的数据作为阴性数据，对模型进行了训练。之后，分析了结果的ROC曲线并计算了AUC。模型最后的AUC达到了0.95，表明模型有比较好的分类效果，能够比较好的预测求职简历和工作描述是否匹配。

以上是本文实现的简历智能推荐算法。由于简历库的搭建进度比预料的慢很多，最后没有来得及获取足够多带标注的求职简历的数据。因此模型最后的性能还不能令人满意。但是，本文实现了一个完整的流程：从数据的预处理，到特征的抽取，再到最后分类模型的训练。这些都为后面进一步的工作打下了基础：这一套完整的流程，可以启发后面对于简历匹配问题的进一步研究，并且提供了一个baseline作为参考。相信在获得更多的数据，以及对于算法进行进一步的分析之后，可以进一步提高算法的效果。

最后，值得一提的是，本文在分析和研究关键词提取的时候，发现在技术关键词抽取的领域，也出现了“长尾效应”。本文具体研究和分析了这个现象的起因，并且通过自编码器试图解决这一问题。最后得到一个还不错的结果：编码器能够学到有语义特征的信息。这表明本文的尝试有一定的成效，可以被更多的相关任务借鉴和参考。“长尾效应”是一个普遍存在的问题：特别在信息抽取和推荐系统的领域。本文采用的方法，背后的理念是：数据驱动的获得信息的抽象的分布式表示。在大量数据下，通过无监督的学习，在低维中得到这样一个更加鲁棒、更加蕴含语义信息的向量表示。笔者相信，这是在正确的方向上走出的一步，期望后续的工作，能过进一步挖掘背后的机制，并以此为基础改进算法。

1. 致谢
2. 参考文献

[1] Morley-Kirk,James,中国企业招聘现状知多少——2009中国选才调查分析报告,人力资源,2009,15:1672-1632

1. 陈晓、王建民，面向网络招聘的个性化简历推荐算法研究--《第二十五届中国数据库学术会议论文集（一）》2008年
2. 罗仕鉴 、陈杭渝，基于Web的高校毕业生就业招聘系统的设计与实现--《计算机应用研究》2002年07期
3. 刘丹，于琨 ，杜静翌，基于ASP的大学生就业招聘网站的设计与实现--《河南机电高等专科学校学报》2009年06期
4. 高峰，招聘网站运营管理模式研究
5. Algorithms and Methods in Recommender Systems
6. Automated Profile Extraction and Classification with Stanford Algorithm
7. Information\_Extraction\_and\_Named\_Entity\_Recognition
8. Taming Text How to Find, Organize, and Manipulate It Grant S. Ingersoll, Thomas S. Morton, and Andrew L. Farris Foreword by Liz Liddy December 2012 ISBN 9781933988382 320 pages printed in black & white
9. http://infolab.stanford.edu/~ullman/mmds/ch9.pdf
10. Daniar Asanov，Berlin Institute of Technology,Berlin, Germany,Algorithms and Methods in Recommender Systems
11. Kalva, Thimma Reddy, "Skill Finder: Automated Job-Resume Matching System" (2013). All Graduate Plan B and other Reports. Paper 343.
12. <https://nlp.stanford.edu/software/CRF-NER.shtml>
13. <http://www.datacommunitydc.org/blog/2013/04/a-survey-of-stochastic-and-gazetteer-based-approaches-for-named-entity-recognition>

Mikheev, Andrei; Moens, Marc; Glover, Claire. 1999. “Named Entity Recognition without Gazetteers.” Proceedings of EACL ’99. HCRC Language Technology Group, University of Edinburgh.

<http://acl.ldc.upenn.edu/E/E99/E99-1001.pdf.>

1. http://www.199it.com/archives/410490.html

【16】John Lafferty, Andrew McCallum, and Fernando C.N. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data", . June 2001.

【17】@article{anderson2008long,

title={Long tail, the, revised and updated edition: Why the future of business is selling less of more},

author={Anderson, Chris},

journal={Hyperion},

year={2008}

}

【18】@article{hinton2012deep,

title={Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups},

author={Hinton, Geoffrey and Deng, Li and Yu, Dong and Dahl, George E and Mohamed, Abdel-rahman and Jaitly, Navdeep and Senior, Andrew and Vanhoucke, Vincent and Nguyen, Patrick and Sainath, Tara N and others},

journal={IEEE Signal Processing Magazine},

volume={29},

number={6},

pages={82--97},

year={2012},

publisher={IEEE}

}

【19】Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pages 384–394. Association for Computational Linguistics

【20】Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In Proceedings of the Thirteenth Conference on Computational Natural Language Learning, pages 147–155. Association for Computational Linguistics.

【21】Alexandre Passos, Vineet Kumar, and Andrew McCallum. 2014. Lexicon infused phrase embeddings for named entity resolution. In Proceedings of the Eighteenth Conference on Computational Natural Language Learning, pages 78–86.

【22】Gang Luo, Xiaojiang Huang, Chin-Yew Lin, and Zaiqing Nie. 2015. Joint entity recognition and disambiguation. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 879–888. Association for Computational Linguistics.

【23】Ronan Collobert, Koray Kavukcuoglu, and Clement 織 Farabet. 2011a. Torch7: A Matlab-like environment for machine learning. In Proceedings of BigLearn, NIPS Workshop, number EPFL-CONF-192376.

【24】Ronan Collobert, Jason Weston, Leon Bottou, Michael ´ Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011b. Natural language processing (almost) from scratch. The Journal of Machine Learning Research, 12:2493– 2537.

【25】Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In Proceedings of the 25th International Conference on Machine Learning, pages 160–167. ACM.

【26】David Rumelhart, Geoffrey Hinton, and Ronald Williams. 1986. Learning representations by back-propagating errors. Nature, pages 323–533.

【27】Domingos, Pedro (2015). "4". The Master Algorithm: How the Quest for the Ultimate Learning Machine Will Remake Our World. Basic Books. "Deeper into the Brain" subsection. ISBN 978-046506192-1.

【28】@inproceedings{fan2013design,

title={Design of an Optical Mechanical System for High-Resolution Encoders},

author={Fan, Yi Hua and Chen, Ching En and Lou, Liao Yong and Chen, Chun Yu},

booktitle={Applied Mechanics and Materials},

volume={284},

pages={2711--2716},

year={2013},

organization={Trans Tech Publ}

}

附录一、工作简历样例：

|  |  |
| --- | --- |
| 更新时间：2017-03-06 | cid:f91b71ce-d3db-4844-9bc0-b7247d66f87d |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  | | --- | | 头像 | | |  |  |  | | --- | --- | --- | | 的=流程状态：     标签： | | ID:536731232 | | |  |  | | --- | --- | | cid:39b88467-76cd-4ff0-bd92-9c0e253b169e | 目前正在找工作 | | |  |  | | --- | --- | | cid:4cbea4c7-b5e3-421f-a05f-a3572b356241 | \* | | |  |  | | --- | --- | | cid:e371c7f1-4eca-425b-9758-a1b9ad002e0c | \*\* | | | cid:93af0f35-e459-4799-a740-9069706ba2ff男 | 32岁（1985年1月1日 ） | 现居住上海 | 6年工作经验 | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | **最近工作** （2年 2个月 ） | | | 职　位： | 高级软件工程师 | | 公　司： | ucloud云计算 | | 行　业： | 互联网/电子商务 | | |  |  | | --- | --- | | **最高学历/学位** | | | 专　业： | 系统科学与工程 | | 学　校： | 华中科技大学 | | 学历/学位： | 硕士 | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  | | --- | |  |  |  | | --- | | **目前年收入：** 28 万元 (包含基本工资、补贴、奖金、股权收益等) | | |  |  | | --- | --- | | |  | | --- | |  | | |  |  | | --- | | **求职意向** | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | 期望薪资： | 30-40万元/年 | | |  |  | | --- | --- | | 地点： | 上海 杭州 武汉 深圳 北京 | | | |  |  | | --- | --- | | 职能： | 高级软件工程师 软件工程师 | | |  |  | | --- | --- | | 行业： | 互联网/电子商务 通信/电信/网络设备 计算机软件 | | | |  |  | | --- | --- | | 工作类型： | 全职 | |  | | |  |  | | --- | --- | | 自我评价： | 爱好建模和编程 | | | |  |  | | --- | | **工作经验** | | |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | 2015/1-至今 | ucloud云计算 (2年 2个月 ) | |  | 互联网/电子商务|500-1000人|民营公司 | | 关系存储开发部 | **高级软件工程师** | | |  |  | | --- | --- | | 工作描述： | 进入公司以来主导开发mysql\postgresql云数据库产品。面向开发者提供易获得、易扩容的普通、高可用数据库产品，同时带领DBA团队向用户提供产品相关的线上线下维护、咨询服务。  目前云postgresql产品是继阿里云之后，国内第二款同类产品，产品相关性能领先于阿里云.  同时负责客户需求采集及分析、竞品分析、产品设计，以及部门代码版本管理及灰度发布。   主要面向Linux服务器后台C/C++开发。涉及分布式、docker 、zookeeper、protobuffer 等开发平台。 精通C/C++编程，对面向对象的编程思想、STL、设计模式等有很好的理解和掌握。熟练使用shell \python\node.js编程。  熟悉Linux系统应用编程、网络应用编程，对TCP/IP协议等有较好的了解和掌握。  熟悉Linux开发、编辑、编译、调试环境。 | | | | |  | | |  |  | | --- | --- | | 2011/7-2015/1 | 华为技术有限公司 (3年 6个月 ) | |  | 通信/电信/网络设备|10000人以上|民营公司 | | 无线开发部 | **软件工程师** | | |  |  | | --- | --- | | 工作描述： | 无线网络MBSC物理传输平台软件设计、开发、测试、维护。  负责开发分布式环境下的网络资源优化管理、运行数据采集及自动监控告警等功能。  负责传输领域业务西迁、能力建设 | | | | |  |  | | --- | | **项目经验** | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | 2016/4-至今 | **云postgresql产品** | | |  |  | | --- | --- | | 所属公司： | ucloud云计算 | | | | |  |  | | --- | --- | | 项目描述： | 新增云postgresql服务。  使得用户可以通过控制台或是RESTful接口快速获得托管的postgresql服务。  能够给公有云用户提供日常维护、监控、告警服务。 | | | | |  |  | | --- | --- | | 责任描述： | 主导功能设计、开发、测试、线上发布   postgresql数据库调研，功能分析、裁剪。平衡postgresql功能的多样性和现有框架局限性之间的矛盾  完成高可用方案调研、设计、开发  完成分布式方案调研、设计、开发  提供用户日常操作维护功能接入能力  运行环境及数据库服务关键监控指标的分析、制定及实现 | | | | |  | | |  |  | | --- | --- | | 2015/6-2016/4 | **云mysql高可用产品** | | |  |  | | --- | --- | | 所属公司： | ucloud云计算 | | | | |  |  | | --- | --- | | 项目描述： | 新增云postgresql服务。  使得用户可以通过控制台或是RESTful接口快速获得托管的postgresql服务。  能够给公有云用户提供日常维护、监控、告警服务。 | | | | |  |  | | --- | --- | | 责任描述： | 主导功能设计、开发、测试、线上发布   postgresql数据库调研，功能分析、裁剪。平衡postgresql功能的多样性和现有框架局限性之间的矛盾  完成高可用方案调研、设计、开发  完成分布式方案调研、设计、开发  提供用户日常操作维护功能接入能力  运行环境及数据库服务关键监控指标的分析、制定及实现 | | | | |  | | |  |  | | --- | --- | | 2015/1-2015/8 | **数据备份模块** | | |  |  | | --- | --- | | 所属公司： | ucloud云计算 | | | | |  |  | | --- | --- | | 项目描述： | 通过RESTful接口向用户提供多样化的备份相关服务。例如sql/nosql数据库主动及定期备份功能、备份下载、从备份（全量、增量）创建数据库实例、备份黑名单等等。 | | | | |  |  | | --- | --- | | 责任描述： | 负责功能设计、开发、测试、线上发布   能够做到兼容sql/nosql数据库。  支持全量及增量备份。  备份文件支持文件存储及对象存储，能够做到2种存储方式的负载均衡、快速切换。  对数据量较大的库支持流式压缩和快照备份。  支持基于增量备份、binlog的数据回档。  能够向重要用户提供定制的备份服务。 | | | | |  | | |  |  | | --- | --- | | 2014/1-2014/7 | **移动鲁棒性优化** | | |  |  | | --- | --- | | 所属公司： | 华为技术有限公司 | | | | |  |  | | --- | --- | | 项目描述： | 基于无线环境测量数据，调整小区切换相关各参数，同时能降低切换抖动，能够很大幅度的提升通话切换成功率。 | | | | |  |  | | --- | --- | | 责任描述： | 研读3GPP信令切换协议，完成数学建模，写出需求分析、概要设计、测试用例等，完成数据采集、上报、切换仲裁，编译版本，在多型基站上烧录版本测试通过。 | | | | |  | | |  |  | | --- | --- | | 2012/4-2012/10 | **分布式系统环境数据采集、上报与统计** | | |  |  | | --- | --- | | 所属公司： | 华为技术有限公司 | | | | |  |  | | --- | --- | | 项目描述： | 在分布式环境中，实时统计系统运行数据。设计了满足大量数据有序高效读写的数据结构，实现了数据的分级存储、上报与统计。通过负载上报与集中裁决保证各系统负载均衡，考虑到了各系统上报数据对齐。 | | | | |  | | |  |  | | --- | --- | | 2011/8-2012/3 | **传输层负载均衡算法** | | |  |  | | --- | --- | | 所属公司： | 华为技术有限公司 | | | | |  |  | | --- | --- | | 项目描述： | 集中实现传输资源的管理和分配。在多种约束情景下，实现资源使用率的最大化。  担任开发人员角色  1、算法所需数据的环境数据采集，传输及数据存储框架。  2、负载均衡算法仿真及开发。  3、算法联调及网上问题接口。 | | | | |  |  | | --- | | **教育经历** | | |  |  |  |  |  | | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | 2008/7-2011/4 | **华中科技大学** | | 硕士|系统科学与工程 | | | |  | | |  |  | | --- | --- | | 2002/9-2006/7 | **北京理工大学** | | 本科|自动化 | | | |  |  | | --- | | **技能特长** （包括IT技能、语言能力、证书、成绩、培训经历） | | |  | | --- | | **▲**技能/语言 | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | |  | | --- | | **C/C++** | | | |  |  | | --- | --- | | |  | | --- | | 精通 | | | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | |  | | --- | | **Python** | | | |  |  | | --- | --- | | |  | | --- | | 良好 | | | | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | |  | | --- | | **Linux** | | | |  |  | | --- | --- | | |  | | --- | | 熟练 | | | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | |  | | --- | | **MySQL** | | | |  |  | | --- | --- | | |  | | --- | | 熟练 | | | | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | |  | | --- | | **Postgresql** | | | |  |  | | --- | --- | | |  | | --- | | 熟练 | | | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | |  | | --- | | **Shell** | | | |  |  | | --- | --- | | |  | | --- | | 精通 | | | | | | | | | |