中图分类号：TP391.1

论 文 编 号 ：10006ZY1506321

北京航空航天大學

专业硕士学位论文

适航领域软件需求跟踪方法研究与应用

作 者 姓 名　　赵 腾

学 科 专 业　　计 算 机 技 术

指 导 教 师　　曹 庆 华　教 授

培 养 院 系　　计 算 机 学 院

Study and Application of the Method of the Software Requirement Traceability in Airworthiness

A Dissertation Submitted for the Degree of Master

Candidate：Zhao Teng

Supervisor：Prof. Cao Qinghua

School of Computer Science & Engineering

Beihang University, Beijing, China

中图分类号：TP391.1

论 文 编 号 ：10006ZY1506321

硕 士 学 位 论 文

适航领域软件需求跟踪方法研究与应用

作者姓名　　赵腾　　申请学位级别　　工程硕士

指导教师姓名　　曹庆华　　职　　称　　教授

学科专业　　计算机技术　　研究方向　　软件适航

学习时间自　　2015 年 09 月 01 日　　起　　至　　2018 年 03 月 09 日止

论文提交日期　2018 年 03 月 12 日　　论文答辩日期　2018 年 03 月 09 日

学位授予单位　北京航空航天大学　　学位授予日期　　年　　月　　日

关于学位论文的独创性声明

本人郑重声明：所呈交的论文是本人在指导教师指导下独立进行研究工作所取得的成果，论文中有关资料和数据是实事求是的。尽我所知，除文中已经加以标注和致谢外，本论文不包含其他人已经发表或撰写的研究成果，也不包含本人或他人为获得北京航空航天大学或其它教育机构的学位或学历证书而使用过的材料。与我一同工作的同志对研究所做的任何贡献均已在论文中做出了明确的说明。

若有不实之处，本人愿意承担相关法律责任。

学位论文作者签名：_____ 日期： 年 月 日

学位论文使用授权书

本人完全同意北京航空航天大学有权使用本学位论文（包括但不限于其印刷版和电子版），使用方式包括但不限于：保留学位论文，按规定向国家有关部门（机构）送交学位论文，以学术交流为目的赠送和交换学位论文，允许学位论文被查阅、借阅和复印，将学位论文的全部或部分内容编入有关数据库进行检索，采用影印、缩印或其他复制手段保存学位论文。

保密学位论文在解密后的使用授权同上。

学位论文作者签名：_____ 日期： 年 月 日

指导教师签名：_____ 日期： 年 月 日

摘　　要

随着国内航空事业的蓬勃发展，机载软件的作用也越来越明显。软件需求跟踪链接是保证软件在整个生命周期过程中能够正确合理的重要依据，也是软件适航审定工作的重要元素之一。随着机载软件的规模越来越大，相关软件文档也越来越多，人工建立需求跟踪链接的难度变得越来越大，并且更容易引入错误的链接。本文从文本的角度出发，对机载软件文档进行分析，提出了基于 word embedding 和学习排序算法的适航领域软件需求跟踪模型。本文的主要研究工作和创新性成果包括如下几个部分：

（1）提出了一种领域相关文本语义相似度计算方法。该方法使用领域相关数据训练 word embedding，并引入领域词典对专业文档进行分词预处理，然后利用查询扩展和设置的加权策略对已有文本相似度算法改进。本文使用该算法完成初步需求跟踪任务，验证了改进后的方法提升了文本语义相似度计算的精确率。

（2）改进了已有的 IR SVM 学习排序算法的损失函数，添加文本语义相似度作为文档对学习排序算法（pairwise）损失函数的权重，使用包含更多信息的文本语义相似度替换 IR SVM 中对不同等级文本的加权方法。利用改进的 IR SVM 学习排序算法对初步的需求跟踪结果二次计算，最终实验结果表明改进的学习排序算法是有效的并且提升了需求跟踪任务的精确率。

（3）综合适航领域文档的特殊性和以上两个部分的改进，本文提出了适航领域软件需求跟踪模型——Tr-WELR（Traceability based on Word Embedding and Learning to Rank）模型。该模型综合考虑了领域特殊性、文本上下文相关性和对初步排序列表的二次利用，同时在适航审定人员明确跟踪关系之后，使用新的需求跟踪关系对模型进行重新训练，可以使模型的精度随着系统的使用不断得到提升。本文将该模型与国际领先水平的模型 ENRL 进行实验对比，证明了该模型在需求跟踪任务上的有效性，并且表现出更好的效果。

（4）实现了基于 Tr-WELR 模型的适航领域软件需求跟踪原型系统。该系统包括适航软件数据管理模块、计算模块和用户管理模块，其中计算模块是核心模块，包括中文分词及翻译、数据预处理、文本相似度计算、软件文档层级关系建立和适航审定目标满足性判定等子模块。

关键词：软件需求跟踪，word embedding，文本语义相似度，学习排序

Abstract

With the development of domestic aerospace industry, airborne software plays an increasingly important role. Software requirement traceability links not only are the key basis to ensure the correctness and rationality of the whole software lifecycle, but also are the one of important elements of software airworthiness certification work. As the scale of software is larger and larger and more and more software documents are created, manually creating requirement traceability links becomes harder and error-prone easier.

From the perspective of text, this paper analyses the airborne software documents, and proposed the software requirement traceability models in airworthiness based on word embedding and learning to rank. The main research and contributions of this paper are as follows:

(1) The paper proposed a domain-specific method to calculate semantic similarity of texts. This method trains word embedding with domain dataset, brings in domain dictionary to segment professional documents, and leverages query expansion and weighting strategy to improve the existing algorithm which calculates semantic similarity of texts. This paper used the improved method to complete preliminary task of requirement traceability, and the results proved that our method improved the precision of calculation of semantic similarity.

(2) We improved the loss function of IR SVM, a learning to rank algorithm, leveraged the text semantic similarity as the weight of pairwise learning to rank algorithm, and used text semantic similarity which contain multi-layer information to substitute the weighting method of IR SVM which relates to different levels' text. We use the improved IR SVM algorithm to recalculate the preliminary results, and the experiment results show that the improved IR SVM algorithm is effective and can improve the precision of final results.

(3) Through the improvement of the above two parts, this paper proposed software requirement traceability model in airworthiness, called Tr-WELR (Traceability based on Word Embedding and Learning to Rank). This model comprehensively take the particularity of airworthiness, the relatedness of text context and the secondary utilization of primary ordered list into consideration and it can be re-trained by more accurate data which is confirmed by authorized personnel. This paper compares our model with the state-of-the-art models ENRL, and it turns out that the effectiveness of our model in the task of requirement traceability and our model shows better performance.

(4) This paper implements the software requirement traceability prototype system based on Tr-WELR model. This system mainly consists of airworthiness software data management module, calculation module and user management module. And calculation module is the key module, which contains Chinese word segmentation and translation, data preprocessing, the calculation of semantic similarity of texts, the establishment of layer relationship of software document and the determination of satisfaction of airworthiness approval target.

Key words: software requirement traceability, word embedding, semantic similarity of texts, learning to rank

目　录

图 目 录

表 目 录

第一章　绪论

本章将首先介绍论文的研究背景和研究意义；其次对目前国内外关于软件需求跟踪任务所使用的方法和软件工程领域文本检索算法进行了总结和分析，指出了每种方法的应用场景和各自的局限性；然后针对这些问题明确了论文的研究目标和研究内容；最后介绍了整篇论文的组织结构。

1.1　课题来源及研究背景

本课题来源于工信部民机专项《航空发动机电子控制系统适航审定关键技术》子项目《航空发动机控制系统软件和电子硬件适航审定方法与流程研究》。

2017 年商飞 C919 客机的首次试飞顺利完成，标志着我国的民用航空事业进入了新的历史阶段。民用航空事业的基本理念是“民机发展，适航先行”[1]。所谓“适航（适航性）”是指航空器的各个部件和子系统能够在预期的环境下安全起飞、着陆和飞行的固有品质，并且这种品质能够由航空器全寿命周期的各个环节来完成和保持[1]。适航性即安全性，是民机能够投入使用、走向市场的重要前提和保证。

随着计算机技术的发展以及在航空领域的应用，机载软件承担了越来越多的功能，同时软件的种类和规模也持续增长。机载软件规模增长和大量使用主要有三个原因：1）通过更加智能和方便的软件来改善人机界面最终减轻飞行员的负担；2）软件设备的重量、功耗、维护成本往往低于其他类型的设备；3）在软硬件的功能分配上，研制人员更青睐于使用软件。机载软件在机载系统乃至整机中起到了越来越重的作用，但是由于规模的增长，增大了软件的研制和审定的工作量，因此如何保证在整个生命周期过程中的软件需求可跟踪性成为影响项目成败的重要因素。2011 年航空无线电技术委员会（RTCA）在 DO-178B 标准的基础上提出 DO-178C，即《机载系统和设备合格审定中的软件考虑》。该标准面向过程和目标，明确了研制满足适航性的软件需要实现的过程和需要达到的目标，但是并没有严格约束开发流程，以可协调的方式为开发适航软件提供了指南。DO-178B/C 在标准层面上制定了适航软件开发的流程以及应该达到的目标，其中制定的目标中明确指出：1）高级需求可追踪到系统需求；2）低级需求可追踪到高级需求；3）源代码可追踪到低级需求[2]。

需求跟踪，即通过建立需求同软件生命周期中各个阶段元素之间的联系，实现对每个需求的监督和跟踪。在工业实践中，现有的需求跟踪技术是通过填写需求跟踪矩阵和

需求跟踪图的方式来进行跟踪，这种方式比较精确，但是工作量也随着软件规模的增长而增多，同时在变更需求和增加需求的时候，难以扩展且容易引入错误。市面上的需求管理软件有：IBM Rational DOORS、IBM Rational RequisitePro 和青铜器 RDM 等，这些软件在记录软件需求跟踪关系的同时，能够帮助软件相关人员增强对项目目标的沟通，增强协作开发的效率。同样的，这些软件也是通过建立需求跟踪矩阵来实现，尽管使用这些软件能够给信息录入人员简化部分操作，但当软件规模逐渐增大时，工作量也会持续增长。因此，能够自动化或者半自动化地帮助软件开发人员简化相关工作具有非常重要的意义。在学术领域，自动化的需求跟踪技术主要基于信息检索思想，使用向量空间模型（Vector Space Model，VSM）、潜在语义索引（Latent Semantic Indexing，LSI）和概率模型（Probabilistic Model，PM）等模型[3]。这些方法基本是通过将软件文本表示成词频向量的形式，然后通过向量计算文本相似度，很少考虑软件文档的语义。近年来越来越热门的自然语言处理（Natural Language Processing，NLP）能够通过统计学和语言学的理论计算得到词和词、句子和句子之间的语义关系，从而提升文档关系计算的准确率。同时，随着机器学习技术的发展，更多研究人员利用文本和文本间的特征对文本分类，并且可以通过训练的模型对未知文本进行预测。因此，在软件需求跟踪的过程中加入语义计算和机器学习算法，一定能够提升需求跟踪关系的精确程度。

综上所述，软件需求跟踪对于适航领域的适航性目标的达成具有重要的意义，同时自动化的需求跟踪技术的研究更是对规模日益增长的适航软件系统的研制和审定具有重要意义。

1.2 国内外研究现状

1.2.1 需求跟踪技术概述

需求跟踪技术分为两类：静态需求跟踪技术、动态需求跟踪技术，其中静态需求跟踪技术主要使用需求跟踪矩阵、需求跟踪图和交叉引用等多种静态文本形式表示跟踪过程，动态需求跟踪技术，即上文提到的自动化建立跟踪关系的技术，是在软件开发过程和需求变更时自动建立或维护跟踪关系，主要使用基于信息检索、基于一定的增强策略和基于规则的方法[4]。

（1）静态需求跟踪技术

传统需求跟踪主要使用静态跟踪技术，但随着软件规模的增长，软件生命周期越来越长，静态跟踪技术易于出错、不易维护的缺点就暴露出来，如果继续使用则成为

软件维护的一个负担。

需求跟踪矩阵（Requirement Traceability Matrix，RTM）用来表示需求和生命周期过程中各个元素的联系，包括与设计阶段、实现阶段和测试阶段等阶段元素的联系。设计阶段元素包括系统设计图、数据流图、类图、数据库 ER 图和设计文档等，实现阶段元素包括源代码文件、类描述文件等，测试阶段元素包括测试用例文档和测试结果报告等。需求跟踪链接和软件生命周期中其他过程的元素可以具有多种关系：一对一、一对多和多对多。举例说明，一个需求可以和一个或多个代码文件相关，同时一个代码文件也可以对应一个或多个需求。需求跟踪矩阵示例如图 1 所示。

需求跟踪矩阵

记录编号： 项目代码-XXX

项目名称： 某型号机载软件

序号	系统名称	需求名称	功能名称		
			顶层功能	细化功能	状态
1	XXX软件系统	系统安全性监测	状态显示	燃油信息显示	
2				温度显示	
3				压力显示	
4		信息系统管理	发送信息		
5			接收信息		

图 1　需求跟踪矩阵示例

需求跟踪图可以直观的表示需求之间的联系、需求和各个阶段元素之间的关系，以及在需求变更时将受到影响的各部分元素。需求跟踪图对跟踪关系的展示比较明显，同时既能够表示需求和系统中各个元素的关系，也能够表示开发过程中产生的中间元素之间的关系。在跟踪图中，用户可以自己定义图中的对象和关系，方便用户使用自己熟悉的语言对关系描述。

交叉引用用于建立软件需求文档之间的跟踪关系，需求文档包括需求规格文档、需求说明文档等，对于适航软件，还有系统需求文档、高级需求文档和低级需求文档等。使用交叉引用建立的跟踪关系较直观，方便实际使用，但只适用于需求文档之间跟踪关系的建立[5]。

（2）动态需求跟踪技术

动态需求跟踪为解决手工建立需求跟踪关系易于出错和难以维护的问题，使用自动化或半自动化的技术建立和维护跟踪关系[4]。

由于软件在全生命周期的设计、开发和维护过程中产生了大量的文本信息，因此动态需求跟踪技术大多从文本角度出发，将需求文档看作信息检索任务中的待查询文档，并计算其与不同软件制品之间的相似度，设置阈值，相似度高于该阈值的软件制

品则被认为与检索的文档具有跟踪关系[6]。

文献[4]中介绍了一种基于句法分析的跟踪关系恢复方法，通过句法分析可以识别出最有可能刻画软件制品特征的部分动词与名词，并减少制品中存在的噪音对需求跟踪关系恢复过程造成的不利影响。方法分为如下步骤：1）将制品中的句子切分为句子块（代码特殊处理）；2）词性标注；3）对句子进行块分析，利用句子的上下文来修正词性标注过程中可能引入的错误；4）减少标引词中存在的噪声；5）通过聚类簇映射为分类建立映射关系。该方法的核心步骤是使用语义聚类对得到的不同分组进行映射，然后使用得到的结果来建立跟踪关系。

文献[7]和文献[4]是同一个作者，文献[7]通过命名实体识别来恢复软件需求和源代码之间的跟踪关系。该文章作者表示基于文本的需求跟踪方法的准确率会随着文本质量的高低而有所不同[7]，因此，该文章提出使用代码上下文建立命名实体识别模型，使用命名实体表示文本中的关键词，解决抽象语法树无法解析自然语言文本的问题。文中使用基于最大熵模型的需求跟踪方法，将源代码中的命名实体识别看作分类问题，同时将代码的类别分为：class、method、invocate、normalText、comment、param；然后对上述已经找到命名实体的制品文本进行预处理，将软件制品转为文档集合，然后使用文献[4]中相同的聚类方法对关键词聚类和映射，根据映射结果建立跟踪关系。

由于之前的工作没有充分利用过代码文件中的代码注释，因此文献[6]中提出使用代码注释来提升需求跟踪任务的准确率，在需求文档和代码之间建立更精确的跟踪关系。文中指出动态需求跟踪任务使用信息检索技术，建立需求文档和其他软件制品的跟踪关系，在跟踪关系的精度上不能满足要求，同时文中认为问题一部分是出在没有利用注释信息上面。该文章中使用的方法和文献[7]相似，主要有两点不同：1）数据预处理阶段，使用自动翻译工具将中文的软件需求文档翻译成英文，因为文中认为代码都是英文，也省去了中文分词的步骤；2）使用向量空间模型表示文本需求和源代码文件，然后使用向量空间模型中的向量计算二者的相似度，设置阈值，达到该阈值即可建立跟踪关系。

以上三个文献分别利用句法分析、命名实体识别和代码注释作为增强策略，从各自的角度和原始的的信息检索方法进行对比，效果都得到一定程度的提升，但是整体效果都比较弱。

文献[8]是比较早被提出通过信息检索的方式来完成动态需求跟踪的文章。文中提出解决需求跟踪任务的主要步骤包括：1）将任务构造成信息检索问题；2）选择 IR 算

法；3）将需求文本输入到 IR 算法中；4）分析算法输出；5）选择合适的策略整理算法输出；6）比较该方法与其他方法的性能。比较重要的是，文中提出可能在这个过程中存在的问题：1）需求文档不全或语义模糊；2）有些缩略语没有定义；3）领域或工程知识缺少；4）高层需求和低层需求中使用不同的术语表示。这篇文章使用 VSM 算法，同时增添了两个扩展：一个是使用关键词列表，另一个使用简单的词典。该论文提供了动态需求跟踪链接的基本思路，也给了我们在该方面研究的启发。

文献[9]提出基于使用 LSI，即潜在语义索引，建立需求和设计制品、测试用例之间的跟踪关系。LSI 是信息检索技术中的一种，可以将信息降维，将所有的文档都表示为 LSI 的子空间，通过余弦相似度方法计算文档间的关系。同时文中指出 LSI 不依赖于提前定义的单词表或语法，也就是说，可以省去部分耗时耗力的数据预处理过程。但是该方法存在着 LSI 方法自身的不足之处，该问题将在 2.2.1 节详细描述。

1.2.2 软件工程领域文本检索算法概述

在软件工程领域，越来越多的任务使用文本检索技术，比如：需求跟踪[10,11]、特征定位[12]、软件复用[12]等。为了提升文本检索的性能，很多方法被提出，下面将列出与本文工作相关的一些方法。

首先是自动查询扩展技术（Automatic Query Expansion，AQE），该技术已经广泛地用在信息检索任务中。自动查询扩展技术通过扩展待查询的单词或短语，去解决“词汇问题”[13]。所谓“词汇问题”指的是由于人类语言的多样性导致的查询语句中的单词与文档集中单词的不一致性问题，这也是“一词多义”或“一义多词”现象。C.Carpineto[14]指出大部分的自动查询扩展都显式地利用词条的依赖特性，比如在建立同义词词典时使用词语的共现关系、语法关系等。除此之外，还有一些关于自动查询扩展的策略，比如对扩展词进行加权，赋予不同的扩展词不同的重要程度，保证扩展后的单词集合或短语集合能更完整的表达待扩展词的含义[15]。

然后是词向量（本文中词向量特指 Word Embedding），其被大量研究者用在词的表示和文档表示上。Word2vec 是 Google 的 Tomas Mikolov 等人提出的文本表示方法，将单词表示成词向量的形式。训练词向量的模型有两种： CBOW 和 skip-gram，其中 CBOW 模型的输入是特定词上下文相关词的词向量，输出是该特定词的词向量，而 skip-gram 模型与 CBOW 的输入输出相反。

在软件工程任务中，很多文本检索任务都使用了 word2vec 或 word2vec 的改进模型。

X. Ye 等人[16]提出一种新的学习词向量的方式，并使用训练得到的词向量去计算文档之间的相似度。在该文章中，出于对“一义多词”现象的考虑，他们提出了两种不同的训练设定：一种是单词典设定，另一种是双词典设定。单词典设定是将自然语言文本和源代码混在一起，而双词典设定是将二者分开。举例说明：对于单词“clear”，在自然语言文本中是形容词“干净的”或者动词“清除”等含义，在源代码文本中，该词为一个方法名。二者的不同是单词典设定会使用方法名“clear”在代码中的上下文去训练形容词“clear”，而双词典设定不会。然而，他们没有考虑很多软件文档是自然语言文本和源代码是混杂在一起的，因此他们的方法并不是对所有的软件制品适用。

Jin G.等人[18]结合了 word2vec 和循环神经网络（Recurrent Neural Networks，RNN）去建立软件制品之间的跟踪关系，通过 RNN 预测两个制品的语义相似度。他们的方法在大规模的工业软件数据集上效果比较好，但是，对于一些相对较小的软件数据集，性能并没有预期的那么好。同时，事实上，得到包含有足够跟踪链接的大规模工业软件数据也是比较困难的。

另一个和文献[18]相似的工作是使用机器学习分类器去估计软件文档中跟踪链接的数量[19]。文献[19]中作者使用了多种不同的分类算法和不同的自然语言处理方法进行两两组合去探寻最预测模型的最优精度，该文献还强调了使用分类器去对跟踪链接进行分类，而不是通过设置阈值来过滤跟踪链接的原因是自然语言处理方法在不同的数据集上表现各不相同，从而对于每一个软件数据集进行恢复跟踪链接时都有一个不同的参数。因此，在本文中我们也被启发结合使用机器学习方法和文本检索，去恢复软件需求跟踪链接。

Tien-Duy B. Le 等人[20]使用文本检索和程序频谱（Program spectrum）解决故障定位问题，文中使用三种方法计算了三种相似度，分别是：使用向量空间模型计算得到的相似度、使用论文中提出的方法 Tarantula 计算得到的相似度和使用基于可疑词的方法计算得到的相似度，然后将三者通过三个参数结合起来，作为最终的相似度。Shaowei Wang 等人[21]提出一个相似的方法去定位故障，从五个维度，即五种不同的源文件去定位故障，然后将五个结果整合在一起。这两种方法的共同点是从不同的角度去解决同一个问题，然后通过参数将不同的解法整合在一起，再通过随机梯度下降的方式对超参数进行求解。故障定位的任务和本文的需求跟踪任务相似，都是根据给定的文本或者代码文件去查找与之相关的文本或代码文件。但是，故障定位任务往往使用一些和程序故障相关的特征去构建相关模型，比如使用程序频谱，即程序执行成功或者失败的路径，这些特征在需

求跟踪任务中无法获得，因此该类任务的方法难以直接应用到软件需求跟踪任务中。

另一个相关的工作是文献[22]，该工作使用主题模型和 PageRank 算法去发现相关指导片段去完善 API（Application Programming Interface，应用程序编程接口）。他们使用非监督学习的方法，克服了监督学习中需要大量标注数据和对数据集高度依赖等缺点，但是由于他们的方法使用了很多文档片段的特征，因此很难迁移到需求跟踪任务中。

除此之外，软件制品，包括需求文档、设计文档和源代码等，包含大量自然语言文本与代码的组合、代码片段和一些专有名词。这就导致了其与普通文章不同，对于普通文章适用的一些文本检索方法并不适用软件制品数据，或不能直接应用在软件制品数据上。

1.3 研究目标和研究内容

本文的研究目标是研究机载软件生命周期数据跟踪关系的构建方法，通过建立模型，能够自动判断各生命周期的数据是否能够满足 DO-178C 中规定的可追溯性目标，为适航软件的审定提供辅助依据。具体来说，包括对 DO-178C 中规定的可追溯性目标的研究和适航审定对该部分的具体要求、软件需求跟踪模型的研究与建立。本文的主要研究内容之间的关系如图 2 所示。

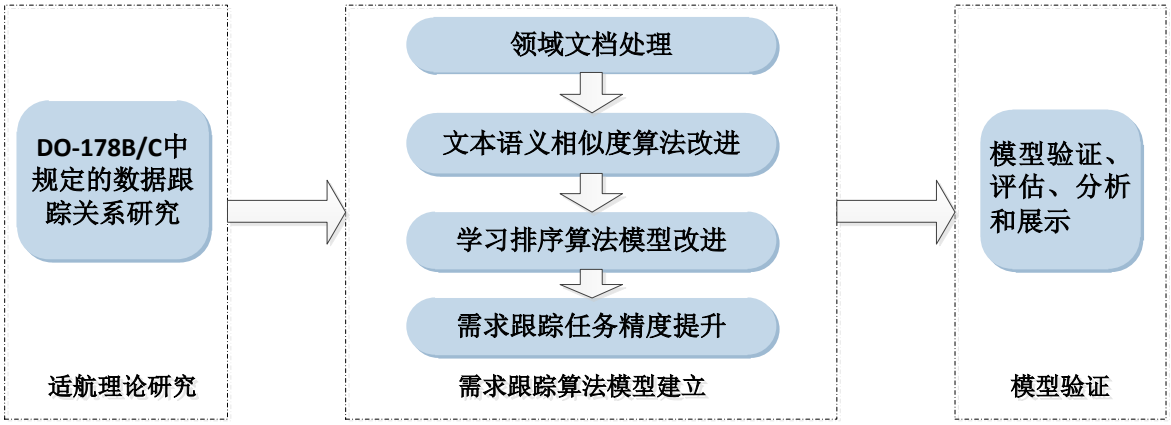


图 2　主要研究内容关系图

具体来说，主要研究内容包括：

（1）适航理论研究。DO-178C 标准中对软件生命周期的定义与普通软件有所不同，对系统和软件的可追溯性具有严格的要求。本文将对适航标准 DO-178C 及其补充标准进行研究，明确在机载软件的软件生命周期，以及每个过程中应该达到的目标，重点在于标准中与需求跟踪有关的部分，和在软件设计、开发到使用的全过程需要注意的问题。

（2）需求跟踪算法模型构建。本文在算法构建阶段重点关注使用基于词向量的文本语义相似度计算的有效性和学习排序在需求跟踪任务中的效果。算法构建过程主要包括：领域文档处理、文本语义相似度算法改进、学习排序算法模型的改进以及对需求跟踪任务精度的提升。

（3）模型验证。从多个维度对提出的模型进行验证，将模型拆分成两个部分，分别对两部分的算法进行验证，并和当前一些领先的方法进行对比；最后将模型整体与国际领先的 ENRL 算法进行效果对比。实验使用开源的、经常用于跟踪恢复的文本数据集，采用 F-measure、MRR 和 MAP 等指标对模型进行评估。同时，实现了基于 word embedding 的适航领域软件需求跟踪原型系统。

综上所述，本文通过在需求跟踪任务中引入词向量、加权策略和查询扩展的方法对文本相似度算法进行了改进，并对 IR SVM 学习排序算法损失函数中的权重进行改进，然后使用改进的学习排序算法对需求跟踪链接进行进一步的优化，最后提出了一个面向适航领域软件文档的需求跟踪模型，并完成了模型的构建、分析、验证和展示。

1.4 论文组织结构

本文共分六章，各章节编排以及主要内容如图 3 所示：

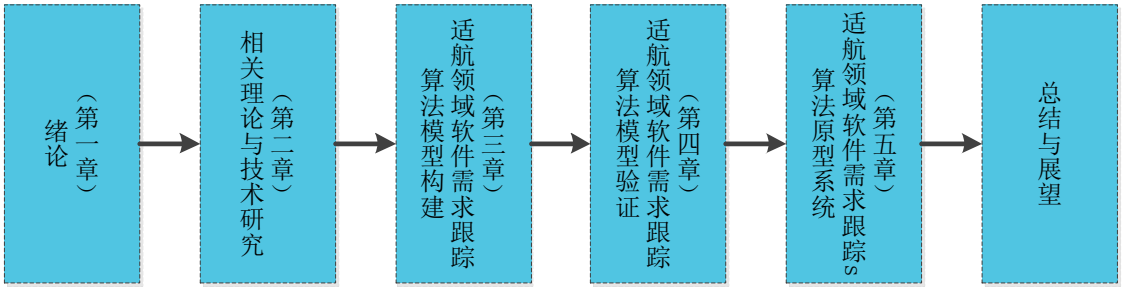


图 3 章节组织及各章节主要内容

第一章，绪论。首先给出了课题来源，介绍了论文研究背景及意义，阐述和总结了软件需求跟踪技术和软件工程领域文本检索算法的国内外研究现状，同时还介绍了本文的研究目的和主要研究内容，最后简述了文章的组织结构。

第二章，相关理论与技术研究。本章围绕软件需求跟踪任务流程中的关键技术，调研了国内外需求跟踪任务的研究现状，以及同样使用信息检索技术解决的与需求跟踪任务相似的一些任务的国内外研究现状，以及软件工程领域对学习排序算法的应用和改进。通过对相关理论和技术的研究，确定了本文研究工作的理论依据和研究方向。

第三章，适航领域软件需求跟踪算法模型构建。本章提出了一个新的基于 word

embedding 和学习排序算法的软件需求跟踪算法模型 Tr-WELR（Traceability based on Word Embedding and Learning to Rank）。首先提出了在软件需求跟踪任务中使用加权策略和查询扩展技术，然后使用 word embedding 改进了文本语义相似度计算算法，接着使用已经计算好的文本语义相似度改进了 IR SVM 算法损失函数中的权重设置方式，最后使用学习排序算法对排序结果进行二次使用，提升需求跟踪任务的精确率。

第四章，适航领域软件需求跟踪算法模型验证。本章设置了多组对比试验对提出的模型进行验证。首先对数据的收集流程和实验数据集做了介绍，然后详细描述了文档预处理的方法，接下来介绍了实验的一些设置，包括对比方法、词向量训练方法和实验环境，之后对模型的评估指标做了介绍，最后对实验结果和运行时间效率进行了分析和说明。

第五章，适航领域软件需求跟踪原型系统构建。本章介绍了原型系统的需求、开发环境和系统的总体设计，然后介绍了原型系统中的主要功能模块的设计与实现，同时展示了该原型系统的主要功能。

总结与展望。总结本论文的研究工作，并提出尚未解决的问题，制定今后的研究工作重点和对该领域发展的展望。

第二章 相关理论与技术研究

本章主要介绍适航领域相关理论、软件需求跟踪领域理论和相关技术。首先介绍了适航领域中机载软件相关的标准 DO-178B/C，详细说明了与需求跟踪相关的适航审定目标；然后对信息检索技术的常用技术和模型进行介绍；接下来介绍了本论文中使用的 word embedding；最后介绍了学习排序算法的基本概念，以及在软件工程领域中的使用情况和已有的方法。

2.1 适航领域相关理论

本节首先介绍适航领域软件相关的标准，以及标准中规定的与普通软件开发不同的软件生命周期和相应的软件生命周期数据，然后介绍标准中与软件需求跟踪相关的内容。

2.1.1 DO-178B/C 标准介绍

20 世纪 70 年代末，计算机软件应用在飞机设备和系统中使用越来越多，因此为了保证这些应用能够满足适航局的审定要求，美国航空无线电技术委员会制定了相应软件规则，用来支持以软件为基础的设备和系统的合格审定[1]。自此之后，DO-178 系列《机载系统和设备合格审定中的软件考虑》产生，并不断补充扩展，经历了 DO-178、DO-178A、DO-178B，现在已经发展到 DO-178C。其中 DO-178B 标准是相对稳定的，其制定汲取了多方面共同意见，包括飞行制造商、设备供应商、工具开发商和适航认证机构等，在使用的 19 年过程中没有发现严重的问题。由于在实际开发软件的过程中，方法会因项目的不同而不同，因此 DO-178B 标准是面向过程和目标的，即尽可能的不涉及到实际的技术，但是，DO-178B 在一些方面，“面向目标”的原则贯彻的不够彻底，因此对 DO-178B 做了一些修正，于是出现了 DO-178C。

DO-178B/C 规定的软件生命周期过程包括：软件计划、软件开发以及软件综合过程，三个过程结构图如图 4 所示。其中软件计划过程规定了五个计划和三个标准。五个计划包括软件审定、开发、验证、构型管理以及质量保证等计划，三个标准包括软件需求、设计以及编码等标准。与开发和设计人员关系最密切的软件开发过程由软件需求、设计、编码和集成等四个过程组成。软件综合过程包括软件验证、构型管理、质量保证以及审定联络等四个子过程，并且综合过程在软件全生命周期中起作用。

在整个软件生命周期中，软件开发过程是主线，软件综合过程与软件开发过程同时执行，并且软件综合过程执行在软件开发过程的各个子过程上，具体的关系如图 5 所示。

虽然标准中对软件生命周期过程规定较多，但是标准并没有强制所有软件活动必须按照规定活动严格执行，只需要在软件生命周期中描述清楚所有过程和活动的先后顺序和执行关系，并定义过程之间的迁移准则即可[2]。

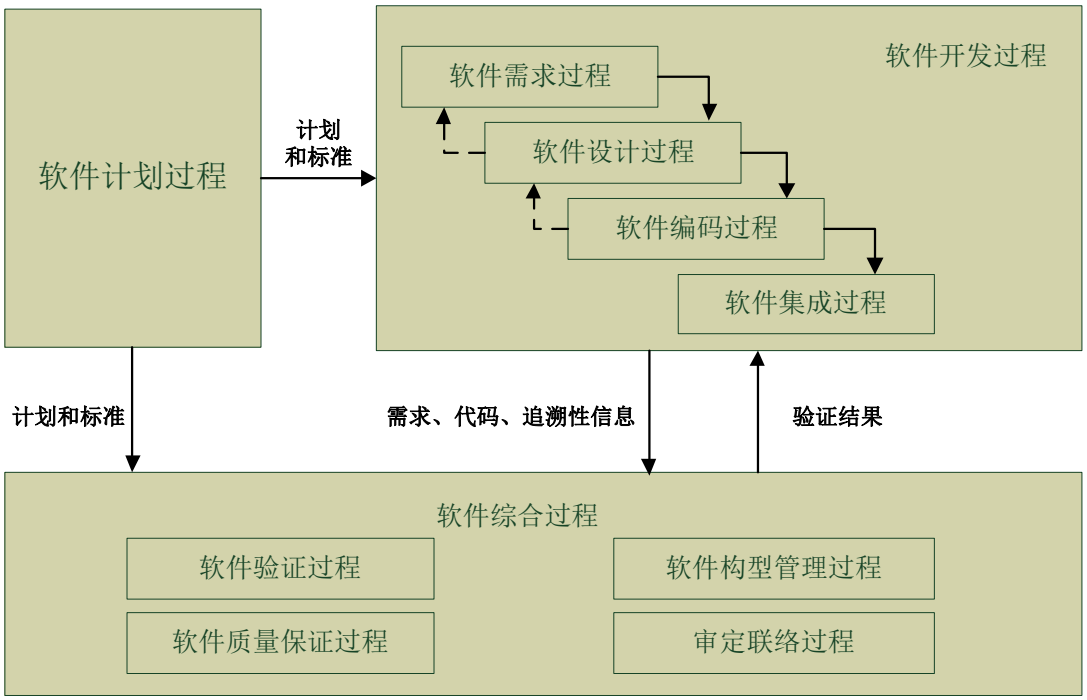


图 4　DO-178B/C 生命周期过程结构图

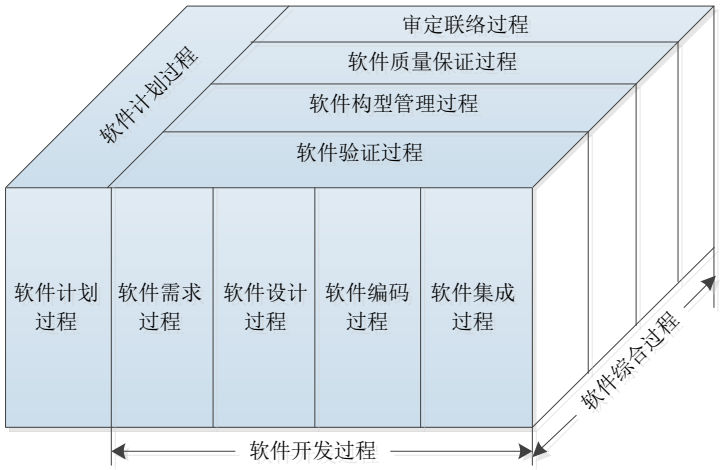


图 5　DO-178B/C 软件生命周期方阵图

在整个软件生命周期过程中，产生了大量的软件生命周期数据，这些软件生命周期数据供审定局方来验证软件的研制开发已经满足了标准中的相应目标。DO-178B/C 中列举了 20 种软件生命周期数据，其中包括软件计划过程中的五个计划数据和三个标准数据，软件开发过程中的软件需求数据、设计描述数据、源代码数据、可执行目标代码数据和软件验证用例和规程，以及软件综合过程中的软件验证结果、环境构型索引、软件

构型索引、软件构型管理记录、问题报告、质量保证记录以及软件完结总数。其中本文将使用到的数据包括软件需求数据和源代码数据。

DO-178B/C 根据软件失效条件将软件分为不同的等级，不同等级的软件需要满足不同数量的目标。标准对于每个过程都制定了相应的需要完成的目标，目标是审定方审定软件是否满足适航要求的最终依据。

DO-178B/C 三个基本要素为上述介绍的过程、软件生命周期数据和目标，三者的关系如图 6 所示。

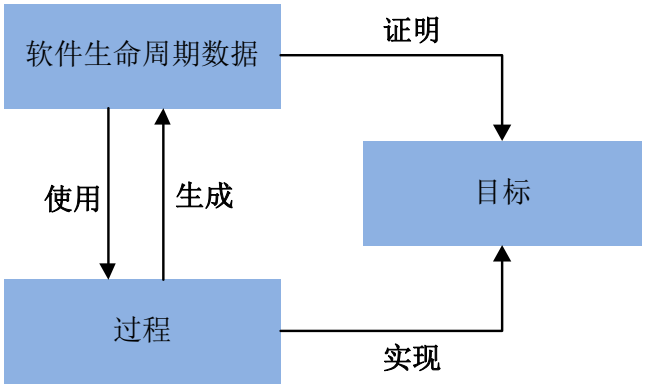


图 6　DO-178B/C 中基本要素的相互关系

2.1.2　DO-178B/C 标准中规定的可追溯性目标

DO-178B/C 标准是面向目标的，标准中提及的目标是适航审定当局对于机载软件是否满足适航性要求的判断指标。针对软件开发过程中的软件需求过程、软件设计过程、软件编码过程和软件集成过程，DO-178B/C 要求至少完成以下三个可追溯性目标：

（1）系统需求和高级需求之间的可追溯性

系统需求指的是机载系统对于某个软件功能的描述，是机载系统（非机载软件）在开发的过程中对软件提出的要求；高级需求是对软件的某个功能的描述，是若干个需求的集合，因此在软件开发过程中软件的高级需求的完整性和正确性对软件的研制起着重要的作用。

系统需求与高级需求之间的追溯性关系连接了系统开发过程和软件开发过程，二者的双向可追溯性保证了系统开发层面要求的性能要求、系统功能和安全性方面的要求已经转换为了软件的高级需求。

（2）高级需求和低级需求之间的可追溯性

分配给软件的高级需求只是在一个抽象层面上表示了软件实现的功能，还需要进一步进行细化为可以进行编码的需求。低级需求即为高级需求具象化之后的需求，对每一

个小功能描述的更详细、对编码应该采用的语言和框架等描述的更清晰。但是在高级需求进行细化时，可能会产生衍生的高级需求[1]，即系统需求可能只提供了一个大致的功能要求，但是部分高级需求并不能直接从系统需求追溯过来，因此需要提供一些衍生的高级需求来完成更具体的功能；同理，有些高级需求描述的功能，能够与其追溯的低级需求并不能实现全部的功能描述，因此也需要提供衍生的低级需求。

高级需求和低级需求之间的追溯性关系是连接了抽象功能和低层实际编码需求之间的纽带，同时通过可追溯性保证所有的高级需求都必须被细化为了低级需求。

（3）低级需求和源代码之间的可追溯性

按照低级需求文档已经可以进行软件的编码工作等具体的软件开发，源代码则是低级需求的呈现结果。低级需求和源代码之间的追溯性关系是为了保证所有的软件低级需求都已经通过代码来实现。

图 7 展示了以上的可追溯性关系。

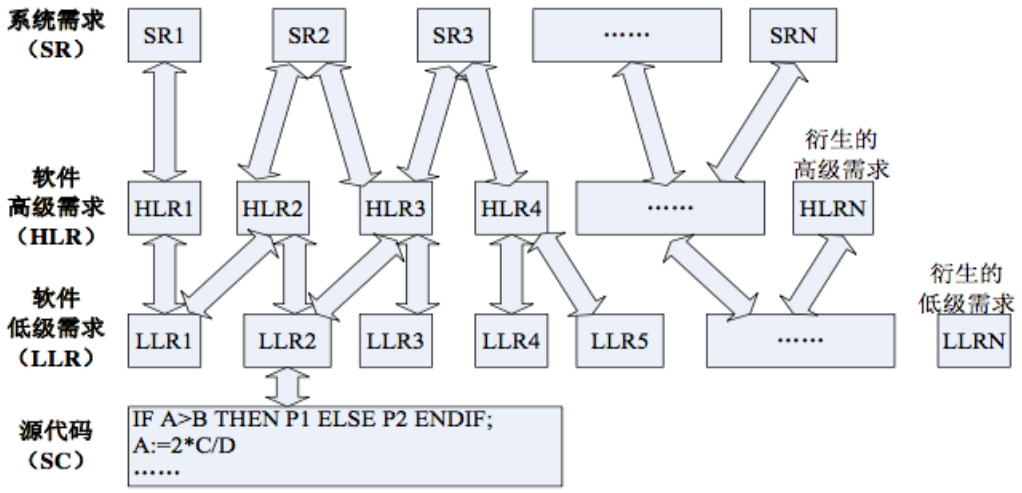


图 7　DO-178B/C 中要求的追溯性关系

当然，除了以上在 DO-178B/C 的目标中严格规定的可追溯性之外，在软件的研发过程中还应该满足一些其他软件生命周期数据之间的可追溯性，比如软件需求与测试用例、测试程序、测试结果之间的追溯性等。

2.2　信息检索技术

信息检索技术是一个比较老的概念，指的是根据给定的查询项在一个数据集合上查找出相似的、相关的内容[23]，最后将结果按照相关度排列。对于信息检索，大部分的工作是基于文本检索，即查询项和待查询集合的内容都是以文本形式存储，主要方法可以

分为两类：基于统计检索和基于语义检索。

2.2.1 基于统计检索

基于统计的方法是根据查询语句和待查询集合的内容的统计结果来确定查询结果。将查询和待查询集合放在同一个统计空间中，对文本进行分词，将所有的内容基于统计信息进行表示，然后计算得到与查询项相关的内容。当前常用的基于统计检索的方法包括向量空间模型和概率模型。

向量空间模型把文本信息用向量空间中的点表示，然后计算向量之间的距离来量化两个文本之间的相似度，最后通过对计算结果排序得到查询结果。文档向量构造最简单的方法是使用词袋模型（Bag of Words，BOW），即文档向量的长度是数据集中所有的词的数量，文档向量每一维的值为当前词在该文档中的出现频率或其他的一些特征，从而将一个文档表示为一个向量。在计算文本间距离时，可以使用余弦公式等方法，得到两个文本之间的相似度值。还有一些方法通过对关键词加权重的方式表示文本向量[24]。

潜在语义索引也是基于向量空间的检索方法。LSI 使用 one-hot 方法表示文档，并将所有文档放在同一个矩阵中，通过奇异值分解（Singular Value Decomposition，SVD）的方法对矩阵降维，并且得到文本的主题和主题之间的关系。但是由于使用向量表示之后矩阵规模巨大、同时矩阵会过于稀疏，因此需要进行降维操作。LSI 方法虽然表示和计算过程都比较简单，但是也有如下几个缺点：（1）通过 SVD 生成的新矩阵解释性较差；（2）无法解决“一词多义”的问题；（3）和词袋模型一样，忽略了文章中单词的先后顺序。

概率模型基于多个随机变量之间概率关系计算查询项和候选文档之间的关系，将查询项与文档集合中的文档同时出现的概率定义为二者的相似度，形式化表示如公式(2.1)所示。在计算完成后，对相似度进行排序，然后设置阈值过滤得到某个查询对应的相关内容，从而完成信息检索任务。

$$Sim(q, D_i) = Pr(D_i|q) = \frac{Pr(q|D_i)Pr(D_i)}{Pr(q)} \tag{2.1}$$

其中D_i表示文档集合中第 i 个文档，q表示查询语句，公式(2.1)使用贝叶斯公式对二者同时出现的概率进行求解。

2.2.2 基于语义检索

基于语义的方法是在一定程度上通过对查询句子进行句法和语义的分析，即通过其他的表现形式对文本的语义进行表示。由于当前大部分信息检索都是基于文本检索，因

此基于语义的信息检索也和自然语言处理技术相关。使用句法分析的方法是借助文本的句式和一些词性信息对文本语义进行强化处理，如在文献[25]中提到的使用子树加速对文本相似度的计算，并将子树与文本特征向量进行比较匹配，从而得到更准确的文本相似度；使用语义分析的方法有基于本体[26]和基于外部语义词典（如 Hownet[27]和 Wordnet[28]）等方法，这些方法基本是将查询文本中的单词或词组抽离出来，然后用本体或者外部语义词典中的含义来表示，从而将整个文档集合中的内容统一表示，然后使用给定的算法计算相似度，从而确定检索列表。

2.3 word embedding 介绍

Word2vec 是 Google 的 Tomas Mikolov[29]等人提出的文本表示方法，将单词表示成 word embedding 的形式，是一种分布式表示模型。Word embedding 的向量表示反应了词的共现关系。在给定的语料库下，通过算法模型将词表示成向量的形式，向量的维数可以由用户指定，通常为几十到几千不等，因此可以远远小于词典的大小，这样就避免了向量空间模型中的矩阵稀疏问题。同时，Mikolov 等人还开发出了训练词向量的工具，训练效率非常高，非常适合在现在大数据集的应用中。

Word embedding 基于分布式假说提出，即出现在相同上下文中的词的含义相近[30]。训练 word embedding 的流程类似前馈神经网络，但是仅包含输入层、隐藏层和输出层三层结构，并且由于 Mikolov 等人对隐藏层和输出层做了优化，使得模型的训练效率更高。训练模型主要包括 skip-gram 和 CBOW 两种，skip-gram 模型的输入是特定词的词向量，输出是特定词上下文相关的词的词向量，而 CBOW 和 skip-gram 的输入输出相反。skip-gram 和 CBOW 模型分别如图 8 和图 9 所示。

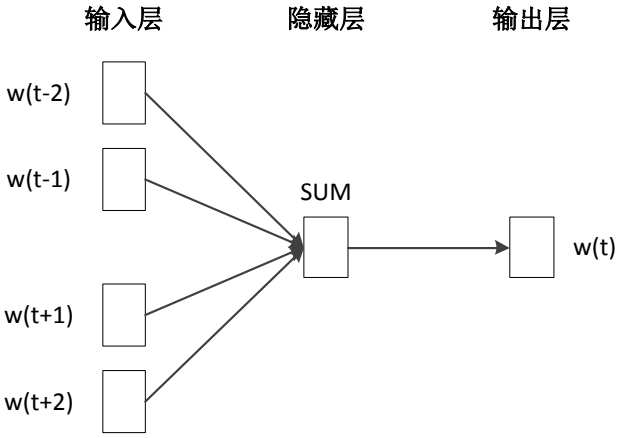


图 8　skip-gram 模型示意图

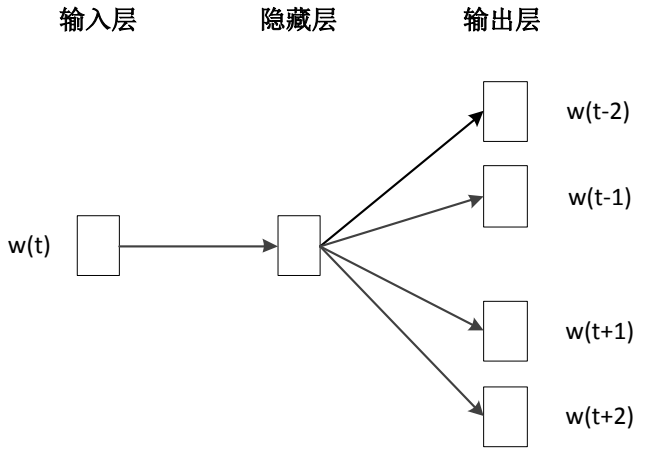


图 9　CBOW 模型示意图

2.4　软件工程领域学习排序算法

学习排序（Learning to Rank，LtR）算法作为一种监督或半监督的机器学习算法[31]，在信息检索、数据挖掘和自然语言处理等领域起着重要的作用。在软件工程领域，学习排序算法经常用在故障定位和数据重复检测等方面。对于每个查询与候选文档的组合，我们能够抽取出若干特征，比如词汇相似度、语义相似度等，作为机器学习模型的输入。学习排序的流程如图 10 所示，更加形式化的表示如公式(2.2)和公式(2.3)所示。

$$R = h(w, \varphi(q_i, D_i)) \tag{2.2}$$

$$D_i = \left\{ d_{i_1}, d_{i_2} \cdots d_{i_j} \cdots d_{i_n} \right\} \tag{2.3}$$

其中q_i表示查询语句，D_i表示相应的候选文档集合，$\varphi(\cdot)$函数将查询-文档对映射为特征向量，w是表示每个特征向量的权重矩阵，$h(\cdot)$是排序方法。排序方法$h(\cdot)$有三种类型：单文档方法（Pointwise）、文档对方法（Pairwise）和文档列表方法（Listwise）。

单文档方法的处理对象是单独的一篇文档，将其转化为特征向量后，该问题转化为分类或回归问题，可以使用等级回归或分类算法来解决问题。该方法只考虑单个文档的绝对相关度，忽略了文档间顺序关系。文档对方法则考虑了文档间的关系，把任意两个文档组成的文档对（文档 A，文档 B）作为机器学习的输入，最后得到文档 A 是否应该排在文档 B 的前边，此时学习排序算法可以转化为二分类问题。该方法通过考虑两两文档对之间的关系进行排序，因此比单文档方法的效果有所提升。文档列表方法的处理对象是每个查询所对应的所有搜索结果的列表，该方法直接训练优化算法模型输出的整体序列，因此其结果能够更接近真实的文档序列。已有的且常用排序算法如表 1 所示。

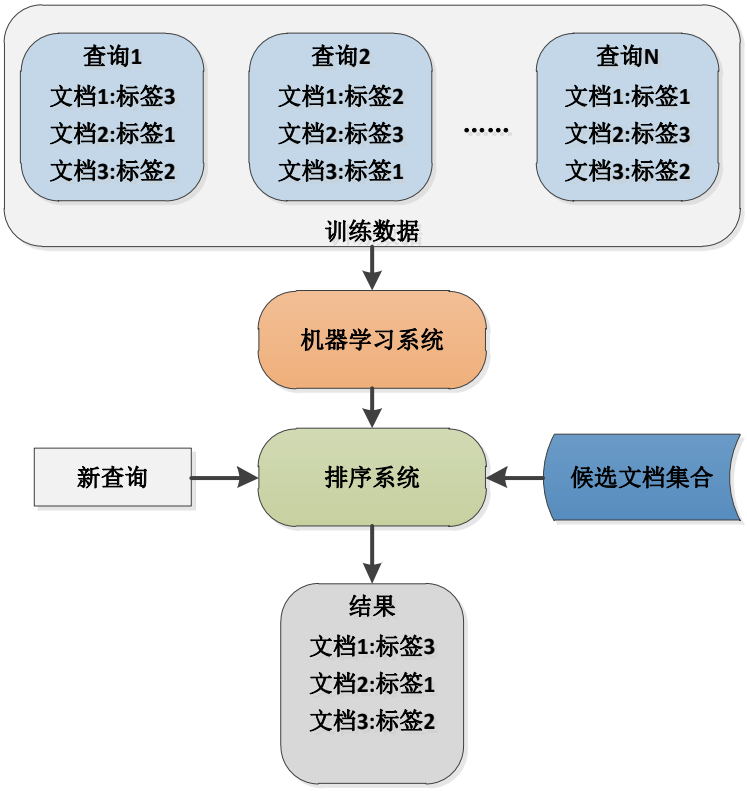


图 10　学习排序的流程

除了排序算法，选择合适的特征作为模型的输入对学习排序算法的性能也有很大的影响。在软件工程领域，大多数使用学习排序算法的应用都选择词或者文本的相似度作为其中一个特征，比如文献[32]中作者使用了词表面信息相似度、通过 API 增强的词相似度和类名的相似度，文献[33]中作者选择文本相似度和上下文相似度作为学习排序算法的特征。值得一提的是，在计算以上相似度时，作者们都是用了余弦相似度。除此之外，特征可以分为两类：依赖查询的特征和不依赖查询的特征[33]，其中依赖查询的特征即与查询文本相关的特征，如相似度特征；不依赖查询的特征则与查询文本无关，仅仅反应候选结果的特点，如候选结果的词频[33]、故障修复频率[32]等。同时，在选择特征时，特征数量不应过大或过小，过大会导致过拟合问题，过小则会降低结果精确度。

表 1　排序方法的常用算法

排序方法分类	常用算法
单文档方法	回归：Subset Ranking[34]　分类：SVM、McRank[35]
文档对方法	IR SVM[36]、Ranking SVM[37]、RankBoost[38]
文档列表方法	ListNet[39]、AdaRank[40]、SVM Map[41]

2.5 本章小结

本章首先介绍了适航领域中的 DO-178B/C 标准中对于软件生命周期的定义、标准中各个基本要素的关系以及标准中对于软件需求跟踪的一些严格的约束；然后介绍了信息检索领域中常用的技术，包括基于统计检索方法和基于语义检索方法；接下来介绍了 word embedding 的概念和两种训练模型的区别；最后对软件工程领域的学习排序算法的应用以及常用排序算法进行介绍。本章通过对适航领域软件特点的分析和需求跟踪技术研究中相关的技术的介绍，为下一章适航领域软件需求跟踪算法模型的构建提供了理论基础和支撑。

第三章　适航领域软件需求跟踪算法模型构建

本章详细的介绍了软件需求跟踪算法模型的构建。首先介绍了需求跟踪算法模型的整体框架 Tr-WELR，并描述了构建模型时应考虑的问题，然后详细描述了模型中文本相似度计算阶段和学习排序阶段两个主要部分，最后对模型构建过程进行总结。

3.1　Tr-WELR 模型框架

适航软件文档组织结构复杂，文档中专业领域名词较多，因此适航文档的预处理工作非常重要。适航软件生命周期与普通软件生命周期不同，在开发过程中，每个子过程完成时，都需要进行软件验证，其中包括追溯性验证。因此，与普通软件需求跟踪不同的是，适航领域软件需求跟踪模型需要能够在软件尚未开发完成时，通过已有的数据自动构建跟踪关系，因此需要针对适航领域软件跟踪关系建立的时机，适当在模型中引入补充数据，用来保证计算过程的合理性和准确性。由于在软件开发过程中的数据基本都可以用文本表示，因此根据生命周期阶段的不同，引入该阶段中应该存在的其他数据。

DO178B/C 标准中描述的软件开发过程包括软件需求过程、软件设计过程、软件编码过程和软件集成过程，其中软件需求过程、软件设计过程和软件编码过程完成时需要与软件验证过程交互，具体交互的数据如表 2 所示。因此可以根据每个过程的不同，补充相应的已有数据。

表 2　开发过程子过程与软件验证过程交互数据

开发过程子过程	交互数据
软件需求过程	软件需求数据、软件开发计划、软件验证计划
软件设计过程	软件设计描述、软件开发计划、软件验证计划
软件编码过程	源代码、软件开发计划、软件验证计划

模型的整体流程如图 11 所示，包括数据预处理阶段、文本相似度计算阶段和学习排序计算阶段。

（1）数据预处理阶段包括适航软件文档的预处理和词向量的训练，其中适航软件文档的预处理包括文档的脱密处理和文本预处理。基于适航文档的重要性和机密性，在处理适航软件文档之前应该先进行脱密处理，常用的脱密方法是将重要的中文名词使用相应汉字拼音首字母表示，如保密数据“北航纠错系统”可以写作“BHJCXT”。具体的

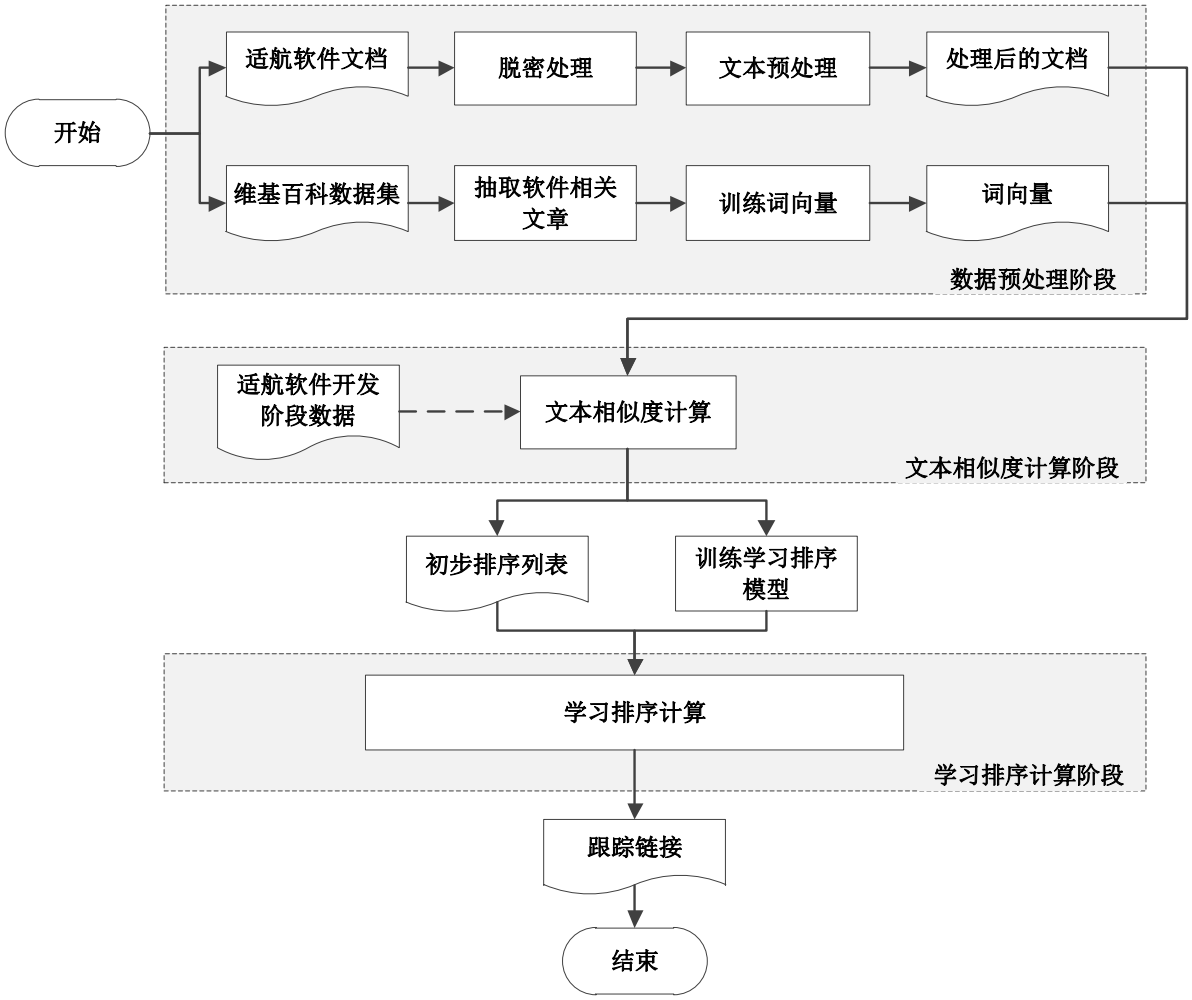


图 11　Tr-WELR 模型流程图

软件文本预处理部分将在实验部分详细描述。由于维基百科数据的数据量庞大，因此为了保证训练的词向量的专业性，仅抽取软件相关文章进行训练，具体操作过程在第四章实验部分详细描述。

（2）预处理完成之后，将软件文档映射为词向量，然后输入到计算文本相似度算法中，该算法命名为 WQI。在算法中，我们将信息检索中查询的概念与待建立跟踪关系的需求文本等价，即把每个需要建立跟踪链接的软件需求文本看作是信息检索中的一个查询语句。算法对每一个查询语句进行文本相似度计算，最终每个查询都得到一个候选的排序列表，排在最前边的是与相应查询相关度最高的软件制品。

（3）最后，将排序列表引入学习排序算法模型中，利用机器学习分类算法将排序后的列表进行二次处理，最终得到相关跟踪链接。

接下来将详细介绍 Tr-WELR 模型中对文本语义相似度计算和学习排序模型中的改进部分。

3.2 文本语义相似度计算

本节主要介绍了通过引入单词加权策略和查询扩展方法来改进的文本语义相似度计算方法，并且为了满足 3.1 节中提到的适航领域软件需求跟踪特点，在使用单词加权策略时，用到的数据集合随着软件生命周期的不同而有所调整。

3.2.1 单词加权策略

为了区分句子中的单词的重要程度，我们选择对句子中的单词进行加权处理。本文使用 TF-IDF 的策略来计算，如公式(3.3)所示。TF（Term Frequency，词频）表示某个词在当前文档中的出现频率，如公式(3.1)所示；IDF（Inverse Document Frequency，逆文档频率）与含有某个单词或短语的文档数量有关，并且包含该单词或短语的文档数量越少，则该值越大，即表示当包含该单词或短语的文档数越少时，该词在当前文档中所占的权重越高，如公式（3.2）所示。当一个词或短语在一个文档中出现的频率较高，而在其他文档中出现频率较低时，则可以认为这个词或短语具有较强的类别区分能力，即能够比其他词更有代表当前文档的能力。

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \tag{3.1}$$

$$idf_i = log\frac{|D|}{|\{j:t_i \in d_j\}|} \tag{3.2}$$

$$tfidf_{i,j} = tf_{i,j} \times idf_i \tag{3.3}$$

其中$tf_{i,j}$指单词t_i在文档d_j中的词频，分子表示在文档中该单词出现的次数，分母表示文档中单词的总数。idf_i指单词或短语t_i在文档集合D中的逆文档频率，分子表示文档总数，分母表示出现过此单词或短语的文档数，一般为了防止分母为 0，会给该文档数加一，最后对比值取对数。

在当文档数量较少时，计算得到的 tfidf 值并不准确，而在适航领域，可能在软件需求相关文档尚未充足的情况下建立需求跟踪关系，因此此时需要进行数据补充。使用当前软件生命周期阶段已有的文档进行数据扩充，能够在一定程度上缓解 tfidf 计算不准确的问题。

3.2.2 查询扩展方法

在论文中，计算两个词之间的语义相似度我们首先将词表示为词向量的形式，然后使用 cosine 相似度计算，如公式(3.4)所示。

$$sim_{w2w}(w_i, w_j) = cos(\boldsymbol{w_i}, \boldsymbol{w_j}) = \frac{\boldsymbol{w_i}}{\|\boldsymbol{w_i}\|} \cdot \frac{\boldsymbol{w_j}}{\|\boldsymbol{w_j}\|} \tag{3.4}$$

其中$\boldsymbol{w_i}$和$\boldsymbol{w_j}$分别表示词w_i和w_j的词向量，$\|\boldsymbol{w_i}\|$和$\|\boldsymbol{w_j}\|$分别表示两个词的词向量的长度。

合适的查询扩展方法和加权策略能够提升信息检索任务的性能[42]。对某个文档s进行查询扩展，是对文档中的各个单词或词组进行扩展，步骤如图 12 所示，描述如下：

（1）首先计算 TFIDF 权重，然后根据 TFIDF 进行排序，选择前$topn\%$的单词或词组进行扩展，组成集合EXT_S，其中$topn\%$参数设置为经验值 0.3。

（2）对以上选择出的前$topn\%$的单词或词组进行扩展，使用单词语义相似度计算公式，选择出相似度大于阈值δ_{qe}的单词或词组，每个被扩展的词组成集合QE_SET_w，形式化的表示如公式(3.5)所示。其中δ_{qe}参数同样设置为经验值 0.7。

$$QE\_SET_w = \{word | sim_{w2w}(w, word) > \delta_{qe}\} \tag{3.5}$$
$$s.t. w \in EXT_S$$

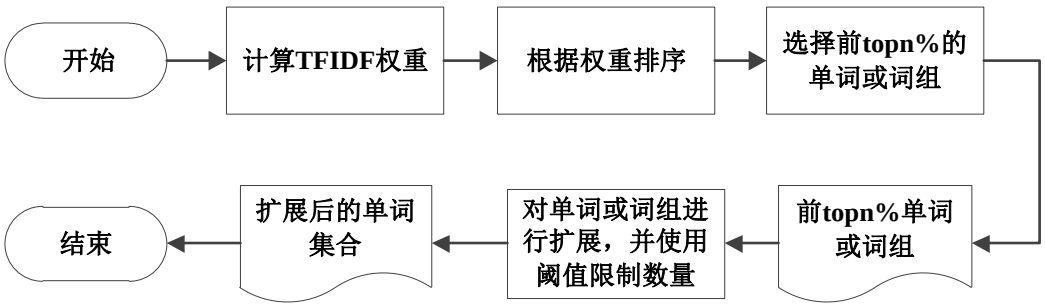


图 12　查询扩展流程图

3.2.3　改进的文本语义相似度算法

在论文[16]中，文本语义相似度的计算有三个步骤：1）计算词之间的语义相似度；2）计算词与文本之间的语义相似度；3）计算文本之间的语义相似度。词与词的相似度计算如公式(3.4)所示，词w与文本T的相似度计算即为词w与文本T中所有词相似度的最大值，如公式(3.6)所示。文本T_i与T_j的相似度为文本T_i中的每个词与文本T_j的相似度的和，然后做归一化处理，如公式(3.7)所示。

$$sim_{w2t\_ori}(w, T) = max_{w' \in T}\, sim_{w2w}(w, w') \tag{3.6}$$

$$sim_{t2t\_asy}(T_i \to T_j) = \frac{\sum_{w \in P(T_i \to T_j)} sim_{w2t\_ori}(w, T_j)}{|P(T_i \to T_j)|} \tag{3.7}$$

$$P(T_i \to T_j) = \{w \in T_i | sim_{w2t\_ori}(w, T_j) \neq 0\} \tag{3.8}$$

其中$P(T_i \to T_j)$表示文本T_i中与文本T_j的语义相似度不为 0 的单词或词组集合。

以上即为论文[16]中提出的方法，我们称之为 W2V 方法。相对于该方法，我们做出如下改进：

（1）在计算词与文本的相似度时，添加上查询扩展，于是重写了公式(3.6)，如公式(3.9)所示。添加上查询扩展后，同时还设置了参数α来分配原词的相似度和扩展的词的相似度之间的权重，该值的取值范围为$(0.0, 1.0)$，在实际的计算过程中，该值在 0.5 到 0.9 之间，以 0.01 的步长增长，遍历整个范围，当效果最好时，设定α值，因此α参数根据数据集的不同而不同。如果当前单词没有扩展词，则使用与论文[16]中相同的方法，即使用公式(3.6)计算。

$$sim_{w2t}(w, T) = max_{w' \in T}\{\alpha \cdot sim_{w2w}(w, w') + (1 - \alpha) \cdot g(w, w')\} \tag{3.9}$$

$$s.t.\ g(w, w') = \frac{\sum_{w_k \in QE\_SET_w} sim_{w2w}(w_k, w')}{|QE\_SET_w|}$$

其中$g(w, w')$表示查询扩展项，即当计算词w与词w'之间的相似度时，如果该词具有扩展词，则扩展项使用词w的扩展词集合中的词与词w'之间的相似度之和，然后做归一化处理，保证$g(w, w')$的值的范围在 0 到 1 之间。举个例子，单词“technique”的扩展词有“technology”、“method”和“approach”等，当计算单词“technique”与句子“The basic requirement of planes is safety.”的相似度时，不仅仅计算“technique”与句子的相似度，同样会计算其扩展词“technology”、“method”和“approach”与该句子的相似度，最后使用比例参数α将两者结合起来。

（2）在计算文本之间的相似度时，原方法的公式是不对称的，即对于同样的一对文本T_i和文本T_j，不同的输入顺序会产生不同的结果。为了保证一致性，以及结果与语句长短的无关性，本方法增加了一个限制，即在输入时使文本T_i的长度长于文本T_j。增加该限制主要有两个原因：第一是使得该公式是对称的，当计算任意一对文本的相似度时，使得结果与二者的前后顺序无关；第二是当较短文本是较长文本的子集时，避免较长文本中的其他词汇被忽视。第二个原因是一种特殊情况，举个例子：文本T_i和文本T_j分别是“Airworthiness is necessary.”、“This document emphasizes that airworthiness is necessary.”，在使用公式(3.9)计算时，文本T_i中的每个词在文本T_j中都出现了，因此在使

用公式(3.7)计算词与文本的相似度时，文本T_i中的每个词与文本T_j中的词的相似度的最大值均为 1，此时文本T_j中的其他词被忽略掉了。为了避免这种情况，限制两个文本的前后顺序是重要的。

3.3 学习排序算法

通过上述的文本语义相似度计算之后，对于每一个查询都可以排序生成一个排序列表，即候选列表，设置相应的阈值$finalThreshold$，即可产生相应的需求跟踪关系。但是，此时仅仅使用了文本语义相似度这一个特征，在数据集中还有一些没有使用的特征。综合使用其他的特征可以提升结果的准确率，因此本文使用学习排序算法来进一步对已有的排序列表进行处理。以下将从两个方面描述学习排序算法：特征选择和排序算法。

3.3.1 特征选择

本文选用了五个特征作为学习排序模型的输入，五个特征可以被分为两组：依赖查询的特征和不依赖查询的特征，如表 3 所示。前三个特征为依赖查询的特征，这几个特征强调查询与候选结果之间的关联关系；后两个为不依赖查询的特征，更强调候选结果文本自身的特点。

（1）语义相似度，通过提出的语义相似度算法 WQI 计算；

（2）广义 Jaccard 系数，通过计算文档向量的关系表示两个文档上下文相似度，在本文应用中即为查询语句和一个候选文档之间的上下文相似度，如公式(3.10)所示。

$$EJ(\boldsymbol{x},\boldsymbol{y}) = \frac{\boldsymbol{x}\cdot\boldsymbol{y}}{\|\boldsymbol{x}\|^2+\|\boldsymbol{y}\|^2-\boldsymbol{x}\cdot\boldsymbol{y}} \tag{3.10}$$

其中，向量$\boldsymbol{x}$和向量$\boldsymbol{y}$均为文档向量，此处文档向量使用词向量的平均值表示。$\|\boldsymbol{x}\|$和$\|\boldsymbol{y}\|$分别表示向量$\boldsymbol{x}$和向量$\boldsymbol{y}$的数值大小。

（3）IDF 之和，在候选结果文本上中计算查询中的词或短语的 IDF 值，并求和，用来表示当前查询在某一候选结果上的重要程度，如公式(3.11)所示。

$$IDF_{sum} = \sum_{t\in Query} idf_{result}(t) \tag{3.11}$$

其中t为查询中的单词或短语，$idf_{result}(t)$表示t在查询结果中的 idf 值。

（4）关键词数量，表示查询候选结果中关键词的数量，关键词的界定使用 TFIDF 的值，计算过程与加权策略中介绍的过程相同。

（5）文本长度，表示查询候选结果的长度，在一定程度上体现该结果的有效程度。

表 3　学习排序算法选择的特征

特征	描述	是否依赖查询
语义相似度	使用 WQI 方法计算的文本语义相似度	是
广义 Jaccard 系数	文本的上下文相似度，文本向量由词向量取平均值表示	是
IDF 之和	候选结果文本上的当前查询中所包含的词或短语的 IDF 值之和	是
关键字数量	候选结果中关键字数量，通过计算 TFIDF 然后排序和取阈值得到	否
文本长度	候选结果文本的长度	否

3.3.2　排序算法

正如第二章中所提到了，单文档方法仅考虑了单个文档与查询的绝对相关度，忽略了文档间的顺序关系；文档对方法考虑了任意两个文档之间的相对前后关系，相比单文档方法的效果更好；文档列表方法需要考虑每次对查询候选结果列表，当文档数量较大时，需要考虑的数量较大，相对而言没有文档对方法的效率高。综合以上原因，Tr-WELR 方法模型中使用了文档对方法，并改进了当前成熟的 IR SVM 算法。

IR SVM 的基本思想和文档对方法处理思想一致，都是将排序问题转化为分类问题。IR SVM 模型将两个样本x_i和x_j表示成一个训练样本，然后使用 SVM 模型训练和预测分类。由于 IR SVM 是对 Ranking SVM 的改进，下面将首先介绍 Ranking SVM，然后再详述 IR SVM 对 Ranking SVM 的改进，最后提出在需求跟踪任务中对 IR SVM 的改进。

3.3.2.1　Ranking SVM

Ranking SVM 算法的思想是将排序问题转化为 pairwise 的分类问题，然后使用 SVM 分类模型进行学习并求解[43]。在排序应用中，假定有两组查询对应的文档集合，每组查询结果集合中有三个等级，分别是等级 1、等级 2 和等级 3。举例说明，在第一组查询结果中有三个对象x_1、x_2和x_3，分别属于等级 1、等级 2 和等级 3，如图 13 所示。图中的w向量是第二章公式(2.2)中 $h(w, \varphi(q_i, D_i))$中的权重向量w。但是，此时对于每一个查询得到的候选文档集合，都有一个各自的权重向量，在应用中非常麻烦。因此，对同一组查询结果集合中的不同等级的对象的特征向量进行组合，形成新的特征向量，即把对

象映射到另一个向量空间，然后根据另一个向量空间的向量进行排序。比如将上述的三个对象重新组合为：$x_2 - x_1$、$x_3 - x_2$和$x_3 - x_1$，并且给这些对象重新打标签，如将$x_2 - x_1$、$x_3 - x_2$和$x_3 - x_1$标记为负相关，相应的$x_1 - x_2$、$x_2 - x_3$和$x_1 - x_3$标记为正相关，如图14所示，即可将以上的排序问题转化为二值分类问题，在该例子中就可以通过训练线性SVM 分类器对上述新的向量空间的向量进行分类，进而可以计算得到同一组查询结果集中的不同等级的向量的前后词序。

训练数据表示为$\left\{\left(\left(x_i^{(1)}, x_i^{(2)}\right), y_i\right)\right\}$，其中$i$为$1 \cdots m$，每一个训练实例由两个特征向量$\left(x_i^{(1)}, x_i^{(2)}\right)$表示，标签由$y_i$表示，并且$y_i$的取值为+1 和-1，分别表示正相关和负相关。Ranking SVM 的最优化问题可以形式化的表示为公式(3.12)。

$$minimize: \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}\xi_i \tag{3.12}$$

$$s.t.\ \ y_i \cdot w^T\left(x_i^{(1)} - x_i^{(2)}\right) \geq 1 - \xi_i$$

$$\xi_i \geq 0, i = 1,2,\cdots,m$$

其中ξ_i为松弛变量，m是训练实例的数量，$\|\cdot\|$是第二范式，把松弛变量代入公式(3.12)中，可以得到：

$$min_w \sum_{i=1}^{m}\left[1 - y_i\langle w, x_i^{(1)} - x_i^{(2)}\rangle\right]_+ + \lambda\|w\|^2 \tag{3.13}$$

其中$\lambda = \frac{1}{2C}$，加和的第一项是合页损失函数，第二项为正则项，防止过拟合。

3.3.2.2 IR SVM

IR SVM 是 Ranking SVM 在信息检索领域的一个改进[36]。Ranking SVM 在信息检索领域的不足之处包括两个方面：

（1）Ranking SVM 是将排序问题转化为分类问题，在学习过程中使用了 0-1 分类损失函数。在信息检索任务中，最终结果列表中排在前面的对检索效果的影响更大，而Ranking SVM 对每两个文档的相对顺序都一视同仁。举例说明，有三个等级，其正确的排序顺序应该为等级 1、等级 2、等级 3，则等级 3>等级 2 和等级 3>等级 1 都是错误的相对顺序，但是二者对 Ranking SVM 的训练过程造成的影响是相同的，显然，这与实际的排序过程有一定的误差。

（2）另一方面，Ranking SVM 同等对待不同查询下的结果对。举例说明：每个文

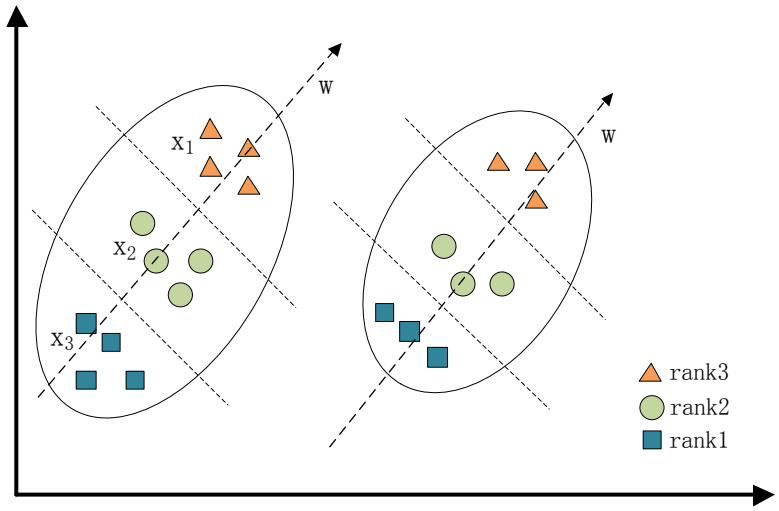


图 13　排序问题示意图

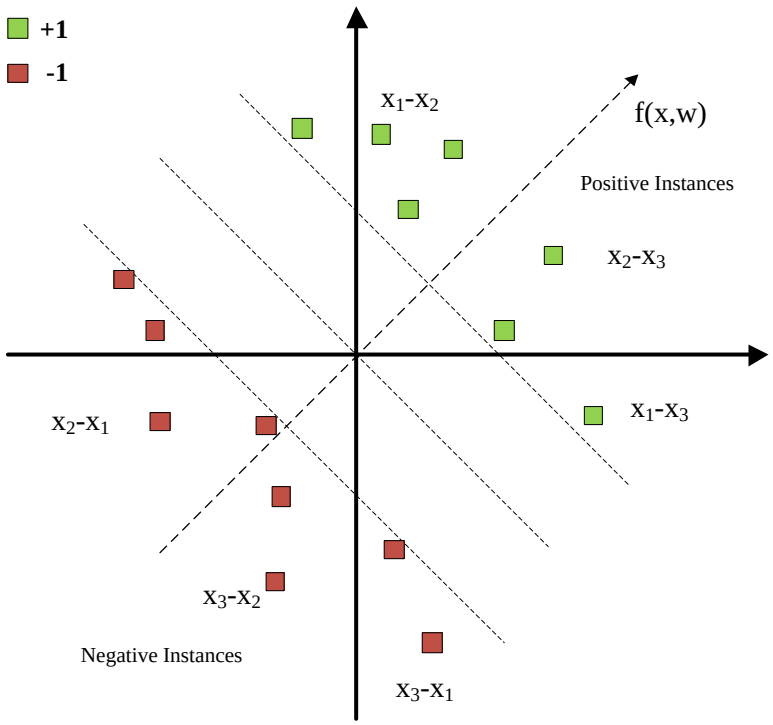


图 14　将排序问题转化为分类问题示意图

档使用相应等级来表示，两个查询的结果如表 4 所示。对于查询一，在转换向量空间时可以有 2 个“等级 3-等级 2”的文档对，4 个“等级 2-等级 1”的文档对，2 个“等级 3-等级 1”的文档对，共 8 个正相关文档对可供训练；对于查询二，则有两个“等级 3-等级 2”的文档对，八个“等级 2 等级 1”的文档对，四个“等级 3-等级 1”的文档对，共有 14 个正相关文档对可供训练。虽然两个查询得到的结果中等级结构相同，但是由于数量不同，查询二对 Ranking SVM 模型的影响会比查询一大，因此最后的结果会有

偏差。同时，这也与信息检索任务中所要求的“每个查询的重要性等同”[23]是不相符的。

表 4 排序列表举例

查询	结果
查询一	五个文档：等级 3、等级 2、等级 2、等级 1、等级 1
查询二	七个文档：等级 3、等级 2、等级 2、等级 1、等级 1、等级 1、等级 1

由于 Ranking SVM 以上两个方面的不足之处，IR SVM 将二值分类问题改进为代价敏感的分类问题，即对来自不同查询的文档对，或者不同等级的文档对设置不同的损失权重。由于在排序结果列表前面的结果比后面的结果的重要程度高，在计算时会对在前边出错的文档对加大损失函数的权重，即出错的文档在列表中顺序越靠前，所需要付出的代价越高；IR SVM 对上述 Ranking SVM 第二点不足的改进是，对排序出错的文档，如果某条查询的结果列表文档数量较少，则加大损失函数的权重，反之，如果结果列表中文档数量较多，则减轻权重，从而减少因为查询本身对查询效果的影响。IR SVM 的最优化问题可以表示成如公式(3.14)所示。

$$min_w \sum_{i=1}^{m} \tau_{k(i)}\mu_{q(i)}\left[1 - y_i\langle w, x_i^{(1)} - x_i^{(2)}\rangle\right]_+ + \lambda\|w\|^2 \tag{3.14}$$

其中$k(i)$表示第 i 个文档对的等级，$\tau_{k(i)}$表示 $k(i)$等级下的权重值，$q(i)$表示第 i 个文档对对应的查询文档，$\mu_{q(i)}$表示 $q(i)$查询对应的相关文档对的参数。$\tau_{k(i)}$和$\mu_{q(i)}$作为惩罚因子，分别用来弥补上述 Ranking SVM 两个方面的不足。参数$\tau_{k(i)}$值的确定是使用一个启发式算法：选定一个评价指标，对每个查询，找到能够使得评价指标最优的排序序列，然后随机交换任意两文档的位置，查看评价指标的降低值，重复该过程，最后对降低值取平均值作为$k(i)$分类下对查询文档对所加的权重。参数$\mu_{q(i)}$值是对查询项$q(i)$对应的相关文档的数量取倒数，当相关文档数目较少时，该值相对较大，进而弥补了在文档数较少时 Ranking SVM 优化函数对其重视程度低的问题。

3.3.2.3 改进的 IR SVM

在 IR SVM 中考虑到了查询结果列表中排在前面的内容对结果影响较大，因此提出了使用公式(3.14)中的权重参数τ来解决这一问题，即对不同等级的查询文档设置不同的权重。如上一小节所说，IR SVM 使用启发式算法计算τ值，但是该方法仅仅考虑了由于顺序不同导致的权重值的不同。

经过 3.3 节的计算，查询文档已经与一个文档序列建立了关系，并且序列中的文档与查询文档通过文本语义相似度值相关联。和查询项相关的文本语义相似度中不仅包含了文档相对顺序的信息，同时还包含了不同位置的文档与查询项的具体关系和一个可以量化的值，因此可以将上述权重参数τ使用文本语义相似度来表示，具体表示如公式(3.15)所示。

$$\tau_i = max(r_i^{(1)}, r_i^{(2)}) \times \left| r_i^{(1)} - r_i^{(2)} \right| \tag{3.15}$$

其中$r_i^{(1)}$和$r_i^{(2)}$分别表示第 i 个文档对（pair）中两个文档与查询项的文本语义相似度值，τ的取值为二者中的较大值与二者差值绝对值的乘积。$max(r_i^{(1)}, r_i^{(2)})$用来表示文档对中排序更靠前的文档在排序列表中的重要程度；$\left| r_i^{(1)} - r_i^{(2)} \right|$指文档对中两个文档与查询项相关度的差值，用来区分不同等级（不同位置）的文档的重视程度。改进之后的参数τ省去了启发式取值的繁琐过程，同时包含了查询结果的相对顺序和每个查询结果与查询项的相关程度。此时，学习排序算法的最优化问题可以表示成公式（3.16）所示。

$$min_w \sum_{i=1}^{m} \tau_i \mu_{q(i)} \left[1 - y_i \langle w, x_i^{(1)} - x_i^{(2)} \rangle \right]_+ + \lambda \|w\|^2 \tag{3.16}$$

其中τ_i为公式（3.15）表示的内容，$\mu_{q(i)}$的意义不变，依然表示查询项$q(i)$对应的查询结果数量的倒数，用来避免某一个查询项有过多的查询结果，从而对训练结果造成偏差。改进的 IR SVM 算法使用 SMO（Sequential Minimal Optimization，序列最小最优化）算法实现，同时由于实例集合非线性可分，因此引入核函数，这里使用高斯核函数，如公式（3.17）所示。

$$K(x, z) = \varphi(x) \cdot \varphi(z) = exp\left(-\frac{\|x - z\|^2}{2\sigma^2} \right) \tag{3.17}$$

其中，$\varphi(\cdot)$为映射函数，在高斯核函数中，并没有指定$\varphi(\cdot)$的具体函数实现，写出该映射函数旨在方便下面对问题的描述。

易知，公式（3.18）是公式（3.16）的等价形式。

$$minimize: \frac{1}{2}\|w\|^2 + \tau_i \mu_{q(i)} \cdot C \sum_{i=1}^{m} \xi_i \tag{3.18}$$

$$s.t.\ \ y_i \cdot \left(w \cdot \varphi\left(x_i^{(1)} - x_i^{(2)} \right) + b \right) \geq 1 - \xi_i$$

$$\xi_i \geq 0, i = 1,2,\cdots,m$$

引入拉格朗日乘子并对其求极大极小问题后，可以得到对偶问题，如公式（3.19）所示。

$$minimize: \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j K\left(x_i^{(1)}-x_i^{(2)}, x_j^{(1)}-x_j^{(2)}\right)-\sum_{i=1}^{m}\alpha_i \tag{3.19}$$

$$s.t.\ \ \sum_{i=1}^{m}\alpha_i y_i = 0$$

$$0 \le \frac{\alpha_i}{\tau_i \mu_{q(i)}} \le C,\ \ i = 1,2,\cdots,m$$

使用 SMO 对上述对偶问题求解的步骤如算法 1 所示，计算得到所有的α值和 b 值，然后可得到 SVM 分类器，分类决策函数如公式（3.20）所示。

$$f(x) = sign\left(\sum_{i=1}^{m}\alpha_i y_i \cdot K\left(x^{(1)}-x^{(2)}, x_i^{(1)}-x_i^{(2)}\right)+b\right) \tag{3.20}$$

其中$sign(z)$是符号函数，当 z 大于等于 0 时，函数值为+1，当 z 小于 0 时，函数值为-1。

算法 1 改进的 IR SVM 中的 SMO 算法部分

输入：训练数据集T = $\left\{\left(x_1^{(1)}, x_1^{(2)}, y_1\right), \left(x_2^{(1)}, x_2^{(2)}, y_2\right), \cdots, \left(x_m^{(1)}, x_m^{(2)}, y_m\right)\right\}$，其中$x_i^{(1)}$和$x_i^{(2)}$均属于$R^n$，$n$ 为特征数量，$y_i \in \{+1, -1\}$，$i = 1,2,\cdots,m$，核函数中参数σ，精度值ε，常数 C；查询项与查询结果的相似度集合$Q = \{(q_1, x_1, s_1^1), (q_1, x_2, s_1^2), \cdots, (q_j, x_k, s_j^k), \cdots\}$，$s_j^k$为查询项$q_j$和文档$x_k$的语义相似度值；每个查询项对应的结果数量集合$W = \{\mu_{q(1)}, \mu_{q(2)}, \cdots, \mu_{q(i)}, \cdots\}$；最大迭代数 *maxIter*

输出：α向量，b 向量

初始化工作：创建 *alpha* 向量并初始化为 0，计算所有的τ值和μ值

```
while 当前迭代次数 < maxIter do
    for all t ∈ T do
        计算当前 alpha 和 b 下分类器的值和y_i的误差值E_i
        if Ei 超过ε的范围 and 当前 alpha[i] 满足限制条件 then
            随机选择另一个 alpha[j]
            同时优化 alpha[i]和 alpha[j]
            if 通过当前 alpha 计算得到的误差值都满足要求精度值ε then
                结束本次循环
            end if
            设置 b 向量
        end if
    end for
    if alpha 在上次循环中没有变动 then
        当前迭代数加一
    end if
end while
return alpha, b
```

完成上述步骤后，即可通过分类器预测查询结果文档的两两相对顺序关系，然后对所有的相对顺序关系进行汇总，即可得到整体的文档排序关系。本文在第四章中将改进的方法与未使用学习排序模型的算法、原 IR SVM 方法分别进行了对比，实验结果表明

在本文任务总，改进的 IR SVM 是有效的，并且比原有的 IR SVM 方法的精确率更高。

3.4 本章小结

本章首先给出了基于 word embedding 和学习排序算法的适航领域需求跟踪算法模型框架，并对框架中包含的几个部分进行了简单介绍。接下来详细介绍了算法模型中改进的部分，其中包括：在计算文本语义相似度时提出的在 word embedding 的基础上添加加权策略和查询扩展，并以此为基础改进了文本语义相似度的计算算法。由于使用单独的信息检索技术在软件需求跟踪关系恢复任务中对数据的特征使用较少，因此提出了使用学习排序算法对结果的精度做进一步的提升，并改进了 IR SVM 算法。

第四章　适航领域软件需求跟踪算法模型验证

本章将通过实验验证和评估第三章所提出的模型 Tr-WELR 的有效性和性能。首先对验证模型所需数据集的来源和相关特征进行详细描述，然后介绍实验设置和实验具体内容，接下来对文档预处理流程进行详细介绍，然后介绍需求跟踪算法模型的评估指标，最后给出了详细的实验结果和分析。

4.1　数据准备

数据获取需要的步骤如图 15 所示，包括（1）相关人员对需求文档、设计文档、代码文件、测试文档、需求变更文档、审定文档和可能有的需求跟踪矩阵文档等项目相关文档的收集，对不存在需求跟踪矩阵或其他形式的需求跟踪文档的情况，还需要相关专家对部分需求的跟踪关系进行标注；（2）对数据进行脱密处理，常用的脱密方法如表 5 所示[44]；（3）对文档进行自动解析，根据文档的结构和标题，将不同生命周期的数据抽取出来，将半结构化数据转为结构化数据，然后录入人员对数据的分类情况进行确认；（4）确认完成，将数据存入数据库中。

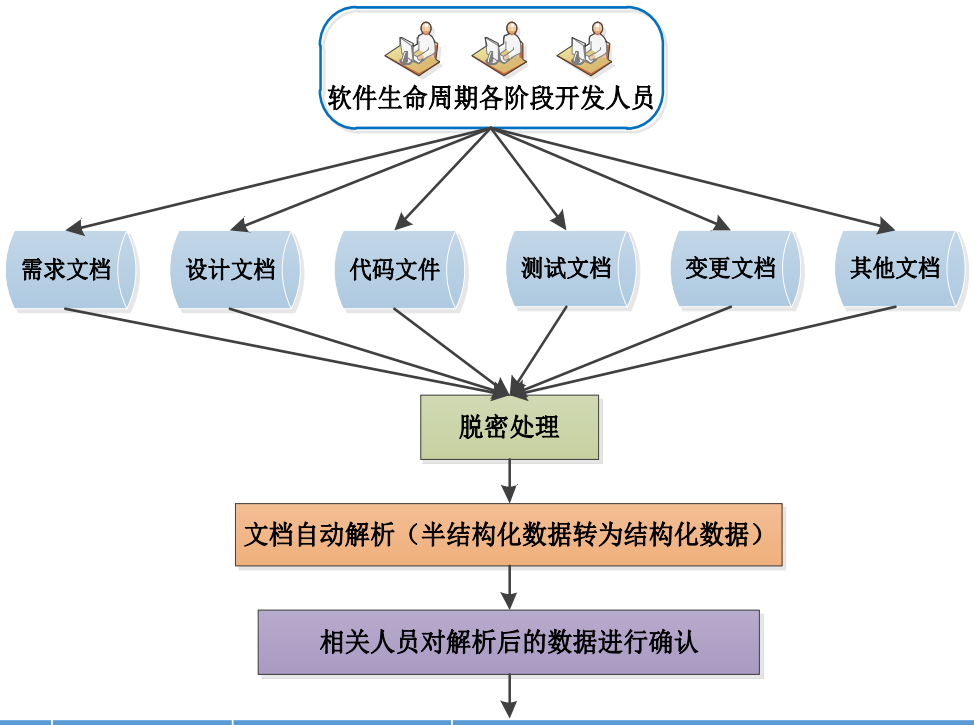
表 5　常用脱密方法

脱密方法	详情
彻底删除法	对工艺技术和关键数据，彻底删除
模糊处理法	对特定的装备，可以省去装备具体名称，使用“某型”等或其他模糊名称；对某个具体数字，可以使用虚数表示
替换取代法	将保密的某个装备表示为“典型装备”或其他特定名字
数据示意法	对于具体某个装备性能的分数或参数，使用合理取值范围内其他数字表示

以上数据获取方式适用于适航领域软件文档，除此之外，本文还选择了几个在需求跟踪领域经常用到的公开数据集。在本文实验中能够使用这些数据集的主要原因包括以下三点：（1）适航领域软件文档主要用于工业部门，保密性较强，能够获取到的适航领域软件数据集数量较少，不足以支撑模型验证过程；（2）在适航领域软件需求跟踪任务中，主要关注软件文档中的系统需求、高级需求、低级需求和源代码，而在公开数据集上，软件文档也被分为几种类型，包括但不限于上述四种，因此公开数据集满足实验模型的输入要求；（3）在需求跟踪领域，已有研究者使用本文选择的公开数据集进行实验来评估提出方法的有效性，因此使用这些数据集更有利于本文设置对比实验，并且得到

的对比结果更有说服力。

公开数据集可以在 CoEST 网站上获取到，包括：CM1-NASA、GANTT、eTOUR、iTrust 和 EasyClinic，其中 CM1-NASA 是美国国家航空航天局的基础项目 CM1 的一个子集；GANTT 是一个使用甘特图来管理项目流程的软件；eTOUR 是由 Salerno 大学使用 java 语言开发研制的一个专用于旅游的电子导航软件；iTrust 是一个用于记录医药信息的 java web 系统；EasyClinic 是用在医疗管理的 java 软件项目，同样由 Salerno 大学开发。每个数据集的具体情况如表 6 所示。CoEST 网站不仅提供了测试数据，而且对每一组测试数据都给出了专业的跟踪链接。使用 CoEST 数据进行实验的优势包括：1）给验证模型的工作带来了极大的便利；2）保证了结果的准确性；3）减少了在学习排序阶段对数据的标注过程。



id	隶属项目	生命周期类别	内容
1	涡轮发动机项目	系统需求	控制系统具有0～10000米起动发动机的能力，起动由手动控制，用机械按钮或虚拟按钮
2	涡轮发动机项目	高级需求	控制系统虚拟按钮应该具备触摸反馈
3	涡轮发动机项目	低级需求	控制系统虚拟按钮触摸反馈使用某中触屏技术实现
4	涡轮发动机项目	设计阶段	确定使用接口，保证控制系统的虚拟按钮安全可靠
5	涡轮发动机项目	编码阶段	相关代码片段

图 15　数据获取过程示意图

表 6　实验数据集

数据集名称	包含的内容	词条（Token）数量
CM1-NASA	22 个高级需求、53 个低级需求、45 个跟踪链接	5767
GANTT	17 个高级需求、69 个低级需求、68 个跟踪链接	2080
eTOUR	58 个用例、116 个代码类、308 个跟踪链接	97452
iTrust	131 个用例、367 个代码类、534 个跟踪链接	123501
EasyClinic	30 个用例、20 个 UML 交互图、63 个测试用例、47 个类描述信息、1257 个跟踪链接	21882

在确定软件数据集之后，还需要生成训练词向量的语料库。Siwei L.等人[45]指出在训练词向量时，语料的领域相关性比语料库的大小更重要，并且语料的领域性越强，词向量的表示效果越好。当然，在相同的领域，语料库越大效果越好。为了保证领域相关性，本文首先下载了 GB/T11457-2006《信息技术软件工程术语》，然后去掉软件术语表中单个词的短语，接下来对每个软件术语添加上“software”，最后使用处理过的软件术语表去过滤维基百科数据集，最终得到一个包含有 43,443,648 个单词的 264M 的软件相关语料库。具体步骤如图 16 所示。

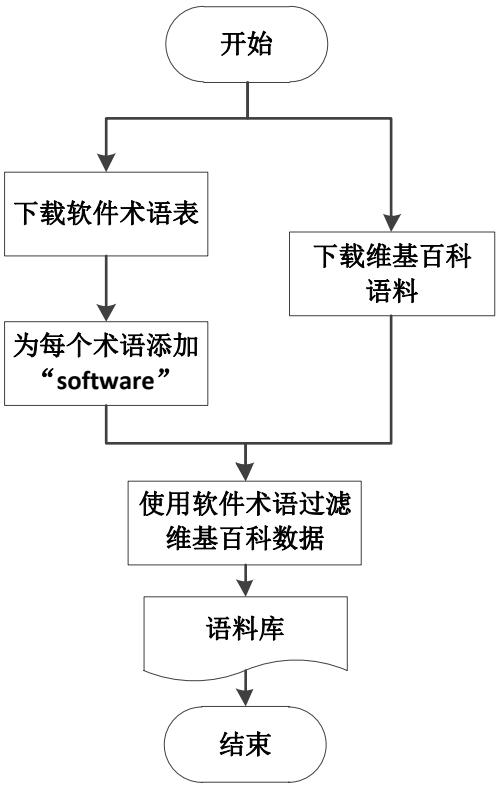


图 16　获取训练词向量语料库流程

4.2 文档预处理

实验中文档根据类型可以分为两类：（1）文本文件，包括高级需求文档、低级需求文档、用例文档、UML 交互图文档、测试用例文档、类描述文档等；（2）源代码文件。具体的文档预处理流程如图 17 所示。

第一步，设置领域词典。本文首先通过光学字符识别（Optical Character Recognition，OCR）将 pdf 版本的《汉英航空发动机工程技术词典》中的汉语和对应的英文解释提取出来，然后把汉语部分作为领域专有名词，并将汉语和英文部分存入数据库表中。

第二步，对中文软件文档进行分词和翻译。步骤如下：（1）使用结巴分词工具，并在分词时加入自定义词典，即第一步中的领域词典对文本进行分词；（2）去掉中文分词结果中的中文停用词；（3）分词得到的结果将优先使用领域词典对单词进行翻译，当领域词典中该词汇不存在时，使用有道云翻译接口进行翻译；（4）将最终的分词和翻译结果存入数据库中。

第三步，对分词并翻译之后的文本和代码文件进行以下处理：（1）以空格为分隔符对文本和代码内容进行分割；（2）去掉标点符号、数字和英文停用词。另外，在处理翻译后的文本时，使用的是 nltk[46]，并在 nltk 提供的英文停用词列表中加入了一些数据集中无用的词；在处理代码文件时，我们将 java 和 C 语言中的关键字加入了停用词列表。

第四步，根据驼峰命名法的规则对复合名词进行分解。需要注意的是，这里只对不在软件术语表中出现的复合名词进行分解，即只分解由开发者命名的复合名词进行。举例说明，单词“DefaultMutableTreeNode”将被分解成四部分：“Default”、“Mutable”、“Tree”和“Node”。

第五步，将所有单词都转换成小写形式。需要注意的是，由于本文中使用 word embedding，在训练使用的语料库中已经包含了单词的各种词形，因此提出的方法中并不需要对单词进行词干提取和词形还原。

最后，将以上的结果保存到数据库中。

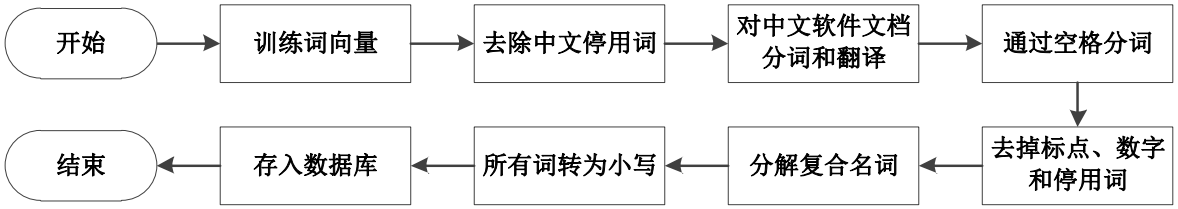


图 17　文档预处理流程

4.3 实验设置

正如第三章提到的，Tr-WELR 模型中计算部分包括使用 WQI 方法计算文本语义相似度阶段和学习排序阶段。本文设置了三组实验，如表 7 所示，第一组实验是使用 LSI[9] 方法和公式(3.7)所示的文献[16]中计算文本相似度的方法 W2V，分别与 WQI 方法做对比，用来检验提出的改进的文本语义相似度算法对结果的提升效果；第二组实验是使用改进的学习排序模型分别与未使用学习排序模型前的结果对比、未改进的 IR SVM 算法对比，用来验证学习排序算法在特定领域软件需求跟踪任务上的有效性和改进后算法性能的提升效果；第三部分实验是对比本文提出的 Tr-WELR 模型和当前国际领先水平的方法 ENRL[17]，用来验证和评估本文提出的 Tr-WELR 模型的有效性和性能。下面将对以上提到的方法进行介绍。

表 7 对比方法

实验	对比方法
实验一	LSI、W2V
实验二	未使用学习排序算法前的结果、IR SVM
实验三	ENRL

在本文中，LSI 方法指的是使用潜在语义索引方法恢复软件跟踪链接的方法，该方法已经被证明是一个在跟踪恢复领域成熟的方法[12]。该 LSI 方法步骤分为六步[9]，分别是：（I）定义一个潜在的跟踪模型；（II）在数据集上使用跟踪模型自动识别出文档概念；（III）预处理文档；（IV）重建跟踪链接；（V）选择相关的跟踪链接；（VI）可视化跟踪链接。使用 LSI 作为对比方法是为了说明使用文本语义相似度计算比使用基于统计的检索方式效果更好。

W2V 方法是使用论文[16]中的方法来解决需求跟踪任务，算法核心在于对文本语义相似度的计算，如公式（3.7）所示。该对比方法的设置是为了从实验的角度说明改进的文本相似度算法的有效性和合理性。

改进的 IR SVM 算法首先按照 Ranking SVM 的方法把所有查询结果映射到文档对的空间中，将每一个文档对看作一个实例，使用 SVM 的方法把这些实例分成两类：+1 和 -1，其中+1 表示文档对中前一个文档在结果列表中的顺序比后一个文档靠前，-1 相反。在应用 SVM 对实例进行分类的过程中，使用了公式（3.16）所示的最优化公式，并且运用 SMO 算法来解决该最优化问题。

ENRL（Estimation of the Number of Remaining Links）方法是论文[17]中为了预测需求跟踪链接数量提出的模型。该模型结合了 NLP 方法和机器学习算法，并且为了探寻预测模型的精度，尝试了 12 种 NLP 方法和 4 个机器学习算法的组合，最终给出了每个数据集在所有组合上的结果。该论文中使用了三组数据，其中两组是本文同样使用到的 eTOUR 和 EasyClinic，因此本文使用论文[17]中这两组数据最好的结果和 Tr-WELR 模型的结果对比。该实验的设置是为了验证提出的 Tr-WELR 模型的在软件需求跟踪任务上具有更好的效果。

本文使用主题模型工具 gensim[47]在生成的语料库上训练词向量，同时选择 word2vec 训练模型中的 CBOW 模型，设置窗口大小为 5，词向量的维度为 200。

本论文中所有的实验都在相同的实验环境下进行，实验所用的机器为八核英特尔 i7 处理器，运行内存为 8G。

4.4 模型评估指标

为了评估模型的性能，本文使用精确率、召回率、F 测度、MAP（Mean Average Precision）和 MRR（Mean Reciprocal Rank）为评估指标。在描述评估指标之前，记：

• N_{tp}为检索列表中与查询项有关联的文档数，即可以与查询的需求建立跟踪关系的文档的数量。

• N_{fp}为检索列表中与查询项无关联的文档数，即与查询的需求无法建立跟踪关系，却错误地与之建立跟踪关系的文档数。

• N_{tn}为数据集中与查询项没有关联的文档，同时也没有在检索列表中出现的文档数。

• N_{fn}为数据集中与查询项有关联的文档，但是不在检索列表中的文档数。

下面给出在软件需求跟踪领域中，常用的模型性能评估指标定义。

精确率（Precision）检索结果列表中与查询项有关联的文档占检索列表文档总数的比。

$$PRE = \frac{N_{tp}}{N_{tp}+N_{fp}} \tag{4.1}$$

召回率（Recall）检索结果列表中与查询项有关联的文档数占数据集中所有与查询项有关联的文档数的比。

$$REC = \frac{N_{tp}}{N_{tp}+N_{fn}} \tag{4.2}$$

F 测度（F-measure）好的模型应该同时有好的精确率和召回率，但是精确率和召回率有时候是矛盾的，即当精确率较高时，召回率低；当召回率较高时，精确率低。F 测度用于计算精确率（Precesion）与召回率（Recall）的调和平均值，当 F 测度值较高时精确率和召回率的结果都相对较高。

$$F = \frac{2\times PRE\times REC}{PRE+REC} \tag{4.3}$$

平均准确率（Average Precision）平均准确率是对单个查询而言，即针对单个查询的结果集的准确率的平均值。

$$AP = \frac{1}{m}\sum_{i=1}^{m}\frac{i}{rank_i} \tag{4.4}$$

MAP 该值为数据集合的平均准确率，是对所有独立查询的平均准确率（Average Precision）取平均值。举例说明：软件数据集中共有两个需求，req1 和 req2，两个需求对应的结果列表分别有 4 个和 5 个文档。与需求 req1 相关的文档在结果列表中的排名为 1、2、4、7，与需求 req2 相关的文档在结果列表中的排名为 1、3、5，则对于需求 req1 的平均准确率为（1/1 + 2/2 + 3/4 + 4/7）/ 4 = 0.83，对于需求 req2 的平均准确率为（1/1 + 2/3 + 3/5 + 0 + 0）/ 5 = 0.45，那么该软件数据集的 MAP = （0.83 + 0.45）/ 2 = 0.64。

$$MAP = \frac{1}{n}\sum_{i=1}^{n}AP_i \tag{4.5}$$

MRR 与 MAP 所关心所有正确的文档位置不同，MRR 仅关心第一个正确的文档的位置。对每个查询对应的候选文档列表，找到每一个集合中第一个正确的文档的位置，然后取倒数，再对所有集合中的倒数取平均值，如公式(4.6)所示。

$$MRR = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{rank_i} \tag{4.6}$$

其中$rank_i$为集合中第一个正确的文档的位置。举例说明：对于软件数据集共有两个需求，分别是需求 req1 和需求 req2，需求 req1 建立跟踪关系的候选文档集合列表为 doc1、doc2、doc3，其中 doc2 是与需求 req1 有跟踪关系的文档，需求 req2 对应的候选文档集合列表为 doc4、doc5、doc6、doc7，其中 doc6 是与需求 req2 有跟踪关系的文档，则 MRR = （1/2 + 1/3）/ 2 = 0.53。

相对误差（Relative Error，RE）相对误差是常用的用来评价预测模型准确率的指标之一[49]，在本文中该值可以表示为公式（4.7）所示。

$$RE = \frac{|(N_{tp}+N_{fn})-(N_{tp}+N_{fp})|}{N_{tp}+N_{fn}} = \frac{|N_{fn}-N_{fp}|}{N_{tp}+N_{fn}} \tag{4.7}$$

平均相对误差（Mean Relative Error，MRE）平均相对误差同样用来评价预测模型的准确率，从全局的角度考虑方法的相对误差，计算方式如公式（4.8）所示。

$$MRE = \frac{1}{n}\sum_{1}^{n} RE_i \tag{4.8}$$

4.5 实验结果与分析

这一节将详细描述三组实验的实验结果，根据结果对提出的适航领域软件需求跟踪算法模型的有效性和性能进行分析，并对一些性能提升点和异常情况进行详细的解释说明。

4.5.1 第一组实验：使用 WQI 算法

本组实验分别使用 LSI 方法、W2V 方法和 WQI 算法对七组数据进行跟踪链接的恢复，并使用精确率、召回率和 F 测度进行模型评估，实验结果如表 8 所示。表 8 中 PRE 和 REC 分别表示精确率和召回率。HL、LL、UC、CC、ID、TC 分别表示高级需求文档、低级需求文档、用例文档、类文件、UML 交互图文档和测试用例文档。

在结果中可以看到，后两种使用词向量的方法 W2V 和 WQI 平均来说比 LSI 的效果更好。其中，WQI 算法和 LSI 方法相比，WQI 算法比 LSI 在精确率上相对提升了 33.3%，在召回率上相对提升了 24.5%；WQI 算法和 W2V 方法相比，WQI 算法比 W2V 方法在精确率上相对提升了 6.6%，在召回率上相对提升了 17.2%。

表 8　三种方法的实验结果

数据集		LSI		W2V		WQI	
		PRE	REC	PRE	REC	PRE	REC
CM1-NASA	HL→LL	0.127	0.41	0.262	0.217	0.371	0.329
GANTT	HL→LL	0.286	0.332	0.278	0.418	0.255	0.563
eTOUR	UC→CC	0.077	0.221	0.098	0.332	0.088	0.415
iTrust	UC→CC	0.009	0.45	0.192	0.363	0.198	0.322
EasyClinic	UC→ID	0.259	0.833	0.338	0.75	0.342	0.806
	UC→TC	0.45	0.755	0.522	0.867	0.499	0.867
	UC→CC	0.317	0.503	0.215	0.677	0.232	0.76

图 18 展示了三种方法在七组数据上 F 测度的平均值，显示了 WQI 算法比 LSI 方法和 W2V 方法在大多数数据集上效果都更好。

从表 8 中可以看出，WQI 算法在七组不同类型的跟踪链接上，相比其他方法都有普遍的性能提升。跟踪链接能够根据软件文档的类型分为两类：一类是文本对文本（text to text，t2t），另一类文本对源代码（text to code，t2c）。表 8 中 HL→LL、UC→ID 和 UC→TC

都属于 t2t，UC→CC 属于 t2c。同样，在图 19 中也能看出，WQI 算法对两类跟踪链接的 F 测度都有提升。对于以上性能的提升，可以做如下解释：对于 t2t 类型的跟踪链接的性能提升，表示本文提出的改进的文本语义相似度算法 WQI 在软件需求跟踪任务是可行的，且效果不错；对于 t2c 类型的跟踪链接的性能提升，表示在实验之前对代码文件的预处理操作是合理的，且通过词向量表示时能够相对准确的表示其在代码中的实际含义。

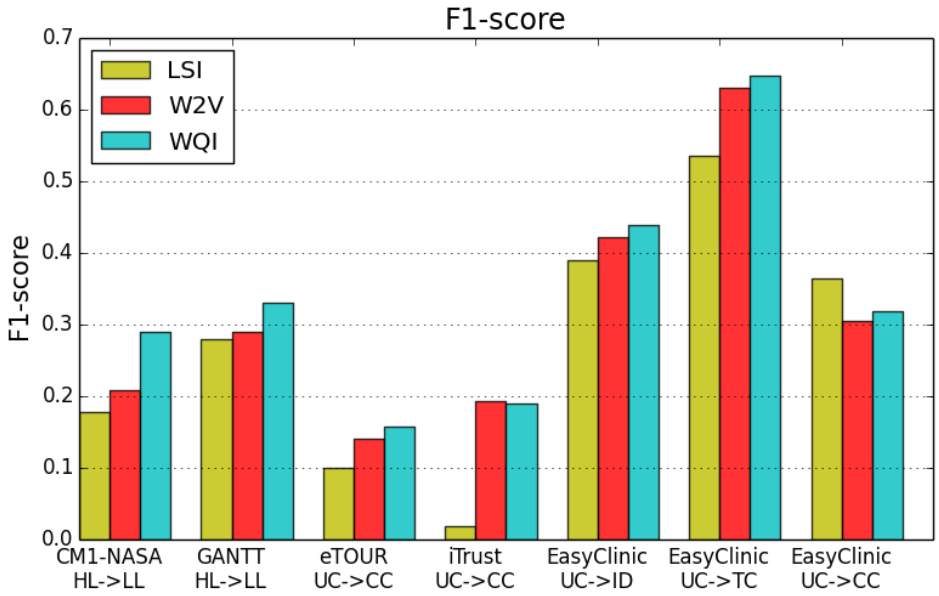


图 18　LSI、W2V 和 WQI 三种方法在七组数据上的 F 测度

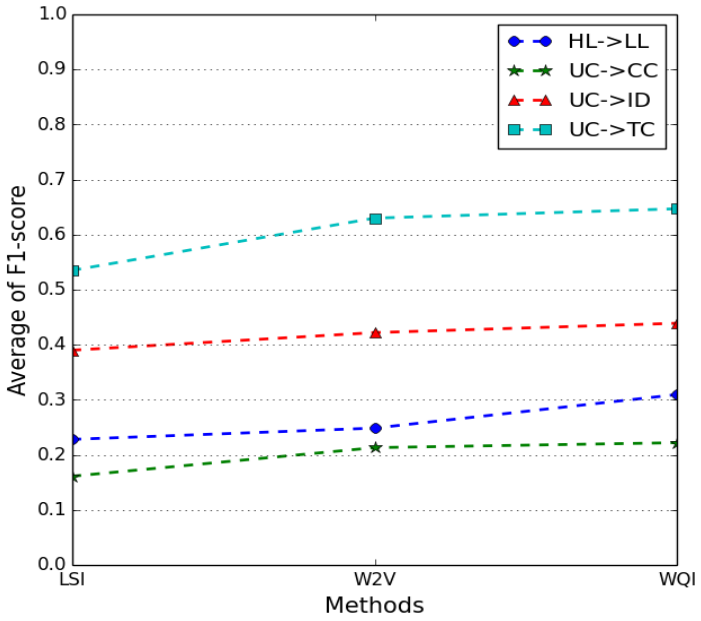


图 19　LSI、W2V 和 WQI 三种方法在不同类型的跟踪链接上的平均 F 测度

相比 W2V 方法，WQI 算法引入了 TFIDF 加权策略和查询扩展技术，并且改进了计

算的过程。从结果上来看，WQI 算法所做的改进提升了结果的精确率和召回率，即提升了需求跟踪任务的性能。查询扩展技术使用和查询项相关或相似的单词或短语对查询项进行扩展，这也就在语义空间上丰富了查询项的含义[48]。从另一个角度来说，因为每一个词向量包含词的频率特征和上下文的信息，因此一组相似的词必定比一个特定的词包含的信息多。另外，权重策略 TFIDF 不仅仅能够减少句子中的一些噪声信息，也通过抽取句子中的关键词减少了待扩展的词，既提升了精度又提高了算法执行效率。

综上所述，相比 LSI 方法和 W2V 方法，文本语义相似度计算算法 WQI 能够在软件需求跟踪任务中得到更好的效果，即生成的排序列表的质量更高。

4.5.2 第二组实验：使用改进的学习排序算法

学习排序算法将利用查询和排序列表的一些特征去学习、预测新的需求跟踪链接。本文使用学习排序算法去提升需求跟踪任务的精确度。为了验证学习排序算法是否可以让软件需求跟踪任务取得更好的效果，本节做实验去对比使用学习排序算法前后对结果的影响；同时为了验证改进的学习排序算法是否能够提升软件需求跟踪任务的性能，本节还将与未改进的 IR SVM 做对比试验。两个实验均使用 MRR 和 MAP 作为模型评估指标。学习排序算法使用之前介绍的五个特征去生成排序模型，同时做了十折的交叉验证，并且最终取这些验证的平均值作为最终结果。

图 20 展示了执行学习排序算法前后的 MRR 和 MAP 指标的值。和执行学习排序算法之前的结果相比，执行算法之后 MRR 和 MAP 的平均值都有所提升，具体来说，MRR 的平均值提升了 7.48%，MAP 的平均值提升了 22.05%。MAP 指标的显著提升，表明 IR SVM 算法能够提升候选结果列表的整体排序水平和整体结果的精度；MRR 指标的提升，说明在候选结果列表中首次出现正确文档的位置更靠前。从另一个角度来说，使用学习排序算法后，模型能够提供更准确的跟踪链接。

图 21 展示了改进的 IR SVM 学习排序模型和 IR SVM 模型的对比结果，同样使用 MRR 和 MAP 两个指标做对比。从图中可以看出改进之后的 IR SVM 在 MRR 值和 MAP 值上都有提升，具体来说，MRR 平均相对提升了 5.38%，MAP 平均相对提升了 9.80%。改进之后的 IR SVM 算法在实验所用的所有的数据集上的效果都有提升，从一定程度上说明改进的权重相比之前的权重机制有更好的效果，也说明了提出的观点——查询项与查询结果的语义相似度既包含了相对顺序关系也包含了决定相对顺序的可量化的测度——是比单独的使用相对顺序关系更有效的。

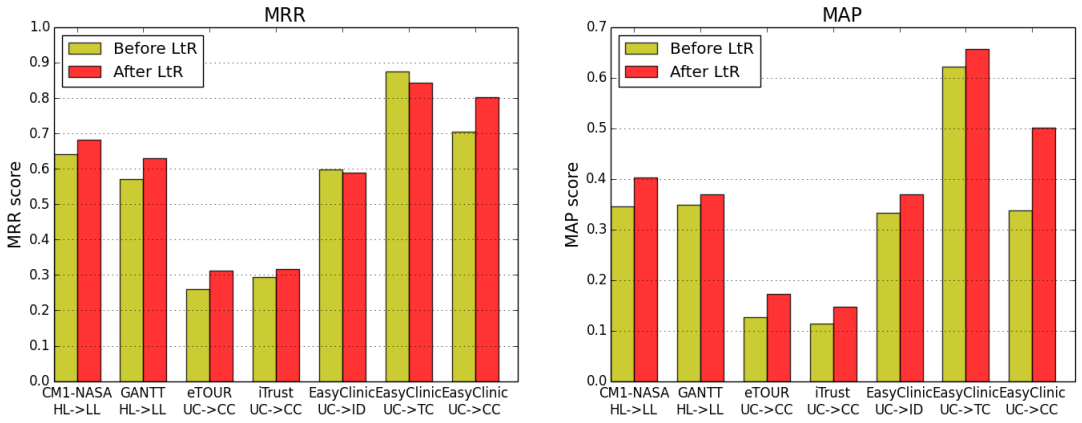


图 20　执行改进的学习排序算法前后的 MRR 和 MAP 比较

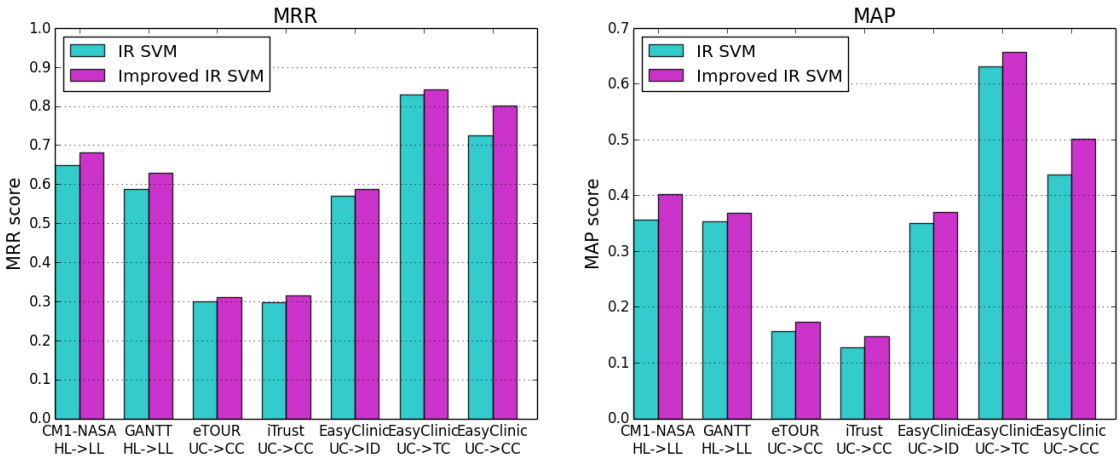


图 21　IR SVM 与改进后的 IR SVM 对比

4.5.3　第三组实验：Tr-WELR 模型对比 ENRL 方法

本组实验采用和 ENRL 中相同的数据集 eTOUR 和 EasyClinic，并且选择 MRE 作为评估指标。ENRL 中尝试了 12 种不同的 NLP 方法和 4 种机器学习算法的所有组合，其中针对每个数据集的最优组合如表 9 所示。表中 stop-stem 表示去除停用词和对单词抽取词干，TF 表示词频统计，LSA100 表示使用潜在语义分析方法并设置概念数为 100，VSM 表示使用向量空间模型，cosine 表示使用 cosine 函数计算相似度值，IBk 表示使用 k 近邻方法分类实例。

表 9　ENRL 实验在 eTOUR 和 EasyClinic 上的最优组合

数据集	NLP 方法	机器学习分类算法
eTOUR	stop-stem, TF, LSA100, cosine	IBk
EasyClinic	stop-stem, TF, VSM, cosine	IBk

使用 Tr-WELR 模型对上述两个数据集计算，使用 MRE 作为评估指标，得到的结果与 ENRL 方法的对比如表 10 所示。从对比结果可以看出，在两个数据集上 Tr-WELR 模

型计算得到的平均相对误差更小。

表 10　Tr-WELR 模型在 eTOUR 和 EasyClinic 上 MRE 值

方法	数据集	MRE 值
ENRL	eTOUR	0.0300
	EasyClinic	0.0100
Tr-WELR	eTOUR	**0.0208**
	EasyClinic	**0.0059**

综合前两组实验的分析，可以说明如下三点：

（1）Tr-WELR 模型使用特殊训练的 word embedding 和改进的文本语义相似度算法，得到了比使用 LSI 和 VSM 等基于统计的信息检索方法更精确的文本相关度，同时在需求跟踪任务中，得到了准确率更高的需求跟踪链接。较好的结果说明在任务中使用的对文本的预处理和对参数的设置能够对实验的结果产生积极的影响，并且提出的文本语义相似度算法比其他的文本相似度算法效果更好，在需求跟踪任务中也是可行的。

（2）Tr-WELR 模型使用基于文本语义相似度的权重表示方法改进了 IR SVM 方法中对不同等级（位置）的文档的惩罚因子，对比改进前后，该方法取得了较好的结果，说明改进的部分是有效的。

（3）对比国际领先水平的方法 ENRL 中对 eTOUR 和 EasyClinic 两个数据集的最优结果，Tr-WELR 能够获得更小的平均相对误差。因此说明 Tr-WELR 在软件需求跟踪任务上具有更好的性能。

4.6　运行时间效率

表 11 显示了 Tr-WELR 模型中在不同数据集上计算部分的运行时间，其中计算部分包括文本语义相似度计算阶段和执行学习排序算法阶段。表中 Mean R.T.(sec)表示在特定数据集上恢复一组跟踪链接需要的平均运行时间，Retrieval Nums 指的是在每个数据集上对任意一个查询项需要检索的文档数，Mean Documents' Length 指的是每个数据集的平均文档长度。根据结果我们可以推断出平均运行时间和检索数目、平均文档长度都是正相关的，即需要检索的文档数越多，平均文档长度越长，需要的计算时间越长。

虽然本文实验中在检索数量最大的 iTrust 数据集上平均需要 79.25 秒才能建立全部的用例到代码类的跟踪关系，但是相比于人工建立需求跟踪关系，这个时间是完全可以

接受的。

表 11 Tr-WELR 模型计算部分的运行时间和相关信息

数据集		Mean R.T.(sec)	Retrieval Nums	Mean Documents' Length
CM1-NASA	HL→LL	2.44	53	339
GANTT	HL→LL	2.05	69	107
eTOUR	UC→CC	49.42	116	1766
iTrust	UC→CC	79.25	367	2867
EasyClinic	UC->ID	18.09	20	604
	UC->TC	27.00	63	604
	UC->CC	29.78	30	604

4.7 本章小结

本章是本文的实验部分，目的是对本文所提出的软件需求跟踪模型 Tr-WELR 进行验证。首先介绍了数据准备流程和实验所用到的数据集的特征和来源，然后根据文本种类的不同分情况介绍了文档预处理过程，接下来对实验中的对比方法、训练词向量的语料库和实验环境做了介绍，并给出了模型的评估标准，最后对实验的结果和现象进行分析，给出结论。实验证明，在多组数据集和多个不同类型的跟踪链接上，本文提出的 WELR 模型能够较好的完成恢复跟踪链接的任务，主要原因有两方面：1）在软件文档上使用提出的基于 word embedding 的文本语义相似度算法准确率有所提高；2）使用学习排序算法使得跟踪结果的精度提高。

第五章　适航领域软件需求跟踪原型系统

本文在之前第三章详细描述了 Tr-WELR 模型的构建过程和算法实现，在第四章通过对比实验验证了模型的有效性和性能。本章在 Tr-WELR 算法模型的基础上，设计并实现了适航领域软件需求跟踪原型系统，系统中主要包括：中文文档录入、存储、中文分词及翻译，数据预处理及存储，文档相似度关系计算，适航领域需求层级跟踪关系建立，适航审定目标满足性检查及检查单导出等。本章将从系统需求分析、系统总体设计、各模块设计与实现和开发环境等四个方面详细介绍系统实现并对 Tr-WELR 模型的结果进行展示。

5.1　系统需求分析

适航领域软件需求跟踪原型系统是一个给适航审定人员提供软件需求跟踪完整路径和给出是否满足适航标准中规定的可追溯性目标结果的辅助系统。该系统通过对项目中软件各个生命周期数据的收集，使用提出的 Tr-WELR 算法模型对数据进行处理，自动判定项目中各个需求是否满足可追溯性目标。系统主要功能包括：文档录入及存储、数据预处理、文档相似度计算、需求层级关系建立及适航目标满足性判定。

（1） 文档录入及存储

文档录入包括两种录入方式：系统录入方式和 word 文件导入方式。系统录入以项目数据、生命周期类型、软件制品数据三个等级的形式录入；word 文件导入是将包含有软件生命周期数据的 word 文本导入，系统自动对 word 内容进行抽取，然后进行人工信息确认。录入和导入的数据都将存储到数据库中。同时，由于 Tr-WELR 模型的要求，需要将中文文档进行分词和翻译。用例图如图 22 所示。

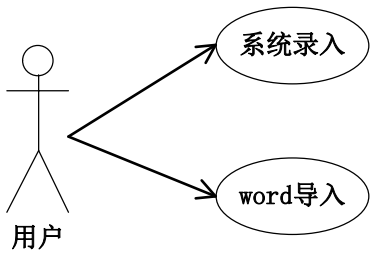


图 22　文档录入用例图

（1） 数据预处理

在使用 Tr-WELR 算法模型进行计算前，需要对数据进行预处理操作，包括去停用

词、计算每个项目中所有数据的 TFIDF 值、抽取关键词以及扩展词。由于当项目中数据发生变更时，TFIDF 的值会随之改变，因此需要数据预处理能够在录入数据、导入数据或修改数据时自动重新执行。

（2） 文本相似度计算

文本相似度计算是算法模型的核心之一。选取同一项目中的两个文本，应当可以单独计算二者相似度，并提供二者是否具有跟踪关系的判定，给适航审定人员提供数据参考。

（3） 需求层级关系建立及适航目标满足性判定

针对适航标准 DO-178B/C 中提到的三种可追溯性目标，包括高级需求可追溯到系统需求、低级需求可追溯到高级需求和源代码可追溯到低级需求，能够给出每个项目中三种跟踪关系的跟踪关系图和跟踪矩阵。同时，需要能够根据跟踪关系的情况，自动判定项目中各个目标的满足情况，并且可以将跟踪关系结果导出为适航审定检查单。用例图如图 23 所示。

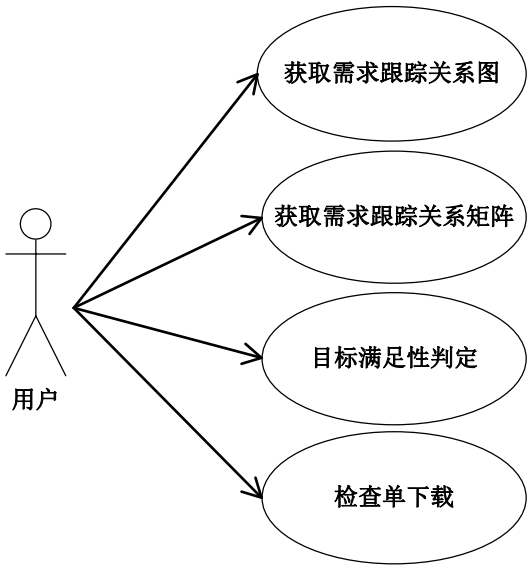


图 23　需求层级关系建立用例图

5.2 系统总体设计

本节介绍适航领域软件需求跟踪原型系统的总体设计，包括系统架构的设计和系统数据库的设计。

5.2.1 系统架构设计

根据对适航领域软件需求跟踪原型算法系统的需求分析，将系统总体架构分为三层：

数据交互层、算法分析层和数据存储层。整体架构如图 24 所示，系统最底层为数据存储层，该层包括本地数据库和词向量数据库，其中本地数据库用来存储与用户、软件项目相关的数据，词向量数据库用来存储事先通过训练维基百科数据得到的词向量；计算分析层是系统的核心，通过运行算法模型来对数据进行计算，并连接数据存储层和数据交互层；数据交互层与用户进行交互，包括数据准备、数据录入、数据格式化和数据展示等。数据交互层使用前端语言 php 实现，计算分析层通过 python 实现，二者交互通过 php 调用 python 的脚本接口实现。

计算分析层对应 Tr-WELR 算法模型，主要包括数据预处理和算法计算两个部分，数据预处理首先对录入的中文软件文档数据进行分词和翻译，然后对生成的项目软件数据集计算 TFIDF 作为每个词的权重，然后通过查询扩展对部分词进行扩展；算法计算部分利用 Tr-WELR 模型中改进的文本语义相似度算法进行计算，然后按照相应策略生成相关文本排序列表，利用 Learning to Rank 对排序列表进行训练，最终生成跟踪关系。将得到的跟踪关系通过 python 接口传入数据交互层，经过数据格式化后对结果进行展示。

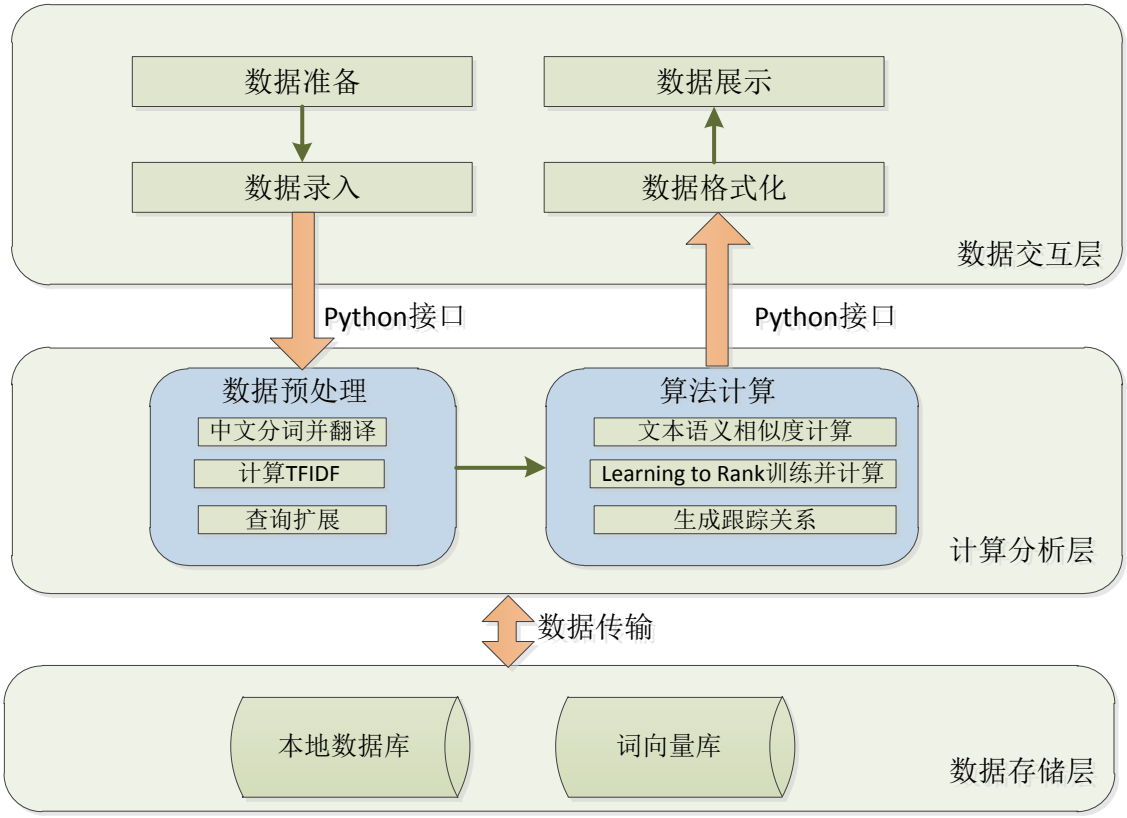


图 24　系统架构图

5.2.2　系统数据库设计

系统数据库中与需求跟踪关系功能相关的数据表可以分为三类，包括预置表、软件

制品相关表和实验中间表。其中预置表包括词向量表和自定义词典表，词向量表中主要存储训练维基百科数据得到的词向量，方便检索；自定义词典表为事先读取领域词典制作的数据表，用来辅助翻译过程。软件制品相关表包括软件项目数据、软件生命周期分类数据和软件制品数据，用来存储用户通过系统录入的软件数据。实验中间表包括存储软件文档分词并翻译后的存储表、计算 TFIDF 后的存储表、执行查询扩展后扩展项的存储表，以及计算结果的临时存储表。上述主要数据表如图 25 所示。

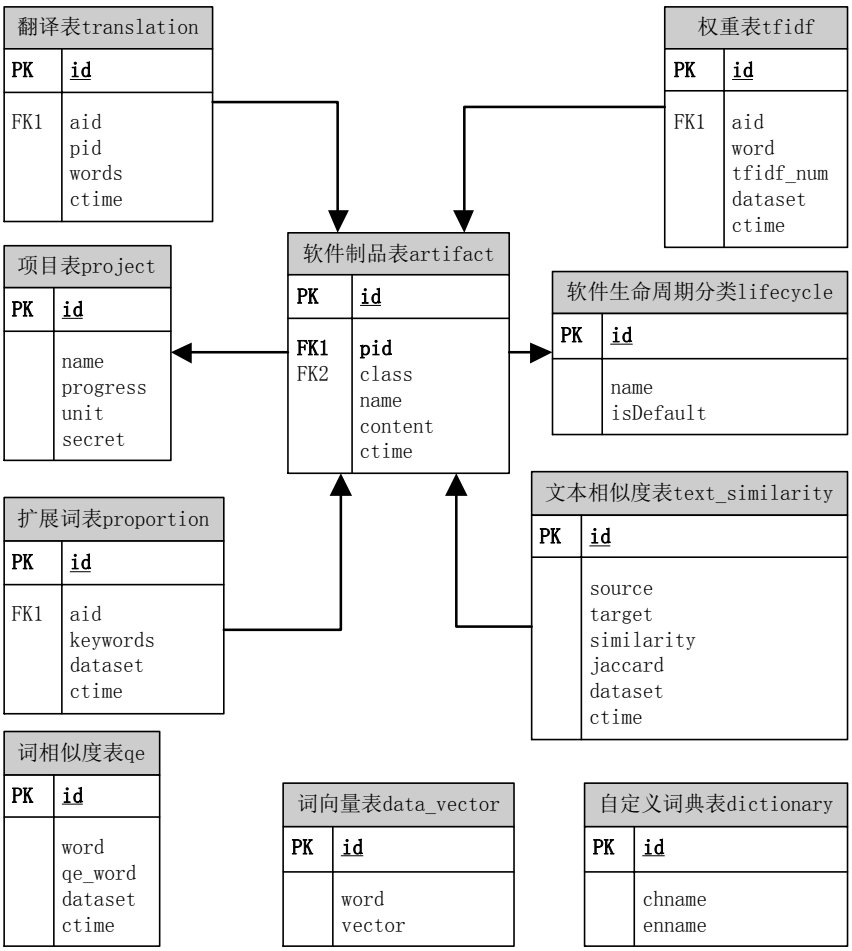


图 25　主要数据表

其中，预置表包括 data_vector、dictionary；软件制品相关表包括 project、lifecycle 和 artifact；实验中间表包括 translation、tfidf、proportion、qe 和 text_similarity。各个表的详细介绍如下：

（1）data_vector：词向量表，用来存储事先训练好的词向量，每个单词对应一个二百维的词向量；

（2）dictionary：自定义用户词典表，用来存储事先在发动机词典上通过 ocr 获取的领域词典，其中 chname 是中文名称，enname 是对应的英文名称；

（3）project：项目表，用来存储软件项目信息，包括项目名称、项目进度、项目单位和项目的秘密等级；

（4）lifecycle：软件生命周期分类数据表，用来存储用户定义的各个软件生命周期，其中由系统定义的软件生命周期包括：系统需求、高级需求、低级需求和源代码，其通过表中的 isDefault 字段实现用户定义和系统定义的区分；

（5）artifact：软件制品表，用来存储软件制品，同时指定该软件制品属于哪个项目的哪个软件生命周期；

（6）translation：翻译表，用来存储中文软件文本经过中文分词和翻译之后内容，用作实验的初始数据；

（7）tfidf：权重表，用来存储文本中各个词的权重值，该值通过计算 TFIDF 得到；

（8）proportion：扩展词表，用来存储每个软件制品中应该被扩展的词的扩展词集合；

（9）qe：词相似度表，用来存储扩展的词和被扩展词之间的词语之间的相似度，该相似度通过加载词向量模型来计算得到；

（10）text_ similarity：文本相似度表，用来临时存储计算过程中的文本相似度，表中字段 source 表示源文本，target 表示目的文本，similarity 表示文本语义相似度，jaccard 表示广义 Jaccard 相似度。

5.3 模块设计与实现

系统模块图如图 26 所示。

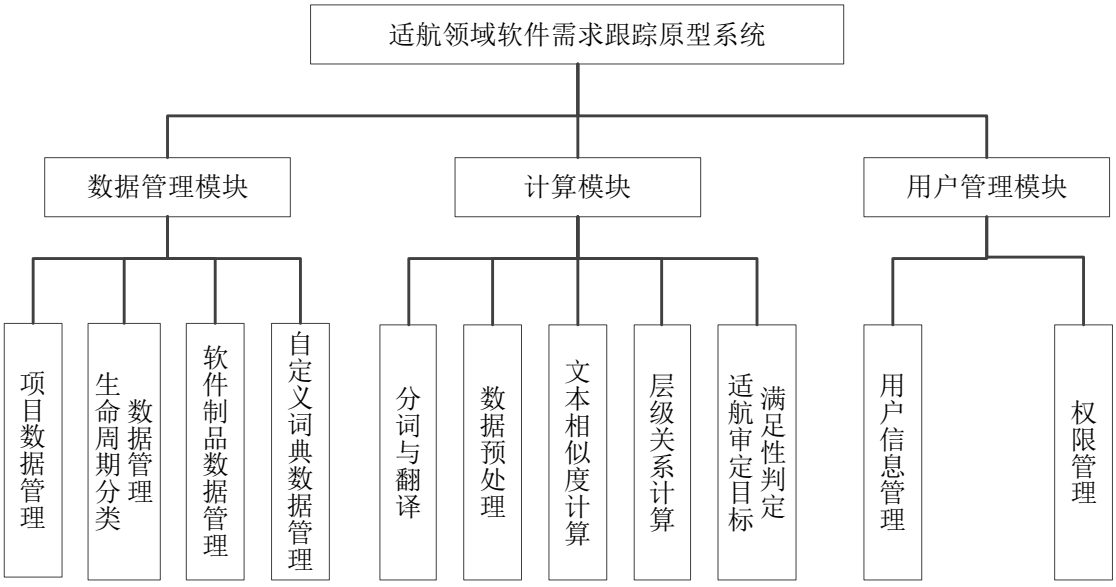


图 26　系统模块图

5.3.1 数据管理模块设计与实现

数据管理模块包括项目数据管理、生命周期分类数据管理、软件制品数据管理和自定义词典数据管理等四个部分，每个部分都包括对数据的“增删改查”四个基本功能。

软件制品数据管理子模块中数据的录入包括用户在界面直接录入和 word 导入两种形式。用户通过 word 导入包括两个步骤：1）后台接收到用户上传的 word 文件后，使用 python 的 docx 类库对文件中的样式进行解析，找到各级标题中的“系统需求”、“高级需求”、“低级需求”和“源代码”，然后将相应标题下的子标题和正文内容抽取出来，将子标题看作相应需求或源代码的具体名称，正文内容看作该分类下的详细描述，然后按照出现的前后顺序呈现在界面上；2）录入需求的用户或者具有专业知识背景的用户对抽取的内容进行确认、修改或删除，全部确认完毕则完成此次上传。对数据库的操作使用 ThinkPHP 提供的与 mysql 交互的类库。通过 word 录入数据的界面如图 27 所示。

自定义词典数据管理是管理自定义的词典数据，该词典数据用于中文文档分词和术语翻译。得到自定义词典的步骤包括如下几步：（1）调用有道云的 ocr 接口，对 pdf 版本的《汉英航空发动机工程技术词典》进行光学字符识别；（2）自定义脚本文件对识别到的内容进行格式化处理；（3）将处理后的中英文对照内容存入到数据表 dictionary 中；（4）使用 sql 脚本过滤掉数据表 dictionary 中中英文单词或短语中出现标点符号的内容。

图 27 用户对录入的 word 确认界面

5.3.2 计算模块设计与实现

计算模块包括分词及翻译、预处理、文本相似度计算、层级关系计算和适航审定目标满足性判定等五个部分。系统后端具体的计算部分使用 python 代码实现，共包括 10 个主要代码类，类图如图 28 所示。

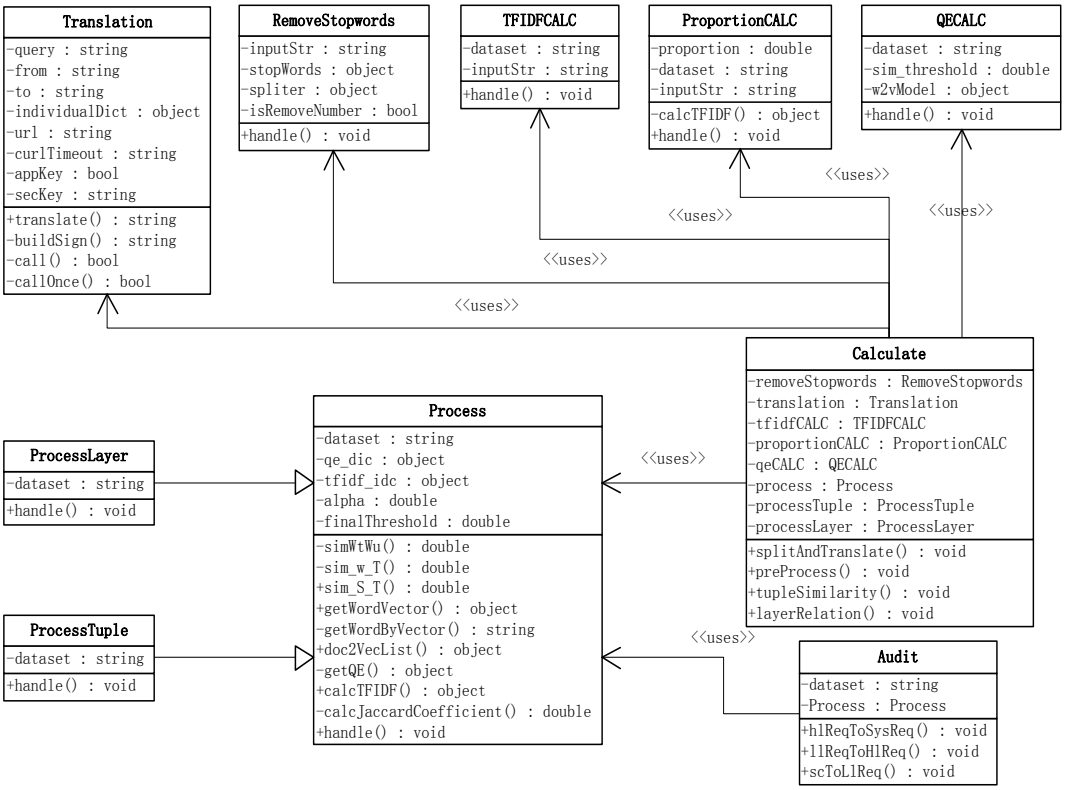


图 28　计算模块类图

Process 类实现了文本语义相似度计算算法、改进的学习排序算法，Calculate 类、Audit 类使用 Process 类实现文本相似度计算功能和适航审定功能，ProcessLayer 类和 ProcessTuple 类均继承 Process 类，分别用来完成文档层级关系计算和两两文档语义相似度计算和跟踪关系判定，而 Translation 类、RemoveStopwords 类、TFIDFCALC 类、ProportionCALC 类和 QECALC 类用于文本预处理过程，分别用于分词并翻译、去停用词、计算 tfidf 值、文本相似度计算前的权重设置和查询扩展。

可以在图 28 中看到，计算过程中将几个关键步骤独立封装成类来实现，包括去停用词、分词和翻译、计算权重、获取扩展词、查询扩展和计算文本相似度等，这样划分的好处包括：1）能够根据功能的不同灵活调用各个阶段的函数，且能够做到互相独立；2）在用户使用时，可以根据软件制品的修改情况，有选择性的调用不同阶段的函数，节省计算时间，提升用户体验。

计算过程中使用的 word embedding 预先训练完成并保存在数据库中，使用 numpy 对向量进行初始化，然后将第三章介绍的算法公式编程实现。使用 php 提供的 exec 函数调用 python 脚本，并将结果返回给用户界面。计算过程中 php 使用 ajax 异步调用 python 脚本，避免页面假死，提升用户使用体验。计算模块中涉及到的文本相似度计算、层级关系建立和适航审定目标满足性判定界面分别如图 29、图 30、图 31 所示。文本相似度计算给出了两个文档需求跟踪关系的参考结果，层级跟踪关系建立给出了跟踪图和跟踪矩阵两种形式结果，适航审定目标判定给出了统计结果和检查单下载。

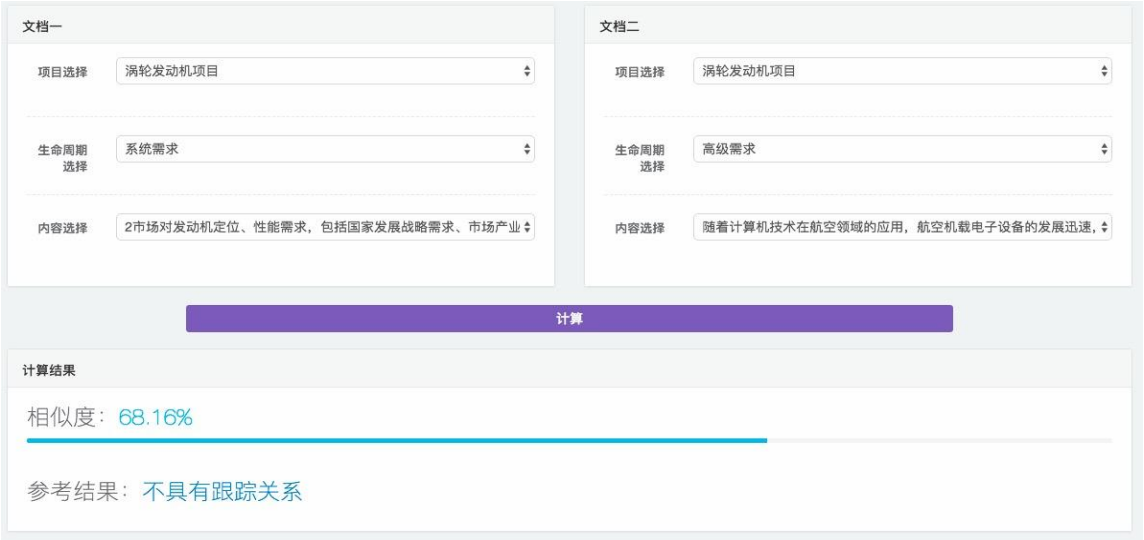


图 29　文本相似度计算界面

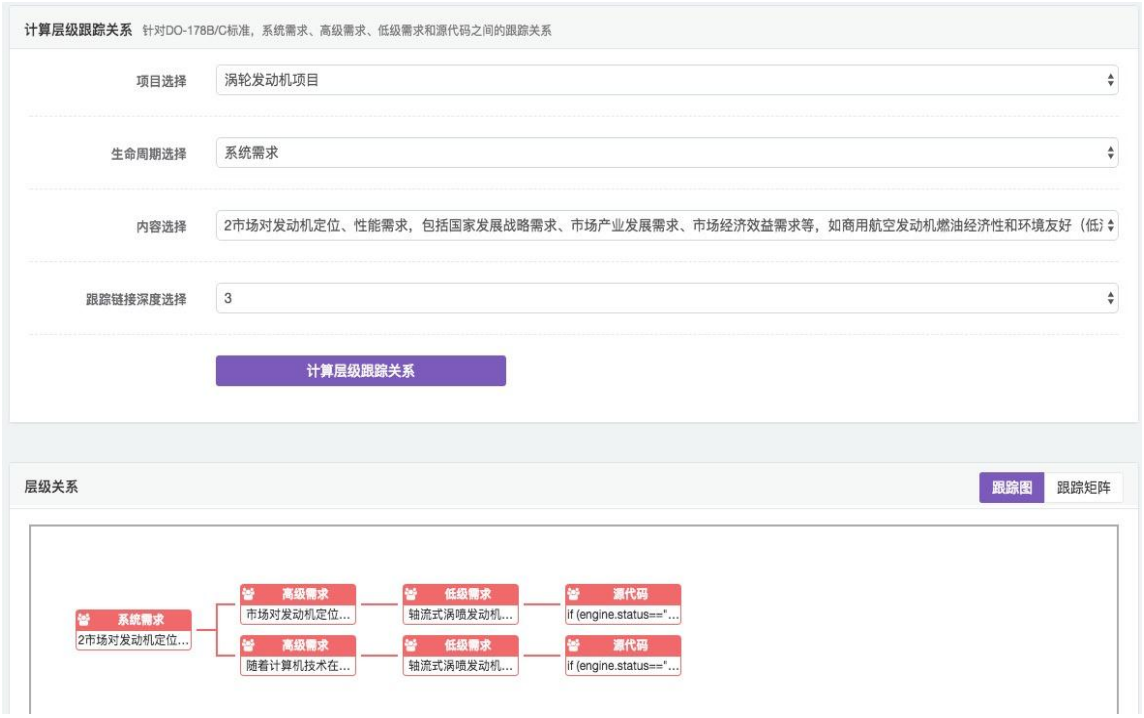


图 30　层级跟踪关系建立界面

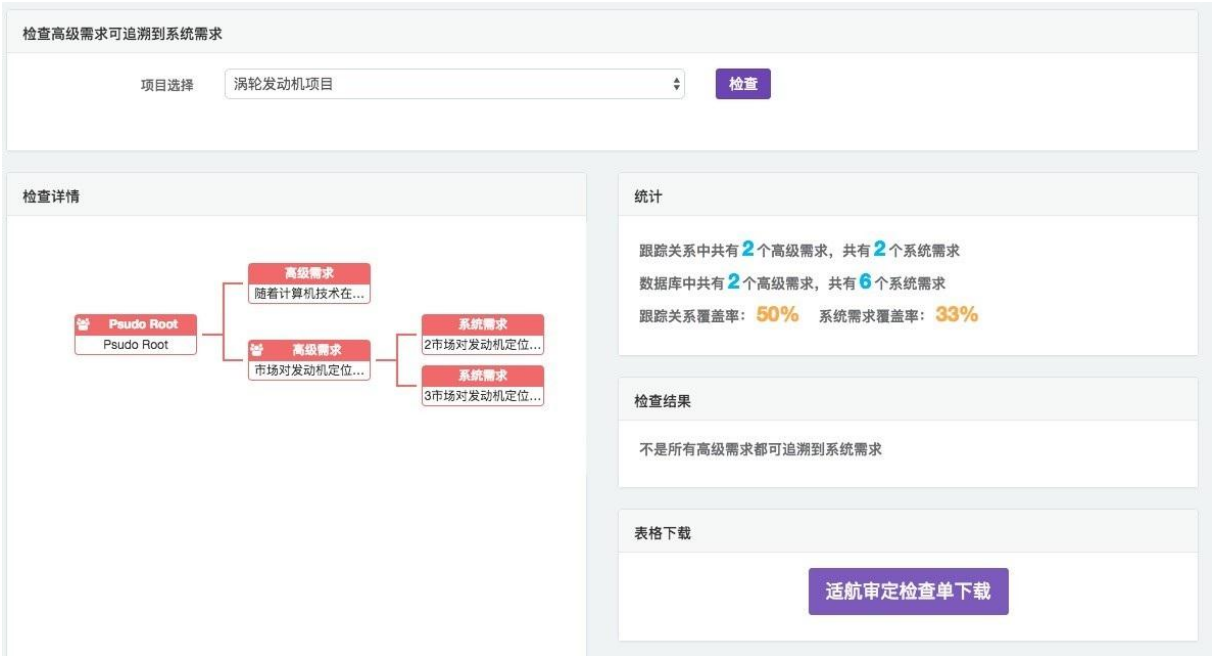


图 31　适航审定目标满足性判定界面

5.3.3　用户模块设计与实现

用户模块包括用户信息管理和权限管理两个部分。用户信息管理实现用户对个人信息的管理，权限管理是系统超级管理员或具有权限管理权限的用户对系统中的用户、角色进行权限分配和管理。

权限管理使用 ThinkPHP 框架提供的 RBAC 模块实现，将系统中的用户划分为不同的角色，系统为角色分配权限后，具有该角色的用户将得到相应的权限。如图 32（省略相关属性和函数）所示，用户和权限之间通过角色来关联，权限为 ThinkPHP 框架中的各个模块、控制器和函数的名称或函数签名。在用户登录时，将该用户具有的权限存入 session 中，在该用户使用系统期间，即 session 有效期间，用户点击某个功能时系统会根据 session 中的权限列表判断当前用户是否具有访问该功能的权限。实现界面如图 33 所示。

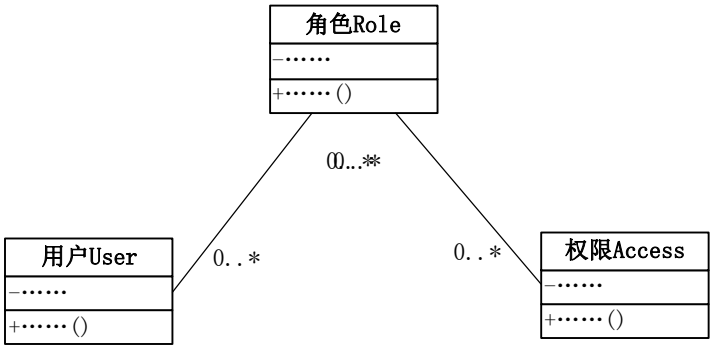


图 32　权限管理类图

图 33　权限分配界面

5.4　开发环境

系统前端与用户交互部分使用基于 php 语言的 ThinkPHP 框架，后端预处理和计算部分使用 python 语言，系统整体运行在 CentOS 系统上。具体开发工具及开发环境详细信息如表 12 所示。

表 12　开发环境详细信息

开发语言及工具	版本号	说明
Php	5.4.16	前端开发语言
ThinkPHP	3.2.3	前端开发框架
Python	2.7.5	实现后端计算部分语言
nltk	3.2.5	自然语言处理工具
gensim	3.2.0	自然语言处理工具
OrgChart	1.0.4	展示跟踪关系图插件
PHPExcel	2.1	生成检查单插件
Bootstrap	3.3.6	前端页面框架
Mysql	5.6.38	数据库
Youdao ocr 服务接口	无	生成自定义词典所用 OCR 接口
Youdao 自然语言翻译接口	无	中文短语翻译接口

5.5 本章小结

本章完整介绍了适航领域软件需求跟踪算法原型系统的需求分析、系统总体设计、模块设计与实现和开发环境。同时展示了系统主要功能的界面，包括数据管理界面、文本相似度计算界面、层级关系建立界面、审定目标满足性判定界面和权限管理界面等。

总结与展望

总结

本文以软件需求跟踪技术流程为切入点，引入 word embedding 和学习排序模型，使用信息检索的思想建立和恢复适航领域软件需求跟踪链接。本文从文本语义相似度角度出发，通过计算文本相似度，得到与需求相关文档列表，然后利用计算得到的文本语义相似度改进学习排序模型的损失函数，通过避免 Ranking SVM 中未考虑的两个问题来提升整个模型的精确度。本文完成的具体工作包括：

（1）通过研究适航标准 DO-178B/C，确定了与适航审定工作相关的软件需求跟踪方面的审定内容，制定审查单，并通过文中提到的算法动态填写其中的内容，给局方审定人员提供参考；

（2）改进已有文本语义相似度计算方法，使用加权策略和查询扩展提升语义相似度的精度，得到算法 WQI；使用信息检索思想，每个软件需求看作是一条查询，该需求的下一层所有文档作为检索库；综合以上两点，执行软件需求跟踪任务，并验证了算法的有效性。

（3）改进 IR SVM 学习排序算法，在损失函数中引入权重，并使用文本语义相似度来设置该权重，用来解决 Ranking SVM 中针对不同等级查询结果同等对待的问题。该模型最初通过已有的跟踪关系进行训练，在审定人员通过系统确定其他的跟踪关系后，可以使用新数据重新训练，提升模型的精度。实验表明，该模型能够更加有效的应用在需求跟踪任务中。

（4）设置三组对比实验，依次验证每一步算法的有效性，包括：提出的文本相似度算法 WQI 与信息检索算法 LSI 和已有文本语义相似度算法 W2V 的对比实验、使用改进的学习排序模型和 IR SVM 模型的对比实验以及 Tr-WELR 模型与国际领先水平的方法 ENRL 的对比试验。三组实验均达到了相应的设置目的，最后证明了提出的模型在本文任务的有效性和准确性。

（5）实现了基于 Tr-WELR 模型的适航领域软件需求跟踪算法原型系统。该系统包括数据获取及录入、数据分析、层级关系建立、适航审定目标满足性判定以及适航审定检查单下载等主要功能，以及数据管理、用户管理等系统功能。

展望

首先，本文提出的模型在文本预处理时，将自然语言的停用词和一些软件文档中经常出现的词去掉了，因此破坏了文本的句法结构。在使用自然语言编写的软件文档中，句子的句法结构对整个语义的表达也起着较重要的作用。因此，虽然使用 word embedding 能够较好的表示句子中单词或短语的上下文语义和词与词之间的共现关系，并且提出的算法能够更有效和更准确的计算文本语义相似度关系，但是，如果能够增加对句法关系的考虑，能够使语义相似度的计算更加精准，这也是本文未来工作之一。

然后，本文在使用学习排序算法时，选取的两类特征是在软件工程领域中常用的特征，这个工作可以再进行下去，探索并尝试使用多个其他的特征。同时应该从不同的角度对特征进行选取，使用更多的、不同类型的特征对模型进行训练可能使实验结果更好。同时，虽然改进后的 IR SVM 模型能够通过实验验证其有效性和其对结果精度的提升，但是还需要在理论上对改进的损失函数进行验证，从数学角度证明其正确性和有效性。

最后，由于适航领域文档数量较少，因此之后的工作仍需要从相关部门获取更多的适航领域软件文档，并对适航领域文档的特点进行更加深入的分析，从而提升模型的精确率。同时从模型运行时间上可以看到，计算时间和文档数量、平均文档长度均成正比，因此对大规模的文档能够使用分布式算法进行加速，对于提升对本文提出的系统的性能一定有很大的帮助。

参考文献

[1] 沈小明, 王云明, 陆荣国. 机载软件研制流程最佳实践适航标准 DO-178B/C 研究[J]. 2013.

[2] RTCA (Firm). SC 167. Software considerations in airborne systems and equipment certification [M]. RTCA, Incorporated, 2012.

[3] 胡成海, 彭蓉, 王帮超. 基于信息检索的需求跟踪方法综述[J]. 计算机应用与软件, 2017, 34(10): 20-28.

[4] 王金水，翁伟，彭鑫. 一种基于句法分析的跟踪关系恢复方法[J]. 计算机研究与发展，2015, 52(3): 729-737.

[5] 王慧灵. 支持动态需求跟踪的构件关联机制研究[D]. 湖南大学, 2014.

[6] 董刘, 李引, 李娟. 基于注释改进动态需求跟踪的方法[J]. 计算机工程与设计, 2009 (1): 113-115.

[7] 王金水，薛醒思，唐郑熠. 一种基于命名实体识别的需求跟踪方法[J]. 计算机应用研究，2016, 33(1): 132-135.

[8] Hayes J H, Dekhtyar A, Osborne J. Improving requirements tracing via information retrieval[C]//Requirements Engineering Conference, 2003. Proceedings. 11th IEEE International. IEEE, 2003: 138-147.

[9] Lormans M, Van Deursen A. Can LSI help reconstructing requirements traceability in design and test?[C]//CSMR. 2006, 6: 47-56.

[10] De Lucia A, Di Penta M, Oliveto R, et al. Improving ir-based traceability recovery using smoothing filters[C]//Program Comprehension (ICPC), 2011 IEEE 19th International Conference on. IEEE, 2011: 21-30.

[11] Capobianco G, Lucia A D, Oliveto R, et al. Improving IR‐based traceability recovery via noun‐based indexing of software artifacts[J]. Journal of Software: Evolution and Process, 2013, 25(7): 743-762.

[12] Cleland-Huang J, Gotel O C Z, Huffman Hayes J, et al. Software traceability: trends and future directions[C]//Proceedings of the on Future of Software Engineering. ACM, 2014: 55-69.

[13] Furnas G W, Landauer T K, Gomez L M, et al. The vocabulary problem in human-system communication[J]. Communications of the ACM, 1987, 30(11): 964-971.

[14] Carpineto C, Romano G. A survey of automatic query expansion in information retrieval[J]. ACM Computing Surveys (CSUR), 2012, 44(1): 1.

[15] Pyshkin E, Klyuev V. A study of measures for document relatedness evaluation[C]//Computer Science and Information Systems (FedCSIS), 2012 Federated Conference on. IEEE, 2012: 249-256.

[16] Ye X, Shen H, Ma X, et al. From word embeddings to document similarities for improved information retrieval in software engineering[C]//Proceedings of the 38th international conference on software engineering. ACM, 2016: 404-415.

[17] Falessi D, Di Penta M, Canfora G, et al. Estimating the number of remaining links in traceability recovery[J]. Empirical Software Engineering, 2017, 22(3): 996-1027.

[18] Guo J, Cheng J, Cleland-Huang J. Semantically enhanced software traceability using deep learning techniques[C]//Proceedings of the 39th International Conference on Software Engineering. IEEE Press, 2017: 3-14.

[19] Falessi D, Cantone G, Canfora G. Empirical principles and an industrial case study in retrieving equivalent requirements via natural language processing techniques[J]. IEEE Transactions on Software Engineering, 2013, 39(1): 18-44.

[20] Le T D B, Oentaryo R J, Lo D. Information retrieval and spectrum based bug localization: better together[C]//Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering. ACM, 2015: 579-590.

[21] Wang S, Lo D. AmaLgam+: Composing Rich Information Sources for Accurate Bug Localization[J]. Journal of Software: Evolution and Process, 2016, 28(10): 921-942.

[22] Jiang H, Zhang J, Ren Z, et al. An unsupervised approach for discovering relevant tutorial fragments for APIs[C]//Proceedings of the 39th International Conference on Software Engineering. IEEE Press, 2017: 38-48.

[23] Greengrass E. Information retrieval: A survey[J]. 2000.

[24] 向广利, 陶然, 林香, 等. 基于向量空间模型的短文本密文检索方法[J]. 计算机工程与设计, 2017, 38(11): 2909-2913.

[25] 张佩云, 陈传明, 黄波. 基于子树匹配的文本相似度算法[J]. 模式识别与人工智能, 2014, 27(3): 226-234.

[26] Uthayan K R, Anandha Mala G S. Hybrid ontology for semantic information retrieval model using keyword matching indexing system[J]. The Scientific World Journal, 2015, 2015.

[27] Zhou H, Jia C, Yang Y, et al. Combining Large-Scale Unlabeled Corpus and Lexicon for Chinese Polysemous Word Similarity Computation[C]//China Conference on Information Retrieval. Springer, Cham, 2017: 198-210.

[28] Jain A, Gaur A. Summarizing Long Historical Documents Using Significance and Utility Calculation Using Wordnet[J]. Imperial Journal of Interdisciplinary Research, 2017, 3(3).

[29] 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.

[30] Harris Z S. Distributional structure[J]. Word, 1954, 10(2-3): 146-162.

[31] Severyn A, Moschitti A. Learning to rank short text pairs with convolutional deep neural networks[C]//Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, 2015: 373-382.

[32] Ye X, Bunescu R, Liu C. Learning to rank relevant files for bug reports using domain knowledge[C]//Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering. ACM, 2014: 689-699.

[33] Niu H, Keivanloo I, Zou Y. Learning to rank code examples for code search engines[J]. Empirical Software Engineering, 2017, 22(1): 259-291.

[34] Cossock D, Zhang T. Subset ranking using regression[C]//International Conference on Computational

Learning Theory. Springer, Berlin, Heidelberg, 2006: 605-619.

[35] Li P, Wu Q, Burges C J. Mcrank: Learning to rank using multiple classification and gradient boosting[C]//Advances in neural information processing systems. 2008: 897-904.

[36] Cao Y, Xu J, Liu T Y, et al. Adapting ranking SVM to document retrieval[C]//Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 2006: 186-193.

[37] Joachims T. Optimizing search engines using clickthrough data[C]//Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2002: 133-142.

[38] Freund Y, Iyer R, Schapire R E, et al. An efficient boosting algorithm for combining preferences[J]. Journal of machine learning research, 2003, 4(Nov): 933-969.

[39] Cao Z, Qin T, Liu T Y, et al. Learning to rank: from pairwise approach to listwise approach[C]//Proceedings of the 24th international conference on Machine learning. ACM, 2007: 129-136.

[40] Xu J, Li H. Adarank: a boosting algorithm for information retrieval[C]//Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 2007: 391-398.

[41] Yue Y, Finley T, Radlinski F, et al. A support vector method for optimizing average precision[C]//Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 2007: 271-278.

[42] Alnofaie S, Dahab M, Kamal M. A novel information retrieval approach using query expansion and spectral-based[J]. information retrieval, 2016, 7(9).

[43] Herbrich R. Large margin rank boundaries for ordinal regression[J]. Advances in large margin classifiers, 2000: 115-132.

[44] 关立哲, 韩纪富, 朱峰, 等. 军校学报稿件的脱密加工方法与技巧[J]. 中国科技期刊研究, 2013, 24(6): 1192-1194.

[45] Lai S, Liu K, He S, et al. How to generate a good word embedding[J]. IEEE Intelligent Systems, 2016, 31(6): 5-14.

[46] Xue N, Bird E. Natural Language Processing with Python[J]. Natural Language Engineering, 2011, 17(3): 419.

[47] Rehurek R, Sojka P. Software framework for topic modelling with large corpora[C]//In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks. 2010.

[48] Roy D, Paul D, Mitra M, et al. Using word embeddings for automatic query expansion[J]. arXiv preprint arXiv:1606.07608, 2016.

[49] Foss T, Stensrud E, Kitchenham B, et al. A simulation study of the model evaluation criterion MMRE[J]. IEEE Transactions on Software Engineering, 2003, 29(11): 985-995.

攻读硕士学位期间取得的学术成果

[1]. **Zhao Teng**, Cao Qinghua, Sun Qing. An Improved Approach to Traceability Recovery Based on Word Embeddings[C]. Asia-Pacific Software Engineering Conference, 2017:81-89. (EI-Indexed, CCF-C)

[2]. Sun Qing, **Zhao Teng**, Yao Zhong, Xu Xi. The prediction model of online consumer reviews performance based on sentiment mining and Tobit regression[C]. CIE 2016: 46th International Conferences on Computers and Industrial Engineering, 2016:158-169. (EI-Indexed)

致　谢

不知不觉即将度过研究生生涯，在两年半的时光里，有收获、有成长、有喜悦、有心酸，过程起起伏伏，但又似波澜不惊。

首先，衷心感谢我的研究生导师曹庆华教授。曹老师治学严谨，学识渊博，不仅在学习和工作中给予很多指导和帮助，在生活中也给予我指引和教诲，让我在研究生的过程不断明确人生方向。同时曹老师极大地工作热情和尽职尽责的工作态度深深的影响着我，让我在今后的成长道路上时刻保持永不松懈的精神，不断以更高标准严格要求自己。在论文研究阶段和实验阶段，曹老师都给予了非常多且重要的指导，让我能够沿着正确的方向研究下去。这些都是我攻读硕士学位期间非常大收获和一生都将受用的财富。在这里祝曹老师工作顺利，生活幸福。

其次，衷心感谢孙青老师在两年半的时光中对我学业和生活的关心和支持。在研究内容尚未清晰时，孙老师耐心地与我探讨相关工作，为我开拓思路，并尽可能地帮助我寻找解决问题的方法。在投递小论文时，孙老师非常耐心细致地帮助修正语法错误和不正确的内容，对我的帮助特别大，在此感谢孙老师的无私的帮助。同时，孙老师解决问题时的钻研精神和一丝不苟的工作态度非常值得我学习。在此祝愿孙老师在未来的工作和生活中一切顺利，幸福美满。

然后，感谢实验室的刘云师兄、庞静雯师姐在我学习和生活中遇到困难和挫折时对我的鼓励和帮助；感谢王鑫治、田庆松、苑铎同学在实验室时常与我讨论问题，开拓了我完成论文时的思路，同时感谢同学们在找工作时对我的帮助。祝愿大家前程似锦。

当然，更重要的要感谢我的父母，感谢你们对我一直以来的支持，感谢你们在我求学生涯过程中对我无微不至的关心和鼓励。无论顺境逆境，只要想到你们会一直站在我的身边，都会充满力量，继续走下去。愿你们身体健康，一切顺利。

还要感谢本文引用文献的众多作者们，你们的文章开拓的我解决问题的思路，也为论文提供了理论依据和基础。

最后，感谢审阅本论文的评审老师和答辩组老师们，感谢你们对论文的指导和宝贵的修改意见。