文本复制检测报告单(全文对照)

№:ADBD2019R_20170516132611201904081814564038568378 21　　检测时间:2019-04-08 18:14:56

检测文献:　6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现

作者:　安磊

检测范围:　中国学术期刊网络出版总库
中国博士学位论文全文数据库/中国优秀硕士学位论文全文数据库
中国重要会议论文全文数据库
中国重要报纸全文数据库
中国专利全文数据库
图书资源
优先出版文献库
学术论文联合比对库
互联网资源(包含贴吧等论坛资源)
英文数据库(涵盖期刊、博硕、会议的英文数据以及德国Springer、英国Taylor&Francis 期刊数据库等)
港澳台学术文献库
互联网文档资源
CNKI大成编客-原创作品库

时间范围:　1900-01-01至2019-04-08

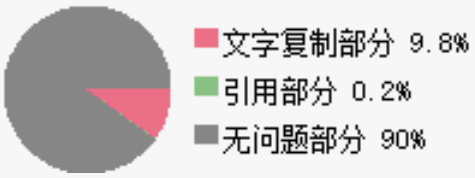
检测结果

去除本人已发表文献复制比：10%　　跨语言检测结果：0%

去除引用文献复制比：9.8%　　总文字复制比：10%

单篇最大文字复制比：4.7%（011_M201570005_聂国平）

重复字数：	[5122]	总段落数：	[12]
总字数：	[51099]	疑似段落数：	[7]
单篇最大重复字数：	[2404]	前部重合字数：	[608]
疑似段落最大重合字数：	[2904]	后部重合字数：	[4514]
疑似段落最小重合字数：	[36]		



指　标：☐疑似剽窃观点　☑疑似剽窃文字表述　☐疑似自我剽窃　☐疑似整体剽窃　☐过度引用

表　格：0　　公　式：等待检测　　疑似文字的图片：0　　脚注与尾注：0

6.8%（195） 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第1部分（总2867字）
0%（0） 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第2部分（总313字）
0%（0） 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第3部分（总441字）
0%（0） 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第4部分（总433字）
0%（0） 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第5部分（总1561字）
8.6%（494） 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第6部分（总5747字）
17%（1256） 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第7部分（总7372字）
0.4%（40） 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第8部分（总9201字）
7.9%（197） 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第9部分（总2480字）
18.4%（2904） 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第10部分（总15793字）
0%（0） 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第11部分（总3437字）
2.5%（36） 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第12部分（总1454字）

（注释：无问题部分 文字复制部分 引用部分）

1. 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第1部分

总字数：2867

相似文献列表

去除本人已发表文献复制比：6.8%(195) 文字复制比：6.8%(195) 疑似剽窃观点：(0)

1	基于GeoMedia的高速公路监控系统的研究和应用 刘溯(导师：史金余) -《大连海事大学硕士论文》- 2010-06-01	3.2%（92） 是否引证：否
2	良好亲子关系构建中的社会工作介入研究 刘雨露(导师：张翠娥;钟华) -《华中农业大学硕士论文》- 2012-06-01	2.5%（73） 是否引证：否
3	移动对等网络若干关键技术的研究 牛新征(导师：周明天) -《电子科技大学博士论文》- 2008-09-01	2.4%（68） 是否引证：否
4	智能算法研究及其在网络中的应用 梁淑萍(导师：毛力) -《江南大学硕士论文》- 2012-03-01	2.2%（62） 是否引证：否
5	基于二维码宁波网上购物物流系统的设计与实现 张猛(导师：余堃;高顺林) -《电子科技大学硕士论文》- 2013-09-25	2.2%（62） 是否引证：否
6	微博用户转发预测特征的特征选择研究 赵领帅(导师：管子玉) -《西北大学硕士论文》- 2018-12-01	2.0%（56） 是否引证：否
7	时间序列特征编码方法改进及在金融数据中的应用 董肖凯(导师：李昌峰) -《南京财经大学硕士论文》- 2018-04-01	1.7%（50） 是否引证：否
8	上海某商品房项目施工成本管理研究 李蕾(导师：刘敬孝) -《中国海洋大学硕士论文》- 2013-05-18	1.7%（48） 是否引证：否

原文内容	相似内容来源
1 此处有 91 字相似 mplementation of information extraction service. After that, the thesis describes the requirements analysis, including detailed requirements analysis, project architecture analysis and detailed analysis of model	基于GeoMedia的高速公路监控系统的研究和应用 刘溯 -《大连海事大学硕士论文》- 2010-06-01（是否引证：否） 1.rmissions, and so on.★For studying the tow subsystems, the thesis describes the requirements analysis, design, and implementation. The final chapter introduces the

2	此处有 104 字相似 ……II 目录 ……III 图目录 ……VI 表目录 ……VIII 第一章绪论 1 1.1 项目背景 1 1.2 国内外研究现状 2 1.2.1 命名实体识别研究现状 2 1.2.2 关系抽取研究现状 4 1.3 本文主要的工作 6 1.4 本文的组织结构 6	良好亲子关系构建中的社会工作介入研究 刘雨露 -《华中农业大学硕士论文》- 2012-06-01（是否引证：否） 1.STRACT -71 绪论 1 1.1 问题的提出 1 1.2 研究目的与意义 1 1.2.1 研究目的 1 1.2.2 研究意义 2 1.3 文献综述 2 1.3.1 国外研究概况 2 1.3.2 国内研究概况 4 1.3.3 文献评述 82 研究设计 10 2.1 研究的理论视角 10 2.2 主要研究方法 移动对等网络若干关键技术的研究 牛新征 -《电子科技大学博士论文》- 2008-09-01（是否引证：否） 1……ns摘要 -5ABSTRACT -7第一章 绪论 11.1 研究背景及研究意义 11.2 研究现状 31.2.1 研究概况 31.2.2 典型研究成果 31.3 关键技术研究分类 61.4 论文主要工作及贡献 131.5 本文结 智能算法研究及其在网络中的应用 梁淑萍 -《江南大学硕士论文》- 2012-03-01（是否引证：否） 1.gy Adaptive Clustering Hierarchy摘要 -3Abstract -4第一章 绪论 11.1 研究背景及意义 11.2 Ad Hoc 网络及无线传感器网络的研究现状 31.2.1 Ad hoc 网络的研究现状 31.2.2 无线传感器网络的研究现状 41.3 存在问题与解决思路 51.4 本文主要工作和结构安排 6 基于二维码宁波网上购物物流系统的设计与实现 张猛 -《电子科技大学硕士论文》- 2013-09-25（是否引证：否） 1.code;;Commodity circulation;;MVC摘要 -5ABSTRACT -6第一章 绪论 1 1.1 引言 1 1.2 项目研发意义 1 1.3 国内外研究现状分析 2 1.3.1 网络购物现状 2 1.3.2 二维码国内外现状 3 1.3.3 物流系统国内外现状 4 1.4 论文组织结构 5 微博用户转发预测特征的特征选择研究 赵领帅 -《西北大学硕士论文》- 2018-12-01（是否引证：否） 1.研究背景和意义 1 1.1.1 研究背景 1 1.1.2 研究意义 1 1.2 国内外研究现状 2 1.3 主要研究内容 4 1.4 本文组织结构 6第二章 相关知识介绍 7 2.1 特征提取 7 2.2 特征选择 8 2.3 因子分解机 9 时间序列特征编码方法改进及在金融数据中的应用 董肖凯 -《南京财经大学硕士论文》- 2018-04-01（是否引证：否） 1.criterion;;expected lower bound摘要 -4ABSTRACT -3第一章 绪论 1 1.1 选题背景 1 1.2 研究意义 2 1.3 研究综述 4 1.3.1 时间序列特征表示研究综述 4 1.3.2 时间序列相似度量研究综述 6 上海某商品房项目施工成本管理研究 李蕾 -《中国海洋大学硕士论文》- 2013-05-18（是否引证：否） 1.背景和意义 1 1.1.1 研究背景 1 1.1.2 研究意义 1 1.2 研究现状 2 1.2.1 国外研究现状 2 1.2.2 国内研究现状 4 1.3 研究目标与内容 5 1.4 研究方法和技术路线 5 1.5 研究的主要创新点 62.

指　标

疑似剽窃文字表述

1. the thesis describes the requirements analysis, including detailed requirements analysis,
2. 第一章绪论 1
   1.1 项目背景 1
   1.2 国内外研究现状 2

2. 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第2部分

总字数：313

相似文献列表

去除本人已发表文献复制比：0%(0)　文字复制比：0%(0)　疑似剽窃观点：(0)

3. 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第3部分

总字数：441

相似文献列表

去除本人已发表文献复制比：0%(0)　文字复制比：0%(0)　疑似剽窃观点：(0)

4. 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第4部分

总字数：433

相似文献列表

去除本人已发表文献复制比：0%(0)　文字复制比：0%(0)　疑似剽窃观点：(0)

5. 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第5部分

总字数：1561

相似文献列表

去除本人已发表文献复制比：0%(0)　文字复制比：0%(0)　疑似剽窃观点：(0)

6. 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第6部分

总字数：5747

相似文献列表

去除本人已发表文献复制比：8.6%(494)　文字复制比：8.6%(494)　疑似剽窃观点：(0)

1	命名实体识别与关系抽取研究及应用 李飞(导师：朱艳辉) -《湖南工业大学硕士论文》- 2018-02-28	3.4%（198） 是否引证：否
2	关系抽取技术的研究 赵立鹏;张若伟; -《计算机产品与流通》- 2018-09-15	2.9%（166） 是否引证：否
3	知识图谱中子图查询技术研究 李新锋(导师：潘鹏) -《华中科技大学硕士论文》- 2017-05-01	1.5%（84） 是否引证：否
4	多特征融合的细粒度图像检索算法 王虹(导师：王智慧) -《大连理工大学硕士论文》- 2018-03-27	0.8%（44） 是否引证：否
5	F类高效率功率放大器的研究与设计 蔡炜波(导师：袁乃昌) -《国防科学技术大学硕士论文》- 2016-11-01	0.7%（41） 是否引证：否
6	医疗风险防控系统设计与实现 田磊(导师：欧阳柳波;刘金朝) -《湖南大学硕士论文》- 2016-04-20	0.6%（37） 是否引证：否
7	基于忆阻器的物理不可克隆函数可重构性研究 文韬(导师：林亚平;杨志新) -《湖南大学硕士论文》- 2016-05-18	0.6%（37） 是否引证：否
8	在线社交网络中的安全朋友推荐方案研究与实现 刘浩文(导师：林亚平;杨志新) -《湖南大学硕士论文》- 2016-06-01	0.6%（37） 是否引证：否

原文内容	相似内容来源
1	此处有 53 字相似 文同时结合了命名实体识别和关系抽取两项任务，并将这两种模型进行组合，搭建了一个信息抽取服务系统来对非结构化文本中的三元组信息进行抽取。 命名实体识别（Named Entity Recognition，简称NER）作为信息抽取	命名实体识别与关系抽取研究及应用 李飞 -《湖南工业大学硕士论文》- 2018-02-28（是否引证：否） 1.以帮助人们更便利的获取信息，也可以对本体构建、垂直搜索、自动问答等自然语言处理领域起到非常重要的作用[1]。信息抽取主要包含命名实体识别（Name Entity Recognition，NER）、关系抽取等内容[2]。命名实体识别第一次在 MUC-6 会议上被提出，MUC-7[3]会议对命名实体在原来的基础上进行了细

	中的一个基本任务，最早是在MUC-6（the Sixth of the Message Understanding Co	
2	此处有 147 字相似 领域的文本中都有着更高的识别准确率，从而受到更多的学者、从业者们的青睐。在命名实体识别任务中表现比较出色的有监督学习方法 主要有：隐马尔科夫模型（Hidden Markov Models，简称HMM）[Bikel et al., 1999]、最大熵模型（Maximum Entropy Models，简称MEM）[Tsai et al., 2004]、条件随机场（Conditional Random Fields， 简称CRF）[McCallum et al., 2003]。基于统计方法的命名实体识别对特征选取的要求比较高，需要从文本	命名实体识别与关系抽取研究及应用 李飞 -《湖南工业大学硕士论文》- 2018-02-28（是否引证：否） 1.究者青睐。以下详细介绍主要的方法。（1）有监督学习方法该方法主要思想是将命名实体识别看做序列标注问题。目前主要模型有：隐形马尔科夫模型（Hidden Markov Models，HMM）[9]、最大熵模型（MaximumEntropy Models，MEM）[10]和条件随机场（Conditional Random Fields，CRFs）[11]等模型。其中，马尔科夫模型容易产生序列标签偏移的问题。条件随机场模型由于特征模板订制比较灵
3	此处有 35 字相似 .2 关系抽取研究现状 从对标注数据的依赖程度可以将关系抽取的方法分为以下三类： （1）有监督的学习方法 把关系 抽取任务当做一个分类问题，从训练数据中提取有效的特征并训练各种分类模型 进行关系预测。有监督学习可以从标注好的训练数据中学习到各种特征，但是这种方法需要大量的人工标注语料，比较耗时耗力，但预测	关系抽取技术的研究 赵立鹏;张若伟; -《计算机产品与流通》- 2018-09-15（是否引证：否） 1.体的对应属性。现有主流的关系抽取技术分为有监督的学习方法、半监督的学习方法和无监督的学习方法三种:有监督的学习方法将关系抽取任务当做分类问题,根据训练数据设计有效的特征,从而学习各种分类模型,然后使用训练好的分类器预测关系。该方法的问题在于需要大量的人工标注训练语料,而语料标注工作通常非常耗时耗力。半监督的学习
4	此处有 134 字相似 需要大量的人工标注语料，比较耗时耗力，但预测效果较好。 （2）半监督的学习方法 主要是用Bootstrapping 方法进行关系抽取。先设定若干种子实例，再迭代地从文本中抽取相应的关系模板和更多关系实例。 （3）无监督的学习方法 无监督学习是假设有相同语义的命名实体且拥有相似的上下文信息，可根据每个命名实体对的上下文信息来代表它们的语义关系，并对所有实体对的语义进行聚类。 自从2000年Miller等提出基于句法解析增强的方法来实现关系抽取[Miller et al., 2000]后，越来越	关系抽取技术的研究 赵立鹏;张若伟; -《计算机产品与流通》- 2018-09-15（是否引证：否） 1.于需要大量的人工标注训练语料,而语料标注工作通常非常耗时耗力。半监督的学习方法主要采用进行关系抽取。对于要抽取的关系,该方法首先手工设定若干种子实例,然后迭代地从数据从抽取关系对应的关系模板和更多的实例。无监督的学习方法假设拥有相同语义关系的实体对拥有相似的上下文信息。因此可以利用每个实体对对应上下文信息来代表该实体对的语义关系,并对所有实体对的语义关系进行聚类。二、研究内容信息抽取的主要目的是将非结构化或半结构化描述的自然语言文本转化成结构化数据,关系抽取是其重要的子任务,主要负
5	此处有 44 字相似 通过递归神经网络学习到句子的词汇特征、句法特征、语义特征再用于关系分类和抽取。[Zeng et al., 2014]提出 使用卷积神经网络（Convolutional Neural Network，简称CNN） 来解决关系抽取问题，采用词向量和词的相对位置作为卷积神经网络的输入，通过卷积、池化、非线性计算等	多特征融合的细粒度图像检索算法 王虹 -《大连理工大学硕士论文》- 2018-03-27（是否引证：否） 1.达图像层次信息的特征。2012 年，Krizhevsky A.等人[10]在 Image Net 图像识别大赛上利用卷积神经网络（Convolutional Neural Network, CNN）取得了极好的对象识别分类效果。通过训练卷积滤波器，CNN 可以自动学习复杂的特征并且达到很好的效果。其中每一层的中

	操作得到句子特征并用于关	
6	此处有 42 字相似 . 将训练好的模型包成app，封装到Docker容器中并部署到Kubernetes集群上，构建一套信息抽取服务系统。 1.4 本文的组织结构 本文的各章节内容安排如下： 第一章绪论。本章介绍了论文的 项目背景，信息抽取任务中的两个子任务命名实体识别和关系抽取，国内外在这两个方向上的研究成果，本文所做的主要工作和论文的组	知识图谱中子图查询技术研究 李新锋 -《华中科技大学硕士论文》- 2017-05-01（是否引证：否） 1.论 文使过滤能力强的顶点优先匹配。（3）提出了剪枝能力更强的剪枝判定规则，可以把不匹配的子图尽早剪枝。1.4 论文的组织结构 论文的章节安排如下： 第一章介绍了本文的研究背景，研究现状，本文的主要工作和论文的组织结构。 第二章介绍图扩展和子图匹配问题的相关基础知识和概念， F类高效率功率放大器的研究与设计 蔡炜波 -《国防科学技术大学硕士论文》- 2016-11-01（是否引证：否） 1.于双频带 F 类功放面临谐波频率冲突问题，巧妙地设计了 F/IF 混合模式的双频带高效率功放。1.3.2 本文的组织结构本文的各章节内容安排如下：第一章，主要介绍 F 类谐波控制功率放大器的研究背景及研究意义，详细阐述国内外单管 F 类功放和双频功放的研究现状，并对本文的研究内
7	此处有 39 字相似 如下： 第一章绪论。本章介绍了论文的项目背景，信息抽取任务中的两个子任务命名实体识别和关系抽取，国内外在这两个方向上的 研究成果，本文所做的主要工作和论文的组织结构。 第二章技术综述。本章详细介绍 了论文涉及的相关技术，包括词向量、BERT模型、双向长短时记忆模型、深度残差网络、远程监督、深度学习框架Tensorfl	知识图谱中子图查询技术研究 李新锋 -《华中科技大学硕士论文》- 2017-05-01（是否引证：否） 1.图尽早剪枝。1.4 论文的组织结构 论文的章节安排如下： 第一章介绍了本文的研究背景，研究现状，本文的主要工作和论文的组织结构。 第二章介绍图扩展和子图匹配问题的相关基础知识和概念，详细介绍子图匹配中查询图、数据图的定义；并对图扩展、子图同构、子图匹配的

指　标
疑似剽窃文字表述
1. 1.4 本文的组织结构 本文的各章节内容安排如下： 第一章绪论。本章介绍了论文的

7. 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第7部分

总字数：7372

相似文献列表

去除本人已发表文献复制比：17%(1256)　　文字复制比：17%(1256)　　疑似剽窃观点：(0)

1	11881446_肖朴_基于大数据的网络安全态势预测方法研究 肖朴 -《学术论文联合比对库》- 2017-10-28	3.8%（282） 是否引证：否
2	Docker与KVM之间的区别 - huaweitman的专栏 - 博客频道 - CSDN.NET -《网络（http://blog.csdn.net）》- 2017	3.7%（271） 是否引证：否
3	Docker整理之简介（一） - timeless的博客 - 博客频道 - CSDN.NET -《网络（http://blog.csdn.net）》- 2017	3.7%（271） 是否引证：否
4	云计算与大数据背后的核心技术研究 -《学术论文联合比对库》- 2017-09-04	3.6%（265） 是否引证：否
5	【深入浅出容器云】关于容器云你不得不知的十大特性 - 云计算 -《网络（http://www.ciotimes.）》- 2016	3.5%（258） 是否引证：否
6	2_肖朴_基于大数据的网络安全态势预测方法研究 肖朴 -《学术论文联合比对库》- 2017-10-16	3.4%（247） 是否引证：否

7	docker【1】docker简介（入门知识）- 既认准这条路，又何必在意要走多久 - 博客频道 - CSDN.NET	3.3%（245）
	-《网络（http://blog.csdn.net）》- 2017	是否引证：否
8	云计算	3.3%（242）
	-《学术论文联合比对库》- 2015-09-30	是否引证：否
9	信息学院-张晓燕	2.8%（205）
	张晓燕 -《学术论文联合比对库》- 2017-10-10	是否引证：否
10	11942841_肖朴_基于大数据的网络安全态势预测方法研究	2.7%（198）
	肖朴 -《学术论文联合比对库》- 2017-12-01	是否引证：否
11	Docker容器及Spring Boot微服务应用 - 小飞侠的博客 - 博客频道 - CSDN.NET	2.7%（196）
	-《网络（http://blog.csdn.net）》- 2017	是否引证：否
12	k8s docker集群搭建	2.2%（160）
	-《互联网文档资源（https://wenku.baidu.）》- 2018	是否引证：否
13	“智慧校园”-教育服务云平台的研究	1.8%（130）
	王芳(导师：赵振东;程建军) -《华北电力大学硕士论文》- 2018-06-01	是否引证：否
14	1334061002_王毫_081202_计算机软件与理论_王晓明	1.5%（113）
	王毫 -《学术论文联合比对库》- 2016-04-23	是否引证：否
15	1334061002_王毫_081202_计算机软件与理论_王晓明	1.5%（113）
	王毫 -《学术论文联合比对库》- 2016-04-23	是否引证：否
16	Kubernetes将统治云端	1.3%（97）
	张迟第 -《人民邮电》- 2017-12-14	是否引证：否
17	基于Word2Vec的主题爬虫研究与实现	1.2%（89）
	宋健(导师：赫枫龄) -《吉林大学硕士论文》- 2018-04-01	是否引证：否
18	社区问答系统中的专家推荐方法研究	1.0%（74）
	孙吉庆(导师：王健) -《大连理工大学硕士论文》- 2017-05-01	是否引证：否
19	基于深度学习的中文词表示学习技术研究	0.9%（64）
	庄航(导师：周学海) -《中国科学技术大学博士论文》- 2018-08-22	是否引证：否
20	浅析基于微服务架构的测试云平台的移动应用兼容性测试实现	0.7%（49）
	张倩;刘侃;周宇; -《科技资讯》- 2018-10-03	是否引证：否
21	基于深度学习的森林资源信息估测模型研究	0.7%（49）
	汪康宁(导师：马庆勋) -《西安科技大学硕士论文》- 2018-06-01	是否引证：否
22	基于深度学习的三维模型补全技术研究	0.7%（48）
	满婧琦(导师：魏小鹏) -《大连理工大学硕士论文》- 2018-06-08	是否引证：否
23	基于事件框架的生物信息抽取的研究	0.4%（29）
	王安然(导师：王健) -《大连理工大学硕士论文》- 2018-05-01	是否引证：否

原文内容		相似内容来源
1	此处有 33 字相似 提出了一种新的词表示方法：分布式表示法（Distributed Reputation），即用向量来表示一个词。词的分布式 表示主要分为两类：基于矩阵的分布表示和基于神经网络的分布表示。 2.1.1 基于矩阵的分布表示 基于矩阵的分布表示主要是构建“词-上下文”矩阵，从矩阵中获取词的分布表示。矩阵的每一行	基于Word2Vec的主题爬虫研究与实现 宋健 -《吉林大学硕士论文》- 2018-04-01（是否引证：否） 1.似上下文的词汇语义也相近。分布假说是统计语言学的基础，基于分布假说获得词向量的方法主要有两种：基于矩阵的分布表示和基于神经网络的分布表示。基于矩阵的分布表示通常用矩阵的一行表示一个词的词向量，但是这种方式维度较高，往往需要利用奇异值分解（SVD）[46]等方法对矩阵降维 基于深度学习的中文词表示学习技术研究 庄航 -《中国科学技术大学博士论文》- 2018-08-22（是否引证：否） 1.现有词表示学习方法 13 2.1 引言 13 2.2 词表示方法 13 2.2.1 离散表示 13 2.2.2 基于矩阵的分布式表示 14 2.2.3 基于神经网络的分布式表示 15 2.3 小结 23第3章 基于笔画的汉字编码 25 3.1 引言 25 3.2 基于深度学习的中文词表示学习技术研究 庄航 -《中国科学技术大学博士论文》- 2018-08-22（是否引证：否） 1.现有词表示学习方法 13 2.1 引言 13 2.2 词表示方法 13 2.2.1 离散表示 13 2.2.2 基于矩阵的分布式表示 14 2.2.3 基于神经网络的分布式表示 15 2.3 小结 23第3章

		基于笔画的汉字编码 25 3.1 引言 25 3.2
2	此处有 61 字相似 ion），即用向量来表示一个词。词的分布式表示主要分为两类：基于矩阵的分布表示和基于神经网络的分布表示。 2.1.1 基于矩阵的分布表示 基于矩阵的分布表示主要是构建“词-上下文”矩阵，从矩阵中获取词的分布表示。矩阵的每一行都表示一个词，列表示上下文信息。常见的上下文有： （1）文档，即“词-文档”矩阵。 （2）上下文的单词，即“词-词”矩阵。	基于Word2Vec的主题爬虫研究与实现 宋健 -《吉林大学硕士论文》- 2018-04-01（是否引证：否） 1.文的词汇语义也相近。分布假说是统计语言学的基础，基于分布假说获得词向量的方法主要有两种：基于矩阵的分布表示和基于神经网络的分布表示。基于矩阵的分布表示通常用矩阵的一行表示一个词的词向量，但是这种方式维度较高，往往需要利用奇异值分解（SVD）[46]等方法对矩阵降维。基于神经网络的分布表示
3	此处有 72 字相似 ennington et al., 2014]。 2.1.2 基于神经网络的分布表示 基于神经网络的分布表示一般通过神经网络训练语言模型得到，[Bengio et al., 2006]提出了一种神经网络语言模型（Neural Network Language Model，简称NNLM），考虑对语言模型进行建模。但是NNLM中词向量只是语言模型训练得到的副产物，并没有指出哪一套向	基于深度学习的中文词表示学习技术研究_庄航 -《中国科学技术大学博士论文》- 2018-08-22（是否引证：否） 1.算法[42][43：1来弥补，进一步降低数据稀疏带来的影响。同时n元模型没有考虑单词之间的语义信息。 '2.神经网络语言模型2003年Bengio等人提出了神经网络语言模型（NNLM，Neural Network Lan-gaugeModelp'在NNLM学习语言模型的过程中得到了稠密的低维词向量。虽然神经网络语言模型同样 基于深度学习的中文词表示学习技术研究_庄航 -《中国科学技术大学博士论文》- 2018-08-22（是否引证：否） 1.算法[42][43：1来弥补，进一步降低数据稀疏带来的影响。同时n元模型没有考虑单词之间的语义信息。 '2.神经网络语言模型2003年Bengio等人提出了神经网络语言模型（NNLM，Neural Network Lan-gaugeModelp'在NNLM学习语言模型的过程中得到了稠密的低维词向量。虽然神经网络语言模型同样
4	此处有 47 字相似 本的后续文本。引入这个任务是为了让模型更好地学习连续的文本片段之间的关系。 2.3 双向长短时记忆模型 2.3.1 循环神经网络 循环神经网络（Recurrent Neural Networks，简称RNN） [Rumelhart et al. 1986]指的是一个序列当前的输出和之前的输出存在某种关联的神经网络。具体的表现形式	11881446_肖朴_基于大数据的网络安全态势预测方法研究 肖朴 -《学术论文联合比对库》- 2017-10-28（是否引证：否） 1.M 进行预训练，判别模型包括深层感知器（deep MLP）、卷积神经网络（CNN）[42]、循环神经网络（Recurrent Neural Network，RNN）[43]等等。值得一提的是，虽然受限玻尔兹曼机、自编码器、深层信念网络、深层玻尔兹曼机都被归类为生成模型，但由于模型 11942841_肖朴_基于大数据的网络安全态势预测方法研究 肖朴 -《学术论文联合比对库》- 2017-12-01（是否引证：否） 1.M需要 RBM 进行预训练，判别模型包括深层感知器（deep MLP）、卷积神经网络（CNN）[47]、循环神经网络（Recurrent Neural Network，RNN）[48][49][50]等等。值得一提的是，虽然受限玻尔兹曼机、自编码器、深层信念网络、深层玻尔兹曼机都被归类为生成 基于事件框架的生物信息抽取的研究 王安然 -《大连理工大学硕士论文》- 2018-05-01（是否引证：否） 1.. 2.3 The model of CNN 基于事件框架的生物信息抽取的研究- 10 -2.2.3 循环神经网络循环神经网络（Recurrent Neural Network, RNN）是包含反馈连接的

		，用于处理序列信息数据的神经网络[34]。不同于卷积网络允许跨时间或区域的共享参数，循环神经网络
5	此处有 76 字相似 经网络（Recurrent Neural Networks，简称RNN）[Rumelhart et al. 1986]指的是一个序列当前的输出和之前的输出存在某种关联的神经网络。具体的表现形式为网络会对前面的信息进行记忆，保存在网络的内部状态中，并应用于当前输出的计算中。 所以循环神经网络常常被用于对序列数据进行建模。循环神经网络的结构图如图2.3所示。 图2.3 循环神经网络的结构图	社区问答系统中的专家推荐方法研究_孙吉庆 -《大连理工大学硕士论文》- 2017-05-01（是否引证：否） 1.etwork），它主要处理序列数据，在序列标注、命名实体识别、序列到序列的模型等很多场景都有应用。RNN 指的是一个序列当前的输出与之前的输出也有关。具体的表现形式为网络会对前面的信息进行记忆，保存在网络的内部状态中，并应用于当前输出的计算中，即隐含层之间的节点不再是无连接而是有连接的，并且隐含层的输入不仅包含输入层的输出还包含上一时刻隐含层的输出。
6	此处有 29 字相似 组提取特征（从所有出现该三元组的句子中），构造训练数据；3.采用多类别逻辑回归进行分类。 在测试使用时，先使用命名实体识别模型对目标文本中的命名实体进行标注，抽取其中的命名实体对和特征。如果多个句子的命名实体对一样，则把它们的特征合并在同一个特征向量中，再利用逻辑回归分类器对关系名称进行识别。	基于事件框架的生物信息抽取的研究_王安然 -《大连理工大学硕士论文》- 2018-05-01（是否引证：否） 1.是文本中实体的复杂关系，目的在于获取这些实体之间细粒度的语义关联。目前主流的方法将事件抽取分为以下几个步骤：首先，识别文本中的生物医学实体，也就是命名实体识别任务；其次，识别事件的触发词，从而获取事件位置与类型；然后，抽取实体与触发词的关系，检测事件元素；最后，根据识
7	此处有 198 字相似 架Tensorflow，Flask框架，Docker容器技术以及Kubernetes集群等。 2.6.1 深度学习框架 Tensorflow Tensorflow既是一个实现机器学习算法的接口，也是执行机器学习算法的框架。它前端支持Python、C++、Go、Java等多种开发语言，后端使用C++、CUDA等写成。除了执行深度学习算法，Tensorflow还可以用来实现很多传统机器学习算法，如逻辑回归、随机森林等。 Tensorflow使用数据流式图来规划计算流程，可以把计算映射到不同的硬件甚至是操作系统。 Tensorflow的计算可以表示为有状态的数据流式图，可以让用户简单地实现并行计算，同时使用不用的硬件资源进行训练，同	2_肖朴_基于大数据的网络安全态势预测方法研究_肖朴 -《学术论文联合比对库》- 2017-10-16（是否引证：否） 1.升；第二是回馈社区，Google希望让这个优秀的工具得到更多的应用，从整体上提高学术界乃至工业界使用深度学习的效率。Tensorflow既是一个实现机器学习算法的接口，同时也是执行机器学习算法的框架。它前端支持Python、C++、Java等多种开发语言，后端使用C++、CUDA等写成。Tensorflow实现的算法可以在众多异构的系统上方便地移植，比如Android手机、iPhone、普通的CPU服务器 2.的算法可以在众多异构的系统上方便地移植，比如Android手机、iPhone、普通的CPU服务器、乃至大规模GPU集群。除了执行深度学习算法，Tensorflow还可以用来实现很多其他算法，包括线性回归、逻辑回归、随机森林等。Tensorflow建立的大规模深度学习模型的应用场景也非常广，包括语音识别、自然语言处理，计算机视觉、机器人控制、信息抽取、分子活动预测等 3.制、信息抽取、分子活动预测等，使用Tensorflow开发的模型也在这些领域获得了最前沿的成果。Tensorflow使用数据流式图来规划计算流程，它可以将计算映射到不同的硬件和操作系统平台。凭借着统一的架构，Tensorflow可以方便的部署到各种平台，大大简化了真实场景中应用机器学习的难度。使用Tensor 11881446_肖朴_基于大数据的网络安全态势预测方法研究_肖朴 -《学术论文联合比对库》- 2017-10-28（是否引证：否） 1.馈社区，Google希望让这个优秀的工具得到更多的应用，从整体上提高学术界乃至工业界使用深度学习的效率。Tensorflow既是一个实现机器学习算法的接口，同时也是执行机器学习算法的框架。它前端支持Python、

		C++、Java 等多种开发语言，后端使用C++、CUDA 等写成。Tensorflow 实现的算法可以在众多异构的系统上方便地移植，比如Android 手机、iPhone、普通的 2.异构的系统上方便地移植，比如Android 手机、iPhone、普通的CPU 服务器、乃至大规模GPU集群。除了执行深度学习算法，Tensorflow还可以用来实现很多其他算法，包括线性回归、逻辑回归、随机森林等。Tensorflow建立的大规模深度学习模型的应用场景也非常广，包括语音识别、自然语言处理，计算机视觉、机器人控制、信息抽取、分子活动预 3.测等，使用Tensorflow开发的模型也在这些领域获得了最前沿的成果。第49页Tensorflow使用数据流式图来规划计算流程，它可以将计算映射到不同的硬件和操作系统平台。凭借着统一的架构，Tensorflow可以方便的部署到各种平台，大大简化了真实场景中应用机器学习的难度。使用 11942841_肖朴_基于大数据的网络安全态势预测方法研究 肖朴 -《学术论文联合比对库》- 2017-12-01（是否引证：否） 1.社区，Google 希望让这个优秀的工具得到更多的应用，从整体上提高学术界乃至工业界使用深度学习的效率。Tensorflow 既是一个实现机器学习算法的接口，同时也是执行机器学习算法的框架。它前端支持 Python、C++、Java 等多种开发语言，后端使用 C++、CUDA等写成。Tensorflow 实现的算法可以在众多异构的系统上方便地移植，比如 Android第 50 页手机、iPho 2.便地移植，比如 Android第 50 页手机、iPhone、普通的 CPU 服务器、乃至大规模 GPU 集群。除了执行深度学习算法，Tensorflow 还可以用来实现很多其他算法，包括线性回归、逻辑回归、随机森林等。Tensorflow 建立的大规模深度学习模型的应用场景也非常广，包括语音识别、自然语言处理，计算机视觉、机器人控制、信息抽取、分子活动预 基于深度学习的森林资源信息估测模型研究 汪康宁 -《西安科技大学硕士论文》- 2018-06-01（是否引证：否） 1.sorflow 深度学习架构 2015 年 11 月 9 日，Google 发布并开源了人工智能系统 Tensorflow。它既是一个实现机器学习算法的接口，也是执行机器学习算法的框架。目前可以在 Windows、Linux与 Mac OS 系统上运行。本文所进行的所有实验均在其上完成。西安科 基于深度学习的三维模型补全技术研究 满婧琦 -《大连理工大学硕士论文》- 2018-06-08（是否引证：否） 1.Tensor Flow 官方提供了核心代码的 C++接口，此外还提供了 Python、Java 和 Go 等接口。它前端支持 Python、C++、Java 等多种开发语言，后端使用 C++、CUDA 等写成。通过 SWIG(Simplified Wrapperand Interface Generator)实现 C/C+
8	此处有 202 字相似 流式图来规划计算流程，可以把计算映射到不同的硬件甚至是操作系统。Tensorflow的计算可以表示为有状态的数据流式图， 可以让用户简单地实现并行计算，同时使用不用的硬件资源进行训练，同步或异步地更新全局共享的模型参数和状态。	信息学院-张晓燕 张晓燕 -《学术论文联合比对库》- 2017-10-10（是否引证：否） 1.涌来无法运行计算时，可利用热迁移创建多个容器处理运算任务，调节信息数据处理峰谷，配置管理负载均衡比例，降低应用延迟。2.3 docker的运用Docker是一个开源的引擎，可以轻松的为任何应用创建一个轻量级的、可移植的、自给自足的容器。开发者在笔记本上编译测试通过的容器可以批量地在生产环境中部署，包括

2.6.2 Docker容器技术

Docker是一个开源的引擎，可以轻松地为任何应用创建一个轻量级的、可移植的、自给自足的容器。开发者编译测试通过的容器可以批量地在生产环境中部署，包括VMs（虚拟机）、bare metal、OpenStack集群和其他基础应用平台。

Docker跟传统的虚拟
化方式相比具有很多优势，其优势主要体现在：

1. 更高效的虚拟化：Docker容器的运行不需要额外的hyperviso

VMs（虚拟机）、bare metal、OpenStack 集群和其他的基础应用平台。2.3.1 虚拟化的含义虚拟化我们可以简单的理解为一种资源管理方式。有如下几种虚拟化的方式：1.完全虚拟化：对底层硬件实现完全的

“智慧校园”-教育服务云平台的研究 王芳 -《华北电力大学硕士论文》- 2018-06-01（是否引证：否）

1.ring + Structs + Hibernate）框架数据能够帮助实现这些应用系统的整合，实现资源共享。2.4 Docker 容器技术Docker 是一个开源的引擎，可以轻松的为任何应用创建一个轻量级的、可移植的、自给自足的容器。开发者在笔记本上编译测试通过的容器可以批量地在生产环境中部署，包括 VMs（虚拟机）、bare metal、Open Stack 集群和其他的基华北电力大学硕士学位论文9础应用平台。Docker 通常用于如下场景：（1）web 应用的自

1334061002_王耄_081202_计算机软件与理论_王晓明 王耄 -《学术论文联合比对库》- 2016-04-23（是否引证：否）

1.2.4.1 什么是DockerDocker是一个开源的引擎，是一个重新定义了程序开发测试、交付和部署过程的开放平台，可以轻松的为任何应用创建一个轻量级的、可移植的、自给自足的容器。开发者在PC上编译测试通过的容器可以批量地在生产环境中部署，包括VMs（虚拟机）、bare metal、OpenStack集群和其他的基础应用平台。Docker的核心底层技术是LXC（Linux Container），Docker在其上面加了薄薄的一层，添加了许多有用的功能。Doc

1334061002_王耄_081202_计算机软件与理论_王晓明 王耄 -《学术论文联合比对库》- 2016-04-23（是否引证：否）

1.2.4.1 什么是DockerDocker是一个开源的引擎，是一个重新定义了程序开发测试、交付和部署过程的开放平台，可以轻松的为任何应用创建一个轻量级的、可移植的、自给自足的容器。开发者在PC上编译测试通过的容器可以批量地在生产环境中部署，包括VMs（虚拟机）、bare metal、OpenStack集群和其他的基础应用平台。Docker的核心底层技术是LXC（Linux Container），Docker在其上面加了薄薄的一层，添加了许多有用的功能。Doc

2_肖朴_基于大数据的网络安全态势预测方法研究 肖朴 -《学术论文联合比对库》- 2017-10-16（是否引证：否）

1.护两套程序的成本。Tensorflow的计算可以表示为有状态的数据流逝图，对于大规模的神经网络训练，Tensorflow可以让用户简单地实现并行计算，同时使用不同的硬件资源进行训练，同步或异步地更新全局共享的模型参数和状态。如今，包括优步、Twitter、京东、小米等国内外科技公司也纷纷加入了使用Tensorflow的行列，正如谷歌在Tens

11881446_肖朴_基于大数据的网络安全态势预测方法研究 肖朴 -《学术论文联合比对库》- 2017-10-28（是否引证：否）

1.Tensorflow的计算可以表示为有状态的数据流逝图，对于大规模的神经网络训练，Tensorflow可以让用户简单地实现并行计算，同时使用不同的硬件资源进行训练，同步或异步地更新全局共享的模型参数和状态。如今，包括优步、Twitter、京东、小米等国内外科技公司也纷纷加入了使用Tensorflow 的行列，正如谷歌

		11942841_肖朴_基于大数据的网络安全态势预测方法研究 肖朴 -《学术论文联合比对库》- 2017-12-01（是否引证：否）
		1.套程序的成本。Tensorflow 的计算可以表示为有状态的数据流逝图，对于大规模的神经网络训练，Tensorflow 可以让用户简单地实现并行计算，同时使用不同的硬件资源进行训练，同步或异步地更新全局共享的模型参数和状态。如今，包括优步、Twitter、京东、小米等国内外科技公司也纷纷加入了使用Tensorflow的行列，正如谷歌在 Te
9	此处有 71 字相似 enStack集群和其他基础应用平台。 Docker跟传统的虚拟化方式相比具有很多优势，其优势主要体现在： 1. 更 高效的虚拟化：Docker容器的运行不需要额外的hypervisor支持，它是内核级的虚拟化，因此可以实现更高的性能和效率。 2. 更安全的 运行应用：Docker容器内运行的进程完全与系统隔离，一些恶意行为对系统造成的影响不会波及宿主系统。 3. 更快速的交	Docker与KVM之间的区别 - huaweitman的专栏 - 博客频道 - CSDN.NET -《网络（http://blog.csdn.net）》-（是否引证：否）
		1.是如何创建和工作的。 Docker 容器很轻很快！容器的启动时间是秒级的，大量地节约开发、测试、部署的时间。 2.2 更高效的虚拟化 Docker 容器的运行不需要额外的 hypervisor 支持，它是内核级的虚拟化，因此可以实现更高的性能和效率。 2.3 更轻松的迁移和扩展 Docker 容器几乎可以在任意的平台上运行，包括物理机、虚拟机、公有云、私有云、个人电脑、服务器等。这种兼
		Docker整理之简介（一）- timeless的博客 - 博客频道 - CSDN.NET -《网络（http://blog.csdn.net）》-（是否引证：否）
		1.应用程序是如何创建和工作的。 Docker 容器很轻很快！容器的启动时间是秒级的，大量地节约开发、测试、部署的时间。 更高效的虚拟化 Docker 容器的运行不需要额外的 hypervisor 支持，它是内核级的虚拟化，因此可以实现更高的性能和效率。 更轻松的迁移和扩展 Docker 容器几乎可以在任意的平台上运行，包括物理机、虚拟机、公有云、私有云、个人电脑、服务器等。这种
		云计算与大数据背后的核心技术研究 -《学术论文联合比对库》- 2017-09-04（是否引证：否）
		1.创建和工作的。Docker容器很轻很快，容器的启动时间是秒级的，因此，可以大量地节约开发、测试、部署的时间。第二，更高效的虚拟化。Docker容器的运行不需要额外的Hypervisor支持，它是内核级的虚拟化，因此可以实现更高的性能和效率。第三，更轻松的迁移和扩展。Docker容器几乎可以运行在物理机、虚拟机、公有云、私有云、个人计算机、服务器等任意的平台。这种兼容性可以
		【深入浅出容器云】关于容器云你不得不知的十大特性 - 云计算 -《网络（http://www.ciotimes.）》-（是否引证：否）
		1.其有用，例如在构建私有云或PaaS。当然，当你想在有限的资源里部署更多的应用时，Docker对于中小型的部署也非常有用。Docker 容器的运行不需要额外的 hypervisor 支持，它是内核级的虚拟化，因此可以实现更高的性能和效率。 7.容器的启动是（毫）秒级的 通常，如果要在一台服务器上运行多个任务，传统的方法是将其划分为多个虚拟机，使用每个虚拟机来运行一个任
		Docker容器及Spring Boot微服务应用 - 小飞侠的博客 - 博客频道 - CSDN.NET -《网络（http://blog.csdn.net）》-（是否引证：否）
		1.以直接使用这个容器来部署代码。 Docker 容器很轻很快！容器的启动时间是秒级的，大量地节约开发、测

		试、部署的时间。 Docker 容器的运行不需要额外的hypervisor(虚拟化管理程序)支持，它是内核级的虚拟化，因此可以实现更高的性能和效率。 Docker 容器几乎可以在任意的平台上运行，包括物理机、虚拟机、公有云、私有云、个人电脑、服务器等。 这种兼容性可以 docker【1】docker简介（入门知识）- 既认准这条路，又何必在意要走多久 - 博客频道 - CSDN.NET - 《网络（http://blog.csdn.net）》- （是否引证：否） 1.解应用程序是如何创建和工作的。 Docker 容器很轻很快！容器的启动时间是秒级的，大量地节约开发、测试、部署的时间。 Docker 容器的运行不需要额外的hypervisor 支持，它是内核级的虚拟化，因此可以实现更高的性能和效率。 Docker 容器几乎可以在任意的平台上运行，包括物理机、虚拟机、公有云、私有云、个人电脑、服务器等。 这种兼容性可以 信息学院-张晓燕 张晓燕 -《学术论文联合比对库》- 2017-10-10（是否引证：否） 1.器负载方面，如果你单独开一个虚拟机，那么虚拟机会占用空闲内存的，docker部署的话，这些内存就会被有效的利用起来。而且Docker容器的运行不需要额外的hypervisor支持，它是内核级的虚拟化，因此可以实现更高的性能和效率。作为一种新兴的虚拟化方式，Docker跟传统的虚拟化方式相比具有众多的优势。首先，Docker容器的启动可以在秒级实现， 云计算 -《学术论文联合比对库》- 2015-09-30（是否引证：否） 1.部署的时间。(2) 更高效的虚拟化由于Docker是内核级的虚拟化，直接使用宿主机的Linux Kernel,不需要额外的hypervisor支持，因此可以实现更高的性能和效率。(3) 更轻松的迁移和扩展Docker容器几乎可以在任意的平台上运行，包括物理机、虚拟机、公有云、私有云、个人电脑、
10	此处有 114 字相似 Docker容器内运行的进程完全与系统隔离，一些恶意行为对系统造成的影响不会波及宿主系统。 3. 更快速的交付和部署： 开发者可以使用一个标准的镜像来构建一套开发容器，开发完成之后，运维人员可以直接使用这个容器来部署代码、Docker可以快速创建容器，快速迭代应用程序，并使整个过程全程可见，使团队其他成员更容易理解应用程序是如何创建和工作的。 4. 更轻松的迁移和扩展：Docker容器几乎可以在任意平台上运行，包括物理机、虚拟机、公有云、私有云、服务器等。这种兼	Docker与KVM之间的区别 - huaweitman的专栏 - 博客频道 - CSDN.NET - 《网络（http://blog.csdn.net）》- （是否引证：否） 1. 2.1 更快速的交付和部署 对开发和运维（devop）人员来说，最希望的就是一次创建或配置，可以在任意地方正常运行。 开发者可以使用一个标准的镜像来构建一套开发容器，开发完成之后，运维人员可以直接使用这个容器来部署代码。 Docker 可以快速创建容器，快速迭代应用程序，并让整个过程全程可见，使团队中的其他成员更容易理解应用程序是如何创建和工作的。 Docker 容器很轻很快！容器的启动时间是秒级的，大量地节约开发、测试、部署的时间。 2.2 更高效的虚拟化 Doc Docker整理之简介（一）- timeless的博客 - 博客频道 - CSDN.NET - 《网络（http://blog.csdn.net）》- （是否引证：否） 1.的优势。 更快速的交付和部署 对开发和运维（devop）人员来说，最希望的就是一次创建或配置，可以在任意地方正常运行。 开发者可以使用一个标准的镜像来构建一套开发容器，开发完成之后，运维人员可以直接使用这个容器来部署代码。 Docker 可以快速创建容器，快速迭代应用程序，并让整个过程全程可见，使团队中的其他成员更容易理解应用程序是如何创建和工作的。 Docker 容器很轻很快！容器的启动时间是秒级的，大量地节约开发、测试、部署的时间。 更高效

的虚拟化 Docker

docker【1】docker简介（入门知识）- 既认准这条路，又何必在意要走多久 - 博客频道 - CSDN.NET - 《网络（http://blog.csdn.net）》- （是否引证：否）

1.如下几个方面具有较大的优势： 对开发和运维（devop）人员来说，最希望的就是一次创建或配置，可以在任意地方正常运行。 开发者可以使用一个标准的镜像来构建一套开发容器，开发完成之后，运维人员可以直接使用这个容器来部署代码。 Docker 可以快速创建容器，快速迭代应用程序，并让整个过程全程可见，使团队中的其他成员更容易理解应用程序是如何创建和工作的。 Docker 容器很轻很快！容器的启动时间是秒级的，大量地节约开发、测试、部署的时间。 Docker 容器的运行不需要

【深入浅出容器云】关于容器云你不得不知的十大特性 - 云计算 - 《网络（http://www.ciotimes.）》- （是否引证：否）

1.（有的人玩Windows，有的玩Ubuntu，抑或是Mac），导致出错率大大提高以及效率的降低。而使用Docker之后，开发者可以使用一个标准的镜像来构建一套开发环境，开发完成之后，运维人员可以直接使用这个容器来部署代码。 Docker 可以快速创建容器，快速迭代应用程序，并让整个过程全程可见，使团队中的其他成员更容易理解应用程序是如何创建和工作的。 5.更易于微服务架构的实现 微服务采用一组服务的方式来构建一个应用，服务独立部署在不同的进程中，不同服务通过一些轻量级

云计算 -《学术论文联合比对库》- 2015-09-30（是否引证：否）

1.较大的优势。(1) 更快速的交付和部署对开发和运维人员来说，最希望的就是一次创建或配置，可以在任意地方正常运行。开发者可以使用一个标准的镜像来构建一套开发容器，开发完成之后，运维人员可以直接使用这个容器来部署代码。Docker可以快速创建容器，快速迭代应用程序，并让整个过程全程可见，使团队中的其他成员更容易理解应用程序是如何创建和工作的。 Docker容器启动很快，容器的启动时间是秒级的，大量地节约开发、测试、部署的时间。(2) 更高效的虚拟化 由于D

云计算与大数据背后的核心技术研究 -《学术论文联合比对库》- 2017-09-04（是否引证：否）

1.er的优势主要体现在以下方面。第一，更快速的交付和部署。开发和运维人员最希望一次创建或配置，可以在任意地方正常运行。开发者可以使用一个标准的镜像来构建一套开发容器，开发完成之后，运维人员可以直接使用这个容器来部署代码。Docker可以快速创建容器，快速迭代应用程序，并让整个过程全程可见，使团队中的其他成员更容易理解应用程序是如何创建和工作的。Docker容器很轻很快，容器的启动时间是秒级的，因此，可以大量地节约开发、测试、部署的时间。第二，更高效的虚拟化。

Docker容器及Spring Boot微服务应用 - 小飞侠的博客 - 博客频道 - CSDN.NET - 《网络（http://blog.csdn.net）》-（是否引证：否）

1.如下几个方面具有较大的优势。 对开发和运维（devop）人员来说，最希望的就是一次创建或配置，可以在任意地方正常运行。 开发者可以使用一个标准的镜像来构建一套开发容器，开发完成之后，运维人员

		可以直接使用这个容器来部署代码。 Docker 容器很轻很快！容器的启动时间是秒级的，大量地节约开发、测试、部署的时间。 Docker 容器的运行不需要额外的 hyperv
11	此处有 92 字相似 以快速创建容器，快速迭代应用程序，并使整个过程全程可见，使团队其他成员更容易理解应用程序是如何创建和工作的。 4. 更 轻松的迁移和扩展：Docker容器几乎可以在任意平台上运行，包括物理机、虚拟机、公有云、私有云、服务器等。这种兼容性可以帮助用户很轻松地将应用程序从一个平台直接迁移到另一个平台上。 2.6.3 Kubernetes集群 Kubernetes（简称k8s）是一个自动化容器操作的开源平台，这些操作包括部	Docker与KVM之间的区别 - huaweitman的专栏 - 博客频道 - CSDN.NET - 《网络（http://blog.csdn.net）》- （是否引证：否） 1.er 容器的运行不需要额外的 hypervisor 支持，它是内核级的虚拟化，因此可以实现更高的性能和效率。 2.3 更轻松的迁移和扩展 Docker 容器几乎可以在任意的平台上运行，包括物理机、虚拟机、公有云、私有云、个人电脑、服务器等。这种兼容性可以让用户把一个应用程序从一个平台直接迁移到另外一个。 2.4 更简单的管理 使用 Docker，只需要小小的修改，就可以替代以往大量的更新工作。所有的修改都以增量的方式被分 Docker整理之简介（一） - timeless的博客 - 博客频道 - CSDN.NET - 《网络（http://blog.csdn.net）》- （是否引证：否） 1.Docker 容器的运行不需要额外的 hypervisor 支持，它是内核级的虚拟化，因此可以实现更高的性能和效率。 更轻松的迁移和扩展 Docker 容器几乎可以在任意的平台上运行，包括物理机、虚拟机、公有云、私有云、个人电脑、服务器等。 这种兼容性可以让用户把一个应用程序从一个平台直接迁移到另外一个。 更简单的管理 使用 Docker，只需要小小的修改，就可以替代以往大量的更新工作。所有的修改都以增量的方式被分发和更新 云计算 -《学术论文联合比对库》- 2015-09-30（是否引证：否） 1.使用宿主机的Linux Kernel,不需要额外的hypervisor支持，因此可以实现更高的性能和效率。(3) 更轻松的迁移和扩展Docker容器几乎可以在任意的平台上运行，包括物理机、虚拟机、公有云、私有云、个人电脑、服务器等。这种兼容性可以让用户把一个应用程序从一个平台直接迁移到另外一个。(4) 更简单的管理使用 Docker，只需要小小的修改，就可以替代以往大量的更新工作。所有的修改都以增量的方式被分发 【深入浅出容器云】关于容器云你不得不知的十大特性 - 云计算 - 《网络（http://www.ciotimes.）》- （是否引证：否） 1.，全自动/半自动弹性伸缩，日志监控，滚动升级等都往往成为了容器云的“标配”，你无需再为这些事情而操心。 10.易于扩展和迁移 容器云的Docker 容器几乎可以在任意的平台上运行，包括物理机、虚拟机、公有云、私有云、个人电脑、服务器等。 这种兼容性可以让用户把一个应用程序从一个平台直接迁移到另外一个。容器云的这种特性类似于Java的JVM，Java程序可以运行在任意的安装了JVM的设备上，在迁移和扩展方面变得更加容易。 云计算与大数据背后的核心技术研究 -《学术论文联合比对库》- 2017-09-04（是否引证：否） 1.ocker容器的运行不需要额外的Hypervisor支持，它是内核级的虚拟化，因此可以实现更高的性能和效率。第三，更轻松的迁移和扩展。Docker容器几乎可以运行在物理机、虚拟机、公有云、私有云、个人计算机、服务器等任意的平台。这种兼容性可以让用户把一个应用程序从一个平台直接迁移到另外一个平台。第四，更简单的管

		2.第三，更轻松的迁移和扩展。Docker容器几乎可以运行在物理机、虚拟机、公有云、私有云、个人计算机、服务器等任意的平台。这种兼容性可以让用户把一个应用程序从一个平台直接迁移到另外一个平台。第四，更简单的管理。更新工作以往是大量而复杂的，使用Docker只需要小小的修改即可完成。所有的修改都以增量的方式被分发 Docker容器及Spring Boot微服务应用 - 小飞侠的博客 - 博客频道 - CSDN.NET - 《网络（http://blog.csdn.net）》 - （是否引证：否） 1.容器的运行不需要额外的 hypervisor(虚拟化管理程序)支持，它是内核级的虚拟化，因此可以实现更高的性能和效率。 Docker 容器几乎可以在任意的平台上运行，包括物理机、虚拟机、公有云、私有云、个人电脑、服务器等。 这种兼容性可以让用户把一个应用程序从一个平台直接迁移到另外一个。 特性 容器(Docker) 虚拟机(VM) 启动 秒级 分钟级 硬盘使用 MB级 GB级 性能 接近原生 弱于 系统支 docker【1】docker简介（入门知识） - 既认准这条路，又何必在意要走多久 - 博客频道 - CSDN.NET - 《网络（http://blog.csdn.net）》 - （是否引证：否） 1. Docker 容器的运行不需要额外的 hypervisor 支持，它是内核级的虚拟化，因此可以实现更高的性能和效率。 Docker 容器几乎可以在任意的平台上运行，包括物理机、虚拟机、公有云、私有云、个人电脑、服务器等。 这种兼容性可以让用户把一个应用程序从一个平台直接迁移到另外一个。 使用 Docker，只需要小小的修改，就可以替代以往大量的更新工作。所有的修改都以增量的方式被分发和更新，从而实现自动 浅析基于微服务架构的测试云平台的移动应用兼容性测试实现 张倩;刘侃;周宇; -《科技资讯》- 2018-10-03（是否引证：否） 1.,它是一个操作系统级的虚拟化方法,能为用户提供更多的资源。过去几年里,Docker的快速发展使其成为容器技术的典型代表。Docker可以运行在任意平台上,包括物理机、虚拟机、公有云、私有云、服务器等,这种兼容性使我们不用担心生产环境的操作系统或者平台的差异性,能够很方便地将Docker的映像部署到任何运行Docker的环境中。除
12	此处有 53 字相似 。这种兼容性可以帮助用户很轻松地将应用程序从一个平台直接迁移到另一个平台上。 2.6.3 Kubernetes集群 Kubernetes（简称k8s）是一个自动化容器操作的开源平台，这些操作包括部署、调度和节点集群间扩展。上一节中介绍的Docker容器就可以看作是Kubernetes内部使用的低级别组件。Kubernetes的主要组件有：	Kubernetes将统治云端 张迟第 -《人民邮电》- 2017-12-14（是否引证：否） 1.Kubernetes是一个自动化容器操作的开源平台，这些操作包括部署、调度、节点集群间扩展。如果用户曾经使用Docker容器技术部署容器，那么可以将Docker看成Kubernetes内部使用的低级别组件。Kub
13	此处有 45 字相似 Kubernetes（简称k8s）是一个自动化容器操作的开源平台，这些操作包括部署、调度和节点集群间扩展。上一节中介绍的Docker容器就可以看作是Kubernetes内部使用的低级别组件。Kubernetes的主要组件有：	Kubernetes将统治云端 张迟第 -《人民邮电》- 2017-12-14（是否引证：否） 1.动化容器操作的开源平台，这些操作包括部署、调度、节点集群间扩展。如果用户曾经使用Docker容器技术部署容器，那么可以将Docker看成Kubernetes内部使用的低级别组件。Kubernetes不仅支持Docker，还支持Rocket。$$自从Kubernetes第一版于2015年7月发布以

	1. etcd:高可用存储共享配置和服务发现，作为与minion机器上的flannel配套使用，使每台	来，其功能发生
14	此处有 83 字相似 群间扩展。上一节中介绍的Docker容器就可以看作是Kubernetes内部使用的低级别组件。Kubernetes的主要组件有： 1. etcd:高可用存储共享配置和服务发现，作为与minion机器上的flannel配套使用，使每台minion上运行的docker拥有不同的ip。 2. flannel：网络结构支持。 3. kube-apiserver：无论通过kubectl还是使用remote	k8s docker集群搭建 -《互联网文档资源（https://wenku.baidu.）》- （是否引证：否） 1.ubernetes是google公司基于docker所做的一个分布式集群，有以下组件组成 1. Kubernetes组成组件 etcd: 高可用存储共享配置和服务发现，作为与minion机器上的flannel配套使用，作用是使每台 minion上运行的docker拥有不同的ip段，最终目的是使不同minion上正在运行的docker containner都有一个与别的任意一个containner（别
15	此处有 80 字相似 套使用，使每台minion上运行的docker拥有不同的ip。 2. flannel：网络结构支持。 3. kube -apiserver：无论通过kubectl还是使用remote api直接控制，都要经过apiserver。 4. kube-controller-mana ger：与apiserver交互，保证controller的工作。 5. kube-scheduler：根据特定的调度	k8s docker集群搭建 -《互联网文档资源（https://wenku.baidu.）》- （是否引证：否） 1.任意一个containner（别的minion上运行的docker containner）不一样的IP地址。 kube-apiserver: 不论通过kubectl还是使用remote api 直接控制，都要经过apiserver kube-controller-manager: 对replication controller,endpoints controller,namespace cont

指　标

疑似剽窃文字表述

1. 基于矩阵的分布表示
   基于矩阵的分布表示主要是构建“词-上下文”矩阵，从矩阵中获取词的分布表示。矩阵的每一行都表示一个词，
2. 具体的表现形式为网络会对前面的信息进行记忆，保存在网络的内部状态中，并应用于当前输出的计算中。
3. Tensorflow
   Tensorflow既是一个实现机器学习算法的接口，也是执行机器学习算法的框架。它前端支持Python、C++、Go、Java等多种开发语言，后端使用C++、CUDA等写成。除了执行深度学习算法，Tensorflow还可以用来实现很多传统机器学习算法，如逻辑回归、随机森林等。
   Tensorflow使用数据流式图来规划计算流程，可以把计算映射到不同的硬件甚至是操作系统。
4. 可以让用户简单地实现并行计算，同时使用不用的硬件资源进行训练，同步或异步地更新全局共享的模型参数和状态。
   2.6.2 Docker容器技术
   Docker是一个开源的引擎，可以轻松地为任何应用创建一个轻量级的、可移植的、自给自足的容器。
5. 高效的虚拟化：Docker容器的运行不需要额外的hypervisor支持，它是内核级的虚拟化，因此可以实现更高的性能和效率。
   2. 更安全的
6. 开发者可以使用一个标准的镜像来构建一套开发容器，开发完成之后，运维人员可以直接使用这个容器来部署代码、Docker可以快速创建容器，快速迭代应用程序，并使整个过程全程可见，使团队其他成员更容易理解应用程序是如何创建和工作的。
7. 轻松的迁移和扩展：Docker容器几乎可以在任意平台上运行，包括物理机、虚拟机、公有云、私有云、服务器等。这种兼容性可以帮助用户很轻松地将应用程序从一个平台直接迁移到另一个平台上。
8. Docker容器就可以看作是Kubernetes内部使用的低级别组件。Kubernetes
9. 组件有：

1. etcd:高可用存储共享配置和服务发现，作为与minion机器上的flannel配套使用，使每台minion上运行的docker拥有不同的ip。

10. -apiserver：无论通过kubectl还是使用remote api直接控制，都要经过apiserver。

4. kube-controller-mana

8. 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第8部分 总字数：9201

相似文献列表

去除本人已发表文献复制比：0.4%(40)　文字复制比：0.4%(40)　疑似剽窃观点：(0)

1	基于迁移学习的无参考视频质量评价 张浩;桑庆兵; -《激光与光电子学进展》- 2018-04-27 1	0.4%（40） 是否引证：否

	原文内容	相似内容来源
1	此处有 40 字相似 据预处理模块传入的文本数据和预训练好的词向量；Model_Builder负责对选择好的模型进行初始化，在模型训练完后保存 训练好的模型；Test负责读取测试集数据，将训练好的模型在测试集上进行预测，预测 完成后返回测试数据；Models部分分为NER_Model和RE_Model，分别提供了2种命名实体识别模型和1种关系抽	基于迁移学习的无参考视频质量评价 张浩;桑庆兵; -《激光与光电子学进展》- 2018-04-27 1（是否引证：否） 1.min,(14)式中x^为归一化后输入数据,x为原始视频数据,xmin和xmax分别为原始视频数据中的最小值和最大值。把训练好的模型在测试集上进行预测。首先,预测每个失真视频块的质量得分,然后,通过求取其所有子块的平均分得到整个视频质量的预测得分。为了验证算法的稳健性,采用不同的随

指　标

疑似剽窃文字表述

1. 训练好的模型；Test负责读取测试集数据，将训练好的模型在测试集上进行预测，预测

9. 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第9部分 总字数：2480

相似文献列表

去除本人已发表文献复制比：7.9%(197)　文字复制比：7.9%(197)　疑似剽窃观点：(0)

1	基于灰色神经网络的直升机旋翼故障诊断技术 周鸿灯(导师：高亚东) -《南京航空航天大学硕士论文》- 2018-03-01	6.1%（152） 是否引证：否
2	基于FPGA的目标检测算法加速与实现 吴晋(导师：王东) -《北京交通大学硕士论文》- 2018-06-04	2.1%（52） 是否引证：否
3	基于LSTM融合多CNN的事件图像分类研究 汤华东(导师：侯建军) -《北京交通大学硕士论文》- 2018-05-01	1.7%（41） 是否引证：否
4	010_21401026_李想 李想 -《学术论文联合比对库》- 2017-05-08	1.4%（35） 是否引证：否

	原文内容	相似内容来源
1	此处有 30 字相似 NN模型结构示意图 输入层包括词向量层和位置向量层；卷积层是对输入层输出的文本特征向量做卷积操作来提取特征；残差卷积块 利用残差学习结合卷积神经网络对卷积层的输出进行进一步处理；最 大池化层对残差卷积块的输出进行最大池化操作并取其中的最大值；最后把这些特征传入Softmax层进行计算得到最终的预测结果	基于灰色神经网络的直升机旋翼故障诊断技术 周鸿灯 -《南京航空航天大学硕士论文》- 2018-03-01（是否引证：否） 1.和小波对旋翼故障振动信号进行处理后提取特征，然后把样本特征输入到灰色神经网络模型进行分类，在灰色神经网络模型里，先使用卷积神经网络对输入特征进行组合并获得深层特征，其输出作为灰色关联模型的输入。使用灰色神经网络模型进行旋翼故障诊断的准确率比单一使用灰色模型要高，其中无
2	此处有 40 字相似	基于LSTM融合多CNN的事件图像分类研究 汤华东 -《北京交通大学硕士论文》- 2018-05-01（是否引证：否）

	9 ReLU激活函数 使用ReLU作为激活函数的好处有： 1. 单侧抑制。即使所有的负值都变为0，而正值的数值不变。 正因为ReLU的单侧抑制才使得神经网络中的神经元具有了稀疏激活性，也就是说当模型 的层数增加了N倍后，ReLU神经元的激活率将降低2的N次方倍。 2. 对于线性函数而言，ReLU的表达能力更强，尤其在	1.图2-2中可以看出ReLU函数的特性。ReLU在负区间输出为0，在非负区间输出值为x,这样的特性被称为单侧抑制。正因为单侧抑制的特性，才使得神经网络中的神经元也具有了稀疏激活性，从而使模型更好地挖掘相关特征，拟合训练数据。由于ReLU非负区间的梯度为常数，因此不存在梯度消失问题（Vanishing 基于灰色神经网络的直升机旋翼故障诊断技术 周鸿灯 -《南京航空航天大学硕士论文》- 2018-03-01（是否引证：否） 1.从上图可以看出，Re Lu 函数是分段线性函数，把所有的负值都变为 0，而正值不变，这种操作称为单侧抑制。有了单侧抑制之后，就使得神经网络中的神经元具有稀疏激活性。当训练一个深度分类模型的时候，和目标相关的特征往往就那么几个，因此通过 Re Lu 实现稀疏后的模型能够更好的
3	此处有 92 字相似 经网络中的神经元具有了稀疏激活性，也就是说当模型的层数增加了N倍后，ReLU神经元的激活率将降低2的N次方倍。 2. 对于线性函数而言，ReLU的表达能力更强，尤其在网络很深的情况下。 3. 对于非线性函数而言，ReLU由于非负区间的梯度为常数，所以不存在梯度消失的问题，而且模型的收敛速度维持在一个 比较稳定的状态。 设残差卷积块中的2层卷积神经网络的卷积过滤器分别为和，。第一个卷积过滤器（filter）：。第二个卷	基于灰色神经网络的直升机旋翼故障诊断技术 周鸿灯 -《南京航空航天大学硕士论文》- 2018-03-01（是否引证：否） 1.挖掘相关特征，拟合训练数据。此外，相比于其他激活函数（如 sigmoid 激活函数等），Re Lu 具有以下优势：对于线性函数，Re Lu 的表达能力更强，尤其在深度网络中；对于非线性函数，由于非负区间的梯度为常数，所以不存在梯度消失问题，使得模型的收敛速度维持在一个稳定状态。梯度消失指的是当梯度小于 1 时，预测值与真实值之间的误差每传播一层会衰减一次，如果在深层模型中使用 s 基于FPGA的目标检测算法加速与实现 吴晋 -《北京交通大学硕士论文》- 2018-06-04（是否引证：否） 1.而言,ReLU的表达能力更强。ReLU函数其实就是一个max(0，x),计算代价小很多，所以速度更快。② 对非线性函数而言，ReLU由于非负区间的梯度为常数，因此不会存在梯度消失的问题，使模型的收敛速度维持在一个稳定状态。2)池化层我们利用卷积得到的特征去做分类，可通常卷积得到的特征向量非常的大。例如我们输入的4
4	此处有 35 字相似 出层 模型的输出层是一个全连接的Softmax层，输入为最大池化层中提取出来的m个特征，输出的是通过使用Dropout 进行全连接得到的关系的概率分布。 3.6 本章小结 本章主要介绍了 信息抽取服务系统的需求分析、服务系统的总体设计，对信息服务系统进行了模块划分，对系统中的每个模块进行详细设计以及信息抽取	010_21401026_李想 李想 -《学术论文联合比对库》- 2017-05-08（是否引证：否） 1.交初始化算法。此外据我们所知，所有的初始化工作中，对卷积核的处理都有欠妥当。本文我们将仔细的深入卷积核，并针对卷积核进行额外的处理。2.6 本章小结本章我们深入介绍了卷积神经网络的所有流行的方法：逐层预训练、渐进式训练、Xavier 算法、正交初始化算法以及LSUV初始化算法。

指　标
疑似剽窃文字表述

1. 正因为ReLU的单侧抑制才使得神经网络中的神经元具有了稀疏激活性，也就是说当模型
2. 对于线性函数而言，ReLU的表达能力更强，尤其在网络很深的情况下。

3. 对于非线性函数而言，ReLU由于非负区间的梯度为常数，所以不存在梯度消失的问题，而且模型的收敛速度维持在一个

10. 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第10部分

总字数：15793

相似文献列表

去除本人已发表文献复制比：18.4%(2904)　　文字复制比：18.4%(2904)　　疑似剽窃观点：(0)

1	011_M201570005_聂国平 聂国平 -《学术论文联合比对库》- 2017-05-18	15.2%（2404） 是否引证：否
2	基于卷积神经网络的中文文本分类研究 聂国平 -《学术论文联合比对库》- 2017-04-26	15.2%（2404） 是否引证：否
3	论文Convolutional Naural Networks for Sentence Classification--TensorFlow实现篇 - liuchonge的博客 - 博客频道 - CSDN.NET -《网络（http://blog.csdn.net）》- 2017	13.7%（2163） 是否引证：否
4	Tensorflow实现的CNN文本分类 - somTian的博客 - 博客频道 - CSDN.NET -《网络（http://blog.csdn.net）》- 2017	11.6%（1827） 是否引证：否
5	基于tensorflow的cnn文本分类 - u013713117的专栏 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	11.2%（1772） 是否引证：否
6	Tensorflow版TextCNN主要代码解析 - u013818406的博客 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	11.1%（1752） 是否引证：否
7	CNN情感分析（文本分类）- 萝卜虫的博客 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	9.6%（1519） 是否引证：否
8	卷积神经网络实践 - Abrohambaby的博客 - CSDN博客 -《网络（http://blog.csdn.net）》- 2017	6.9%（1084） 是否引证：否
9	基于深度学习的评论文本情感分类系统设计与实现 周全(导师：维尼拉·木沙江;吐尔地·托合提) -《新疆大学硕士论文》- 2018-06-01	5.3%（843） 是否引证：否
10	基于集成算法的密级文本分类系统设计 顾凯文(导师：孙国梓) -《南京邮电大学硕士论文》- 2018-11-14	5.2%（828） 是否引证：否
11	1049721403027_鞠亮_学术性硕士_信息与通信工程_周祖德 鞠亮 -《学术论文联合比对库》- 2017-06-07	4.9%（772） 是否引证：否
12	基于关键词自学习的中文网页分类技术研究与实现 鞠亮(导师：周祖德) -《武汉理工大学硕士论文》- 2017-03-01	4.9%（770） 是否引证：否
13	纠错： 深度学习模型优化时快速收敛 - Sherry -《网络（http://blog.csdn.net）》- 2017	3.4%（536） 是否引证：否
14	020+1642+刘恒旺 刘恒旺 -《学术论文联合比对库》- 2017-12-18	1.1%（170） 是否引证：否
15	经管院-115107000850-刘恒旺-吴鹏nm 经管院 -《学术论文联合比对库》- 2017-12-27	1.0%（160） 是否引证：否
16	Hadoop分布式集群的自动化容器部署研究 李杰;刘广钟; -《计算机应用研究》- 2016-01-08 1	0.7%（118） 是否引证：否
17	2014262593_尹羿_基于OSGi的政务云组件运行环境研究与实现_软件工程_武君胜 尹羿 -《学术论文联合比对库》- 2017-02-24	0.5%（77） 是否引证：否
18	荣科科技今年跌逾24% 实控人承诺兜底增持 证券时报记者 康殷 -《证券时报》- 2018-09-14	0.4%（67） 是否引证：否
19	docker 标签 -《网络（http://blog.51cto.co）》- 2016	0.4%（62） 是否引证：否
20	煤炭板块利润连续下滑 重庆能源欲转型“卖”咖啡	每经网 -《网络（http://www.nbd.com.c）》- 2016	0.4%（56） 是否引证：否

	原文内容	相似内容来源
1	此处有 56 字相似 ": "煤炭", "start": 35, "end": 37, "type": "IND"}], "text": "《 每日经济新闻》记者注意到，上述评级报告在阐述重庆能源的关注点时提及“煤炭板块盈利能力较弱，拖累公司经营性业务利润	煤炭板块利润连续下滑 重庆能源欲转型“卖”咖啡	每经网 -《网络（http://www.nbd.com.c）》-（是否引证：否） 1.12亿元、360.08亿元、75.74亿元、净利润分别为0.67亿元、0.06亿元、-1.77亿元和-5.02亿元。《每日经济新闻》记者注意到，上述评级报告在阐述重庆能源的关注点时提及“煤炭板块盈利能力较弱，拖累公司经营

	”"} 图4.1 命名实体识别标注数据示例 金力泰吴国政股东金力泰公告，12月13日，公司控股股东吴国政先生协议转让	性业务利润”。 据了解，截至2016年3月底，重庆能源的煤炭可采储量达35.96万吨，重庆本地可采储量为12.82亿吨；重庆地区的

2	此处有 65 字相似
	apped_grads_vars, self.global_step)# saver of the modelself.
	saver = tf.train.Saver(tf.global_variables(), max_to_keep=5)
	
	图4.
	8 基于word2vec的命名实体识别模型结构实现
	
	输入层将输入单词序列和文本的额外特征做嵌入操作，得到输入文本的特征
		论文Convolutional Naural Networks for Sentence Classification--TensorFlow实现篇 - liuchonge的博客 - 博客频道 - CSDN.NET - 《网络（http://blog.csdn.net）》- （是否引证：否）
		1.os.path.exists(checkpoint_dir): os.makedirs(checkpoint_dir) saver = tf.train.Saver(tf.global_variables(),max_to_keep=FLAGS.num_checkpoints) # Write vocabulary保存vocabulary vocab_proces

3	此处有 258 字相似 on进行参数配置，再构建ResCnn模型，设置Adam优化器和学习率并利用计算得到的loss对网络中的参数进行更新。 with tf.Graph().as_default(): session_conf = tf.ConfigProto(allow_soft_placement=FLAGS.allow_soft_placement, log_device_placement=FLAGS.log_device_placement) sess = tf.Session(config=session_conf) with sess.as_default(): cnn = ResCNN(FLAGS.sequence_length, len(datamanager.relations), FLAGS.embedding_dim, 5, list(ma	基于卷积神经网络的中文文本分类研究 聂国平 -《学术论文联合比对库》- 2017-04-26（是否引证：否） 1.ning# ==========with tf.Graph().as_default():session_conf = tf.ConfigProto(allow_soft_placement=FLAGS.allow_soft_placement,log_device_placement=FLAGS.log_device_placement)sess = tf.Session(config=session_conf)with sess.as_default():cnn = TextCNN(sequence_length=x_train.shape[1],num_classes=y_train.shape[1],vocab_ 011_M201570005_聂国平 聂国平 -《学术论文联合比对库》- 2017-05-18（是否引证：否） 1.ning# ==========with tf.Graph().as_default():session_conf = tf.ConfigProto(allow_soft_placement=FLAGS.allow_soft_placement,log_device_placement=FLAGS.log_device_placement)sess = tf.Session(config=session_conf)with sess.as_default():cnn = TextCNN(sequence_length=x_train.shape[1],num_classes=y_train.shape[1],vocab_ Tensorflow实现的CNN文本分类 - somTian的博客 - 博客频道 - CSDN.NET - 《网络（http://blog.csdn.net）》- （是否引证：否） 1.显式创建 Session和Graph可确保在不再需要资源时正确释放资源。 FLAGS = tf.flags.FLAGS with tf.Graph().as_default(): session_conf = tf.ConfigProto(allow_soft_placement=FLAGS.allow_soft_placement,log_device_placement=FLAGS.log_device_placement) sess = tf.Session(config=session_conf) with sess.as_default(): 当优选设备不存在时，allow_soft_placement设置允许TensorFlow回退到具有特定操作的设备上。 论文Convolutional Naural Networks for Sentence Classification--TensorFlow实现篇 - liuchonge的博客 - 博客

		频道 - CSDN.NET - 《网络（http://blog.csdn.net）》- （是否引证：否）
		1.CNN神经网络已经搭建完毕，接下来就是传入数据开始一步步的训练了。训练部分代码在train.py文件中，代码如下所示： with tf.Graph().as_default(): session_conf = tf.ConfigProto(#允许TensorFlow回退到特定设备，并记录程序运行的设备信息 allow_soft_placement=FLAGS 2.on_conf = tf.ConfigProto(#允许TensorFlow回退到特定设备，并记录程序运行的设备信息 allow_soft_placement=FLAGS.allow_soft_placement,log_device_placement=FLAGS.log_device_placement) sess = tf.Session(config=session_conf) #指定session with sess.as_default(): #初始化CNN网络结构 cnn = TextCNN(sequence_
4	此处有 89 字相似 ResCNN(FLAGS.sequence_length, len(datamanager.relations), FLAGS.embedding_dim, 5, list(map(int, FLAGS.filter_sizes.split(","))), FLAGS.num_filters, FLAGS.l2_reg_lambda) # Define Training procedure global_ste	论文Convolutional Naural Networks for Sentence Classification--TensorFlow实现篇 - liuchonge的博客 - 博客频道 - CSDN.NET - 《网络（http://blog.csdn.net）》- （是否引证：否）
		1.,vocab_size=len(vocab_processor.vocabulary_),embedding_size=FLAGS.embedding_dim,filter_sizes=list(map(int,FLAGS.filter_sizes.split(","))),num_filters=FLAGS.num_filters,l2_reg_lambda=FLAGS.l2_reg_lambda) # t
		基于卷积神经网络的中文文本分类研究_聂国平 -《学术论文联合比对库》- 2017-04-26（是否引证：否）
		1.ulary_),embedding_size=embedding_dimension,filter_sizes=list(map(int, FLAGS.filter_sizes.split(","))),num_filters=FLAGS.num_filters,l2_reg_lambda=FLAGS.l2_reg_lambda)
		011_M201570005_聂国平_聂国平 -《学术论文联合比对库》- 2017-05-18（是否引证：否）
		1.ulary_),embedding_size=embedding_dimension,filter_sizes=list(map(int, FLAGS.filter_sizes.split(","))),num_filters=FLAGS.num_filters,l2_reg_lambda=FLAGS.l2_reg_lambda)
5	此处有 306 字相似 ap(int, FLAGS.filter_sizes.split(","))), FLAGS.num_filters, FLAGS.l2_reg_lambda) # Define Training procedure global_step = tf.Variable(0, name="global_step", trainable=False) optimizer = tf.train.AdamOptimizer(1e-3) grads_and_vars = optimizer.compute_gradients(cnn.loss) train_op = optimizer.apply_gradients(grads_and_vars, global_step=global_step) saver = tf.train. Saver(tf.global_variables()) sess.run(tf.global_variables_in	基于卷积神经网络的中文文本分类研究_聂国平 -《学术论文联合比对库》- 2017-04-26（是否引证：否）
		1.plit(","))),num_filters=FLAGS.num_filters,l2_reg_lambda=FLAGS.l2_reg_lambda)# Define Training procedureglobal_step = tf.Variable(0, name="global_step", trainable=False)optimizer = tf.train.AdamOptimizer(1e-3)grads_and_vars = optimizer.compute_gradients(cnn.loss)train_op = optimizer.apply_gradients(grads_and_vars, global_step=global_step)# Keep track of gradient values and sparsity (optional)grad_
		011_M201570005_聂国平_聂国平 -《学术论文联合比对库》- 2017-05-18（是否引证：否）
		1.plit(","))),num_filters=FLAGS.num_filters,l2_reg_lambda=FLAGS.l2_reg_lambda)# Define Training procedureglobal_step = tf.Variable(0, name="global_step", trainable=False)optimizer = tf.train.AdamOptimizer(1e-3)grads_and_vars = optimizer.compute_gradients(cnn.loss)train_op = optimizer.apply_gradients(grads_and_vars,

		global_step=global_step)# Keep track of gradient values and sparsity (optional)grad_ Tensorflow实现的CNN文本分类 - somTian的博客 - 博客频道 - CSDN.NET - 《网络（http://blog.csdn.net）》- （是否引证：否） 1.ers) 接下来，我们定义如何优化网络的损失函数。TensorFlow有几个内置优化器。 我们正在使用Adam优化器。 # Define Training procedure global_step = tf.Variable(0,name="global_step",trainable=False) optimizer = tf.train.AdamOptimizer(1e-4) grads_and_vars = optimizer.compute_gradients(cnn.loss) train_op = optimizer.apply_gradients(grads_and_vars,global_step =global_step) 在这里，train_op这里是一个新创建的操作，我们可以运行它们来对我们的参数执行更新。train_op的每次执行都是一个训练步骤。 基于tensorflow的cnn文本分类 - u013713117的专栏 - CSDN博客 - 《网络（http://blog.csdn.net）》- （是否引证：否） 1.HE LOSS 利用TensorFlow 内置的optimizers，例如Adam optimizer，优化网络损失。 global_step = tf.Variable(0,name="global_step",trainable=False) optimizer = tf.train.AdamOptimizer(1e-4) grads_and_vars = optimizer.compute_gradients(cnn.loss) train_op = optimizer.apply_gradients(grads_and_vars,global_step =global_step) train_op 是一个新建的操作，我们可以在参数上进行梯度更新。每执行一次 train_op 就是一次训练步骤。 TensorFlow 论文Convolutional Naural Networks for Sentence Classification--TensorFlow实现篇 - liuchonge的博客 - 博客频道 - CSDN.NET - 《网络（http://blog.csdn.net）》- （是否引证：否） 1.bda) # trainable=False表明该参数虽然是Variable但并不属于网络运行参数，无需计算梯度并更新 global_step = tf.Variable(0,name="global_step",trainable=False) optimizer = tf.train.AdamOptimizer(1e-3) grads_and_vars = optimizer.compute_gradients(cnn.loss) #计算梯度并根据AdamOptimizer优化函数跟新网络参数 train_op = optimizer.apply_g 2.compute_gradients(cnn.loss) #计算梯度并根据AdamOptimizer优化函数跟新网络参数 train_op = optimizer.apply_gradients(grads_and_vars,global_step =global_step) # 将一些想要在TensorBoard中观察的网络参数记录下来，保存到Summary中 grad_summaries =
6	此处有 62 字相似 训练流程代码 ResCNN的网络层代码如图4.12所示。通过tf.nn.conv2d()实现卷积层，其中卷积块的尺寸为[filter_size, embedding_size+2*position_size, 1, num_filters]， 分别是卷积核的大小、卷积核的维度、通道数和卷积核的数量；strides表示在4个维度上的卷积块的滑动步长	基于关键词自学习的中文网页分类技术研究与实现 鞠亮 - 《武汉理工大学硕士论文》- 2017-03-01（是否引证：否） 1.axpool-%s" % filter_size): filter_shape = [filter_size, embedding_size, 100, num_filters] W = tf.Variable b = tf.Variable 基于卷积神经网络的中文文本分类研究 聂国平 - 《学术论文联合比对库》- 2017-04-26（是否引证：否） 1.ol-%s" % filter_size):# Convolution Layerfilter_shape = [filter_size, embedding_size, 1, num_filters]W =

	都为1；padd	tf.Variable(tf.truncated_normal(filter_shape, stddev=0.1), 011_M201570005_聂国平_聂国平 -《学术论文联合比对库》- 2017-05-18（是否引证：否）
		1.ol-%s" % filter_size):# Convolution Layerfilter_shape = [filter_size, embedding_size, 1, num_filters]W = tf.Variable(tf.truncated_normal(filter_shape, stddev=0.1), 1049721403027_鞠亮_学术性硕士_信息与通信工程_周祖德_鞠亮 -《学术论文联合比对库》- 2017-06-07（是否引证：否）
		1..name_scope("conv-maxpool-%s" % filter_size):filter_shape = [filter_size, embedding_size, 100, num_filters]W = tf.Variableb = tf.Variableconv = tf.nn.conv2dh = tf.nn.rel 基于深度学习的评论文本情感分类系统设计与实现_周全 -《新疆大学硕士论文》- 2018-06-01（是否引证：否）
		1.v-maxpool-%s" % filter_size): filter_shape = [filter Size, embedding_size, 1,num_filters] W = tf.Variable(tf. normal(filter_shape), name="W")
7	此处有 483 字相似 换成[-1, num_filter_total]得到最终的隐藏层特征矩阵，再在后面接一层Dropout层以防过拟合。 pooled_outputs = [] for i, filter_size in enumerate(filter_sizes): with tf.name_scope("conv-maxpool-%s" % filter_size): # Convolution Layer filter_shape = [filter_size, embedding_size+2*position_size, 1, num_filters] W = tf.Variable(tf.truncated_normal(filter_shape, stddev=0.1), name="W") b = tf.Variable(tf.constant(0.1, shape=[num_filters]), name="b") conv = tf.nn.conv2d(self.embedded_chars_expanded, W, strides=[1, 1, 1, 1], padding="VALID", name="conv") # Apply non-linearity, shape of h is [batch_size, sequence_length-2, 1, num_filter	Tensorflow实现的CNN文本分类 - somTian的博客 - 博客频道 - CSDN.NET - 《网络（http://blog.csdn.net）》-（是否引证：否）
		1.er。 因为每个卷积产生不同形状的张量，我们需要迭代它们，为它们中的每一个创建一个层，然后将结果合并成一个大特征向量。 pooled_outputs = [] for i,filter_size in enumerate(filter_sizes): with tf.name_scope("conv-maxpool-%s" %filter_size): # Convolution Layer filter_shape = [filter_size,embedding_size,1,num_filters] W = tf.Variable(tf.truncated_normal(filter_shape,stddev=0.1) ,name="W") b = tf.Variable(tf.constant(0.1,shape=[num_filters]),name=" b") conv = tf.nn.conv2d(self.embedded_chars_expanded,W,strides=[1,1,1,1],p adding="VALID",name="conv") # Apply nonlinearity h = tf.nn.relu(tf.nn.bias_add(conv,b),name="relu") # Max-poo
		Tensorflow版TextCNN主要代码解析 - u013818406的博客 - CSDN博客 - 《网络（http://blog.csdn.net）》-（是否引证：否）
		1.# Create a convolution + maxpool layer for each filter size pooled_outputs = [] for i,filter_size in enumerate(filter_sizes): with tf.name_scope("conv-maxpool-%s" % filter_size): # Convolution Layer filter_shape = [filter_size,embedding_size,1,num_filters] W = tf.Variable(tf.truncated_normal(filter_shape,stddev=0.1) ,name="W") b = tf.Variable(tf.constant(0.1,shape=[num_filters]),name=" b") conv = tf.nn.conv2d(self.embedded_chars_expanded,W,strides=[1,1,1,1],p adding="VALID",name="conv") # Apply nonlinearity h = tf.nn.relu(tf.nn.bias_add(conv,b),name="relu") # Maxpool
		基于tensorflow的cnn文本分类 - u013713117的专栏 - CSDN博客 - 《网络（http://blog.csdn.net）》-（是否引证

		：否） 1.filters。因为每次卷积都会产生不同形状的张量，所以我们要遍历每个filter，然后将结果合并成一个大的特征向量。 pooled_outputs = [] for i,filter_size in enumerate(filter_sizes): with tf.name_scope("conv-maxpool-%s" % filter_size): # Convolution Layer filter_shape = [filter_size,embedding_size,1,num_filters] W = tf.Variable(tf.truncated_normal(filter_shape,stddev=0.1),name="W") b = tf.Variable(tf.constant(0.1,shape=[num_filters]),name="b") conv = tf.nn.conv2d(self.embedded_chars_expanded,W,strides=[1,1,1,1],padding="VALID",name="conv") # Apply nonlinearity h = tf.nn.relu(tf.nn.bias_add(conv,b),name="relu") # Max-poo CNN情感分析（文本分类）- 萝卜虫的博客 - CSDN博客 -《网络（http://blog.csdn.net）》- （是否引证：否） 1.的局基层conv2d需要一个4维的tensor（batch，width，height，channel）3、卷积池化层 pooled_outputs = [] for i,filter_size in enumerate(filter_sizes): with tf.name_scope("conv-maxpool-%s" % filter_size): # Convolution Layer filter_shape = [filter_size,embedding_size,1,num_filters] W = tf.Variable(tf.truncated_normal(filter_shape,stddev=0.1),name="W") b = tf.Variable(tf.constant(0.1,shape=[num_filters]),name="b") conv = tf.nn.conv2d(self.embedded_chars_expanded,W,strides=[1,1,1,1],padding="VALID",name="conv") # Apply nonlinearity h = tf.nn.relu(tf.nn.bias_add(conv,b),name="relu") # Max-poo 基于卷积神经网络的中文文本分类研究 聂国平 -《学术论文联合比对库》- 2017-04-26（是否引证：否） 1. Create a convolution + maxpool layer for each filter sizepooled_outputs = []for i, filter_size in enumerate(filter_sizes):with tf.name_scope("conv-maxpool-%s" % filter_size):# Convolution Layerfilter_shape = [filter_size, embedding_size, 1, num_filters]W = tf.Variable(tf.truncated_normal(filter_shape, stddev=0.1), name="W")b = tf.Variable(tf.constant(0.1, shape=[num_filters]), name="b")conv = tf.nn.conv2d(self.embedded_chars_expanded,W,strides=[1, 1, 1, 1],padding="VALID",name="conv")# Apply nonlinearityh = tf.nn.relu(tf.nn.bias_add(conv, b), name="relu")# Maxpoo 011_M201570005_聂国平 聂国平 -《学术论文联合比对库》- 2017-05-18（是否引证：否） 1. Create a convolution + maxpool layer for each filter sizepooled_outputs = []for i, filter_size in enumerate(filter_sizes):with tf.name_scope("conv-maxpool-%s" % filter_size):# Convolution Layerfilter_shape = [filter_size, embedding_size, 1, num_filters]W = tf.Variable(tf.truncated_normal(filter_shape, stddev=0.1), name="W")b = tf.Variable(tf.constant(0.1,

shape=[num_filters]), name="b")conv = tf.nn.conv2d(self.embedded_chars_expanded,W,strides=[1, 1, 1, 1],padding="VALID",name="conv")# Apply nonlinearityh = tf.nn.relu(tf.nn.bias_add(conv, b), name="relu")# Maxpoo

基于深度学习的评论文本情感分类系统设计与实现 周全 -《新疆大学硕士论文》- 2018-06-01（是否引证：否）

1.疆大学硕士研究生学位论文34通过 Tensorflow 自有的应用程序接口，将数据准备好然后调用即可。pooled Outputs = [] for i, filter Size in enumerate(filter Sizes): with tf.name_scope("conv-maxpool-%s" % filter_size): filter_shape = [filter Size, embedding_size, 1,num_filters] W = tf.Variable(tf. normal(filter_shape), name="W") b = tf.Variable(tf.constant(0.1), name="b") conv = tf.nn.conv2d(self.embedded_chars_expanded, W, strides=[1, 1, 1, 1], padding="VALID", name="conv") h = tf.nn.relu(tf.nn.bias_add(conv, b), name="relu")

论文Convolutional Naural Networks for Sentence Classification--TensorFlow实现篇 - liuchonge的博客 - 博客频道 - CSDN.NET -《网络（http://blog.csdn.net）》-（是否引证：否）

1.dded_chars,-1) #为每一个filter_size创建卷积层+池化层。并将最后的结果合并成一个大的特征向量 pooled_outputs = [] for i,filter_size in enumerate(filter_sizes): with tf.name_scope("conv-maxpool-%s" % filter_size): #构建卷积核尺寸，输入和输出channel分别为1和num_filters filter_shape = [filte

2.l-%s" % filter_size): #构建卷积核尺寸，输入和输出channel分别为1和num_filters filter_shape = [filter_size,embedding_size,1,num_filters] W = tf.Variable(tf.truncated_normal(filter_shape,stddev=0.1),name="W") b = tf.Variable(tf.constant(0.1,shape=[num_filters]),name="b") conv = tf.nn.conv2d(self.embedded_chars_expanded,W,strides=[1,1,1,1],padding="VALID",name="conv") #非线性操作，激活函数：relu(W*x + b) h = tf.nn.relu(tf.nn.bias_add(conv

基于集成算法的密级文本分类系统设计 顾凯文 -《南京邮电大学硕士论文》- 2018-11-14（是否引证：否）

1.神经网络方面的研究，他的广泛性和扩展性也使得其可以用于其他领域。卷积神经网络训练核心代码如下：for i, filter_size in enumerate(filter_sizes): with tf.name_scope("conv-maxpool-%s" % filter_size): # 卷积层 filter_shape = [filter_size, embedding_size, 1, num_filters] # 设置权重大小 W = tf.Variable(tf.truncated_normal(filter_shape, stddev=0.1), name="W") b = tf.Variable(tf.constant(0.1, shape=[num_filters]), name="b")# 设置卷积 conv = tf.nn.conv2d(self.embedded_chars_expanded, W, strides=[1, 1, 1, 1], padding="VALID", name="conv") # 应用非线性映射 南京邮电大学专业学位硕士研究生学位论文 第四章 密级文本分类系统设计35

卷积神经网络实践 - Abrohambaby的博客 - CSDN博客 -《网络（http://blog.csdn.net）》-（是否引证：否）

1.on + maxpool layer num_filters = 128 filter_sizes = [3,4,5] pooled_outputs = [] for i,filter_size in

		enumerate(filter_sizes): with tf.name_scope("conv-maxpool-%s" % filter_size): filter_shape = [filter_size,embedding_size,1,num_filters] W = tf.Variable(tf.truncated_normal(filter_shape,stddev=0.1)) b = tf.Variable(tf.constant(0.1,shape=[num_filters])) conv = tf.nn.conv2d(embedded_chars_expanded,W,strides=[1 ,1,1,1],padding="VALID") h = tf.nn.relu(tf.nn.bias_add(conv,b 纠错： 深度学习模型优化时快速收敛 - Sherry - 《网络（http://blog.csdn.net）》- （是否引证：否） 1.NN模型。 # define a convolution function def conv(input_data): pooled_outputs = [] for i,filter_size in enumerate(filter_sizes): with tf.name_scope("conv-maxpool-%s" % filter_size): # Convolution Layer filter_shape = [filter_size,embedding_size,1,num_filters] W = tf.Variable(tf.truncated_normal(filter_sh 2.ol-%s" % filter_size): filter_shape = [filter_size,embedding_size,1,num_filters] W = tf.Variable(tf.truncated_normal(filter_shape,stddev=0.1) ,name="W") b = tf.Variable(tf.constant(0.1,shape=[num_filters]),name="b") conv = tf.nn.conv2d(self.embedded_chars1_expanded,W,strides=[1,1,1,1],padding="VALID",name="conv") h = tf.nn.relu(tf.nn.bia 基于关键词自学习的中文网页分类技术研究与实现 鞠亮 - 《武汉理工大学硕士论文》- 2017-03-01（是否引证：否） 1.2_reg_lambda=0.0): l2_loss = tf.constant(0.0) pooled_outputs = []for j in 2 for i, filter_size in enumerate(filter_sizes): with tf.name_scope("conv-maxpool-%s" % filter_size): filter_shape = [filter_size, embedding_size, 100, num_filters] W = tf.Variable b = tf.Variable conv = tf.nn.conv2d 1049721403027_鞠亮_学术性硕士_信息与通信工程_周祖德 鞠亮 -《学术论文联合比对库》- 2017-06-07（是否引证：否） 1., num_filters, l2_reg_lambda=0.0):l2_loss = tf.constant(0.0)pooled_outputs = []for j in 2for i, filter_size in enumerate(filter_sizes):with tf.name_scope("conv-maxpool-%s" % filter_size):filter_shape = [filter_size, embedding_size, 100, num_filters]W = tf.Variableb = tf.Variableconv = tf.nn.conv2dh = tf.nn.relupooled = tf.nn
8	此处有 61 字相似] h = tf.nn.relu(tf.nn.bias_add(conv, b), name="relu") self. l2_loss += tf.nn.l2_loss(W) self.l2_loss += tf.nn.l2_loss(b) # nine convolution layers for i in range(3): h2 = self.Cnnbl	论文Convolutional Naural Networks for Sentence Classification--TensorFlow实现篇 - liuchonge的博客 - 博客频道 - CSDN.NET - 《网络（http://blog.csdn.net）》- （是否引证：否） 1. tf.Variable(tf.constant(0.1,shape=[num_classes]),name="b") l2_loss += tf.nn.l2_loss(W) l2_loss += tf.nn.l2_loss(b) self.scores = tf.nn.xw_plus_b(self.h_drop,W,b,name="scores") Tensorflow版TextCNN主要代码解析 - u013818406的博客 - CSDN博客 - 《网络（http://blog.csdn.net）》- （是否引证

		：否 ） 1. tf.Variable(tf.constant(0.1,shape=[num_classes]),name ="b") l2_loss += tf.nn.l2_loss(W) l2_loss += tf.nn.l2_loss(b) self.scores = tf.nn.xw_plus_b(self.h_drop,W,b,name="scores") 基于关键词自学习的中文网页分类技术研究与实现_鞠亮 -《武汉理工大学硕士论文》- 2017-03-01（是否引证：否） 1.e("output"):W = tf.get_variableb = tf.Variable l2_loss += tf.nn.l2_loss(W) l2_loss += tf.nn.l2_loss(b) self.scores = tf.nn.xw_plus_b self.predictions = tf.a 基于卷积神经网络的中文文本分类研究_聂国平 -《学术论文联合比对库》- 2017-04-26（是否引证：否） 1..Variable(tf.constant(0.1, shape=[num_classes]), name="b")l2_loss += tf.nn.l2_loss(W)l2_loss += tf.nn.l2_loss(b)self.scores = tf.nn.xw_plus_b(self.h_drop, W, b, name="scores") 011_M201570005_聂国平_聂国平 -《学术论文联合比对库》- 2017-05-18（是否引证：否） 1..Variable(tf.constant(0.1, shape=[num_classes]), name="b")l2_loss += tf.nn.l2_loss(W)l2_loss += tf.nn.l2_loss(b)self.scores = tf.nn.xw_plus_b(self.h_drop, W, b, name="scores") 1049721403027_鞠亮_学术性硕士_信息与通信工程_周祖德_鞠亮 -《学术论文联合比对库》- 2017-06-07（是否引证：否） 1.h tf.name_scope("output"):W = tf.get_variableb = tf.Variablel2_loss += tf.nn.l2_loss(W)l2_loss += tf.nn.l2_loss(b)self.scores = tf.nn.xw_plus_bself.predictions = tf.argmaxwith tf.
9	此处有 324 字相似 in range(3): h2 = self.Cnnblock(num_filters, h, i) h = h2+h pooled = tf.nn.max_pool(h, ksize=[1, sequence_length - filter_size + 1, 1, 1], strides=[1, 1, 1, 1], padding='VALID', name="pool") pooled_outputs.append(pooled) num_filters_total = num_filters * len(filter_sizes) self.h_pool = tf.concat(pooled_outputs, 3) self.h_pool_flat = tf.reshape(self.h_pool, [-1, num_filters_total], name="hidden_feature")with tf.name_scope("dropout"): self.h	Tensorflow版TextCNN主要代码解析 - u013818406的博客 - CSDN博客 -《网络（http://blog.csdn.net）》-（是否引证：否） 1.bias_add(conv,b),name="relu") # Maxpooling over the outputs pooled = tf.nn.max_pool(h,ksize=[1,sequence_length - filter_size + 1,1,1],strides=[1,1,1,1],padding='VALID',name="pool") pooled_outputs.append(pooled)建立了一个pooled_outputs来保存每次卷积结果，在不同的卷积核大小进行卷积、relu激活函数和max_po 2.rs*len(filters_sizes)个数字。 # Combine all the pooled features num_filters_total = num_filters * len(filter_sizes) self.h_pool = tf.concat(3,pooled_outputs) self.h_pool_flat = tf.reshape(self.h_pool,[-1,num_filters_total]) # Add dropout with tf.name_scope("dropout"): self.h_drop = t 基于tensorflow的cnn文本分类 - u013713117的专栏 - CSDN博客 -《网络（http://blog.csdn.net）》-（是否引证：否） 1.ias_add(conv,b),name="relu") # Max-pooling over the outputs pooled = tf.nn.max_pool(h,ksize=[1,sequence_length - filter_size + 1,1,1],strides=[1,1,1,1],padding='VALID',name="pool") pooled_outputs.append(pooled) # Combine all the pooled features num_filters_total = num_fi

		2.ed_outputs.append(pooled) # Combine all the pooled features num_filters_total = num_filters * len(filter_sizes) self.h_pool = tf.concat(3,pooled_outputs) self.h_pool_flat = tf.reshape(self.h_pool,[-1,num_filters_total]) 这里W 是 filter 矩阵，h 是对卷积结果进行非线性转换之后的结果。每个 filter都从整个embedding划过
		CNN情感分析（文本分类）- 萝卜虫的博客 - CSDN博客 -《网络（http://blog.csdn.net）》- （是否引证：否）
		1.ias_add(conv,b),name="relu") # Max-pooling over the outputs pooled = tf.nn.max_pool(h,ksize=[1,sequence_length - filter_size + 1,1,1],strides=[1,1,1,1],padding='VALID',name="pool") pooled_outputs.append(pooled) # Combine all the pooled features num_filters_total = num_fi 2.ed_outputs.append(pooled) # Combine all the pooled features num_filters_total = num_filters * len(filter_sizes) self.h_pool = tf.concat(3,pooled_outputs) self.h_pool_flat = tf.reshape(self.h_pool,[-1,num_filters_total]) 我们先建立卷积层，然后是最大池化层。因为我们卷积层的过滤器选择了多个，所以我们需要对它们进行迭代，为每一个过滤器都建立一
		基于卷积神经网络的中文文本分类研究 聂国平 -《学术论文联合比对库》- 2017-04-26（是否引证：否）
		1._add(conv, b), name="relu")# Maxpooling over the outputspooled = tf.nn.max_pool(h,ksize=[1, sequence_length - filter_size + 1, 1, 1],strides=[1, 1, 1, 1],padding='VALID',name="pool")pooled_outputs.append(pooled)# Combine all the pooled featuresnum_filters_total = num_fil 2._outputs.append(pooled)# Combine all the pooled featuresnum_filters_total = num_filters * len(filter_sizes)self.h_pool = tf.concat(pooled_outputs, 3)self.h_pool_flat = tf.reshape(self.h_pool, [-1, num_filters_total])# Add dropoutwith tf.name_scope("dropout"):self.h_drop = t
		011_M201570005_聂国平 聂国平 -《学术论文联合比对库》- 2017-05-18（是否引证：否）
		1._add(conv, b), name="relu")# Maxpooling over the outputspooled = tf.nn.max_pool(h,ksize=[1, sequence_length - filter_size + 1, 1, 1],strides=[1, 1, 1, 1],padding='VALID',name="pool")pooled_outputs.append(pooled)# Combine all the pooled featuresnum_filters_total = num_fil 2._outputs.append(pooled)# Combine all the pooled featuresnum_filters_total = num_filters * len(filter_sizes)self.h_pool = tf.concat(pooled_outputs, 3)self.h_pool_flat = tf.reshape(self.h_pool, [-1, num_filters_total])# Add dropoutwith tf.name_scope("dropout"):self.h_drop = t
		基于深度学习的评论文本情感分类系统设计与实现 周全 -《新疆大学硕士论文》- 2018-06-01（是否引证：否）
		1.n.relu(tf.nn.bias_add(conv, b), name="relu") pooled = tf.nn.max_pool(h, ksize=[1, sequence_length - filter

		Size + 1,1,1], strides=[1, 1, 1, 1], padding='VALID', name="pool") pooled_outputs.append(pooled) # 将 max-pooling 层的各种特征整合在一起 num_filters_total = num_fi 2.puts.append(pooled) # 将 max-pooling 层的各种特征整合在一起 num_filters_total = num_filters * len(filter Sizes) self.h_pool = tf.concat(3, pooled_outputs) self.h_pool_flat = tf.reshape(self.h_pool, [-1,num_filters_total]) 新疆大学硕士研究生学位论文 354.5 文本情感分类的实现有了上面的基础，文本的情感分类就容易了，这里使
		基于集成算法的密级文本分类系统设计 顾凯文 -《南京邮电大学硕士论文》- 2018-11-14（是否引证：否）
		1.e="relu") # 最大池化输出设置 pooled = tf.nn.max_pool(h, ksize=[1, sequence_length - filter_size + 1, 1, 1], strides=[1, 1, 1, 1], padding='VALID', name="pool") pooled_outputs.append(pooled)# 合并所有池化特征 num_filters_total = num_filters * len(filter_sizes)self.h_pool = tf.concat(3, pooled_outputs)self.h_pool_flat = tf.reshape(self.h_pool, [-1, num_filters_total])为了实例化类，我们需要传递以下参数到类中。embedding_size 设置每个单词的词向量长度，nu
		Tensorflow实现的CNN文本分类 - somTian的博客 - 博客频道 - CSDN.NET -《网络（http://blog.csdn.net）》-（是否引证：否）
		1.ias_add(conv,b),name="relu") # Max-pooling over the outputs pooled = tf.nn.max_pool(h,ksize=[1,sequence_length - filter_size +1,1,1],strides=[1,1,1,1],padding="VALID",name="pool") pool 2.=[1,sequence_length - filter_size +1,1,1],strides=[1,1,1,1],padding="VALID",name="pool") pooled_outputs.append(pooled) # Combine all the pooled features num_filters_total = num_fi 3.ed_outputs.append(pooled) # Combine all the pooled features num_filters_total = num_filters * len(filter_sizes) self.h_pool = tf.concat(3,pooled_outputs) self.h_pool_flat = tf.reshape(self.h_pool,[-1,num_filters_total]) 这里，W是我们的滤波器矩阵，h是将非线性应用于卷积输出的结果。 每个过滤器在整个嵌入中滑动，但是它涵盖的字数有所不同。
		卷积神经网络实践 - Abrohambaby的博客 - CSDN博客 -《网络（http://blog.csdn.net）》-（是否引证：否）
		1.,1],padding="VALID") h = tf.nn.relu(tf.nn.bias_add(conv,b)) pooled = tf.nn.max_pool(h,ksize=[1,input_size - filter_size + 1,1,1],strides=[1,1,1,1],padding='VALID') pooled_outputs.append(pooled) num_fi 2.e - filter_size + 1,1,1],strides=[1,1,1,1],padding='VALID') pooled_outputs.append(pooled) num_filters_total = num_filters * len(filter_sizes) h_pool =

		tf.concat(pooled_outputs,3) h_pool_flat = tf.reshape(h_pool,[-1,num_filters_total]) # dropout with tf.name_scope("dropout"): h_drop = tf.nn.drop 论文Convolutional Naural Networks for Sentence Classification--TensorFlow实现篇 - liuchonge的博客 - 博客频道 - CSDN.NET - 《网络（http://blog.csdn.net）》- （是否引证：否） 1.name="relu") # 池化，选取卷积结果的最大值 pooled的尺寸为[None,1,1,128](卷积核个数) pooled = tf.nn.max_pool(h,ksize=[1,sequence_length - filter_size + 1,1,1],strides=[1,1,1,1],padding='VALID',name="pool") _outputs最终为一个长度为3的列表。每一个元素都是[None,1,1,128]的Tensor张量 pooled_outputs.a 2.h_pool = tf.concat(pooled_outputs,3) #展开成两维Tensor[None,384] self.h_pool_flat = tf.reshape(self.h_pool,[-1,num_filters_total]) # Add dropout with tf.name_scope("dropout"): self.h_drop = t 纠错：深度学习模型优化时快速收敛 - Sherry - 《网络（http://blog.csdn.net）》- （是否引证：否） 1.ias_add(conv,b),name="relu") # Max-pooling over the outputs pooled = tf.nn.max_pool(h,ksize=[1,sequence_length - filter_size + 1,1,1],strides=[1,1,1,1],padding='VALID',name="pool") # shape of pooled is [batch_size,1,1,num_filters] pooled_out
10	此处有 105 字相似 self.h_pool, [-1, num_filters_total], name="hidden_feature") with tf.name_scope("dropout"): self.h_drop = tf.nn.dropout(self.h_pool_flat, self.dropout_keep_prob) 图4. 12 ResCNN网络层的实现 经过网络层处理后的数据传入给分类器，图4.13为分类器的实现。分类器中的两个主要参数权	论文Convolutional Naural Networks for Sentence Classification--TensorFlow实现篇 - liuchonge的博客 - 博客频道 - CSDN.NET - 《网络（http://blog.csdn.net）》- （是否引证：否） 1.f.reshape(self.h_pool,[-1,num_filters_total]) # Add dropout with tf.name_scope("dropout"): self.h_drop = tf.nn.dropout(self.h_pool_flat,self.dropout_keep_prob) #全连接层计算输出向量(w*h+b)和预测(scores向量中的最大值即为预测结果) with tf.name_scop Tensorflow版TextCNN主要代码解析 - u013818406的博客 - CSDN博客 - 《网络（http://blog.csdn.net）》- （是否引证：否） 1.f.reshape(self.h_pool,[-1,num_filters_total]) # Add dropout with tf.name_scope("dropout"): self.h_drop = tf.nn.dropout(self.h_pool_flat,self.dropout_keep_prob) # Final (unnormalized) scores and predictions with tf.name_s 基于tensorflow的cnn文本分类 - u013713117的专栏 - CSDN博客 - 《网络（http://blog.csdn.net）》- （是否引证：否） 1.定。在训练的时候设为 0.5 ,测试的时候设为 1 (disable dropout) . # Add dropout with tf.name_scope("dropout"): self.h_drop = tf.nn.dropout(self.h_pool_flat,self.dropout_keep_prob) SCORES AND PREDICTIONS 利用特征向量，我们可以用矩阵相乘计算两类的得分，也可以用 softmax函 卷积神经网络实践 - Abrohambaby的博客 - CSDN博客 - 《网络（http://blog.csdn.net）》- （是否引证：否） 1._flat = tf.reshape(h_pool,[-1,num_filters_total]) # dropout with tf.name_scope("dropout"): h_drop =

		tf.nn.dropout(h_pool_flat,dropout_keep_prob) # output with tf.name_scope("output"): W = tf.get_variable(" CNN情感分析（文本分类）- 萝卜虫的博客 - CSDN博客 -《网络（http://blog.csdn.net）》-（是否引证：否） 1.地方，就是我们文字不能跨行过滤，因为那样会没有意义。 4、设置dropout（训练时候用） # Add dropout with tf.name_scope("dropout"): self.h_drop = tf.nn.dropout(self.h_pool_flat,self.dropout_keep_prob) 5、评分和预测（最后的全连接层） with tf.name_scope("output"): W = tf.Variab 基于卷积神经网络的中文文本分类研究 聂国平 -《学术论文联合比对库》- 2017-04-26（是否引证：否） 1.shape(self.h_pool, [-1, num_filters_total])# Add dropoutwith tf.name_scope("dropout"):self.h_drop = tf.nn.dropout(self.h_pool_flat, self.dropout_keep_prob)# Final (unnormalized) scores and predictionswith tf.name_sc 011_M201570005_聂国平 聂国平 -《学术论文联合比对库》- 2017-05-18（是否引证：否） 1.shape(self.h_pool, [-1, num_filters_total])# Add dropoutwith tf.name_scope("dropout"):self.h_drop = tf.nn.dropout(self.h_pool_flat, self.dropout_keep_prob)# Final (unnormalized) scores and predictionswith tf.name_sc
11	此处有 808 字相似 xw_plus_b()对网络层输出的特征矩阵进行矩阵乘法并添加偏置项，再调用交叉熵函数，对其求平均值后加上l2正则项得到 最后的损失值。 with tf.name_scope("output"): W = tf.get_variable("W", shape=[num_filters_total, num_classes], initializer=tf.contrib.layers.xavier_initializer()) b = tf.Variable(tf.constant(0.1, shape=[num_classes]), name="b") self.l2_loss += tf.nn.l2_loss(W) self.l2_loss += tf.nn.l2_loss(b) self.scores = tf.nn.xw_plus_b(self.h_dropout, W, b, name="scores") self.predictions = tf.argmax(self.scores, 1, name="predictions") with tf.name_scope("loss"): losses = tf.nn.softmax_cross_entropy_with_logits(logits=self.scores, labels=self.input_y) self.loss = tf.reduce_mean(losses) + l2_reg_lambda * self.l2_loss with tf.name_scope("accuracy"): correct_predictions = tf.equal(self.predictions, tf.argmax(self.input_y, 1)) self.accuracy = tf.reduce_mean(tf.cast(correct_predictions, "float"), name="accuracy") 图4.13 关系抽取模型分类器的实现 4.5 信息抽取服务的实现与部署 4.5.1 模型格式转换 在信息抽取模型模块中通过模型	Tensorflow版TextCNN主要代码解析 - u013818406的博客 - CSDN博客 -《网络（http://blog.csdn.net）》-（是否引证：否） 1.ut_keep_prob) # Final (unnormalized) scores and predictions with tf.name_scope("output"): W = tf.get_variable("W",shape=[num_filters_total,num_classes],initializer=tf.contrib.layers.xavier_initializer()) b = tf.Variable(tf.constant(0.1,shape=[num_classes]),name="b") l2_loss += tf.nn.l2_loss(W) l2_loss += tf.nn.l2_loss(b) self.scores = tf.nn.xw_plus_b(self.h_drop,W,b,name="scores") self.predictions = tf.argmax(self.scores,1,name="predictions") d Tensorflow lib cant handle 64 bit integers,adding an extra 2.层与softmax层将特征映射成不同类别上的概率 # CalculateMean cross-entropy loss with tf.name_scope("loss"): losses = tf.nn.softmax_cross_entropy_with_logits(self.scores,self.input_y) self.loss = tf.reduce_mean(losses) + l2_reg_lambda * l2_loss # Accuracy with tf.name_scope("accuracy"): correct_predictions = tf.equ 3._lambda * l2_loss # Accuracy with tf.name_scope("accuracy"): correct_predictions = tf.equal(self.predictions,tf.argmax(self.input_y,1)) self.accuracy = tf.reduce_mean(tf.cast(correct_predictions,"float"),name="accuracy")损失函数使用的交叉熵加上L2正则损失，准确度用的非常多就不特别说明了

		基于卷积神经网络的中文文本分类研究_聂国平 -《学术论文联合比对库》- 2017-04-26（是否引证：否）
		1._keep_prob)# Final (unnormalized) scores and predictionswith tf.name_scope("output"):W = tf.get_variable("W",shape=[num_filters_total, num_classes],initializer=tf.contrib.layers.xavier_initializer())b = tf.Variable(tf.constant(0.1, shape=[num_classes]), name="b")l2_loss += tf.nn.l2_loss(W)l2_loss += tf.nn.l2_loss(b)self.scores = tf.nn.xw_plus_b(self.h_drop, W, b, name="scores")self.predictions = tf.argmax(self.scores, 1, name="predictions")# CalculateMean cross-entropy losswith tf.name_scope("loss") 2.1, name="predictions")# CalculateMean cross-entropy losswith tf.name_scope("loss"):losses = tf.nn.softmax_cross_entropy_with_logits(logits=self.scores, labels=self.input_y)self.loss = tf.reduce_mean(losses) + l2_reg_lambda * l2_loss# Accuracywith tf.name_scope("accuracy"):correct_predictions = tf.equal(self.predictions, tf.argmax(self.input_y, 1))self.accuracy = tf.reduce_mean(tf.cast(correct_predictions, "float"), name="accuracy")
		011_M201570005_聂国平_聂国平 -《学术论文联合比对库》- 2017-05-18（是否引证：否）
		1._keep_prob)# Final (unnormalized) scores and predictionswith tf.name_scope("output"):W = tf.get_variable("W",shape=[num_filters_total, num_classes],initializer=tf.contrib.layers.xavier_initializer())b = tf.Variable(tf.constant(0.1, shape=[num_classes]), name="b")l2_loss += tf.nn.l2_loss(W)l2_loss += tf.nn.l2_loss(b)self.scores = tf.nn.xw_plus_b(self.h_drop, W, b, name="scores")self.predictions = tf.argmax(self.scores, 1, name="predictions")# CalculateMean cross-entropy losswith tf.name_scope("loss") 2.1, name="predictions")# CalculateMean cross-entropy losswith tf.name_scope("loss"):losses = tf.nn.softmax_cross_entropy_with_logits(logits=self.scores, labels=self.input_y)self.loss = tf.reduce_mean(losses) + l2_reg_lambda * l2_loss# Accuracywith tf.name_scope("accuracy"):correct_predictions = tf.equal(self.predictions, tf.argmax(self.input_y, 1))self.accuracy = tf.reduce_mean(tf.cast(correct_predictions, "float"), name="accuracy")
		论文Convolutional Naural Networks for Sentence Classification--TensorFlow实现篇 - liuchonge的博客 - 博客频道 - CSDN.NET -《网络（http://blog.csdn.net）》-（是否引证：否）
		1.测(scores向量中的最大值即为预测结果) with tf.name_scope("output"): W = tf.get_variable("W",shape=[num_filters_total,num_classes],initializer=tf.contrib.layers.xavier_initializer()) b =

tf.Variable(tf.constant(0.1,shape=[num_classes]),name="b") l2_loss += tf.nn.l2_loss(W) l2_loss += tf.nn.l2_loss(b) self.scores = tf.nn.xw_plus_b(self.h_drop,W,b,name="scores") self.predictions = tf.argmax(self.scores,1,name="predictions") # CalculateMean cross-entropy loss计算scores和input_y的交叉熵损失函数 w

2. # CalculateMean cross-entropy loss计算scores和input_y的交叉熵损失函数 with tf.name_scope("loss"): losses = tf.nn.softmax_cross_entropy_with_logits(logits=self.scores,labels=self.input_y) self.loss = tf.reduce_mean(losses) + l2_reg_lambda * l2_loss # Accuracy计算准确度，预测和真实标签相同即为正确 with tf.name_scope("accuracy"): correct_

3. Accuracy计算准确度，预测和真实标签相同即为正确 with tf.name_scope("accuracy"): correct_predictions = tf.equal(self.predictions,tf.argmax(self.input_y,1)) self.accuracy = tf.reduce_mean(tf.cast(correct_predictions,"float"),name="accuracy") 最终网络结构在TensorBoard中可视化结果如下图所示： 对上述代码中使用的常用函数做个总结 ： 1，tf.nn.co

基于tensorflow的cnn文本分类 - u013713117的专栏 - CSDN博客 -《网络（http://blog.csdn.net）》-（是否引证：否）

1. PREDICTIONS 利用特征向量，我们可以用矩阵相乘计算两类的得分，也可以用 softmax函数计算两类的概率值。 with tf.name_scope("output"): W = tf.Variable(tf.truncated_normal([num_filters_total,num_classes],stddev=0.1),name="W") b = tf.Variable(tf.constant(0.1,shape=

2.ormal([num_filters_total,num_classes],stddev=0.1),name="W") b = tf.Variable(tf.constant(0.1,shape=[num_classes]),name="b") self.scores = tf.nn.xw_plus_b(self.h_drop,W,b,name="scores") self

3. tf.Variable(tf.constant(0.1,shape=[num_classes]),name="b") self.scores = tf.nn.xw_plus_b(self.h_drop,W,b,name="scores") self.predictions = tf.argmax(self.scores,1,name="predictions") LOSS AND ACCURACY 可以用得分定义损失值。损失计算的是网络的误差，我们的目标是将其最小化，分类问题标准的损失函数是

4.alculate mean cross-entropy loss with tf.name_scope("loss"): losses = tf.nn.softmax_cross_entropy_with_logits(self.scores,self.input_y) self.loss = tf.reduce_mean(losses) 计算正确率 # Calculate Accuracy with tf.name_scope("accuracy"): c

5. 计算正确率 # Calculate Accuracy with

		tf.name_scope("accuracy"): correct_predictions = tf.equal(self.predictions,tf.argmax(self.input_y,1)) self.accuracy = tf.reduce_mean(tf.cast(correct_predictions,"float"),name="accuracy") MINIMIZING THE LOSS 利用 TensorFlow 内置的optimizers，例如 Adam optim
		CNN情感分析（文本分类）- 萝卜虫的博客 - CSDN博客 - 《网络（http://blog.csdn.net）》- （是否引证：否）
		1.t(self.h_pool_flat,self.dropout_keep_prob) 5、评分和预测（最后的全连接层） with tf.name_scope("output"): W = tf.Variable(tf.truncated_normal([num_filters_total,num_classes],stddev=0.1),name="W") b = tf.Variable(tf.constant(0.1,shape= 2.ormal([num_filters_total,num_classes],stddev=0.1),name="W") b = tf.Variable(tf.constant(0.1,shape=[num_classes]),name="b") self.scores = tf.nn.xw_plus_b(self.h_drop,W,b,name="scores") self 3. tf.Variable(tf.constant(0.1,shape=[num_classes]),name="b") self.scores = tf.nn.xw_plus_b(self.h_drop,W,b,name="scores") self.predictions = tf.argmax(self.scores,1,name="predictions") 6、设置损失函数和准确率计算 # Calculate mean cross-entropy loss with tf.n 4.ctions") 6、设置损失函数和准确率计算 # Calculate mean cross-entropy loss with tf.name_scope("loss"): losses = tf.nn.softmax_cross_entropy_with_logits(self.scores,self.input_y) self.loss = tf.reduce_mean(losses) 这里使用的交叉熵损失函数 # Calculate Accuracy with tf.name_scope("accura 5.叉熵损失函数 # Calculate Accuracy with tf.name_scope("accuracy"): correct_predictions = tf.equal(self.predictions,tf.argmax(self.input_y,1)) self.accuracy = tf.reduce_mean(tf.cast(correct_predictions,"float"),name="accuracy") 剩下的，按照常规步骤走就可以了。
		Tensorflow实现的CNN文本分类 - somTian的博客 - 博客频道 - CSDN.NET - 《网络（http://blog.csdn.net）》- （是否引证：否）
		1.阵乘法并选择具有最高分数的类来生成预测。 我们还可以应用softmax函数将原始分数转换为归一化概率，但这不会改变我们的最终预测。 with tf.name_scope("output"): W = tf.Variable(tf.truncated_normal([num_filters_total,num_classes],stddev=0.1),name="W") b = tf.Variable(tf.constant(0.1,shape= 2. tf.Variable(tf.constant(0.1,shape=[num_classes]),name="b") self.scores = tf.nn.xw_plus_b(self.h_drop,W,b,name="scores") self.predictions =

tf.argmax(self.scores,1,name="prediction") 这里，tf.nn.xw_plus_b是执行Wx + b矩阵乘法的便利包装器。 3.6 LOSS AND ACCURACY 使用分数我们可以

3.-entropy loss。 # Calculate mean cross-entropy loss with tf.name_scope("loss"): losses = tf.nn.softmax_cross_entropy_with_logits(self.scores,self.input_y) self.loss = tf.reduce_mean(losses) 这里，tf.nn.softmax_cross_entropy_with_logits是一个方便的函数，计算每个类的交叉熵

4.的有用数值。 # Calculate Accuracy with tf.name_scope("accuracy"): correct_predictions = tf.equal(self.predictions,tf.argmax(self.input_y,1)) self.accuracy = tf.reduce_mean(tf.cast(correct_predictions,"float"),name="accuracy") 3.7 TRAINING PROCEDURE 在我们为网络定义训练程序之前，我们需要了解一些关于TensorFlow如何

1049721403027_鞠亮_学术性硕士_信息与通信工程_周祖德_鞠亮 -《学术论文联合比对库》- 2017-06-07（是否引证：否）

1.h tf.name_scope("output"):W = tf.get_variableb = tf.Variablel2_loss += tf.nn.l2_loss(W)l2_loss += tf.nn.l2_loss(b)self.scores = tf.nn.xw_plus_bself.predictions = tf.argmaxwith tf.name_scope("loss"):losses = t

2.(b)self.scores = tf.nn.xw_plus_bself.predictions = tf.argmaxwith tf.name_scope("loss"):losses = tf.nn.svm_cross_entropy_with_logitsself.loss = tf.reduce_mean(losses) + l2_reg_lambda * l2_losswith

3.tf.nn.svm_cross_entropy_with_logitsself.loss = tf.reduce_mean(losses) + l2_reg_lambda * l2_losswith tf.name_scope("accuracy"):correct_predictions = tf.equal(self.predictions, tf.argmax(self.input_y, 1))self.accuracy = tf.reduce_mean(tf.cast(correct_predictions, "float"), name="accuracy")图3-11 级联网络算法生成图算法的训练过程如图3-12所示，测试过程如图3-13所示。def train_step(x_b

基于关键词自学习的中文网页分类技术研究与实现_鞠亮 -《武汉理工大学硕士论文》- 2017-03-01（是否引证：否）

1.e("output"):W = tf.get_variableb = tf.Variable l2_loss += tf.nn.l2_loss(W) l2_loss += tf.nn.l2_loss(b) self.scores = tf.nn.xw_plus_b self.predictions = tf.argmax with tf.name_scope("loss"):

2.= tf.nn.xw_plus_b self.predictions = tf.argmax with tf.name_scope("loss"): losses = tf.nn.svm_cross_entropy_with_logits self.loss = tf.reduce_mean(losses) + l2_reg_lambda * l2_losswi

3. losses = tf.nn.svm_cross_entropy_with_logits self.loss = tf.reduce_mean(losses) + l2_reg_lambda * l2_losswith tf.name_scope("accuracy"): correct_predictions =

4.mbda * l2_losswith tf.name_scope("accuracy"):

		correct_predictions = tf.equal(self.predictions, tf.argmax(self.input_y, 1)) self.accuracy = tf.reduce_mean(tf.cast(correct_predictions, "float"),name="accuracy")图 3-11 级联网络算法生成图算法的训练过程如图 3-12 所示，测试过程如图 3-13 所示。def train_st 卷积神经网络实践 - Abrohambaby的博客 - CSDN博客 - 《网络（http://blog.csdn.net）》- （是否引证：否） 1.rop = tf.nn.dropout(h_pool_flat,dropout_keep_prob) # output with tf.name_scope("output"): W = tf.get_variable("W",shape=[num_filters_total,n_output_layer],initializer=tf.contrib.layers.xavier_initializer()) b = tf.Variable(tf.constant(0.1,shape=[n_output_layer])) output = tf.nn.xw_plus_b(h_drop,W,b) return 020+1642+刘恒旺 刘恒旺 -《学术论文联合比对库》- 2017-12-18（是否引证：否） 1._y, 1) 返回的是模型对于输入x预测的标签值。然后可用tf.equal来检测预测的结果与实际是否匹配，函数代码如下：correct_predictions = tf.equal(self.predictions, tf.argmax(self.input_y, 1))。这里correct_predictions返回一个布尔数组。为了计算分类的准确率，本文将布尔值转换为浮点数来代表对、错， 2.如：[True，True，False，True，True]变为[1，1，0，1，1]，计算出平均值为0.80。代码如下：self.accuracy = tf.reduce_mean(tf.cast(correct_predictions, "float"), name="accuracy")。5.4.6 卷积神经网络与支持向量机模型对比分析除了通过正确率的评价度量寻找卷积神经网络模型的最优超参数之外，本文还需要验证卷积 经管院-115107000850-刘恒旺-吴鹏nm 经管院 -《学术论文联合比对库》- 2017-12-27（是否引证：否） 1.返回模型对于输入 x预测的标签值。（3）本文用“tf.equal”方法检测预测的结果与实际是否匹配，函数代码如下：“correct_predictions= tf.equal(self.predictions, tf.argmax(self.input_y, 1))”。这里“correct_predictions”返回一个布尔数组。（4）为了计算网络舆情情感分类的准确率，本文将布 2.均值。如：[True，True，False，True]变为[1，1，0，1]，计算出平均值为 0.75。代码如下：“self.accuracy = tf.reduce_mean(tf.cast(correct_predictions,"float"), name="accuracy")”。5.4.6 情感标注的对比分析为了验证 OCC 情感规则体系的科学性，本文将训练集分成两类，一类是用 OCC 基于集成算法的密级文本分类系统设计 顾凯文 -《南京邮电大学硕士论文》- 2018-11-14（是否引证：否） 1.ed_normal(filter_shape, stddev=0.1), name="W") b = tf.Variable(tf.constant(0.1, shape=[num_filters]), name="b")# 设置卷积 conv = tf.nn.conv2d(se
12	此处有 95 字相似 meta_name, pbmodel_path, pb_model_name, model_output_nodes): with tf.Session() as sess: sess.run(tf.global_variables_initializer()) latest_ckpt = tf.train.	卷积神经网络实践 - Abrohambaby的博客 - CSDN博客 - 《网络（http://blog.csdn.net）》- （是否引证：否） 1.amOptimizer().minimize(cost_function) #ok,开始组织数据 epochs = 2 with tf.Session() as sess: sess.run(tf.global_variables_initializer()) x_data = train_data[:,0] y_data = train_data[:,1] for epoch in

	latest_checkpoint(ckpt_path) restore_saver = tf.train.import	range(epochs): epo
13	此处有 83 字相似 为pb格式后，就可以将所有的模块和依赖都封装到一个Docker镜像中方便后面的部署。本文的Docker镜像的构建使用的是 Dockerfile文件，Dockerfile是一种可以被Docker程序解释的脚本，由一条条指令组成，每条指令对应Linux下面的一条命令，Docker程序解决这些 命令间的依赖关系，根据指令生成定制的镜像。当有额外需求需要重新定制镜像时，只需要在Dockerfile中增加或者修改指令	Hadoop分布式集群的自动化容器部署研究 李杰;刘广钟; -《计算机应用研究》- 2016-01-08 1（是否引证：否） 1.int在Docker容器集群上自动化部署Hadoop集群。其基本流程如图7所示。图7 Hadoop容器化部署流程2.1 DockerfileDockfile是一种被Docker程序解释的脚本,Dockerfile由一条一条的指令组成,每条指令对应Linux下面的一条命令。Docker程序将这些Dockerfile指令翻译成真正的Linux命令。Dockerfile有自己书写格式和支持的命令,Docker程序解决 docker 标签 - 《网络（http://blog.51cto.co）》- （是否引证：否） 1.创] 使用Dockerfile构建镜像 标签： 虚拟化 docker 导入镜像	博主：aaron428 Dockfile是一种被Docker程序解释的脚本，Dockerfile由一条一条的指令组成，每条指令对应Linux下面的一条命令。Docker程序将这些Dockerfile指令翻译真正的Linux命令。Dockerfile有自己书写格式和支持的命令，Docker程序解决这 2014262593_尹羿_基于OSGi的政务云组件运行环境研究与实现_软件工程_武君胜 尹羿 -《学术论文联合比对库》- 2017-02-24（是否引证：否） 1.file可以在基础镜像层之上构建新的镜像层。Dockfile是一种被Docker程序解释的脚本，Dockerfile由一条一条的指令组成，每条指令对应Linux下面的一条命令。Docker程序将这些Dockerfile指令翻译真正的Linux命令。Dockerfile有自己书写格式和支持的命令，Docker程序解决这
14	此处有 41 字相似 条指令组成，每条指令对应Linux下面的一条命令，Docker程序解决这些命令间的依赖关系，根据指令生成定制的镜像。当有 额外需求需要重新定制镜像时，只需要在Dockerfile中增加或者修改指令并运行， 就可以重新生成镜像。本文的Dockerfile内容如图4.15所示。 from 172.16.1.41:5000/mo	2014262593_尹羿_基于OSGi的政务云组件运行环境研究与实现_软件工程_武君胜 尹羿 -《学术论文联合比对库》- 2017-02-24（是否引证：否） 1.ile这种显而易见的脚本更容易被使用者接受，它明确的表明image是怎么产生的。有了Dockerfile，当需要定制自己额外的需求时，只需在Dockerfile上添加或者修改指令，重新生成image即可，省去了敲命令的麻烦。在Docker Hub公共镜像仓库中有许多的已经实现的基础镜像，比如一个 Hadoop分布式集群的自动化容器部署研究 李杰;刘广钟; -《计算机应用研究》- 2016-01-08 1（是否引证：否） 1.ockerfile这种显而易见的脚本更容易被使用者接受,它明确地表明image是怎么产生的。有了Dockerfile,当需要定制额外的需求时,只需在Dockerfile上添加或者修改指令,重新生成image即可。Dockerfile的指令根据作用可以分为两种,即构建指令和设置指令。构建指令用于构建image
15	此处有 68 字相似 pi请求调用信息抽取服务返回的命名实体识别的结果，可以看到输入文本中的实体和实体类别都被正确预测、返回并显示。如输入为“ 9月9日美芝股份公告，控股股东、实际控制人、董事长李苏华向全体员工发出增持公司股票倡议书，鼓励公司及全资子公司全体员工积极买入公司股票。 ”，在调用信息抽取服务后返回的结果为标签为company的实体为美芝股份，标签为person的实体为李苏华，结果符合预期	荣科科技今年跌逾24% 实控人承诺兜底增持 证券时报记者 康殷 -《证券时报》- 2018-09-14（是否引证：否） 1.司员工平均薪酬为13.6万元/年。$$值得注意的是，今年9月以来已有美芝股份、花园生物推出兜底增持方案。$$9月9日晚美芝股份公告，控股股东、实际控制人、董事长李苏华向全体员工发出增持公司股票倡议书，鼓励公司及全资子公司全体员工积极买入公司股票，承诺凡在9月10日至10月15日期间净买入公司股票，且连续持有12个月以上并在职的员工，若因在前述期间买入公司股票产生的

指　标
疑似剽窃文字表述

1. Dockerfile文件，Dockerfile是一种可以被Docker程序解释的脚本，由一条条指令组成，每条指令对应Linux下面的一条命令，Docker程序解决这些
2. 额外需求需要重新定制镜像时，只需要在Dockerfile中增加或者修改指令并运行，

11. 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第11部分

总字数：3437

相似文献列表

去除本人已发表文献复制比：0%(0)　　文字复制比：0%(0)　　疑似剽窃观点：(0)

12. 6549072_安磊_基于深度学习的信息抽取服务系统的设计与实现_基于深度学习的信息抽取服务系统的设计与实现.doc_第12部分

总字数：1454

相似文献列表

去除本人已发表文献复制比：2.5%(36)　　文字复制比：2.5%(36)　　疑似剽窃观点：(0)

1	浆细胞样树突状细胞和LL37在IgG4相关性疾病中的研究 朱亚男(导师：张文) -《北京协和医学院博士论文》- 2017-06-01	2.5%（36） 是否引证：否

原文内容	相似内容来源
1	此处有 36 字相似 ，今后我即将从学生转变为一个从业者，希望我在学生时代学习到的知识、完成过的项目可以学以致用，为母校争光，为自己添彩。 最后，再次感谢所有在我学习和工作中帮助过我的所有人，感谢你们的付出与帮助 ，也由衷地感谢在百忙之中评阅论文的各位专家和教授。	浆细胞样树突状细胞和LL37在IgG4相关性疾病中的研究_朱亚男-《北京协和医学院博士论文》- 2017-06-01（是否引证：否） 1.无旁虑，一直在这条路上走下去，作为他们的女儿和妹妹，我真的很幸运，他们是我永远的动力。51/53最后，再次感谢所有在我的学习工作中帮助过我的所有人，感谢你们无私的帮助，才有了现在的我，愿你们永远身体健康，工作顺利，心想事成。52/53独创性声明本人声明所呈交的学位

说明：1.总文字复制比：被检测论文总重合字数在总字数中所占的比例

2.去除引用文献复制比：去除系统识别为引用的文献后，计算出来的重合字数在总字数中所占的比例

3.去除本人已发表文献复制比：去除作者本人已发表文献后，计算出来的重合字数在总字数中所占的比例

4.单篇最大文字复制比：被检测文献与所有相似文献比对后，重合字数占总字数的比例最大的那一篇文献的文字复制比

5.指标是由系统根据《学术论文不端行为的界定标准》自动生成的

6.红色文字表示文字复制部分;绿色文字表示引用部分;棕灰色文字表示作者本人已发表文献部分

7.本报告单仅对您所选择比对资源范围内检测结果负责

amlc@cnki.net
http://check.cnki.net/
http://e.weibo.com/u/3194559873/