**2018年全国大学生信息安全竞赛**

**作品报告**

**作品名称： 文档指纹及其在微信公众平台上的应用**

**电子邮箱： github上传清除**

**提交日期： 2018年5月29日**

填写说明

1. 所有参赛项目必须为一个基本完整的设计。作品报告书旨在能够清晰准确地阐述（或图示）该参赛队的参赛项目（或方案）。

2. 作品报告采用A4纸撰写。除标题外，所有内容必需为宋体、小四号字、1.5倍行距。

3. 作品报告中各项目说明文字部分仅供参考，作品报告书撰写完毕后，请删除所有说明文字。(本页不删除)

4. 作品报告模板里已经列的内容仅供参考，作者可以在此基础上增加内容或对文档结构进行微调。

5. 为保证网评的公平、公正，作品报告中应避免出现作者所在学校、院系和指导教师等泄露身份的信息。

**目 录**

[摘要 1](#_Toc515625663)

[Abstract 3](#_Toc515625664)

[第一章 作品概述 5](#_Toc515625665)

[1.1背景介绍 5](#_Toc515625666)

[1.2相关研究 6](#_Toc515625667)

[1.3特色描述 6](#_Toc515625668)

[1.4应用前景分析 7](#_Toc515625669)

[1.5本文组织结构 8](#_Toc515625670)

[第二章 作品设计与实现 10](#_Toc515625671)

[2.1系统结构 10](#_Toc515625672)

[2.2技术原理 11](#_Toc515625673)

[2.2.1 Function-Word LDA（FWLDA） 12](#_Toc515625674)

[2.2.2同义词林 17](#_Toc515625675)

[2.2.3 GloVe & TFIDF 21](#_Toc515625676)

[2.2.4变分自动编码器 24](#_Toc515625677)

[2.2.5异常检测（孤立森林算法） 26](#_Toc515625678)

[2.3实现方案 30](#_Toc515625679)

[2.3.1数据集介绍 30](#_Toc515625680)

[2.3.2算法实现 32](#_Toc515625681)

[2.3.3小程序部署 35](#_Toc515625682)

[2.3.4服务器配置 37](#_Toc515625683)

[第三章 作品测试与分析 38](#_Toc515625684)

[3.1测试方案 38](#_Toc515625685)

[3.2观察与结论 47](#_Toc515625686)

[第四章 创新性说明 50](#_Toc515625687)

[4.1 文档指纹概念的提出 50](#_Toc515625688)

[4.2 应用场景的创新 50](#_Toc515625689)

[4.3 算法创新 52](#_Toc515625690)

[4.4产品创新 53](#_Toc515625691)

[第五章 总结 54](#_Toc515625692)

[参考文献 55](#_Toc515625693)

# 摘要

当前网络已经是大众获取信息的主要来源，微信、知乎等公众发布平台规模也日益庞大。与此同时，网络虚假信息、文章审核与文章抄袭等信息内容安全问题也愈发突出。上述问题在现阶段依赖“算法预处理+人工识别”等方式解决，无法及时处理当下公众发布平台上海量的文章。本项目组认为，上述问题的核心是对文档信息的抽取。为此，我们提出了“文档指纹”的概念。

“文档指纹”旨在通过对文章内容与用词的分析，从实词和虚词两个维度来衡量文章作者的写作风格、写作主题以及写作习惯并将其使用向量进行表示，成为该文档独一无二的“文档指纹”。

项目组自主设计了一套算法流程，实现了上述“文档指纹”。首先，虚词和标点的使用是作者写作风格的一部分体现，因此我们创新地将隐狄利克雷分布（LDA）算法应用在了文档的虚词，得到了一部分写作风格维度。之后，由于作者对于同义词的使用有不同的习惯，因此我们设计了“同义词林”算法进行另一部分写作风格维度的提取。接着，对于文档的主题信息，训练了GloVe词向量并将其用文章中词的TF-IDF作为权重叠加，得到关于文章主题的向量。最后，使用变分自动编码器（VAE）对上述向量进行混合和降维，最终得到“文档指纹”。

项目组在微信公众发布平台为“文档指纹”找到了三种重要应用场景：文章审核、抄袭检测、相似推荐。第一，当前微信等公众平台的文章审核均使用人工手段，效率低成本高，对此可根据公众号新发布文章“指纹”与历史“文档指纹”的比较结果，达到对文章快速审核的效果，实验中平均正确率在88%左右；第二，当前公众信息平台对文章抄袭均依赖于事后举报，本项目对此情况可进行前期检测。我们通过将新发布文章的“文档指纹”快速同数据库已有“文档指纹”对比，从而检测是否存在抄袭嫌疑；第三，由于微信当前并没有十分有效的公众号文章推荐算法，无法利用微信公众号的大量资源，因此我们根据用户阅读文章喜好，选择与当前文章“文档指纹”距离最近的部分文章，为用户提供文章的推荐信息。

最后，我们使用微信小程序实现上述应用场景。项目组设计的小程序界面友好，同时能够兼容微信公众平台的信息格式，从而便于用户在微信公众平台直接实践“文档指纹”的文章审核、抄袭检测和相似推荐的功能。

**关键词：**文档指纹，内容安全，自然语言处理

# Abstract

Internet has already been the main source used by the public to get information. Wechat, Zhihu and other information platforms are growing rapidly. However, false information, article review, and plagiarism have been serious problems. At the present stage, solving these problems is mainly relied on "algorithm preprocessing + manual recognition". However, it is impossible for people to check mass data on the platform timely. Our project team believes that the core of these problems is extraction of document information. So we propose the concept of "document fingerprint".

Our project team has designed a set of algorithmic processes to implement our "document fingerprint". Firstly, function words and punctuations can show information of an author's writing style. Therefore, we innovatively apply Latent Dirichlet Allocation (LDA) to function words of one document and obtain a vector which represents a part of the writing style. Later, because different authors have different writing habits on synonyms, we design the "Synonym forest" algorithm to extract another part of the author’s writing style. Then, to get subject information of documents, we train the GloVe word vectors and overlap with the TF-IDF of words in the article to obtain a vector describing the subject of the article. Finally, we use the VAE (Variational Auto Encoder) to blend and reduce the dimensions of these vectors, and get “document fingerprint”.

Our project team has found three important applications for “document fingerprinting” on the WeChat Admin Platform: article review, plagiarism detection, and similar article recommendation. First, WeChat and other public platforms use artificial means to review articles, but this is inefficiency and costly. To review articles quickly, we design to compare the newly published article’s “fingerprint” and historical “document fingerprints”, and if they are similar, we consider the new article is normal which can pass artificial review. The average accuracy rate in the experiment is about 88%.

 Second, current public platforms rely on post-reporting for article plagiarism. Our project can detect this situation earlier than traditional methods. We quickly detect plagiarism by comparing the “document fingerprint” of the newly published article with the “document fingerprints” in the database. Third, WeChat cannot take full advantage of WeChat admin platform’s articles because WeChat does not have a valid recommendation algorithm. In order to solve this problem, we select articles which have closest distance measured by “document fingerprint”, and then provide users with article recommendation information.

**Keywords:** Document Fingerprint, Content Security, Natural Language Processing

# 第一章 作品概述

1.1背景介绍

随着互联网的快速发展和移动终端的迅速普及，网络逐渐成为了人们获取信息和资讯的首要方式。在此基础上，依赖于网络媒介的新媒体迅速发展起来，同时微信公众号和微博等为新媒体和自媒体提供平台服务的企业也越来越多。目前数以亿计的用户每天通过移动互联网发布自己的作品，移动互联网俨然已经成为现阶段最重要的信息传播媒介及渠道之一。

在媒体运营者范围极广且文章发布量爆炸式增长的情况下，文章发布平台如何进行有效的版权管理，以及媒体运营者如何防止账号被盗发恶意文章，平台如何对账户的异常行为进行监测就成为了重要且亟待解决的问题。网络时代媒体撰写的文章很容易被其他媒体在没有授权的情况下随意转载，侵犯了版权的同时也助长了抄袭的不良风气，损害了发布平台的声誉。而账号被盗发则可能带来粉丝的流失，造成经济损失。腾讯2017年发布的《知识产权保护白皮书》显示，自2015年1月至2016年12月，微信共收到针对公众号文章侵权投诉61000余件，其中著作权侵权投诉占总投诉量的41%。2016年3月20日，包括“科技每日推送”、“视觉志”、“小果哥哥”、“深圳微时光”、“法制晚报”等知名公众号被盗，向用户推送了大量商品广告链接，导致被多名用户投诉。2018年1月，今日头条招募大量内容审核编辑来监控平台内容是否合规，然而这种人工检测方式高费用低效率。2018年5月28号，刚刚接受腾讯投资的“差评”公众号运营团队因“洗稿”事件被曝光，而申请退还投资。微信公众平台的版权保护再次成为焦点问题。解决上述三类问题，对于保护发布平台、媒体运营者的权益，维护良好网络环境，都有十分积极且重大的意义。

这类问题的一个解决思路是使用技术手段将文章和发布媒体进行精准匹配。打个比方，也就是使用媒体运营者发布的文章生成一个数字的“文档指纹”，相同发布者的文章“文档指纹”相似度很高，而我们认为同一个账户的创作运营者会保持基本稳定，从而可以将不同文章划分到同一媒体，即实现文章和发布媒体的精准匹配。利用该指纹对该媒体发布的文章进行实时监测，就能达到有效管理的目的。本产品也将基于此进行探索和实践。

1.2相关研究

文档指纹概念可以理解为一篇文章生成唯一的标识符号，对于不同目标的应用有不同的实现方式。本项目“文档指纹”概念旨在通过对文章内容与用词的分析，来衡量文章作者的写作风格、写作主题以及写作习惯。现有其他算法和产品虽然名字与文档指纹相关，在目标、效果、实现方面与本项目“文档指纹”均有所差异，在此进行一定的分析：

首先，文档指纹的概念在复制保护系统COPS (copy protection system)中有所提及，COPS使用基于比较字符串相似性的算法。这类算法使用字符串匹配和句子匹配算法来找到重叠文档[1]。

此外，在抄袭检测方面，也有使用文档指纹的概念进行判断其使用部分文件来检查是否抄袭的例子。检查的部分将通过特定的机制（字符匹配）来确定欺诈行为。有两种基于指纹的算法是指纹算法(fingerprint)和筛选算法(Winnowing)[2][3]。其中, fingerprint不依赖于原始整体文章，而是对文章关键词组分布频率与其对应的词组类进行分析和比较。文章关键词与先验频率之间的误差经过编码之后成为文章的文档指纹。winnowing 算法则使用k-gram对文档切分，每个序列计算hash值作为一个文档指纹。接着对文档分窗，选择hash值代表文档指纹，以此保证重复部分可被有效检测。

最后，文档指纹的概念在结构匹配方法中也有提及。上述两类算法侧重于利用文档的词法特性，但结构匹配方法侧重于结构上的特征，如标题、段落等[4]。此类方法已经有较为成熟的应用，现有包括微软的Document Fingerprint[5]、 Digital Guardian公司的FILE FINGERPRINTING[6]等产品。

然而，上述研究虽然也使用文档指纹以及类似的名字，但其对文档指纹具体的实现方式与本项目提出的基于写作风格、用词习惯与写作主题的文档指纹大相径庭，在原理与应用等方面均有显著差异。

1.3特色描述

由于微信公众号作为媒体平台的快速成长，本研究产品选择了公众号平台作为主要实验对象，采用LDA、GloVe、同义词林、VAE等算法生成“文档指纹”，并通过微信小程序进行项目展示。本产品的特色主要有以下四点：

1. 概念上，创新性地提出了“文档指纹”的概念。“文档指纹”旨在通过对文章内容与用词的分析，来衡量文章作者的写作风格、写作主题以及写作习惯，从而实现文章与作者的精准匹配。
2. 功能上，我们将“文档指纹”应用到了微信公众号平台上，成功实现了微信公众号的异常发文检测和抄袭/转发检测。同时，由于“文档指纹”本质上实现了文章和作者的精准信息抽取，因此其实际应用的可扩展性极强。据此，我们还通过文章向量之间的相似度的分析，同时实现了相似推荐的扩展功能。
3. 算法上，我们采用了NLP与机器学习领域的一些算法，并自主设计了一整套指纹生成的算法流程。与以往单独分析文章主题不同，我们同时考虑了文章作者的写作主题和写作风格。一方面利用GloVe算法，将文章中主题词的GloVe词向量以其TFIDF值作为权重相加，得到关于主题的维度；另一方面，创新性地使用Function-Word LDA，对虚词提取写作风格的部分维度；此外我们还设计了同义词林进行同义词统计与降维，从而来得到写作风格的另一部分维度。最终我们使用基于神经网络的VAE编码器实现了信息的混合与降维。
4. 在演示产品上，我们选择了推出时间不久的微信小程序作为展示平台。一方面，微信小程序与我们的实验对象微信公众号平台一致，另一方面，这样的方式抛开网页等较为传统的展示方式，直接在手机端进行演示，方便快捷、用户友好且展示效果更佳。

1.4应用前景分析

本项目总体思路希望能够通过计算“文档指纹”，从文章主题、作者写作风格这两个维度对一篇文章进行综合评价。之后用于需要同时对文章以上两个维度进行综合评价的环境。应用领域包括舆论安全方面、知识产权保护方面、推荐系统等，具体的应用场景包括但不限于文档审核，抄袭/转发预警，相似推荐。下面进行针对性的分析。

1. 文档审核。如何对文章内容进行鉴别一直是信息发布平台较为棘手的问题之一。当前技术条件难以实时有效对文章内容进行审核，相关删除措施往往是亡羊补牢，效果不好。文档审核旨在维护良好网络环境，在萌芽阶段进行发现异常信息，检测保护公众平台账户安全，提高对虚假信息的响应速度。对账户新发布的文章进行“文档指纹”的计算，如果与该账户之前“文档指纹”类似，则可以快速通过，如果差异过大，则可以提请人工审核。可以规避现有人工审核效率低成本高的缺陷，同时也可以有效提高检测的精确程度。
2. 抄袭/转发检测。抄袭/转发检测目的在于保护各类文章作者的著作权。在当前的技术条件下，各类信息平台保护知识产权只能依靠事后举报等人工手段。时间延迟大，处理效率低，判断标准难以掌握。这些缺陷在客观上导致侵权成本低，作者维权成本较高。“文档指纹”则在这些方面有一定的优势。信息平台可以对所有新上传的文章计算“文档指纹”，之后与数据库中已有“文档指纹”数据进行比对，如果发现同某些文章相似程度过高，则可以进行人工审核等方式进行进一步甄别，可以有效遏制对他人知识产权的侵犯。
3. 相似推荐。相似推荐则是应用在现有各类信息平台用户推荐算法的一种补充。当前推荐系统主要使用各类协同过滤算法或深度学习方法，主要考虑文章主题、用户关系等。准确性较高，但却没有体现作者的写作风格。另一方面，从微信公众平台的角度，其“看一看”推荐功能始终并不成熟，未能对海量的微信文章资源进行合理的利用。我们因此设计了基于“文档指纹”的推荐系统，其能够对主题与风格进行兼顾，因此不仅与现有推荐算法形成互补关系，且能利用到微信的文章资源。

通过上述分析，“文档指纹”在现有环境中拥有较好的实际应用场景，也在测试中拥有良好的测试表现，能够满足以上各类应用的要求。

1.5本文组织结构

本文共分为五章。

第一章是作品概述。这一部分项目背景，说明了“文档指纹”希望解决的异常发布检测、抄袭/转发预警、相似推荐问题。之后介绍了其他学者的相关工作，对比说明了我们项目的特色之处，同时对“文档指纹”进行了应用场景分析。

第二章是作品设计与实现。首先概要地介绍了系统方案，详细介绍了各个系统模块和生成“文档指纹”的流程。然后说明了本产品的技术原理，对本产品实现过程中主要使用的五类算法——Function-Word LDA、GloVe、同义词林、VAE与孤立森林进行了深入介绍，并说明了使用这些算法的原因和优势。最后介绍了本产品的实现方案。这一部分首先说明了本产品实验过程中采用的数据集。之后分三个步骤详细地介绍了使用数据集成功生成“文档指纹”向量的具体过程，包括对文章的分词、分别得到文章写作主题维度和写作风格维度以及使用VAE自动编码器进行信息混合。最后介绍了用于演示的微信小程序的编写，包括前端交互界面的编写以及后端服务器端的搭建，同时说明了该微信小程序的演示运行过程。

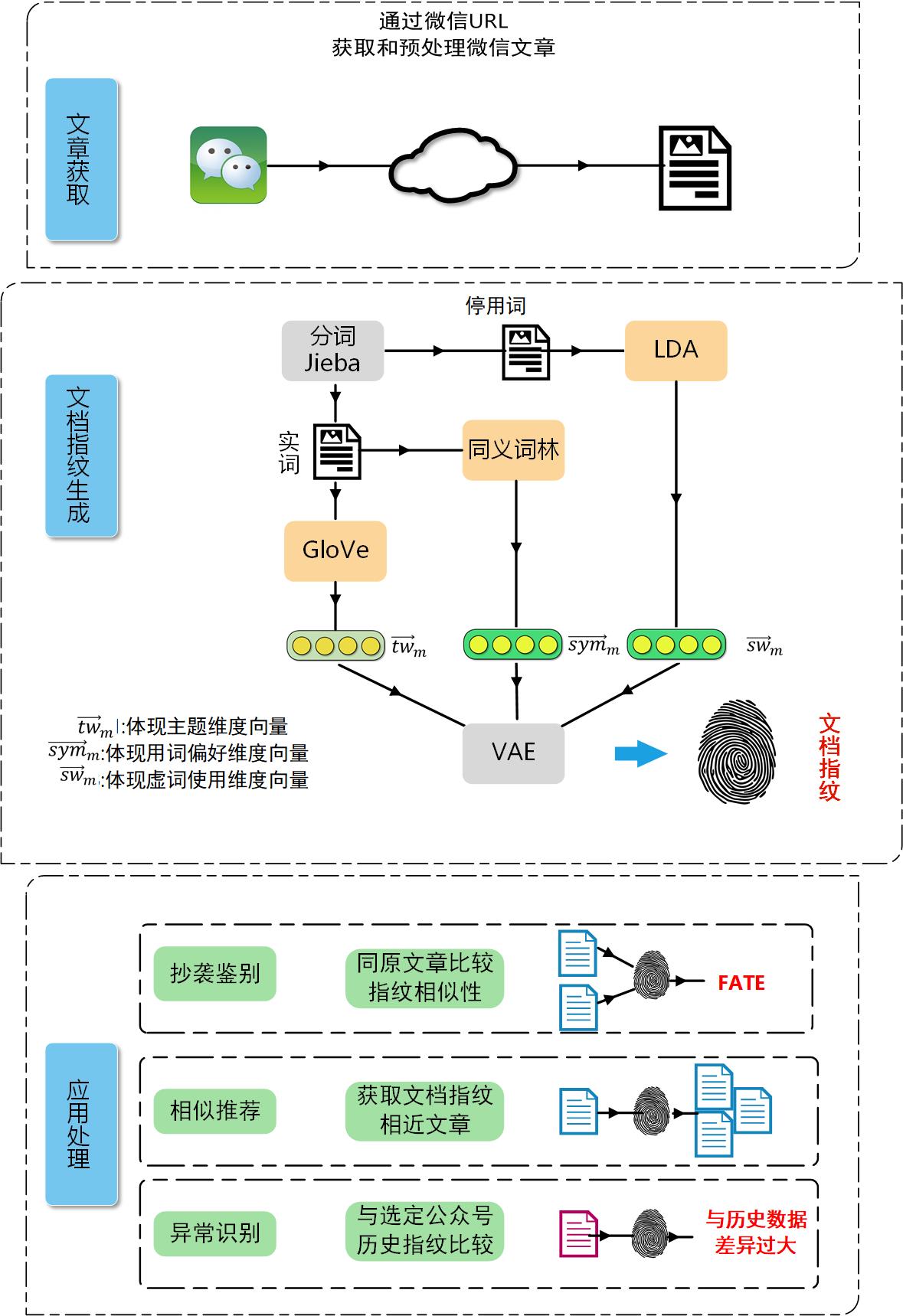
第三章是测试结果与分析。本章首先介绍了我们的测试环境和测试设备，接着对于异常发布检测、抄袭/转发检测和相似推荐三个功能分别在微信小程序上进行了实际测试。然后分析基于数据集得到的算法结果，用详细的动态流程图展示了实际使用中本产品提取出“文档指纹”并进行判别的每个步骤的具体结果。通过对算法总体结果数据的分析说明了我们算法的有效性。最终在观察与结论中指出了本产品仍存在的不足。

第四章是创新性说明。这一部分是对于本产品特色的更加详尽的阐述，从概念创新、功能创新、算法创新和产品创新四个维度深入地介绍了本产品的创新之处，同时在功能创新和算法创新的维度上还与现有的解决方案进行了对比。

第五章是总结。这一部分对本文的工作进行了总结，总结了设计本产品的总体思路和方案，以及本产品实际测试的效果，并指出了本产品仍存在的不足之处。

# 第二章 作品设计与实现

2.1系统结构



**图 2-1 系统结构框图**

图2-1为本系统工作的流程框图，系统分为三个模块，分别是数据收集以及预处理模块、文档指纹生成模块以及应用处理模块。数据收集模块和应用处理模块是实际应用过程中与用户交互的过程，生成“文档指纹”部分是整个系统的核心部分。

每个模块内部是自身的实现流程。在数据收集和预处理部分，我们的程序会根据微信的URL去获取对应URL的文章 ，之后将文章传递到服务器端的数据库，之后开始进行“文档指纹”的分析计算。

“文档指纹”生成部分。生成“文档指纹”的第一步是分词。我们认为“实词反映文章的主题，虚词代表文章作者的写作风格”。因此使用jieba对文章分词之后，将实词和虚词分别存储进行下一步处理。

针对于虚词与标点，我们使用LDA(Latent Dirichlet Allocation)进行特征提取。LDA本身用于主题提取，但我们这次将LDA算法活用在虚词与标点上进行计算，得到一个代表作者语言风格的向量。

针对实词部分，我们将其再分为两个部分来处理。在汉语言中有大量的同义词，我们认为不同作者有自己的行文习惯，在同义词使用上有差异，这种习惯不能用简单的词袋模型获取。对于这部分信息我们通过同义词林进行同义词使用的统计，并进行基本的降维，得到与写作风格相关的另一部分维度。针对主题方面的分析，我们使用GloVe（Global Vectors for Word Representation）获得词向量，之后使用TF-IDF（Term Frequency-Inverse Document Frequency）计算得到代表主题的向量。

经过处理之后我们分别得到了代表语言风格和文章主题的向量，之后我们采用变分自编码器（Variational auto-encoder，VAE）将这三个向量进行降维和混合，得到能够代表整篇文章的一个向量，这就是我们项目所希望获得的“文档指纹”。

在应用处理模块，我们会根据所需要的应用来对“文档指纹”进行比对、检索等处理。以之前提到的三个应用为例。抄袭鉴别我们会对怀疑文档和数据库内文档进行“文档指纹”的对比，根据相似程度给出是否抄袭的判断；相似推荐则会在数据库中检索已有文章推荐“文档指纹”相近的文章；异常检测应用则是将最近文章与此账户已有文章“文档指纹”进行比对，如果相差不大，则可以快速通过审核，否则提请监管人员进行手动审核，确保账号的安全。

2.2技术原理

2.2.1 Function-Word LDA（FWLDA）

对于一篇文章而言，存在一个写作作者的写作风格的特点，这个特点应当要体现在文档指纹之中。根据已有研究[7][8]，作者的虚词、标点甚至俚语的使用对于作者的写作风格的提取相当有效，并且虚词与标点的特征能够有效避开文章主题对于写作风格提取的影响。

传统的对于虚词或标点的特征提取多基于直接的统计，之后将这个高维的向量输入SVM或神经网络进行进一步操作[9]。然而，这样的做法却不能提取出隐含在作者虚词与标点背后的抽象出的写作风格。基于这样的问题，在本项目中，我们创新性地将著名的主题提取算法隐狄利克雷分布（LDA）运用在了文章的写作风格的提取上，并获得了非常优秀的效果。

LDA（Latent Dirichlet Allocation）[10]是一种文档主题生成模型，由David M. Blei、Andrew Y. Ng与Michael I. Jordan三人于2003年提出，提出后便受到广泛关注，其各种变体也非常多。

从模型上来说，LDA本质上是一个三层贝叶斯概率模型，包含词(w)、主题(t)和文档(d)三层结构。所谓生成模型，就是说，我们认为一篇文章的每个词都是通过“文章以一定概率选择了某个主题，并从这个主题中以一定概率选择某个词语”这样一个过程得到。通过不断地进行上述采样与生成过程，我们就能得到我们所看到的一篇文章。

实际应用场景中，词(w)与文档(d)都已知，LDA则能够通过上述采样生成过程的逆过程，将每一篇文档归化为一些主题所构成的一个概率分布，而每一个主题又归化为很多单词所构成的一个概率分布。因此从学习算法的分类角度而言，LDA是一种无监督机器学习技术。

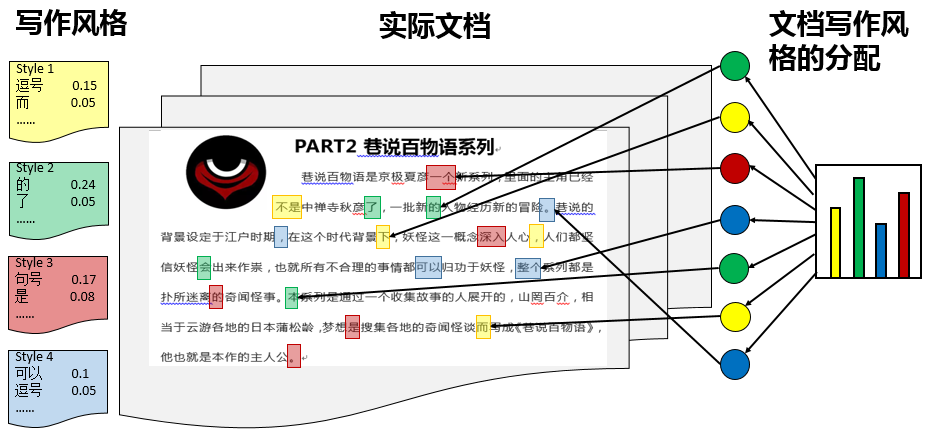
LDA与此前的类似主题算法如PLSA[11]相比，最大的区别在于，文档到主题服从多项式分布，主题到词服从多项式分布，而这两个多项式分布是分别从两个Dirichlet分布中采样得来的。也就是说，这个三层结构每两层之间的分布过程均采用一个Dirichlet-Multinomial共轭结构。这样的结构有益于我们引入先验知识，减少过拟合的可能，且对于我们采用Gibbs采样进行快速的LDA实现也有着极为关键的作用。

本项目组经过对LDA算法和现有的写作风格提取算法的考察，认为LDA算法能够被应用在写作风格的提取上。这里，我们创新性地将LDA算法应用在虚词与标点符号上，用来提取作者对于虚词的使用所反映的其独特写作风格，这个新应用场景的LDA被我们称为Function-Word LDA(FWLDA)。

Function-Word LDA (FWLDA)的基本思路是从传统的LDA引申而来的。传统的LDA是一个词(w)、主题(t)和文档(d)三层结构，是“文章以一定概率选择了某个主题，并从这个主题中以一定概率选择某个词语”的过程。这样的LDA主要对于文章中的主题词（实词）进行某种隐藏特征的提取，最终提取出来的就是文章的主题特征。

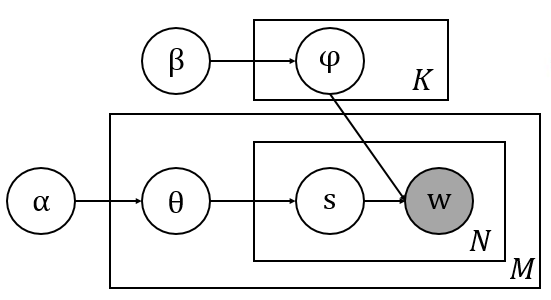
而FWLDA则认为，文章生成时对于虚词的提取，同样满足LDA所提出的生成模型，即作者对虚词与标点的使用，是从自己隐含的写作风格中进行抽样。因此，FWLDA的三层结构分别是词(w)、写作风格(s)和文档(d)，是“作者以一定概率选择了某个写作风格，并从这个写作风格中以一定概率选择某个虚词或标点”的过程。因此，如果我们将一篇文章中所有的虚词与标点提取，并使用LDA算法进行特征提取，此时文章对应隐层(写作风格style)的概率分布就应当是该文章的写作风格的向量。FWLDA的基本思路见图2-2所示。

值得一提的是，FWLDA这样的LDA提取写作风格的方式在以往研究中很少有人采用，属于本项目组在算法上的独立想法与创新。



**图 2-2 FWLDA的基本思路图**

接下来我们阐述一下FWLDA的算法。我们假设文档集中有M篇文档，每一篇文章对应的写作风格的概率为：，在选定了文章的写作风格后，对于一篇文章，可能会有几个写作风格，此时，在写作风格确定的条件下，选择每个写作风格对应的虚词（或标点，下文简称为虚词），对于第m篇文章中的第n个虚词，有其所属写作风格的编号，假设有K个写作风格，在每个写作风格下选择虚词的概率为：，在对应的第k个写作风格下依据概率选择虚词。这里，文档与文档之间是相互独立的，同一个文档中的词与词之间也是相互独立的，如图2-3所示。



**图 2-3 FWLDA的贝叶斯网络图**

值得一提的是，通过FWLDA算法，我们最终的目标是获得一个向量，这个向量能反应文档m的写作风格。在这里，我们认为LDA中，的期望，即文档m相对各隐风格的概率分布，就能体现其真实写作风格。同时如果该算法有效，同一作者的文档与，与应当相当相似（实验中我们证实了这一点，并且效果非常好）。

这里，我们需要明确以下定义与标记。词库中（文档中所有的虚词与标点符号，下文中有时简称虚词）共有V个虚词，而词库中每一个词出现的次数记为：，所有词出现的总次数为N。第k个写作风格下对应的词的概率为：，其中。第m篇文档对应的写作风格的概率记为：，其中。此外，对于每一篇文章中对应的虚词所属写作风格的编号为：。

原始的LDA算法的特点在于三层结构每两层之间的分布过程均采用一个Dirichlet-Multinomial共轭结构，FWLDA也沿用了这个特性。正如图2-3所示，在FWLDA中同样有两个共轭结构。表示的是对于第m篇文档，我们首先根据参数计算出其对应的写作风格的概率，然后生成该文档中的第n个虚词对应的写作风格的编号；表示的是根据参数计算出在写作风格编号确定的条件下该写作风格对应的虚词的概率，依据这个概率选择出每个虚词（这里指的虚词均包含标点）。对于上述过程中的两个共轭结构，每个结构的第一阶段都是从Dirichlet分布中抽取多项式分布，因为服从参数为的Dirichlet分布，服从参数为的Dirichlet分布。而每个结构的第二阶段，则是利用多项式分布实际抽样写作风格s与词w。如此，FWLDA中Dirichlet分布与多项式分布总共构成了两组共轭结构。

在LDA中，我们最为关心的是在已知词W的情况下，获得写作风格S的分布，即条件分布。为了获得这个条件分布，我们首先需要获得联合分布。下面我们将导出这个联合分布。

对于整个文档集来说，文档与文档之间是相互独立的，单个文档中词与词之间也是相互独立的，因此上述的两个共轭过程我们可以分解成如下的两个过程：

1. 首先对于M篇文档生成其对应的词对应的写作风格的编号。

2. 对于K个写作风格，生成所有的文档虚词（包括标点）。

有了上述的两个过程的分解，对于整个文档集，我们可以得到下述的文档生成概率：

（2-1）

其中，表示的是在文章写作风格确定的条件下生成词的概率，表示的是文档对应写作风格的概率。对于上述的第一个过程有：

（2-2）

已知服从的是多项式分布，而服从的是Dirichlet分布，因此，上式可以转换成为：

（2-3）

其中，先验分布为Dirichlet分布，有：

（2-4）

这里，称为归一化因子：

（2-5）

此外，表示的是第m篇文档中属于第k种写作风格的虚词（或标点）的个数。则对于所有的M篇文档有：

（2-6）

对于第二个过程，有下式成立：

（2-7）

其中，服从的是多项式分布，而服从的是Dirichlet分布，则上式可以转化为：

（2-8）

其中，，表示的是属于第k种写作风格的虚词中词v的个数，对于所有的V个词，有下面的公式成立：

（2-9）

因此，对于整个文档，有：

（2-10）

接下来，有了联合分布，我们便可以通过联合分布来计算在给定可观测变量 w 下的隐变量 z 的条件分布（后验分布）来进行贝叶斯分析，得到我们需要的条件分布，这部分一般用Gibbs采样来实现。Gibbs采样是一种用来获取一系列近似等于指定多维概率分布的方法。由于文档篇幅限制，我们略去一定步骤，仅展示Gibbs采样的关键迭代式（这里表示除以外）：

（2-11）

经过Gibbs采样的迭代，我们最终可以得到Dirichlet的分布期望，也就是我们最终得到的某文档对应于某写作风格的概率矩阵θ，以及某写作风格对应于某虚词的概率矩阵φ。其各元素的值由下式获得：

（2-12）

在得到上述期望之后，我们就得到了FWLDA的最终结果。我们认为，对于给定的某文档m，其相对于各写作风格的概率期望就反映了其写作风格，可以将作为反应写作风格的维度。因此，我们最终得到的文档m的反映虚词的写作风格的维度为：

（2-13）

2.2.2同义词林

本部分使用汉语的同义词语义元数据，通过输入的文章对文章作者的语言偏好特征进行提取，并给出了比较给定文章语言偏好与指定作者语言偏好的算法，为判断作者身份提供参考。前期实验与整合后的测试显示，本算法对于识别不同作者的语言偏好具有较高的灵敏度。除了部署前的学习过程外，本过程处理新请求的算法的整体时间复杂度不会超过，具有良好的可伸缩性。以下讨论算法的细节。

每位作者的语言习惯受到其成长、生活环境影响，随着经验的积累日渐独特，并在较长的时间段内可以认为是稳定量。这种语言习惯很大程度上反映在作者在行文中对同义词使用的偏好，故某个自然人创作的文字集合中的同义词使用偏好包含可用于鉴别作者身份的信息，即作者的“思维指纹”。

本部分基于汉语同义词的自然语言语义信息，对输入文字集合中反映的同义词使用偏好进行统计，产生不同作者的同义词偏好特征信息。在应用中，希望能够对不同文字集合之间的相似性进行评价，并对声称为相同作者的文章集进行判别，给出异常或正常的提示。

本算法与目前常见的文本相似度比较算法存在两大区别：首先，本算法实现思路是基于自然语言语义的，其分析过程具有明确的实际意义，其他许多算法则基于统计，不考虑词语意义，或使用难以理解其实际意义的机器学习“黑箱”。其次，本方法进行特征识别的对象不是一篇文章，而是文章作者。文章计算结果是识别出文章作者特征。同指纹检验过程相似，指纹检验不是为了分类收藏指纹，而是为了获得指纹对应自然人的特征。

本算法需要已有自然语言语义信息作为先验知识，即获取经过语言学家论证的同义词词库。本项目使用哈尔滨工业大学社会计算与信息检索研究中心发布的同义词林，该词林收录了1400个同义词类别、共77343个词语的同义关系信息[12]。

算法首先对输入的文本集合进行切词操作，之后将词列表看作词袋，统计各个同义词类内部的概率分布信息。对于任意一个输入词袋，每一个同义词类C内部各个词语w的概率的总和满足如下关系，

(2-14)

这样统计结果与某个词类的总词频脱离了关系，这使得我们的统计结果不受文章主题的影响。因为总词频会与文章主题有较大关联。例如，“警察局”和“公安局”属于同一个同义词类（部分），某作者在某篇社会治安文章中大量使用了“公安局”一词，与此同时，该作者另一篇其他主题的文章中可能仅使用一次“公安局”，这里的词频差别不应该反映在提取的特征中。但是，如果一位作者从不习惯使用“警察局”一词，它却大量出现在声称是此人创作的文章中，我们的算法就应该敏锐地报告这个异常。

统计实现过程：首先，选取词林中第一个词语作为代表元唯一表示一个等价类。将信息记录在同义词字典（称为词字典）中。之后，根据同义词林建立各个同义等价类的字典（称为类字典）。词字典的词条对应一个词，类字典的词条对应一个等价类。对于类字典中的一个类词条，我们在其中维护等价类内成员的总出现频率计数器,还为类内任一个词语w维护一个计数器。每当从词袋里取出一个词，我们查询词字典（查询操作的时间复杂度为），获得该词的类信息。然后再查询类字典，将类计数器加1，再将类内的对应词计数器加1。最终与的关系应该满足：

(2-15)

由于既统计了等价类内部一个词语w的频率，又统计了整个等价类C的频率，故实际上统计了等价类内部的概率分布。每一个词语在一个同义等价类中被选用的概率为：

(2-16)

这样，当分析完整个词袋时，我们以时间复杂度完成了统计，其中n是文章内的实词总数。统计的结果为词袋中出现的所有同义等价类内所有出现的词语被选用的概率分布。根据之前的分析，这个分布应该能够体现作者在文章中遗留的同义词选用习惯的特征，即作者的“指纹”。

由于之前统计的结果有很明确的实际意义，故我们可以很自然地得出比较两个词袋与中同义词选择偏好差异的信息。这个过程如下：

首先，由上述算法，我们已经得到了两个词袋与各个共有的同义等价类内部的概率分布。我们遍历一个词袋中出现的同义等价类，根据上述过程建立的词典在另一个词袋中检查该等价类是否出现，并记录下所有共同出现的等价类（借助散列表，该过程的时间复杂度为）。

然后，我们找出这些等价类内出现的词语，求出与间各个词被选用的概率的差别（如果一文中出现了一个词语，另一文中没有，就将后者中该词的概率设为0）：

(2-17)

由于

故

定义词语w在、中任一个出现的标识函数如下：

(2-18)

将这个值累加，最后，我们根据累加的次数，求出累加值的平均：

(2-19)

最终的结果具有明确的意义，即两篇文章作者在一个同义等价类中选择用不同词的概率差距的平均值。整个计算过程的时间复杂度为，为所有同义词类内词语的总个数。

使用上述方法，我们可以依据清晰地意义计算出两个词袋对应的作者同义词偏好差距。但是，由于上面的方法是基于一对一的比较，当两边各有多个词袋时，对于两个词袋集合进行完全检查的时间复杂度就会变成，此处为一个词袋集合里的词袋数目。可见，上述方法相对于词袋集大小的可伸缩性很差。因此，有必要寻找新的方法解决大量词袋的比较问题。

我们将使用上述的统计数据制作文章的特征向量，来对相似性进行初步的筛选。上面的统计结果有一个天然的优势，就是概率的取值范围是规范的。故应该找出一些具有代表性的、最能够体现不同作者之间语言习惯差异的等价类，并以一个确定的序关系，依照上面的统计结果制作特征向量。

通过下述操作找出最能区分作者的等价类。我们通过比较不同作者的词袋集、中的词袋，将每一个词语对平均差异的贡献进行累加，得到每一个类的总贡献，于是，我们就能够依序找到最能够区分不同作者的等价类，并对它们的重要性进行排序。其中，统计总贡献的公式如下：

(2-20)

得到了同义词类的重要性顺序后，我们就可以根据类的排序制作向量。由于同一个类的成员数总是确定的，故向量的各个维度的意义得以与一个确定的词对应起来。我们选择一个恰当的维数，把上述词语差异的统计结果映射到一个高维空间中。由于同一篇文章并不会出现所有等价类，我们所得到的向量是稀疏的，即有许多维度的值为零。因此需要通过一个降维算法来从稀疏的向量中提取出重要的信息。

我们选择使用主成分分析（Principal Components Analysis，PCA）来对向量进行降维。这是考虑到PCA能够有效地处理稀疏的向量，并能将高维数据最有代表性的信息投影到一个维数低得多的空间上。通过这个方法，我们以令人满意的效率将原来3000维的向量集投影到了一个低维的空间中（实践中为30维）,得到的第m篇文章的同义词林维度为 。通过K-Mean聚类分析发现，上述方法有效地将文章的同义词偏好差异问题转换为低维空间里向量的距离问题。

2.2.3 GloVe & TFIDF

对于一篇文章而言，存在一个写作作者的关注点的问题，这个特点应当要体现在文档指纹之中。所谓作者的关注点，主要体现在作者对于哪些主题与领域比较感兴趣，而这样的兴趣往往会以文章主题的方式体现在文档之中。因此，有关文档主题的信息需要被提取出来，并以某种方式加入文档指纹之内。

对于主题提取，其方法非常众多。最早比较著名的有基于贝叶斯的LDA[10]、CTM[13]等模型。近年来，由于深度学习的火热，基于LSTM、GRU、CNN[14]、Recursive NN[15][16]甚至RCNN[17]、Attention机制[18]的文档主题提取与分类算法也越发成熟，并且取得了超过传统主题模型的效果。

而本项目组在文档指纹的主体部分维度提取中，采用了基于词向量叠加的主题提取方式，同样取得了非常优异的效果。大概而言，我们首先采用了GloVe算法[19]进行词向量训练，之后提取一篇文章内TFIDF值最高的个词汇，将这些“关键词”对应的词向量以正比其TFIDF值的权重进行加权，最终通过词向量叠加得到了关乎于文档主题的一部分向量维度，如图2-4所示。

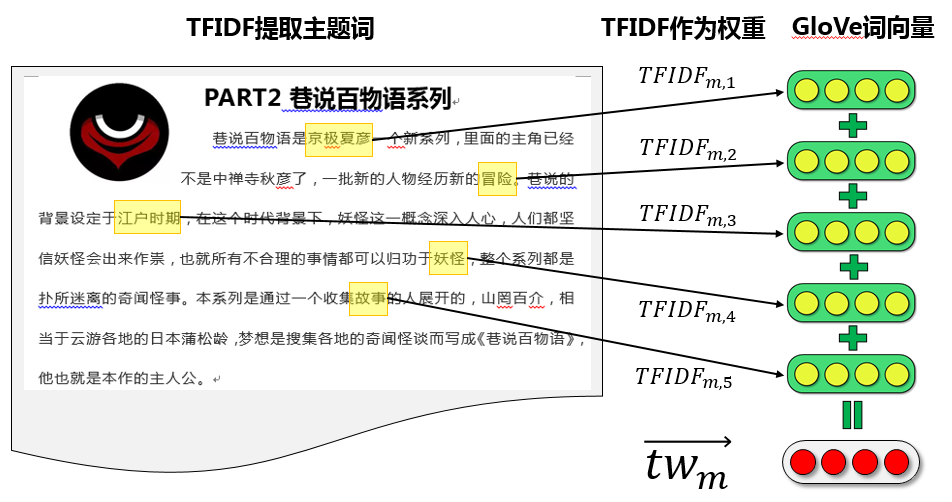


图2-4 GloVe+TF-IDF基本思想

词向量的作用简而言之，就是将自然语言中的词汇转为计算机可以理解的稠密向量，从而更好地体现出词汇之间的相似性与相互关系。此外，词向量的向量形式也能更好地被进一步模型（如各类神经网络、SVM等）所接受。

词向量的概念在上世纪八十年代被首次提出，中间经历了Bengio2003[20]，C&W[21]以及Huang2012[22]几次大的进步，该领域研究不断推进。但一直到2013年Mikolov提出word2vec[23]后，由于词向量的效果和训练速度得到巨大提升，词向量才开始被广泛使用。

本项目中，我们选用的词向量算法是GloVe算法。GloVe算法[24]是2014年斯坦福大学的Pennington等人提出的新一代词嵌入算法。相比以往的词嵌入算法，其训练速度快，对于大或小的语料库均能适应，对于长或短的词向量生成效果均不错。因此从实践角度，其效果往往更胜过word2vec(skip-gram)，这也成为了我们选用GloVe的原因。

GloVe算法的核心思想是，其不仅考虑了语料库的全局统计特征，也使用了局部的上下文特征（即滑动窗口），这是之前的词嵌入方法从未做到的。而为了做到这一点，GloVe模型引入了共现概率矩阵的概念，我们会在下面简述。

GloVe首先引入词与词的共现矩阵，的元素是语料库中出现在词上下文中（window内）的的次数；定义 是出现在词上下文中的所有词的总次数； 是词出现在词上下文的概率。由以上定义，就能引申出共现概率矩阵的概念。

我们假设，GloVe的作者发现，的取值是有一定的规律的，其规律如表2-1所示。

**表2-1 GloVe所使用的词汇共现性规律**

|  |  |  |
| --- | --- | --- |
|  | 与相关 | 与相关 |
| 与相关 | 趋近1 | 很大 |
| 与不相关 | 很小 | 趋近1 |

也就是说，值能够反映词之间的相关性，而GloVe模型就是利用了这个的值。不妨假设现在已经得到了、、的词向量（不妨就记为、、，其中）。GloVe认为，这三个向量通过某种函数的作用后所呈现出来的规律和 具有一致性，即相等，也就可以认为词向量中包含了共现概率矩阵中的信息。因此，我们需要定义一个函数使得。

GloVe中，该函数采用了指数形式进行定义，即：

(2-21)

两边取对数，则有：

(2-22)

此时，为了保证模型的对称性，并去除，GloVe的作者分别引入了两个偏执项，从而将模型变成了：

(2-23)

以上是理想情况。现实情况下，我们只能让左右两边更加接近，而不大可能完全相等。因此，我们引入了如下的代价函数J：

(2-24)

其中，函数用于给代价函数的项加权。如果两个词共同出现的次数越多，那么这两个词在代价函数中的影响就应该越大。原文给出的定义如下：

(2-25)

通过这个代价函数，我们不断迭代训练，最终即可得到我们想要的GloVe词向量。实践中，选择，。

得到GloVe词向量训练的结果后，我们需要提取一篇文章内TFIDF值最高的个词汇，将这些“关键词”对应的词向量以正比其TFIDF值的权重进行加权，最终通过词向量叠加得到文档主题相关的一部分向量维度。

TF-IDF（词频-逆文档频率）算法是一种统计方法，用以评估某一字词对于一个文件集或一个语料库中的其中一份文件的重要程度。我们记文档m中，词汇表内第i个词的TFIDF值为，其计算公式为：

(2-26)

其中意为在文档m中出现的数量，M为文档总数。得到在文档m中的TFIDF值后，我们就可以得到主题维度：

我们定义为一篇文章内TFIDF值最高的个词汇的序号的集合，即：

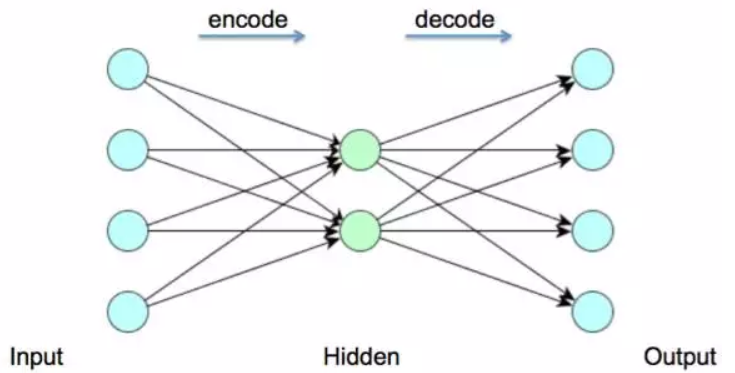
那么主题维度就可以如下得到：

(2-27)

2.2.4变分自动编码器

变分自动编码器（Variational Auto-Encoder），简称为VAE，是一类重要的生成模型，它于2013年由Diederik P.Kingma和Max Welling提出[25]，本质上是自动编码器（Auto-Encoder）的一个变种。

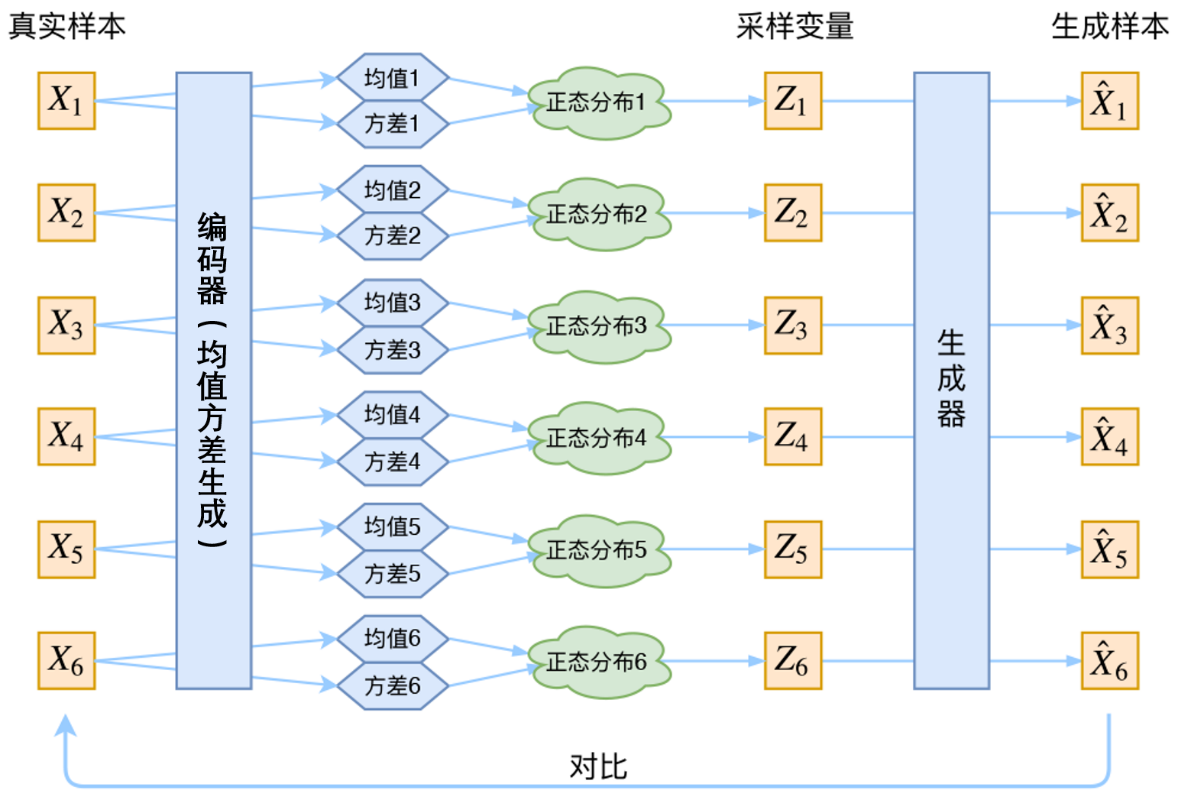
首先简要介绍自动编码器，它广泛应用在数据降维、数据降噪等方面。自动编码器是一种利用神经网络的无监督学习技术，其目的是得到原始数据的压缩表示。用下图来解释，左侧输入是我们的原始数据，在训练过程即编码过程中我们限制隐藏层的维度得到隐藏层向量，再通过隐藏层向量解码得到输出。如果输入输出之间的差异足够小，我们就可以认为隐藏层向量用更小的维度表达了原始数据足够多的信息，从而实现了对原始数据的压缩表示,如图2-5所示。



**图 2-5自动编码器示例**

自动编码器的问题在于它只能基于现有的数据集进行压缩表示，其得到的隐藏层变量可能不连续，也就无法通过对隐藏层变量的构造来生成不同的相似数据。换言之，它仅能压缩，不能进行数据的生成。变分自动编码器，即VAE，对隐藏层变量施加了一定限制，训练得到的是隐藏层变量的分布情况。我们通过对隐藏层变量进行取样，输入解码部分，就能构造一个满足要求的数据。下面我们会阐述VAE的主要思想。

假设观测到的数据集为，其中每个数据用来表示。我们可以认为存在一组变量，这组变量彼此间相互独立，而且决定了的表现，每给定一个就可以得到相对应的一个。这里的就是VAE训练过程中的隐藏变量。为了取样的方便，我们假设这些变量满足正态分布。



**图 2-6 VAE示例**

VAE的主要过程可以用图2-6加以解释。可以看到，通过原始数据经过编码器得到的是每一个隐藏变量的均值和方差，而解码器的输入是从每个隐藏变量的正态分布情况中取样得到的。由此可以看出，VAE相较于自动编码器有更强的表现能力。

我们用数学方式来描述VAE的工作。解码器的过程可以用下面的函数来表述：

(2-28)

上式中的是解码器对应的函数的参数。

我们希望得到的可以尽可能还原数据集，也就是说要使得得到数据集的概率最大，即最大化。采用最大似然估计的方式，即

(2-29)

由于右侧的积分难以计算，VAE在实际计算中使用了一个技巧。它通过编码器来尽量去逼近真实的后验概率分布，两个分布之间的差异由KL散度刻画，经过推导可以得到

(2-30)

由于与理想情况下趋同，从而上式左侧第二项越小越好。我们原本的优化目标是上式左侧第一项，该项越大越好。因此上式整个左边都可以看成我们的优化对象，该对象越大符合要求。这样我们可以只关注上式右侧的优化，而忽略上式难以计算的左侧第二项。

上式右侧第一项表征通过重建的分布，与原数据集越接近其值越大。在神经网络的实现中可以通过交叉熵刻画，越小表示与原数据集越接近。

(2-31)

上式右侧第二项可以表示为

(2-32)

因此在神经网络中实际优化的损失函数可以表示为

(2-33)

得到此损失函数尽可能小的结果为优。

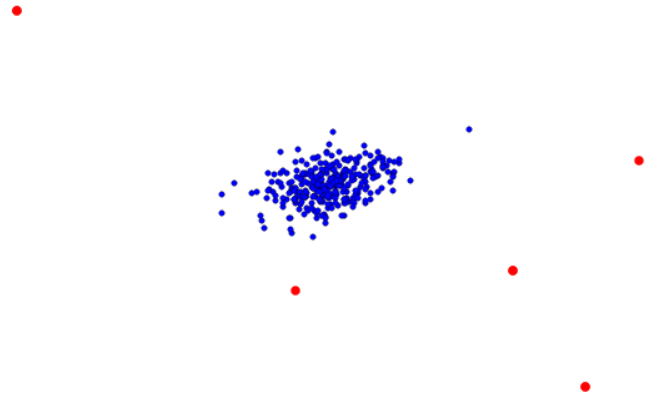
本项目中主要使用VAE来进行各方面信息的混合和降维表示。由于VAE相较于标准自动编码器有更好的表现能力，但工程实现开销不大，因此非常适用于本项目。

用VAE最终降维得到第m篇文章的“文档指纹”.

2.2.5异常检测（孤立森林算法）

异常检测，又被称为离群点检测，被广泛地应用在网络攻击检测、金融欺诈挖掘、医疗图像检测等领域。顾名思义，异常检测就是在大量正常数据的基础上检测出与大多数数据的情况偏差较大的数据，这类数据在整体数据样本中占比较小。用数学语言描述如下：

给定一组数据，训练一个模型，表示数据异常的概率。对于一个新的实例，如果，其中为异常阈值概率，则认为为异常数据。



**图 2-7异常检测说明图**

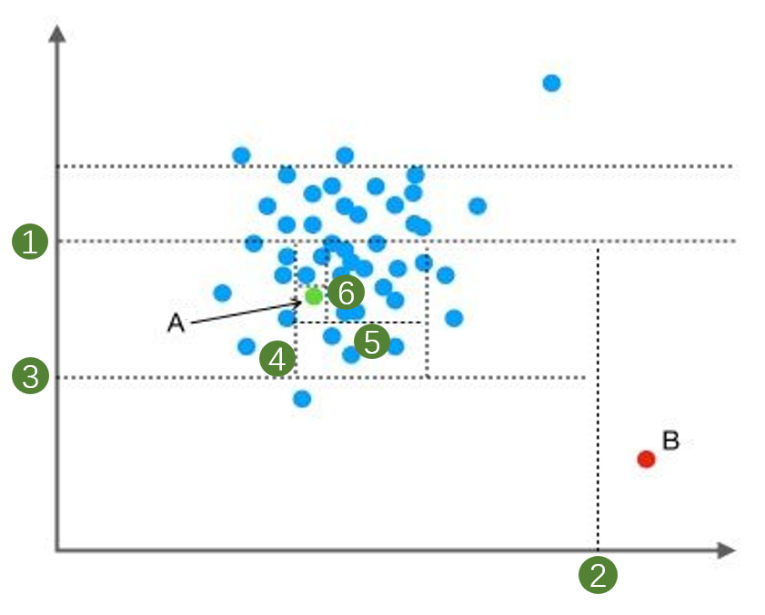
如图2-7所示，图中的红色数据点在距离上明显区别于蓝色点，我们就将这些红色数据点看作异常数据，或者说是离群点。

常见的异常检测算法包括Isolation Forest、Gaussian Distribution Detection、Nearest Neighbor、LOF、RNNs等。在本产品中异常检测应用在异常发文检测当中，选择使用的算法为Isolation Forest。在异常发文检测中，该算法的输入向量是训练得到的某篇公众号文章“文档指纹”向量的集合以及待检测文章的“文档指纹”向量，输出结果是一个百分比数值，该数值越大，异常可能性越高。

**Isolation Forest**

Isolation Forest，又被称为iForest，中文译为孤立森林，是一种无监督的异常检测算法。该算法在2008年由澳大利亚莫纳什大学的两位教授Fei Tony Liu, Kai Ming Ting和南京大学的周志华教授提出[26]。Isolation Forest能够快速处理大规模的数据，具有线性时间复杂度和高精准度，因此在异常检测算法中享有盛名。

刻画异常数据最直接的做法是利用各种统计的、距离的、密度的量化指标去描述数据样本跟其他样本的疏离程度。而Isolation Forest尝试直接去刻画数据的“疏离”程度，而不借助其他量化指标。它的基本思想是假设我们用一个随机超平面来切割数据空间, 切一次可以生成两个子空间。之后我们再继续用一个随机超平面来切割每个子空间，如此循环，直到每个子空间里面只有一个数据点为止。直观上来讲，那些密度很高的簇被切很多次才会停止切割，但是那些密度很低的点很容易很早地就停到一个子空间里了。



**图 2-8 iForest算法思想说明**

如图2-8所示，该数据集是一个二维数据集，每次随机选取一个维度并随机选取一个维度值做分割，图中绿色圈中的数字表示分割的次数。可以看到点B在第二次分割的时候已经被“孤立”，而点A在第六次分割时仍未被“孤立”。我们可以认为，点B代表异常数据，而点A代表了正常数据。

该算法的具体实现过程分为模型训练过程和异常检测过程。

在模型训练过程中，依照上面描述的算法思想构造一定数量的iTree，具体过程为：

（1）训练数据随机选择个样本（一般不放回）作为构造iTree的根节点；

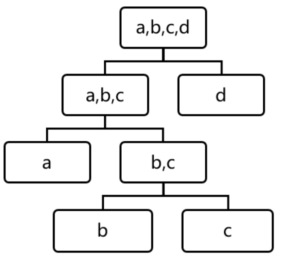
（2）随机指定一个维度并随机产生该维度最大值与最小值之间的切割点；

（3）指定维度小于切割点的数据放在左子节点，大于等于的放在右子节点，完成一次划分；

（4）在子节点中递归（2）和（3），直到子节点中只有一个数据或达到指定的高度，至此一个iTree生成完毕。

（5）继续（1）—（4）过程直到生成指定数量的iTree，训练结束。

异常检测过程就是对每一个测试数据遍历每一棵iTree，记录其落在每一棵iTree的叶子节点的高度，该高度定义为到达叶子节点经过的边数。



**图 2-9 iTree异常检测**

简单地用图2-9进行说明。图中a、b、c、d均为测试数据，表示的是四个数据遍历一棵iTree的情况。可以看到b和c的高度为3，a的高度为2，d的高度为1，从而d最有可能是异常数据。

实际过程中需要用下面的公式计算测试数据的异常得分：

(2-34)

其中表示该测试数据在多棵iTree上的高度均值，是构造iTree需要的样本数，表示用个样本构造的iTree的平均路径长度，用来做归一化，可以用下面的公式计算：

(2-35)

上式中的。

如果测试数据的异常得分越接近于1，其是异常点的可能性越高；如果分数小于0.5，基本可以确定为正常数据。如果所有测试数据的分数都在0.5附近，说明数据集中不包含明显的异常样本。

由于我们关心的异常数据在iTree中的高度相对较低，为了节约训练时间，可以在生成iTree时设定一个限定高度，超过该高度iTree的生成可以结束。

iForest算法主要有三个参数：生成iTree的个数、训练单棵 iTree 时的样本数以及单棵iTree的限定高度。实验表明，当设定为100棵iTree，256且限高为8时，iForest算法在大多数情况下就可以取得不错的效果了。

本产品的实现过程中使用iForest算法实现了异常发文检测。算法输入是某一公众号文章“文档指纹”向量的集合和待检测文章的“文档指纹”向量，输出是一个值，该值越大，待检测文章异常的可能性越高。该值计算方法为：

(2-36)

由于异常得分不同文章间的差距不明显，我们通过引入值进行调整，可以使得异常检测的效果更加突出。上式中的为调整变量，本产品中选择取值为20。

2.3实现方案

2.3.1数据集介绍

因本产品需要进行抄袭/转发检测和异常发文检测，一方面要保证数据所属平台的权威性，另一方面也要保证数据的时效性。基于这两点考虑，本团队选择了微信公众号作为实验对象。由于微信公众号文章数据不对外开放，为了进行有效实验，本团队需要对公众号文章数据进行爬取。由于本产品需要对单个公众号的文章进行提取指纹操作，因此选择的进行实验的公众号发文应尽可能频繁。考虑到使用的算法需要同时考虑作者的写作风格和写作主题，爬取的公众号中既要包括主题鲜明的，也要包括风格鲜明的。

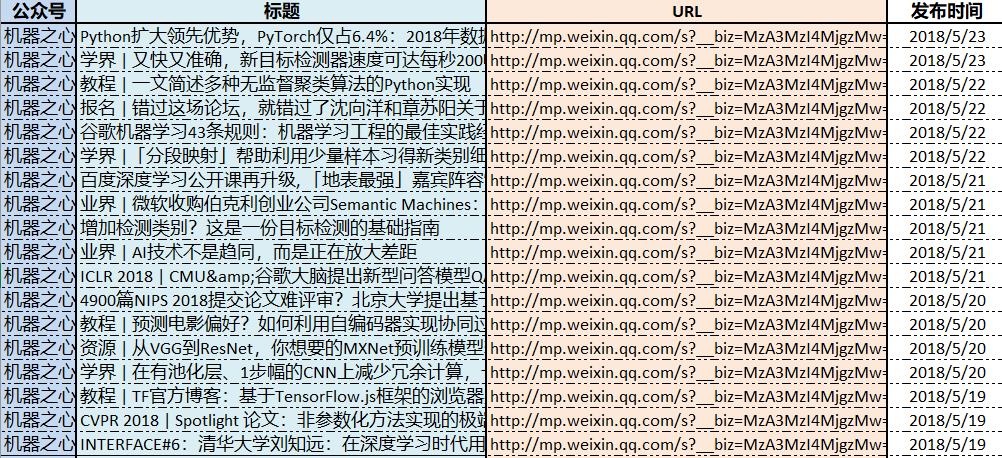
经过实际调研，本团队选择了100个公众号，并成功爬取了公众号近三个月来发布的共计9634篇文章，作为数据集来进行研究。下面是我们选取的一些典型公众号：

（1）主题相关：36氪、大数据文摘、机器之心、独立鱼电影……

（2）风格相关：vista看天下、咪蒙、新世相、三联生活周刊……

（3）主题和风格均相关：果壳网、占豪、侠客岛、知识分子……

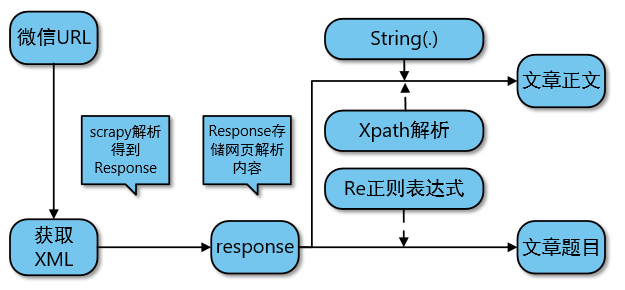
实验数据的爬取使用了清博大数据平台（http://www.gsdata.cn/）所提供的微信文章采集的产品。该产品可以搜素指定公众号并返回该公众号近三个月发布的文章真实URL，并将利用该接口采集到的数据整理成xlsx文件，图2-10是采集到的xlsx文件部分，文件中的URL一列为文章的真实URL，可以进行爬取。



**图 2-10 清博大数据采集的微信文章xlsx文件**

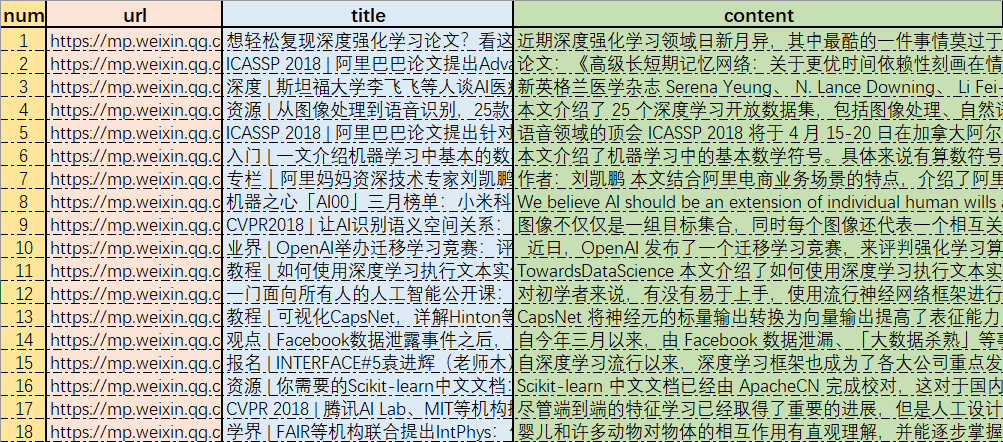
爬取公众号文章时使用了Python3.5+Scrapy+Openpyxl的框架，主要编程语言为Python3.5，使用Openpyxl打开从清博大数据获得的xlsx文件并获取URL，使用Scrapy编写爬虫，利用获取的URL爬取文章。Scrapy具有很好的异步性，对于大规模爬虫有很大优势。

对于一篇公众号文章而言，我们主要获取了该文章的标题和内容。爬取算法的流程图如图2-11。



**图2-11 数据爬取流程简图**

这样，一篇公众号文章的标题和内容就可以成功得到。在批量工作的情况下，我们爬取了100个微信公众号的文章，并将每个公众号的文章保存保存为一个csv文件，图2-12为其中一个公众号的结果。



**图 2-12 爬取csv文件示例**

**2.3.2算法实现**

**2.3.2.1 FWLDA**

在FWLDA的实现中，我们主要是采用了python库lda实现了该算法，并将其作用于一篇文章的虚词与标点上，最终获得了该文章写作风格的一部分维度。

具体而言，我们首先对于训练集中的文章用python库jieba分词，并分析词性，抽取其中的虚词与标点组成一个只有虚词与标点的字符串。对于这个字符串，我们使用了numpy提供的vectorizer生成文章的虚词/标点词汇表，从而方便后续的计算。

算法主体部分，我们采用了python的lda库实现了LDA算法，并将其作用在刚才的虚词信息上。通过LDA的fit函数，我们将模型根据刚才获得的虚词语料库进行拟合，从而得到一个训练过的LDA模型。其中，我们选择的迭代次数为500，最终的隐写作风格维度为 。

当新上传一篇文章，我们采用lda库所提供的transform函数，基于极大似然估计，最终能够得到新文章相对于之前训练找到的50个隐写作风格的概率分布，从而也就得到了该文档的。

**2.3.2.2 同义词林**

实践中，首先对输入的文本集合使用jieba进行切词操作，之后将词列表看做词袋，统计各个同义词类内部的概率分布信息。之后通过计算统计总贡献度寻找最能够区分不同作者的等价类，统计总贡献为公式2.2.2-7。

此步骤计算复杂度较高，但只运行一次，因此系统开销处于可以接受的范围。为了降低运算的复杂程度，我们让{W}里元素的每一个有序对(,)中的只包含一个词袋，其内容是同一个作者所有文章的词袋。这可以理解为，始终拿某一个作者的所有文章与看另一个作者的一篇文章作比较。事实上，这个简化运算的选择不会对结果造成负面影响：由于一个作者的所有文章中同义词偏好的统计结果能够反映作者在一个时间段内的平均偏好，这种统计能够光滑掉作者短时间内的语言波动，这正是我们所期望的。

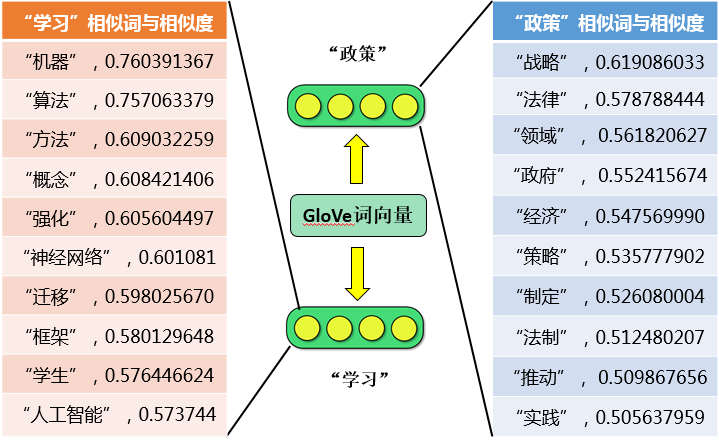
得到了同义词类的重要性顺序后，我们就可以根据类的排序制作向量。我们之后把上述词语差异的统计结果映射到一个3000维的高维空间中。这其中大多数词语所属等价类能够很好的说明语言特征，但是并不会全部出现在每一篇文章中，即我们得到的向量是系数的，之后我们选择主成分分析（PCA）来对向量进行降维，将3000维的向量投影到30维的空间，上述方法有效地将文章的同义词偏好差异问题转换为30维空间里向量的距离问题。在此获得代表作者用词偏好的一个向量。

**2.3.2.3 GloVe & TFIDF**

GloVe & TFIDF算法的实际实现中，我们使用了python的glove-python库进行GloVe词向量的训练。

我们的词向量训练语料库为我们爬取的近10000篇微信公众号文章，分别来自不到100个大大小小不同主题不同写作实体的公众号。通过这样的方式，我们能使我们的词汇表更加全面，词向量也更加符合微信公众号的实际情况。

GloVe词向量的训练中，我们的迭代次数选择为600，window大小选取10，学习率为0.07。由于我们的语料库相对较小，我们初步选择的词向量维度为50。由于GloVe算法对于小的语料库以及较小的词向量维度同样非常有效，因此最终词向量训练的效果依旧非常不错。为了证明这一点，下图中我们分别列出了与“政策”与“学习”两个词最为接近的15个词其对应余弦相似度。可以看到，这些词与“政策”或“学习”的语义关系普遍非常接近。



**图2-13 GloVe词向量中与“政策”或“学习”最接近的词汇**

在得到词向量之后，我们通过python库sklearn提供的TFIDF功能进行文档中词的TFIDF值的计算，并且选取每篇文章的关键词数量为，依据此权重对文章内这个关键词进行加权，最终得到了代表主题的维度。

**2.3.2.4 变分自动编码器（VAE）**

由于VAE是一个基于神经网络的降维算法，因此在实践中，我们主要采用了keras框架实现了VAE。其中，keras的backend为TensorFlow。

具体而言，首先，对于SWLDA、同义词林和GloVe & TFIDF所得到的三个维度的词向量，由于其互相之间的向量幅度不同，我们先对这三组维度分别作了0-1之间的归一化，之后再将三组向量拼接，从而保证其信息混合时的效果均等。

此后，我们用keras实现了变分自动编码器VAE。我们以这个130维的向量作为输入，经过两层全连接层（Dense），进入VAE的采样（Sample）部分。采样过后，再将采样后的结果经过两层全连接层（Dense），得到一个被恢复的130维向量。其中，隐采样层的维度为40（均值与方差各是四十维），全连接的中间层维度为80。之后，我们对于encoder部分（输入到均值的两层全连接层）进行权重的保存，保存到encoder.h5文件中。

当一篇新的文章输入模型需要运行VAE时，我们重新构建上述encoder，并将h5文件中的权重导入。从而，我们就是知道经过VAE降维之后，新来的文章的“文档指纹”为。

**2.3.2.5 异常检测 ---Isolation Forest**

Isolation Forest算法在本项目中主要用于根据一个公众号的历史文档指纹，检测一篇新文章（的指纹）是否仍属于该公众号，从而决定这篇文章是否可以直接快速通过审核。

实践中，我们采用了python的sklearn实现了Isolation Forest。我们首先根据用户输入的公众号的名称，找到该公众号所有历史文章的文档指纹（在数据库中提前跑完并和文章一起存储）。之后，我们将这些文档指纹输入sklearn的Isolation Forest，训练出一个算法模型，再将新进入的文章的文档指纹用该模型进行预测，得到一个百分比，该变量的值就是新文章属于此公众号的概率。

**2.3.2.6 相似推荐与转发、抄袭检测**

本项目中，我们进行相似推荐与转发、抄袭检测的核心思想如下：

由于文档指纹能够完整反映文章的主题和写作风格，那么如果两篇文章的文档指纹的相似度过大（超过某个阈值），那么这两篇文章极有可能存在抄袭或转发关系；而在非抄袭前提下，我们需要推荐给用户的文章，应当就是与用户所上传的文章文档指纹最接近的那些文章，因为用户会更加喜欢那样的内容与写作方式。

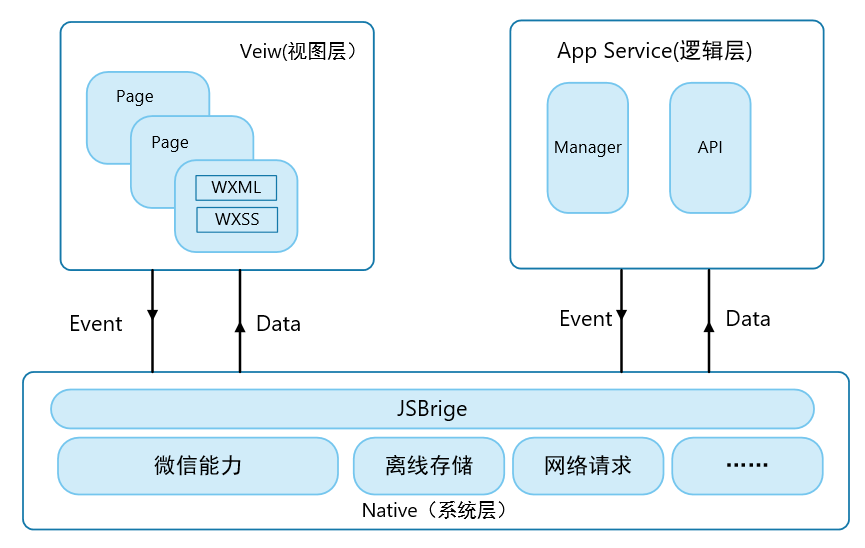
我们在实现这两个功能时，会首先计算出新文章的文档指纹。我们假定我们需要根据此新文章推荐最多篇文章，那么我们将构建一个长度为30的列表，将新文章的指纹与数据库中的M篇文章的指纹进行比对，最终将相似度最高的30篇历史文章留在该列表中。值得注意的是，我们的列表内的文章是根据其相似度从大到小排列，因此每一次比较都是从列表尾部往前比较，且大部分情况下比较次数都不多，从而该搜索算法复杂度为。

在找到这个30篇最相似的文章之后，我们将从列表头部开始考察它们与新文章的文档指纹的相似度。如果相似度大于阈值，就判定为抄袭。

2.3.3小程序部署

本项目研究最终以微信平台下微信小程序的形式体现研究成果。小程序作为项目展示的前端操作平台，同运行我们项目成果的服务器端进行交互。小程序使用类JS+HTML+XSS的WXS+WXML+WXSS编写，后端服务器以Python实现，前端应用程序可以部署在任何微信支持的移动终端。

使用小程序有下述优点，对于开发者而言，微信小程序开发模式同传统玩也应用开发相似；同时微信应用平台便于在不同手机品牌，不同操作系统层面上及逆行兼容，开发维护难度较低。对用户来讲，小程序基于微信生态圈，用户省去下载应用程序的步骤，方便快捷。同时微信本身作为一个大型自媒体平台，小程序的使用也便于我们对微信公众号等主体的分析研究。小程序开发框架如图2-14：

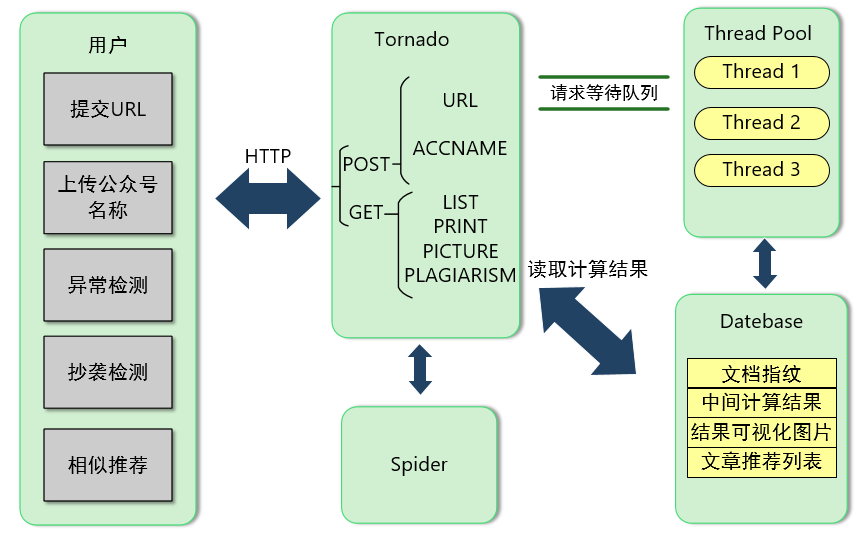


**图 2-14 微信小程序开发框架**

为了增加应用的可扩展性，以及体现微信小程序轻量级的特点，我们在前端只做最基本的信息输入和结果输出显示，输入包括文章的URL、公众号名称，输出包括异常检测的数值结果、基于“文档指纹”的推荐列表和抄袭判定结果。

小程序前端与服务器端交互逻辑如下，将URL传到服务器端进行处理。考虑到服务器端进行数据获取、数据处理的时间较长，因而采用异步的HTTP交互。客户端在上传URL后，并不在本次HTTP会话中返回结果，而是返回这篇文章的唯一标识码，之后经过服务器端处理后，在客户端使用这个唯一标识码进行分析结果的查询。

2.3.4服务器配置



**图 2-15服务器配置逻辑图**

服务器设计的主要目标是，具有产品级的稳定性，最大化适应异步请求，并保证良好的可伸缩性。

Web服务器采用异步非阻塞Web框架tornado实现，利用了该框架产品级的稳定性与可扩展性。这确保了Web服务器能够高效处理并发请求，并具有良好的可伸缩性。 Web服务器检查提交的请求合法性，不合法返回服务器内部错误，合法请求被无阻塞地放到请求队列中。之后，服务器将根据请求ID从数据库查询计算结果。

考虑到算法运行需要一定时间，服务器与小程序的交互采用异步的方式实现：小程序提交请求获得一个请求编号，之后采用此请求编号查询计算结果。

运行算法的后台组件由一个线程池组成。活跃线程的数量可以适当调整。各个线程从一个线程安全的队列获取请求。当一个线程被I/O或计算阻塞时，其他线程可以处理新的请求。这样可以最大化系统资源的利用率，减少延迟，并保证了可伸缩性。

考虑到部分结果例如异常检测，用户从数据中无法直观的获得信息。服务器端将对输出结果进行可视化处理，之后用户请求查询结果就会直接返回可视化结果。

# 第三章 作品测试与分析

3.1测试方案

3.1.1 测试设备

   本产品使用微信小程序进行测试和展示，测试设备为安装微信5.3版本以上的Android/ios手机，调试使用了微信web开发者工具v1.02。

3.1.2 测试环境

**前端**

* 搭建工具：微信web开发者工具v1.02
* 开发语言：微信小程序官方提供的语言，类似XML+CSS+JS的方式。
* 主要框架：

1. JS操作：微信小程序官方提供的API接口，如界面跳转和网络请求，
2. 界面设计：微信小程序官方提供的组件
3. 动态可视化图形：百度echarts团队提供的echarts-for-weixin开源库。

**后端：**

* 后台环境：腾讯云服务器 Debian 9.0 x64
* 开发语言：Python 3.5
* 主要框架：

（1）网络请求处理：Tornado框架

（2）图片可视化：词云使用wordcloud生成，向量极坐标图使用matplotlib生成

（3）爬虫：BeautifulSoup与re正则表达式

（4）算法基本框架：Tensorflow 和 Keras

（5）主要算法库：glove-python，lda，jieba，sklearn

3.1.3测试流程

**基本流程**

（1）测试搭建的微信小程序的产品效果

我们的微信小程序实现了基于“文档指纹”的微信公众平台的异常发文检测，抄袭/转发检测和相似推荐。我们会通过计算某些微信公众号文章“文档指纹”的方式，对微信小程序实现的这三类功能进行检测。最终异常发文检测会产生该文章对于指定微信公众号的异常值和同义词分析结果，抄袭/转发检测会给出我们数据库内存储的疑似抄袭文章，相似推荐会给出与该文章“文档指纹”相似度高的微信文章，并且可以通过网络访问来查看文章内容。

同时，我们在微信小程序中集成了算法流程演示。我们会通过展示图片结果的方式，将输入一篇微信公众号文章，产生对应“文档指纹”的过程演示出来，以便更好地展示我们的算法。

（2）测试算法效果

我们会在测试的后半部分通过三类应用来展示算法效果。由于抄袭/转发检测和相似推荐没有明确的机制进行准确度的度量，我们仅展示了算法异常发文检测的准确率，并对抄袭检测进行了简要测试。异常检测过程中，我们会在数据训练完成后，给定一个微信公众号，将本属于这篇公众号的文章和其他公众号的文章混合起来，进行准确率的检测。在抄袭检测过程中，如果对一篇文章进行简单修改后仍能在数据库中找到原文，就能说明抄袭检测算法的有效性。

**产品测试细节**

1. 打开微信小程序“文档数字指纹”，其图标如图3-1。用户可以从选定的微信公众号的文章列表中选择一篇文章，复制其链接到小程序URL输入框，上传URL，如图3-2。此时服务器端将爬取该URL对应的文章，并计算其文档指纹。



**图3-1 “文档数字指纹”微信小程序**



**图3-2 复制链接并上传URL**

（2）推荐与抄袭检测如图3-3。小程序跳转到指定界面，显示出若干篇微信文章的列表。显示的文章共有两种，疑似或转发的抄袭文章背景标红并置顶，而相似推荐所得的文章背景呈白色，显示在下方。如果用户点击其中任意一篇文章，即可跳转到该文章公众号链接。



**图3-3 抄袭与推荐检测**

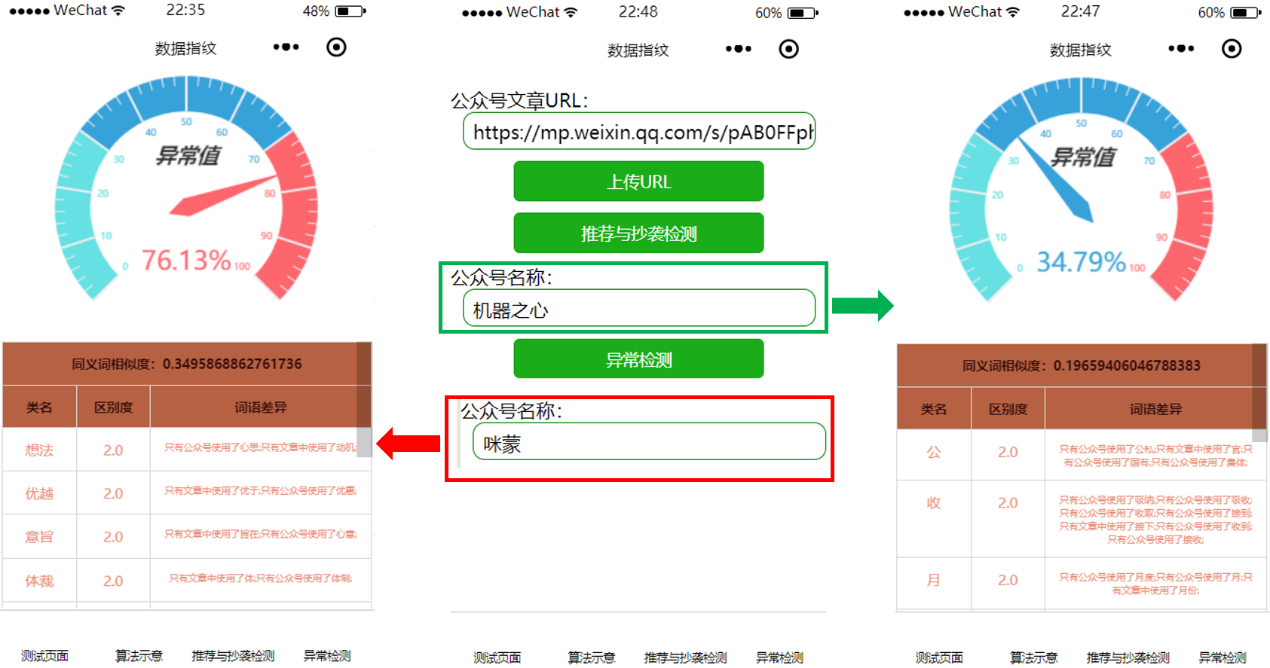
从图3-3中可以看出，该判定为抄袭的文章即为上传URL所对应的原文，由此可见抄袭检测算法有效；而根据“文档指纹”相似推荐的文章主要是公众号“大数据文摘”或“机器之心”的若干篇文章，可以看出被推荐的文章比较相似，推荐算法有效。最后，用户可以手动点开文章列表上的某一篇文章，即可根据该文章的URL跳转到其原文。



**图3-4 对“咪蒙”抄袭与推荐检测**

由于公众号“机器之心”的写作主题偏IT技术类，是明显的写作主题为主的公众号。为了验证本产品的抄袭检测算法，我们选择了和“机器之心”完全不同、以写作风格为主的鸡汤类公众号“咪蒙”进行测试，结果如图3-4所示。同样成功找到了上传URL所对应的原文，并且推荐的“新世相”同样是写作风格为主的鸡汤类公众号，“侠客岛”作为写作风格和主题均有侧重的公众号，其部分文章也被推荐。

（3）异常检测。返回主界面，在公众号名称输入框中输入指定公众号名称，点击异常检测跳转到指定界面。该文章相对公众号的异常值由仪表盘显示，下端以表格形式展示了同义词分析结果，图3-5为测试截图。

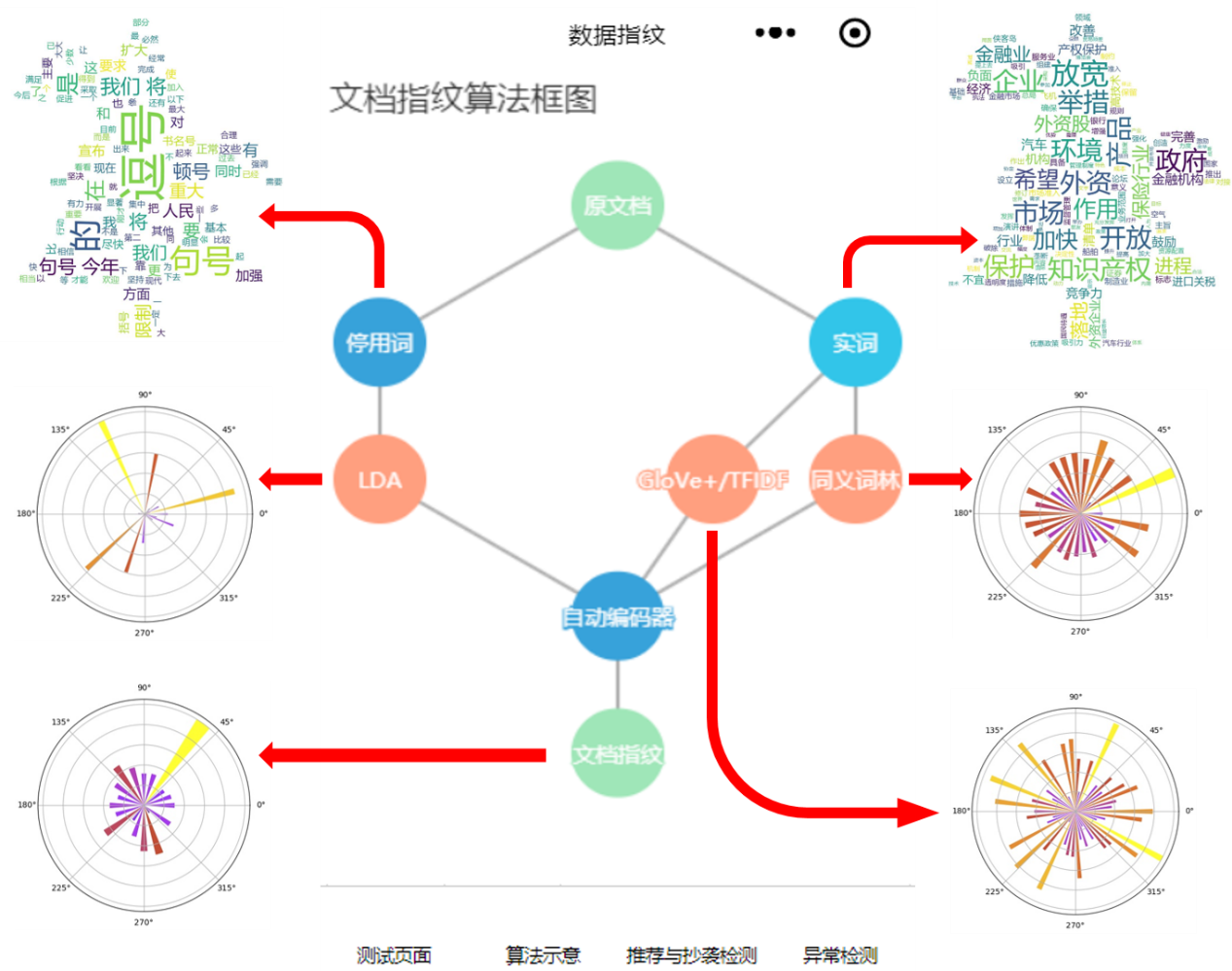


**图3-5 异常检测**

由于不同公众号写作团队不一，写作风格不完全相同，作者主题范围可大可小，因此我们没有设定一个绝对的可判定为异常的异常值阈值，而只是提供异常值的结果。用户可以根据自己经营的公众号文章的检测结果进行判定。

图3-5中上传的URL对应的文章所属公众号为“机器之心”。根据我们的算法，我们认为对于“机器之心”公众号而言，异常值大于50%以上有较大可能为异常，大于60%以上可以判定为异常。由此规则，我们发现当输入对应的公众号“机器之心”进行检测时，异常值只有34.79%，非异常，判断正确；而输入另一个公众号“咪蒙”时，异常值为76.13%，为异常，判断正确。

（4）查看算法运行机理展示。进入底部导航的“算法示意”一栏，展示出算法运行机理图后，点击图中的虚词（或实词，LDA，GloVe+/TFIDF，同义词林，文档指纹），跳转到指定界面并展示该步算法可视化结果。

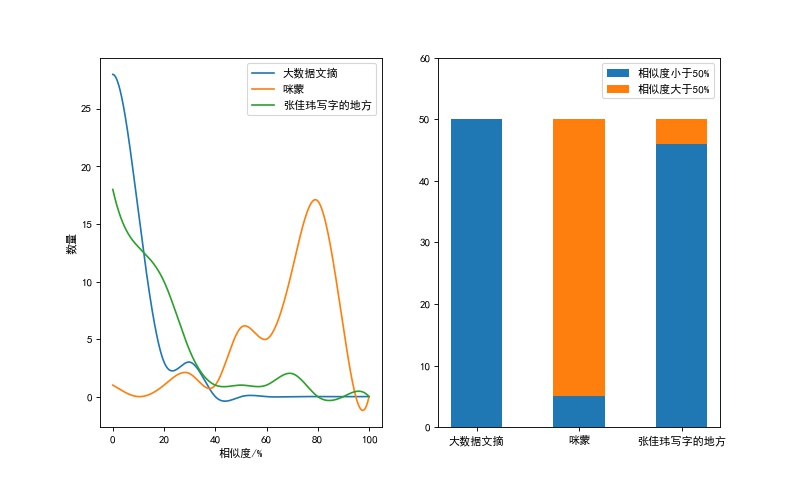


**图3-6 算法运行机理展示**

如图3-6所示，我们采用了词云图的方式来展示分词的结果——虚词和实词，使用极坐标向量图来分别展示Function-Word LDA得到的向量、GloVe&TF-IDF得到的向量、同义词林算法的向量，以及最终VAE编码器得到的结果（即“文档指纹”向量）。

**算法测试细节——异常检测准确率测试**

我们选择待检测公众号为“咪蒙”，其训练数据为200篇该公众号文章。进行检测的数据是公众号“大数据文摘”、“咪蒙”和“张佳玮写字的地方”的各50篇文章。150篇文章打乱顺序混合起来进行异常检测，最终结果为图3-7。



**图3-7 三个公众号相对于咪蒙的相似度对比**

左图显示的是和“咪蒙”不同相似度对应的文章数量，可以看出随着只有“咪蒙”本身的文章绝大多数相似度大于60%，而其他两个公众号的相似度多数文章很低。右图显示的是这三个公众号文章和“咪蒙”比较，相似度大于和小于50%的文章数量，可以看出“咪蒙”绝大多数文章相似度大于50%，其他两个绝大多数小于50%。

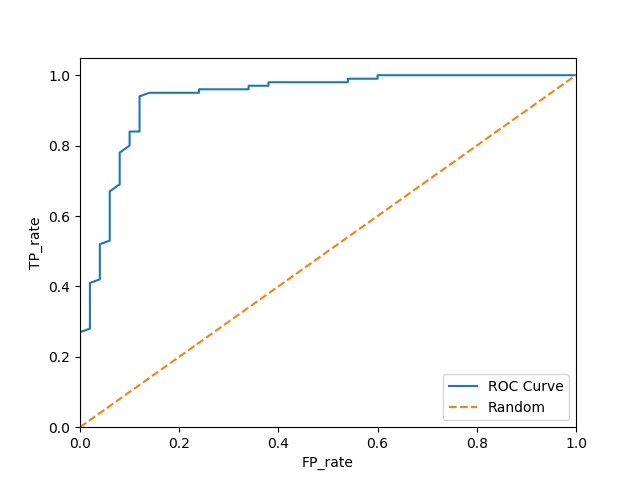
具体数据可见表3-1。

**表3-1 异常检测实验具体数据**

|  |  |  |
| --- | --- | --- |
| 待检测文章所属 公众号名称 | 该公众号文章中被 判定为异常的文章数量 | 准确率 |
| 大数据文摘 | 50 | 100.00% |
| 咪蒙 | 4（误判） | 92.00%（误判率：8.00%） |
| 张佳玮写字的地方 | 46 | 92.00% |

表3-1第一列代表用于异常检测实验的150篇文章所属的三类公众号，其中“咪蒙”是训练文章对应的公众号，即“咪蒙”的文章应被判定为非异常，“大数据文摘”和“张佳玮写字的地方”的文章应被判定为异常。从表3-1中可以看到，属于“大数据文摘”的文章全部判定为异常，“张家玮写字的地方”92%的文章被判定为异常，同时“咪蒙”本身的文章也有8%被误判为异常。

根据本次实验的数据，将“文档指纹”异常检测算法的ROC图绘制如图3-8，由该图可知，计算得到的AUC=0.9346 > 0.9，可以认为准确度很高，性能优秀。



**图3-8 ROC曲线**

深究上述结果，我们可以发现文档指纹的设计思路被很好地体现在了实际效果上。异常检测的训练集公众号“咪蒙”是一个鸡汤类公众号，因此“咪蒙”自己的文章能够自然地被判定正确。而“大数据文摘”与之相对，不仅写作团队截然不同，且主题也大相径庭，因此所有文章的匹配度都非常之低。最后的“张佳玮写字的地方”，虽然也经常发鸡汤类文章，但写作风格与“咪蒙”完全不同，因此虽然匹配度比“大数据文摘”要高，但大部分文章的匹配都依旧无法超越50%的阈值。由此可见，文档指纹算法非常有效。

我们在近百个公众号上进行了进一步的实验。实验中，我们每一次都随机选择一个公众号，并对其输入50篇属于它的文章与100篇不属于它的文章，查看其最终异常检测的效果。平均而言，基于文档指纹的异常检测正确率在88.37%。

**算法测试细节——异常检测算法对比**

本产品使用了Function-Word LDA、GloVe&TF-IDF和同义词林算法，三种算法结合生成“文档指纹”向量，基于此来进行异常检测。为了评估异常检测算法的有效性，本团队将我们的算法与其他一些用于对文章生成向量的算法在异常检测上进行了对比，包括只使用Function-Word LDA、GloVe&TF-IDF、同义词林统计算法，以及使用基于实词的传统LDA算法、skip-gram&TF-IDF算法、BOW算法和doc2vec算法[27]。

算法的训练数据同样是200篇“咪蒙”的文章。为了更好地检测算法的效果，我们选择了350篇公众号文章进行异常检测，其中300篇为我们从后台数据库中随机抽取其他公众号的文章，即正例，应当被判定为异常；其余50篇为我们从“咪蒙”的公众号中选取了新文章，即负例，不应被判定为异常。我们使用准确率, 精确率, 召回率和F1-Measure对算法结果进行统计，结果见表3-2：

**表3-2 异常检测算法对比统计结果**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 算法名称 | 准确率 (Accuracy) | 精确率 (Precision) | 召回率 (Recall) | F1-score |
| “文档指纹” | 90.00% | 98.89% | 89.33% | 93.87% |
| 仅LDA | 66.29% | 97.89% | 62.00% | 75.92% |
| 仅Function-Word LDA | 55.71% | 98.66% | 49.00% | 65.48% |
| 仅GloVe&TF-IDF | 60.57% | 94.02% | 57.67% | 71.49% |
| 仅同义词林统计 | 58.86% | 98.15% | 53.00% | 68.83% |
| skip-gram&TF-IDF | 78.57% | 97.47% | 77.00% | 86.03% |
| BOW | 70.86% | 99.50% | 66.33% | 79.60% |
| doc2vec | 84.86% | 99.20% | 83.00% | 90.38% |

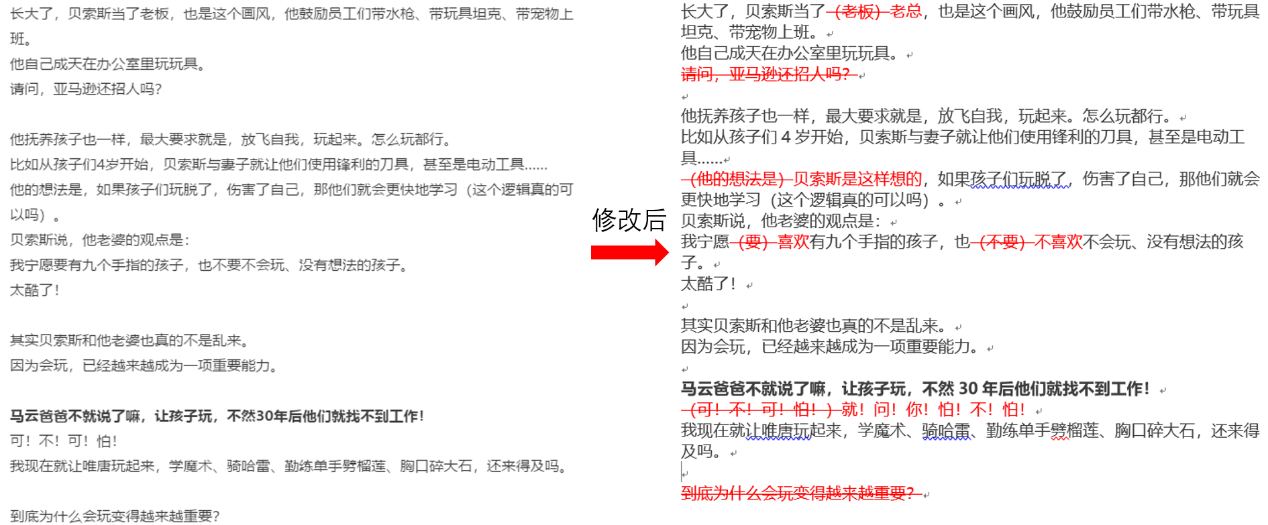
从表3的数据可以看出，“文档指纹”异常检测算法的F1-score值最高，比doc2vec算法还高3.49个百分点，这是因为虽然doc2vec性能很好，但是没有突出作者写作风格的影响。同时从表3可以看出“文档指纹”算法的准确率最高。上述两点说明本项目组设计的“文档指纹”异常检测算法更为有效。不仅如此，我们还可以发现“文档指纹”异常检测算法的F1-score值明显高于仅使用Function-Word LDA、GloVe&TF-IDF和同义词林算法，说明本项目组的算法设计非常有效。

上图的数据是在本次比较实验中的数据。经过多次实验，最终得到的“文档指纹”异常检测算法的准确率达到88.37%，与上面的数据相差无几。

**算法测试细节——推荐与抄袭检测**

对于推荐而言，由于没有明确的技术指标来确定推荐效果的好坏，我们只能从推荐文章的列表中看出对于技术类文章推荐的技术类文章多，对于情感类文章推荐的情感类文章多，难以进行有效的数据对比。因此没有给出推荐算法的具体测试数据。

对于抄袭检测，由于后台所有训练文章的“文档指纹”向量都被存储，而从中选取的URL爬取的文章与训练文章完全相同。因此如果完全按照从数据集中抽取文章进行检测，一定可以找到与之完全符合的“文档指纹”，即原文。这样的检测没有意义。因此我们进行了一定的改进，从后台文章中抽取了几篇文章，对其内容进行了简单的修改，在后台运行抄袭检测算法，示例可见图3-9。



**图3-9 修改文章进行抄袭检测**

简单的修改测试后，抄袭算法仍然会检测出原本的文章。我们进行了同义词替换、部分段落删除等操作，抄袭算法仍然有效。只有当删除的段落数量过多（如删除了超过20%的段落）后，抄袭算法才有可能无法检测出抄袭。

3.2观察与结论

从前面所述的测试结果来看，基于Function-Word LDA、GloVe和同义词林统计等算法生成的“文档指纹”在各项检测中均表现不俗，都达到了88%左右的准确率。但在与机器学习的其他领域，如语音识别、图像处理等相比，本产品的准确率仍有一定的提升空间。经过团队的研究与分析，本产品的算法实现在以下三个方面还存在不足，可以进行改进。

**1、数据集的不足**

本产品选择了使用当前火热且非常重要的信息发布平台——微信公众平台作为实验对象。由于微信公众平台的文章数据没有提供API等获取方式，团队只能使用爬虫的方式进行数据集的爬取，且只能爬取有限时间段内的公众号文章数据，虽然在某种程度上保证了数据集良好的时效性，但同时也存在以下问题：

（1）单个公众号的文章数据较少。我们通过很多公众号的数据来训练我们的模型，但是单个公众号的文章数据很难超过200篇。因此对于单个公众号而言，团队的数据集是比较小的。一方面我们无法在这么小的数据集上使用深度学习的方法，这非常容易过拟合；另一方面也会导致我们使用机器学习算法在不同公众号的表现上出现比较大的差异，在一些公众号上的表现不是非常满意。不过由于一般来说有较大影响力的公众号发文频率和单次发文数量均较高，因此对于我们提出的用于文章快速通过的异常检测方法，其思路是相对合理的。

（2）很多公众号文章末尾会出现广告。这些信息往往和文章主题，作者写作风格并不相关。我们在爬取的过程中对于特征明显的广告信息进行了摘除，但是在数据集中仍然存在这类信息，对算法的实际结果可能产生一定影响。

**2、汉语语言特殊性**

汉语同印欧语系语言的差异主要体现在汉语是一种表意文字，而英语等语言是表音文字。计算机输入是以字为单位，但是在语义表达过程中则是以词为单位，而且同一个字、同一个词在不同的语义环境下具有不同的含义，一字之差产生的意义就可能天差地别。

本产品生成“文档指纹”的第一步是分词，并完成对虚词和实词的划分。为了方便后面的LDA、GloVe以及同义词林等算法过程，我们使用了词袋模型进行了分词。由于有的词汇可能不在词袋模型中，因此分词的粒度与准确度并不完美，影响到最终的结果。比如“机器学习”这个词被分成了“机器”和“学习”两个词。

另一方面，团队使用了哈尔滨工业大学提供的同义词词林进行同义词的统计，但由于汉语的一字之差可能产生完全不同的意义，因此在同义词统计过程中出现了错误分类和统计的情况。比如我们在使用的过程中出现了将“媳妇”和“儿媳妇”归属到同义词的情况。我们尽可能对一些错误的归类进行了调整，但错误统计的情况仍然存在，先验知识无法保证高完整度也是制约分析效果提高的一大瓶颈。

**3、维度选择**

在“文档指纹”最后一步中需要对之前通过Function-Word LDA、GloVe和同义词林统计生成的向量在VAE自动编码器中实现信息混合和降维。维度的选择就成为了一个需要调整的参数，一方面是三种机器学习算法生成的向量的维度，另一方面是VAE自动编码器最终生成的“文档指纹”向量的维度。维度较低对数据信息的损失较大，但提高维度对设备的要求更高。经过不断尝试，团队在现有条件下选定了从(50,50,30)混合降维到40维度的方式。但是选择更高的维度可能会对实验结果产生一定的积极影响。

**4、开发应用**

除了上述三点算法方面需要改进之处外，本项目在产品演示上存在一些限制，以致我们只能在调试模式下演示我们的产品，而不能实际发布。这些限制主要是由于团队选择微信小程序作为演示平台，但微信官方对于微信小程序的把控非常严格，其中主要有以下三点：

（1）一些小程序控件实际发布后只有使用企业号并且获得授权后才可以使用，比如网页浏览控件web-view；

（2）微信小程序和微信公众号在文章交互方面不打通，微信小程序不能直接获取微信公众号文章，也无法监控微信公众号的发文情况，从而无法演示实时监测的效果。并且由于微信小程序只提供了访问手机视频文件和图片文件的接口，传递微信公众号文章只能通过传递对应URL的方式来进行；

（3）在微信小程序开发调试模式中，只有被管理员认定为开发者的微信账号才可以使用真机登录预览微信小程序。

总体来说，团队使用的机器学习算法以及演示产品都取得了不错的效果。如果获取更大更全面的数据集，该产品可以发挥出更好的作用。

# 第四章 创新性说明

4.1 文档指纹概念的提出

“文档指纹”概念旨在通过对文章内容与用词的分析，来衡量文章作者的写作风格、写作主题以及写作习惯。之后通过“文档指纹”来进行一系列有关于不同作者、不同文章的分析应用。

在项目开展前期的资料查询与可行性分析上，我们广泛搜集了同文档识别、自然语言处理方面相关的科研理论资料，但并没有发现同我们定义的“文档指纹”概念类似的已有研究成果。已经发布的算法理论基本是针对不同实际应用的算法，包括自然语言处理中的主题提取方面和搜索引擎方向的LSH算法等。

自然语言主题提取方向的算法主要有：比较经典的LDA、CTM以及各类延伸算法，基于词嵌入Word2Vec的主题提取，近年来随着深度学习的火热，例如LSTM，GRU，CNN，RecursiveNN,等算法也在主题提取方面有着不俗的表现。但是这些算法的聚类方式与写作风格无关，仅仅是反映了文章的主要内容，而不能在反映文章内容的同时反映作者的文风以及写作习惯。

搜索引擎方面，Google等公司使用了Simhash和Minhash等LSH（Locality Sensitive Hashing）算法。这类LSH最根本的作用，就是高效处理海量高维数据的最近邻问题，因此被搜索引擎用于筛选辨别相同网页，并在搜索结果中予以删除。这类型算法在有些文献中也被称为fingerprint，但是它们关注不同文档之间的整体内容的相似程度，是hash函数针对海量应用数据的改进。这类型的算法也并未考虑文章与作者的关系，没有关注同一作者、不同文章之间的相似性。

我们提出的“文档指纹”概念，是通过将文章进行实词和虚词的区分，实词反映主题和写作风格，虚词反映作者的语言习惯。针对不同的词语类型使用不同的算法进行向量化，之后通过变分自动编码器（VAE）生成整篇文章的文档指纹。我们的文档指纹兼顾了现有算法的优势，将之前分别处理、无法有效综合的主题提取和文章相似性分析进行有效结合，取得了良好的分辨效果。

4.2 应用场景的创新

本次参赛作品成果在当前的环境中预计能够应用于较多领域，包括但不限于文章审核、抄袭检测、相似推荐等。当前上述领域并没有通用高效的算法理论进行支持，绝大多数情况下依赖于人工操作，繁琐且效率低。

**文章审核：**

互联网时代为消息的传播提供了极大的便利，但与此同时，恶意、虚假信息的危害也因此被急剧放大。自媒体平台能在极短时间内将信息扩散到整个互联网。因此一旦出现恶意信息扩散，对于整个社会的影响是不可估量的。与此同时，各类平台的账户也容易受到攻击而导致攻击者使用他人的账户进行虚假恶意信息的发布。在文章审核方面，公众信息平台只有在虚假信息形成一定规模时，才能通过响应举报的方式进行遏制，然而这种模式低效率低效果，同时对萌芽阶段的异常信息无法进行及时反应。

而“文档指纹”则可以作为一种文章审核的工具，对虚假恶意信息进行异常检测，尤其是在虚假异常信息的萌芽阶段有良好的效果。公众信息平台在每一篇文章发布的时候计算其“文档指纹”，并将其数据与该账户历史“文档指纹”进行比较分析，如果新发布文章的“文档指纹”与历史记录有较大差距，则发出预警并提交系统管理人员进行进一步的人工审核，否则允许文章快速发布。使用“文档指纹”可以减少大量的人工审核工作量，同时也能够及时发现平台上的异常文章，以此提高公众平台对于异常信息的响应速度和效率。

**抄袭/转发预警（知识产权保护）：**

知识产权保护的重要性日益增强。微信公众号等信息平台通过集成大量用户流量获得不菲的经济收益。但是洗稿、抄袭、未授权转发的行为也日益猖獗。但在当前的技术水平下，无法通过程序做到对文档是否抄袭进行有效鉴定审核。

现有较为主流的抄袭检测主要使用在学术论文的查重上。基本的检测方法包括基于字符串比较的方式和基于词频统计的方式，它们根据重复比例或者根据单词拼读构成文档向量，之后采取点积、余弦等方法进行度量，例如COPS、SCAM、CHECK、MDR等模型算法。而这类型算法在面对抄袭作者使用同义词替换，部分语序颠倒等方式时，效果会有一定的折扣。

“文档指纹”算法在此领域也有一定应用价值，可以有效提高针对这类型抄袭的分析能力。在“文档指纹”的计算生成过程中，结合了文章主题、虚词、同义词等多个维度进行分析，抄袭者在仅仅使用同义词替换等方式将无法抹去原作者的虚词使用习惯，因此能够弥补现有抄袭检测算法的部分漏洞。

**推荐系统：**

推荐系统一直是如今各大平台吸引用户，增加用户粘性的重点。目前已有的文章推荐算法主要基于文章内容主题方面推荐类似的文章，使用协同过滤算法（Collaborative Filtering），其有不同侧重的变种，例如基于用户的协同过滤算法，还有一种是基于Item的协同过滤算法，此外和深度学习相结合的基于模式的协同过滤算法，包括Aspect Model，PLSA，聚类，SVD，Matrix Factorization等。这类算法在当下有较为良好的应用，但是也不乏缺点，例如和深度学习有关的算法需要依赖较大的数据集，否则效果不慎良好；在对于每一个账户推荐的风格将随着历史记录的累计而变化减少，随机性降低。同时上述算法均值基于文章内容进行推荐，而关于文章风格方面的信息则全部丢弃。

基于文档指纹的推荐则可以将主题信息和作者写作风格信息糅合在一起进行推荐。其推荐结果即能够满足追求内容主题相似性的推荐要求，又对文章风格有所考察，能够满足追求文章写作风格相似的推荐要求，目前的推荐算法均忽略掉此点。因此“文档指纹”可以满足不同推荐需求的要求，与当下推荐算法形成互补关系。

4.3 算法创新

我们的“文档指纹”的一大特色在于，其生成过程同时考虑了写作风格与写作主题的信息，这是其他研究中是极少同时涉及的，而这样的信息多样性对于文档指纹又是至关重要的。因此，我们选择自主设计一整套算法来满足我们所提出的文档指纹的要求。最终完成的这套算法，无论是整体流程还是算法细节，均体现了我们的思考与创新。

第一个算法创新在于对于虚词与标点使用LDA算法，从而提取出作者的写作风格维度。LDA(Latent Dirichlet Allocation)作为一个久负盛名的主题模型，传统上完全忽略文章中的语气词、虚词、标点符号，仅仅是对一篇文章的实词加以分析，从而获得文章的写作主题。然而我们认为，如果实词对应的是文章主题，那么虚词就应当对应于写作风格。因此，我们创新性地将LDA算法运用在虚词与标点符号上，用这样的Function-Word LDA提取文章作者的写作风格维度。经过实践证明，Function-Word LDA对于作者写作风格的提取非常有效。

第二个算法创新在于我们自主设计了一个基于同义词的写作风格提取算法——同义词林，用以获得文章反应的作者实词使用方面的习惯与特点。同义词部分之所以重要，原因主要在于自然语言传递的信息不仅有文本本身，实际上还隐含着人类的部分知识。例如文章中使用了“巨大”一词，我们不止获得了其词义，同时其实还包含了作者更喜欢用“巨大”而非“庞大”这样的隐含信息。我们设计的同义词林算法通过对于同义词信息的统计和降维，从而很好地提取出作者的用词习惯，反应出作者的写作风格特点。

第三个算法创新在于整个算法的框架与流程都是我们自主设计和实现的。我们查阅了很多资料，对于同时明确考虑写作风格与文章主题的文本嵌入方式，基本上屈指可数。因此，我们将这个问题拆分重组，用Function-Word LDA与同义词林提取写作风格，用GloVe&TFIDF提取主题，最后采用变分自动编码器（VAE）进行信息混合与降维。这样的思路是我们自己研究设计的，没有任何一篇paper用过一样的算法框架。或许算法并不够完美，也只是“文档指纹”的一种实现方式，但这样的设计却明确体现了我们的创新性和创造力。

4.4产品创新

产品作为项目最终代表，对项目的成功有直接影响。产品也是项目应用场景的直接体现。本项目需要基于公众发布平台进行文章的获取与分析，同时需要便于用户操作。经过对多种产品展现形式的分析，本次项目产品最终选择目前流行的微信小程序。微信小程序同公众号同属一个平台，同时微信公众平台也是当下较大的信息发布平台，采用小程序形式易于我们对微信公众号文章的分析；微信小程序使用类浏览器模式开发，开发简单，同时依靠微信应用程序屏蔽不同操作系统及不同手机型号的差异；微信小程序具有轻量级的特点，方便用户即开即用，减轻用户安装软件负担。

# 第五章 总结

本项目提出了具有实用价值的“文档指纹”概念，在主题和作者写作风格两个维度对一篇文章进行评价并使用自然语言处理方面的算法将其实现。实现结果显示“文档指纹”能够有效体现作者风格，并将项目成果以微信小程序的形式产品化并作为展示平台。项目组设计了“文档指纹”的实现流程，一篇文章从实词和虚词两个部分进行分析，对虚词部分使用隐狄利克雷分布（LDA）算法获得了一部分写作维度；实词部分，使用同义词林算法得到另一部分关于作者写作风格；再对实词使用GloVe&TF-IDF算法得到关于主题方面的分析结果。最后，我们使用变分自动编码器（VAE）对上述向量进行混合和降维，最终得到“文档指纹”。经过测试，本次项目达到了预期的目标。

根据本文的研究结果，“文档指纹”的概念符合客观事实，本项目所设计的系统算法能较为有效地区分不同作者的写作风格，并以此对文章相似度进行衡量。“文档指纹”将现有的主题提取与作者风格提取融为一体，具有较强的创新性。同时“文档指纹”的概念在当前互联网环境也有较多应用，在知识产权保护、异常信息检测和文章推荐等方面均有一定应用空间。

但是本系统仍需继续改进。同一个账户/公众号文章数量有限，这是人类写作的客观规律，因此如何能在少量数据上取得更加优秀的效果，是我们之后工作的方向。此外，汉语自身的特点也导致很多算法在汉语上应用效果产生折扣，之后我们也会针对汉语语言特点进行针对性的尝试，看是否可以有更好的判断结果。同时未来会对算法根据具体应用场景进行相应优化，根据系统在实际中的运行情况，在阈值等数值量上进行一定的调整。相信本项目在未来能够在异常信息检测，抄袭检测等方面有所贡献。

# 参考文献

1. Brin S, Davis J, Garcia-Molina H. Copy detection mechanisms for digital documents[C]//ACM SIGMOD Record. ACM, 1995, 24(2): 398-409.
2. Stein B, zu Eissen S M. Fingerprint-based Similarity Search and its Applications[J]. Universität Weimar, 2007.
3. Schleimer S, Wilkerson D S, Aiken A. Winnowing: local algorithms for document fingerprinting[C]//Proceedings of the 2003 ACM SIGMOD international conference on Management of data. ACM, 2003: 76-85.
4. Osman A H, Salim N, Binwahlan M S, et al. An improved plagiarism detection scheme based on semantic role labeling[J]. Applied Soft Computing, 2012, 12(5): 1493-1502.
5. Data Loss Prevention Using Document Fingerprint[EB/OL].(2014-09-11)[2018-05-29].https://technet.microsoft.com/en-us/library/dn635176(v=exchg.150).aspx.
6. Nate L.Introduction of File Fingerprint[EB/OL].(2017-07-27)[2018-05-29].https://digitalguardian.com/-blog/what-file-fingerprinting.
7. Rocha A, Scheirer W J, Forstall C W, et al. Authorship attribution for social media forensics[J]. IEEE Transactions on Information Forensics and Security, 2017, 12(1): 5-33.
8. Overdorf R, Greenstadt R. Blogs, twitter feeds, and reddit comments: Cross-domain authorship attribution[J]. Proceedings on Privacy Enhancing Technologies, 2016, 2016(3): 155-171.
9. Afroz S, Islam A C, Stolerman A, et al. Doppelgänger finder: Taking stylometry to the underground[C]//Security and Privacy (SP), 2014 IEEE Symposium on. IEEE, 2014: 212-226.
10. Blei D M, Ng A Y, Jordan M I. Latent dirichlet allocation[J]. Journal of machine Learning research, 2003, 3(Jan): 993-1022.
11. Gaussier E, Goutte C, Popat K, et al. A hierarchical model for clustering and categorising documents[C]//European Conference on Information Retrieval. Springer, Berlin, Heidelberg, 2002: 229-247.
12. 哈工大社会计算与信息检索研究中心.《哈工大信息检索研究室同义词词林扩展版》说明.<http://ltp.ai>.
13. Blei D M, Lafferty J D. Correlated topic models[C]//Proceedings of the 18th International Conference on Neural Information Processing Systems. MIT Press, 2005: 147-154.
14. Zhang X, Zhao J, LeCun Y. Character-level convolutional networks for text classification[C]//Advances in neural information processing systems. 2015: 649-657.
15. Socher R, Huang E H, Pennin J, et al. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection[C]//Advances in neural information processing systems. 2011: 801-809.
16. Socher R, Perelygin A, Wu J, et al. Recursive deep models for semantic compositionality over a sentiment treebank[C]//Proceedings of the 2013 conference on empirical methods in natural language processing. 2013: 1631-1642.
17. Lai S, Xu L, Liu K, et al. Recurrent Convolutional Neural Networks for Text Classification[C]//AAAI. 2015, 333: 2267-2273.
18. Yang Z, Yang D, Dyer C, et al. Hierarchical attention networks for document classification[C]//Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2016: 1480-1489.
19. Pennington J, Socher R, Manning C. Glove: Global vectors for word representation[C]//Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014: 1532-1543.
20. Bengio Y, Ducharme R, Vincent P, et al. A neural probabilistic language model[J]. Journal of machine learning research, 2003, 3(Feb): 1137-1155.
21. Collobert R, Weston J, Bottou L, et al. Natural language processing (almost) from scratch[J]. Journal of Machine Learning Research, 2011, 12(Aug): 2493-2537.
22. Huang E H, Socher R, Manning C D, et al. Improving word representations via global context and multiple word prototypes[C]//Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1. Association for Computational Linguistics, 2012: 873-882.
23. Mikolov T, Sutskever I, Chen K, et al. Distributed representations of words and phrases and their compositionality[C]//Advances in neural information processing systems. 2013: 3111-3119.
24. Mikolov T, Chen K, Corrado G, et al. Efficient estimation of word representations in vector space[J]. arXiv preprint arXiv:1301.3781, 2013.
25. Kingma D P, Welling M. Auto-encoding variational bayes[J]. arXiv preprint arXiv:1312.6114, 2013.
26. Liu F T, Ting K M, Zhou Z H. Isolation-based anomaly detection[J]. ACM Transactions on Knowledge Discovery from Data (TKDD), 2012, 6(1): 3.
27. Le Q, Mikolov T. Distributed representations of sentences and documents[C]//International Conference on Machine Learning. 2014: 1188-1196.