**中图分类号：TP****311.5**

**论文编号：10006ZY1504608**



专业硕士学位论文

**软件适航审定过程中需求可追踪性方法的研究与实现**

作者姓名 田庆松

学科专业 航空工程

指导教师 曹庆华 教授

培养院系 能源与动力工程学院

**Research and Implementation of Requirements Traceability in Software Airworthiness Certification**

A Dissertation Submitted for the Degree of Master

**Candidate：Tian Qingsong**

**Supervisor：Prof. Cao Qinghua**

School of Energy and Power Engineering

Beihang University, Beijing, China

**中图分类号：TP311.5**

**论文编号：10006ZY1504608**

硕 士 学 位 论 文

软件适航审定过程中需求可追踪性方法的研究与实现

作者姓名 田庆松 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 航空工程 研究方向 软件适航

学习时间自 年 月 日 起 至 年 月 日止

论文提交日期 年 月 日 论文答辩日期 年 月 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“两机”重大专项全面启动，我国航空发动机产业将加速发展。作为其控制系统，全权限数字式电子控制系统(Full Authority Digital Engine Control, FADEC)也将面临更多挑战。FADEC系统如何在航空发动机控制需求日益复杂的情况下，能够保证对其高效正确的控制显得更加重要。因此，在投入商用之前，必须通过适航取证，以保证软件的正确性和可靠性。DO-178B/C标准是国际广泛认可的适航认证安全标准，它明确指出，机载软件在开发过程中必须满足需求的可追溯性。可追溯性可以描述软件开发过程中每个设计元素间的追溯关系，并确保要求的功能已被实现。

适航审定过程中通常会涉及到大量文本文件，人工从这些文件中构造需求间的追踪链是一项需要大量精力，枯燥乏味，并且容易出错的任务。因此，利用软件数据动态的构建需求间的追踪关系，对于适航审定过程具有重要意义。本文在需求追踪领域的研究基础之上，结合领域文本的特点，使用词嵌入技术和机器学习排序算法来动态的构造需求间的追踪链，并通过与多个算法进行对比进行了验证。本文取得的主要成果如下：

第一，通过对适航认证体系和标准的研究，明确了DO-178B/C对于可追溯性目标的要求，分析了软件研发过程与软件生命周期数据的交互关系，为FADEC的适航审定提供了理论依据。

第二，将当前流行的词向量技术应用于需求追踪领域，并通过分析领域文本的特点合理利用了在使用词向量时的缺失词，提出了一种在运行时处理缺失词的方法，并改进了文本相似度算法，最终得到按相似度排列的排序列表。考虑到文本检索时常常会结合多个文本特征，因此在本文中，将在文本检索领域成熟的学习排序算法应用于需求间可追踪链构建的任务中。对排序列表进一步处理，提高可追踪链的质量。

第三，针对所提出的面向适航领域的需求可追踪链构建方法的每个阶段设计了对比实验，其中包括与对文本相似度算法，学习排序算法和整体模型与国际领先的ENRL算法进行了对比。实验结果优于对比实验算法。

最后，根据所提出的需求追踪算法模型，设计并实现了一个面向适航领域的需求追踪原型系统，包括数据交互，计算模块，用户管理等模块。

**关键词：适航审定，需求可追踪性，文本相似度，学习排序**

# **Abstract**

With the launch of major special projects for aero-engines and gas turbines, the aero-engine industry in our country will develop rapidly. As its control system, FADEC also faces more challenges. How the FADEC system can ensure efficient and correct control is more important with the increasingly complex requirements of the aero-engine control system. For commercial use, it must be certified by airworthiness to ensure the correctness and reliability of the software. Traceability can describe the traceability of each design element in the software development process and ensure that the required functionality has been implemented.

There are a lot of text files during the airworthiness certification. Traceability links recovery from these files by manually is a task that requires lots of efforts, tedious, and error-prone. Therefore, automating the traceability links with software data is of great significance to airworthiness certification process. Based on previous researches in traceability, we take the word embeddings and learnig to rank to recovery the traceability links dynamically in the domain field. And we have conducted several contrast experiments to verify our method. The main achievements of this paper are as follows:

Firstly, summarize the requirement of DO-178B/C for traceability targets according the study of airworthiness certification system and standard. we and analyzed the interactive relationship between the software development process and the software lifecycle data by the research on the airworthiness certification system and standards, which provided a theoretical basis for the airworthiness certification of FADEC.

Secondly, we approach a novel method to calculate the text similarity between two artifacts. We adapt the popular word embeddings to traceability tasks, and for the out-of-vocabulary words, we map it to vector space respectively at a test time taking into consideration of domain texts. In the end, we applied a mature machine learning method to rank the list we get from last step, which can be used to improve the result performance in precision and provide us an excellent prediction model by combining features in datasets.

Thirdly, we design contrastive experiments for each pharse of the method, including the text similarity, learn to rank and we compare our method with the state-of-art method ENRL. And our method outperforms.

Finally, we develop a prototype system based on our method for the airworthiness field, including data interaction module, computing module, user management moudle and other modules.

**Keywords:** Airworthiness certification, Requirements traceability, Text similarity, Learn to rank

# 目 录

[第一章 绪论 1](#_Toc515955741)

[1.1课题研究背景和意义 1](#_Toc515955742)

[1.2国内外研究现状 2](#_Toc515955743)

[1.2.1 FADEC系统软件概述 3](#_Toc515955744)

[1.2.2 软件需求追踪技术 3](#_Toc515955745)

[1.3研究目标与研究内容 7](#_Toc515955746)

[1.4论文组织结构 7](#_Toc515955747)

[第二章 相关理论与技术研究 9](#_Toc515955748)

[2.1 适航标准研究 9](#_Toc515955749)

[2.1.1 适航认证体系 9](#_Toc515955750)

[2.1.2 DO-178B/C标准研究 10](#_Toc515955751)

[2.1.3 DO-178B/C标准中规定的可追溯性目标 12](#_Toc515955752)

[2.2 文本相似度技术 14](#_Toc515955753)

[2.2.1 向量空间模型 14](#_Toc515955754)

[2.2.2 潜在主题模型 17](#_Toc515955755)

[2.3 词嵌入简介 17](#_Toc515955756)

[2.4 学习排序 19](#_Toc515955757)

[2.5 本章小结 20](#_Toc515955758)

[第三章 适航领域软件需求追踪算法模型 21](#_Toc515955759)

[3.1 模型框架概述 21](#_Toc515955760)

[3.2 文本相似度算法 23](#_Toc515955761)

[3.2.1 面向领域的缺失词处理方法 23](#_Toc515955762)

[3.2.2 改进的文本相似度算法 25](#_Toc515955763)

[3.3 学习排序算法 26](#_Toc515955764)

[3.3.1 排序算法选择 26](#_Toc515955765)

[3.3.2 特征选择 31](#_Toc515955766)

[3.4 本章小结 32](#_Toc515955767)

[第四章 适航领域软件需求追踪算法模型验证 33](#_Toc515955768)

[4.1 数据准备及预处理 33](#_Toc515955769)

[4.2 实验设置 36](#_Toc515955770)

[4.3 模型评价指标 37](#_Toc515955771)

[4.4 实验结果与分析 38](#_Toc515955772)

[4.4.1 文本相似度对比实验 39](#_Toc515955773)

[4.4.2 学习排序算法效果对比实验 40](#_Toc515955774)

[4.4.3 与ENRL方法对比 41](#_Toc515955775)

[4.5本章小节 42](#_Toc515955776)

[第五章 需求追踪算法原型系统的设计与实现 44](#_Toc515955777)

[5.1 需求追踪算法原型系统的需求分析 44](#_Toc515955778)

[5.2 系统总体设计 45](#_Toc515955779)

[5.2.1 系统总体架构设计 45](#_Toc515955780)

[5.2.2 系统数据库设计 46](#_Toc515955781)

[5.3 主要功能模块介绍 48](#_Toc515955782)

[5.3.1 数据管理模块 49](#_Toc515955783)

[5.3.2 计算模块 50](#_Toc515955784)

[5.3.3 用户管理模块 52](#_Toc515955785)

[5.4 开发环境 53](#_Toc515955786)

[5.5 本章小结 53](#_Toc515955787)

[总结与展望 55](#_Toc515955788)

[总结 55](#_Toc515955789)

[展望 56](#_Toc515955790)

[参考文献 57](#_Toc515955791)

[攻读硕士学位期间取得的学术成果 61](#_Toc515955792)

[致 谢 62](#_Toc515955793)

# 图 目 录

[图 1 FADEC软件系统功能模块图 3](#_Toc515895193)

[图 2 需求追踪矩阵示意图 4](#_Toc515895194)

[图 3 基于信息检索的需求追踪算法经典模型 5](#_Toc515895195)

[图 4 本文组织结构图 8](#_Toc515895196)

[图 5 DO-178B/C软件过程间关系 10](#_Toc515895197)

[图 6 DO-178B/C软件生命周期方阵图 11](#_Toc515895198)

[图 7 数据、过程和目标间关系 12](#_Toc515895199)

[图 8 DO-178B/C中的可追溯性关系 13](#_Toc515895200)

[图 9 数据追踪示意图 14](#_Toc515895201)

[图 10 Skip-gram模型示意图 18](#_Toc515895202)

[图 11 CBOW模型示意图 18](#_Toc515895203)

[图12 需求追踪算法模型流程图 22](#_Toc515895204)

[图13 某项目中高层需求的描述 24](#_Toc515895205)

[图14 辅助向量计算流程 24](#_Toc515895206)

[图15 原始排序问题 27](#_Toc515895207)

[图16 排序问题转化为分类问题 28](#_Toc515895208)

[图17 软件生命周期数据获取 33](#_Toc515895209)

[图18 维基百科数据获取流程 36](#_Toc515895210)

[图19 不同算法下的F1值 40](#_Toc515895211)

[图20 使用IR SVM前后MAP对比 41](#_Toc515895212)

[图21 使用IR SVM前后MRR对比 41](#_Toc515895213)

[图22 数据输入用例图 44](#_Toc515895214)

[图23 功能模块用例图 45](#_Toc515895215)

[图24 系统架构总体设计图 46](#_Toc515895216)

[图25 系统数据库设计图 47](#_Toc515895217)

[图26 系统功能模块图 48](#_Toc515895218)

[图27 用户对录入的word确认界面 49](#_Toc515895219)

[图28 计算模块UML类图 50](#_Toc515895220)

[图29 文本相似度界面 51](#_Toc515895221)

[图30 追踪关系结果界面 51](#_Toc515895222)

[图31 生成的检查单 52](#_Toc515895223)

[图32 RBAC类图 52](#_Toc515895224)

[图33 用户管理界面 53](#_Toc515895225)

# 表 目 录

[表1 适航审定体系 9](#_Toc515895226)

[表2 常见的排序算法 19](#_Toc515895227)

[表3 开发过程子过程与软件验证过程交互数据 21](#_Toc515895228)

[表4 排序列表实例 29](#_Toc515895229)

[表5 常用脱密方法 34](#_Toc515895230)

[表6 实验数据集描述 35](#_Toc515895231)

[表7 设计的对比实验 36](#_Toc515895232)

[表8 四种算法的precison和recall值 39](#_Toc515895233)

[表9 ENRL算法在eTOUR和EasyClinic上的最优组合方法 42](#_Toc515895234)

[表10 与ENRL算法的MRE值对比 42](#_Toc515895235)

[表11 开发环境详细信息 53](#_Toc515895236)

# 第一章 绪论

## 1.1课题研究背景和意义

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》，主要针对大型涡扇发动机电子控制系统软件的适航审定过程中对在适航审定过程中软件需求的可追踪性目标要求的问题，开展相关的基础技术研究。

适航从字面理解即适合飞行，在中国民用航空局适航司颁发的《中国民用航空器适航管理》中的定义是，是指民用航空器，包括其子系统整体性能和操纵性能在预期的环境和使用限制下的安全性和物理完整性的一种固有品质[1]。我国虽然已经具备自主研制航空发动机的能力，但是在对航空发动机适航审定过程中，仍在面临一些困难：一方面，由于我国航空发动机研制技术相对国际领先水平仍有较大差距，与之配套的适航审定经验不足；另一方面，对航空发动机控制系统的适航审定是的一个重要步骤，这也是未来通过适航审定、取得型号合格证的基础，更是产品进入市场、取得商业成功、实现国家战略的基础。

航空发动机控制系统是保证发动机稳定工作、安全可靠的重要保证 [2]，目前其控制系统中主要采用的FADEC。作为机载软件，在对其的适航审定过程中，一个重要的参考是2011年由航空无线电技术委员会（RTCA）在DO-178B标准的基础上提出的DO-178C，即《机载系统和设备合格审定中的软件考虑》。在我国航空工业中，DO-178C适航符合性标准被引入适航审定过程，如西安航空计算技术研究所(中国航空工业第631研究所)，中国航空发动机控制系统研究所（614 所）的一些项目。DO-178B/C 面向适航审定的过程和目标，明确了研制满足适航性的软件的目标，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。在软件研制的生命周期中，机载软件的研制单位要求对其研制的软件产品及其开发过程按照 DO-178B/C 的要求进行内部评审，以发现不符合 DO-178B/C 目标的问题并及时解决。本论文主要针对在DO-178B/C标准所规定的对适航软件开发过程中，软件需求间可追溯性应该达到的目标和适航审定过程中软件需求追踪链接的构建的相关方法进行了研究，DO-178B/C在所规定的目标中明确指出：1）高层需求可追踪到系统需求；2）低层需求可追踪到高层需求；3）源代码可追踪到低层需求[3]。

广义的软件需求追踪除了系统需求，高层需求及低层需求之间的追踪关系还包括如测试用例，用户样例与原代码等的追踪关系等，用于辅助验证软件的可追溯性。Center of Excellence for Software and System Traceability (CoEST)把软件可追踪性定义为：能够将任何唯一可识别的软件工程制品与其他软件制品关联起来，并在一段时间内维护所需的链接，并使用这些追踪来解决产品及其开发过程前向后向追踪的问题。它是评估软件和系统，尤其是安全攸关软件系统的一个重要的评价指标。完整和准确的可追溯性链接可以确保在变更需求时对每个相关的元素都进行了相关操作，并确保每一个需求都得到实现[5]。在实际的软件开发过程中，常常使用需求追踪矩阵，实体联系模型和追踪图[6]等静态的方式来维持需求间的追踪关，IBM Rational DOORS、IBM Rational Requisite Pro和青铜器RDM等工具可以用来辅助需求的管理。随着软件规模和周期的增长，创建和维护需求追踪不仅需要较高的成本，而且人工判断容易出错，耗时长。而这恰恰是动态建立需求追踪链的方法的优势。动态需求追踪中比较常用的思想是，分析软件制品的相关文件，使用信息检索中的相关技术，如向量空间模型（Vector Space Model，VSM）、隐语义模型（Latent Semantic Indexing，LSI）和主题模型（Topic Model）[7]等方法来计算软件制品间的相似度，然后通过设置相似度阙值作为需求追踪判断条件的依据。他们有一个共同的缺点是，他们把词汇当作最小独立的“单元”，导致这些方法往往会面临“词汇鸿沟”的问题。

词嵌入（word embeddings）是近年来自然语言处理领域的一个热门分支，它可以通过使用神经网络模型对大量无标记的数据进行训练来得到，这种技术的一个重要特性是携带了语义特性，在向量空间距离相近的词语具有相似的语义，在文本相似度计算上具有较好的效果。同时，软件工程领域的相关问题也开始引入机器学习方法，其输入是将文本的不同特征的组合，这种方法也为处理动态需求追踪相关的问题提供了借鉴思路。

综上所述，可追溯性作为DO-178B/C所要求必须实现的目标，在航空发动机控制系统的开发和适航审定过程中有着重要作用。因此，研究适航领域的需求可追踪算法来辅助审查人员半自动或自动化的构建需求间的追踪链，能极大减轻适航审查人员的工作负荷，减少出错率，对航空发动机系统的适航审定有重要意义。

## 1.2国内外研究现状

### 1.2.1 FADEC系统软件概述

从上世纪70年代FADEC开始出现，国外已经在军用发动机F100和民用发动机JT9D上对系统的性能、使用和维护指标进行了验证。经过30多年的发展，FADEC系统已经成为目前航空发动机的主要控制系统[8]，随着我国成功研制出某型号发动机的控制系统，说明我国在发动机控制系统的研制上具有了一定实力。

目前FADEC软件系统主要包括信号采集和处理模块，核心机控制模块，系统监控模块，信号输出模块，冗余管理模块和支撑软件模块等六部分。其模块图如图1所示。



图 1 FADEC软件系统功能模块图

可以看到，FADEC软件系统由多个子模块构成，涉及到数据的采集和处理，系统控制等等方面的功能，软件系统复杂。随着我国航空领域技术的不断发展，对航空发动机的要求也越来越高，随之而来的是航空发动机控制系统软件的需求越来越复杂。

### 1.2.2 软件需求追踪技术

可追溯性作为航空发动机适航符合性审查的目标之一，其重要性早已软件业界认可。软件能力成熟度集成模型(Capability Maturity Model Integration, CMMI)对需求管理提出的明确的指导性的方法，目前已逐渐被软件工程领域所采用，需求追踪作为软件需求管理的重要组成部分，其重要性也是不言而喻。

需求追踪技术主要分为两种类型：静态需求追踪技术和动态需求追踪技术。静态需求追踪技术通常是由手工构建和维护追踪链，常用的方法有：需求追踪矩阵、交叉引用和追踪图。动态追踪技术是为了以自动或半自动化技术帮助开发人员建立和维护可追踪链，这个过程中各种软件制品被看作静态文本文件，根据文本间的关系来构建可追踪链。

#### 1.2.1.1 静态需求追踪技术

静态需求追踪方法优点易于构建，是传统需求追踪中主要使用的方法，但由于只能静态构造，在大型项目或软件需求变更频繁的项目中，这种方法却难以维护和扩展。

需求追踪矩阵（Requirement Traceability Matrix，RTM）是展现和表示需求间追踪信息的最简单方法，它的主要的表现方式是通过矩阵或者表格来表示两个或者多个软件制品间的追踪关系，需要注意的是，软件需求间存在的关系不仅仅是“一对一”的关系，还有“一对多”，“多对多”的关系。一个软件需求追踪矩阵实例如图2所示。



图 2 需求追踪矩阵示意图

追踪图与追踪矩阵类似，它可以用图形的表现形式把用户自定义的对象和关系，将软件需求、测试用例、及源代码等等软件制品表述出来，这种方法直观，灵活，易于理解。

交叉引用则是通过对相互关联的软件制品（实体），将其中一个实体的关联属性中给出另一个相关实体的引用的方式表示。可以在需求文档之间建立追踪关系，需求文档包括需求规格文档、需求说明文档等，对于发动机控制软件，还有系统需求、高层需求和低层需求等不同文档。这种方法表示追踪关系比较直观，方便实际使用。只适用于需求文档间关系的建立与处理是它的一个主要缺点[9]。

#### 1.2.2.2 动态需求追踪技术

为了解决在人工构建并维护软件需求的可追踪关系时费时费力，容易出错，难以维护的问题，有些学者开始尝试自动化或者半自动话的构建和维护需求间的追踪关系。但直到第13届需求工程(Requirements engineering, RE)大会上，“动态需求追踪”这个概念才首次由J. Cleland-Huang提出。

在构建可追踪链的过程中，学者们从不同角度提出了方法来自动构建需求间的追踪关系，主要有基于事件的追踪技术（Event-Based Traceability，EBT），基于场景的追踪技术（Scenario-Based Traceability，SBT），基于程序运行时（Runtime-Based Tracebility，RBT）及基于信息检索技术(Information-Based，IR)[7]等。由于软件在设计、开发和维护过程中产生的生命周期数据大部分为文本信息，而需求之间的关系可以认为是文本间的关系。在这种情况下，基于信息检索技术的动态的需求追踪算法中有广泛应用，基于信息检索的需求追踪算法经典模型[11]如图3所示。

软件工程师

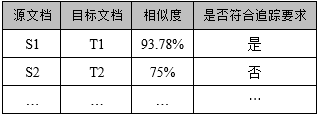
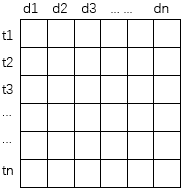
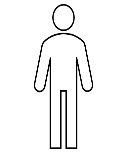
文本向量

结果分析

选择文件

软件制品

文本处理



相似度计算

图 3 基于信息检索的需求追踪算法经典模型

这种方法的主要思想是通过比较文档间的相似度确定需求间可追踪性关系的，从图3可以看到，基于信息检索算法的动态需求追踪算法流程为：首先，需要对软件制品数据作为文档进行预处理，并建立文本的向量表示；然后通过文本相似度算法来计算由软件工程师所选定的查询文件和候选文件间的相似度，最后，通过将文本相似度与事先设定好的阙值去比较，如果相似度足够大，则说明两个文件间具有可追踪关系。显然如何计算文本相似度是需求追踪算法的一个关键问题。

Dag等人[12]首先提出通过使用文本分析的方法来判断需求间的追踪关系，它们通过自定义一些衡量标准，主要通过比较文本间单词的关系计算文本相似度，最后通过调整阙值，来找到最优结果，并取得了一定的结果，并指出了这种方法很有发展前景。同年，G. Antoniol等人[13]使用基于概率模型(Probability Model, PM)和向量空间模型的需求追踪方法为源代码和文档间的追溯性关系，并在自己的数据集上取得了很好的效果。目前的很多基于信息检索的需求追踪算法都是以他们所做的工作为基础的。

Huffman 等人[15]在向量空间模型算法的基础上，有两点改进，一个是手工构建关键词库，另一个是使用简单词典，将一些有关联的词对应起来，起到了扩展查询的作用，最后她们将这种算法应用于NASA的项目，通过与手工和单纯的向量空间算法进行了对比，有较好的效果提升。在这篇文章中她们还整理了NASA项目的需求子集，将高层需求、低层需求和追踪关系整理出来提供给其他研究者继续进行研究。

随后，很多学者通过不同的方法对文本进行表示，从而计算相似度来更快和获取更高质量的需求追踪链的方法，隐语义模型(LSI)和主题模型(LDA)[15,16]被应用于需求追踪，建立需求、软件制品和测试用例之间的跟踪关系，他们通过将具有相同意义的词语映射到同一空间以降低文本向量的维数，以此来降低运算复杂度并获得较高质量的可追踪链。直到现在这些方法已经成为需求追踪领域比较成熟技术[17]。

一些学者在此基础上，进行了改进，将蚁群算法，加入过滤器，本体[18-20]等方法与传统的文本相似度算法相结合，用于提升结果精度。到现在通过改进这种基于统计的信息检索方法所获得的提升已经很小。已经有学者尝试将机器学习和深度学习技术应用于这一领域。

David等人[21]通过用机器学习分类器的方法应用于需求追踪算法，并通过将不同的信息检索算法与多个分类器进行组合，给出了在不同数据集上的最佳实践。它指出可以使用分类器来对追踪链接进行分类，从而不需要在不同的数据集上设置阙值。这说明可以将机器学习方法应用于需求追踪的动态构建中。

而Jin G等人[17]则是将目前流行的word embedddings计算和深度学习技术应用于需求追踪领域。他们将文本用词向量特征表示，并通过循环神经网络（Recurrent Neural Networks，RNN）训练并预测文本间的相似度，他们的实验效果很好，但是它需要一个高质量的领域本体和大量被人工验证过的可追踪链接来进行训练，因此这种方法并不适用与较小的数据集，不具有普遍性。

除此之外，研究者还提供了一些基于信息检索技术的工具，如处理非结构化文本的NETRO，可拓展的实验工具TraceLab，候选链接聚类工具TraCter和支持投票的FaceTrace[22-25]等工具，来帮助学者进行研究。

## 1.3研究目标与研究内容

本文主要对航空发动机控制软件在开发过程中生命周期数据间追踪关系的构建方法展开了研究，根据DO178B/C的中关于需求可追踪性的要求，通过提出的算法自动化的建立软件生命周期数据间的追踪关系，为发动机控制软件的适航审定提供辅助依据。本文的主要工作内容详述如下：

（1）对航空发动机适航理论的研究，DO-178C标准中对软件生命周期的定义与普通软件有所不同，软件研发过程中要求大量的软件生命周期数据，并且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中需要注意的问题。

（2）需求追踪算法模型构建。动态构建需求追踪链有很多方法。本文提出了一种基于词向量技术的适航领域文本相似度算法，在考虑语义的同时，在算法过程中对词向量缺失词的动态处理；然后结合机器学习领域的学习排序技术进一步提升了需求追踪结果的精度。

（3）算法模型验证。针对提出的需求追踪算法，通过将提出的适航领域文本相似度算法与目前需求追踪中常用的相似度算法进行比较(LSI)，并将最终模型与文献[21]提出的ENRL方法进行对比，对提出的算法模型进行了验证。

（4）最后，基于算法模型，实现了一个需求追踪的原型系统，包括数据交互，数据管理，用户管理及需求可追踪性算法实现模块等。

## 1.4论文组织结构

本文一共分为六个章节，各个章节的编排和主要内容如图4所示。

第一章，绪论。首先给出了课题的来源，介绍了论文的选题背景及意义，同时介绍了当前国内外航空发动机控制系统的发展情况和软件需求追踪领域的研究情况，最后介绍了本文的研究目标和研究内容，并给出整个文章的组织结构。

第二章，相关理论与技术研究。本章围绕航空发动机软件控制系统适航审定所依据的法律，对需求可追溯性目标的要求展开了研究，并对目前基于信息检索的需求跟踪领域常用的关键技术进行了研究，这些技术是本文所提出需求追踪算法的基础。



图 4 本文组织结构图

第三章，面向航空发动机适航的软件需求追踪模型方法。本章通过结合目前较为先进的词嵌入（word embedding）技术和机器学习排序技术，提出了一个需求追踪算法模型，在这个过程中，通过分析适航领域文本特点，提出了一种在使用词向量时动态处理缺失词的方法，并辅助最后文本相似的的计算，接着通过结合学习排序算法，对通过计算相似度得到的候选列表再次处理，从而构建需求间的可追踪链。

第四章，算法模型验证。在本章中，首先为第三章算法模型的两个个阶段分别设置了对比实验来验证各个阶段的有效性，然后与最新提出的需求追踪领域算法进行对比来说明算法的实用性。在本章中还介绍了数据处理，训练词向量及评价指标的实验过程细节。

第五章，需求追踪原型系统的设计与实现。本章按照软件开发过程中的系统需求分析，总体架构设计，数据库设计和具体模块设计与实现依次进行介绍，并在最后给出了该系统的开发环境及工具的详细信息。

最后，总结与展望。本章主要是对全文工作进行总结，并介绍了今后的研究方向和工作重点。

# 第二章 相关理论与技术研究

## 2.1 适航标准研究

### 2.1.1 适航认证体系

多个国家和组织通过建立适航认证体系来保证航空器的适航性，航空器在投入使用前要通过相关地区的适航审定标准，主要的相关标准有：美国联邦航空管理局的联邦航空条例 (FAR)、欧洲航空安全委员会的联合航空规则(JAR)及中国民航总局的中国民用航空规章(CCAR) [26]等。

目前国际上比较权威的适航规章体系是美国的联邦适航规章以及欧洲适航认证标准，美国的适航审定体系分为四个层次：“适航法规”、“咨询通告”、“标准”以及“应用”。对其解释和内容举例见表1。

表1 适航审定体系

|  |  |  |
| --- | --- | --- |
| 文件类型 | 文件解释 | 与发动机控制软件有关的内容举例 |
| 法规（FAR） | 必须遵守，强制执行 | FAR25部：运输类飞机适航规定 |
| 咨询通告（AC） | 认证机构对法规的理解及提出使用法 | AC20-115B:推荐RTCA DO-178B  AC20-115C:推荐RTCA DO-178C |
| 标准（ARP series） | 实施时过程、活动应该达到的要求 | SAE ARP 4754A：民用飞机和系统开发指南  DO-178B/C：机载系统和设备合格审定中的软件考虑  SAE ARP 4761：民用飞机机载系统和设备安全性评估过程的指南和方法 |
| 实践 | 选择特定的方法，采用相应的工具，实现一系列的活动，满足标准，符合咨询通告，遵守法规 | FAA　Job Aid及补充文档：软件审查工作指南 |

由表1可以看到，在咨询通告AC-115B/C[27,28]中，就机载软件的符合性证明中分别推荐使用RCTA DO-178B/C，我国整个适航体系架构和内容都和美国联邦航空局(FAA)基本相同。DO-178B/C是整个民航适航认证体系的一个非常重要的标准，一直以来，DO178B/C都在国内外民用飞机机载软件系统的研制过程中被采用，对于航空发动机控制软件系统的适航审定也是如此。

### 2.1.2 DO-178B/C标准研究

20世纪70年代末期，随着数字计算机被越来越多的应用到飞机设备和系统中，美国航空无线电技术委员会制定了相应的软件规则，用来支持指导管理当局对机载软件系统的适航审定。自此之后，《机载系统和热备合格审定中的考虑》即DO-178系列诞生，经历了DO-178、DO-178A、DO-178B和 DO-178C等阶段[3]，通过与实践结合，不断检验和完善。其中，DO-178B在其使用的近20年里没有发现严重问题，经受住了时间和实践的考验。随着软件研发技术的不断发展，如今的技术水平已经与20年前有了较大的变化，这些新技术不能完全满足面向过程和目标的DO-178B的要求。因此，FAA和EASA在2005年共同授权小组对DO-178B内容进行修订，并于2011年底通过审核并正式颁布，即DO-178C。DO-178C是对DO-178B标准的继承和扩充，并没有实质上增加机制软件适航符合性的要求。新补充条款包括：DO-330软件工具验证考虑、DO-331基于模型驱动的开发和验证、DO-332面向对象和相关技术、DO-333形式化方法[29-32]等。



图 5 DO-178B/C软件过程间关系

软件生命周期是指软件产品从概念提出到产品“死亡”的整个周期。就机载软件而言，其生命周期根据DO-178B/C标准可分为软件计划，软件开发及软件综合三个过程。其中软件计划过程主要起到对其软件开发和综合过程的活动的定义和协调。而软件开发过程与开发和设计人员关系最密切，它包括软件需求、设计、编码和集成等四个过程。软件综合过程则用于保证软件生的命周期及软件输出正确，受控和可信的过程。它包括软件验证、构型管理、质量保证以及审定联络等四个子过程[3]，三个过程间的关系如图6所示。

在满足DO-178B/C标准的软件开发过程中，软件计划过程制定软件生命周期活动，是所有过程的起点，软件开发过程则是软件生命周期的主线，软件综合过程与软件开发过程同时执行，并且软件综合过程与软件开发过程的各个子过程上交叉执行。DO-178B/C软件生命周期方阵图6所示。虽然标准中对软件生命周期过程规定较多，但是并没有强制所有软件活动必须按照规定活动严格执行，只需要保证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[4]。



图 6 DO-178B/C软件生命周期方阵图

在整个软件生命周期中会产生的大量软件制品（Software Artifacts），即软件生命周期数据，它们是软件产品的表现形式，切实记录了软件的整个生命过程，局方以此基本依据对软件进行适航审定。在DO-178B/C中，列举了软件计划、软件开发和软件综合过程中的对应的多种软件生命周期数据。本文的目标主要是构建软件需求数据，设计描述数据，验证用例和源代码等数据间的追踪关系。

DO-178B/C的主要关注软件失效对于安全性的影响，例如，每个组件失效有多严重以及对于整个系统的安全性意味着什么。这种影响被分为五类，即灾难性的，危害性的，严重的，不严重的，以及没有影响的。分别记为Level A、B、C、D、E五个等级，其中Level A的认证是最严格，它需要提交大量的软件生命周期数据文档以软件满足DO-178C标准所要求71个目标的证明。之后等级所需满足目标数目逐渐减少，Level E表示对飞行的安全性没有影响，DO-178C的需求对它不适用。DO-178B/C作为目前航空领域和适航审定当局普遍认可的一种适航符合性审查标准，一经采用，则必须满足该标准对该等级软件要求实现的所有目标。航空发动机控制系统所对应的级别为最高等级Level A，也就是必须实现DO-178C所要求的71个目标。



图 7 数据、过程和目标间关系

### 2.1.3 DO-178B/C标准中规定的可追溯性目标

从图7可以看出，我们通过使用软件过程中生成的软件生命周期数据来证明其是否满足适航审定目标。DO-178B/C标准规定，在软件的生命周期中必须实现需求可追溯性目标。可追溯性是软件和系统在开发和维护过程中的重要因素，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要求至少完成以下几个可追溯性目标[3]：

1. 系统需求和高层需求之间的可追溯性

系统需求指的是机载系统在开发的过程中分配给软件的要求，通常是机载系统对于某个软件功能的描述；高层需求是软件某个功能的描述，用于满足功能、接口、性能和安全相关需求，它是若干个需求的集合，通常在软件需求规格说明书中描述。软件高层需求的正确性和完整性在软件的研制有重要的作用。需要注意的是，在需求细化的过程中还会产生衍生需求，这些衍生需求无法追溯到上层需求。

系统需求与高层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层需求。

1. 高层需求和低层需求之间的可追溯性

高层需求是描述了软件实现的某个功能，是若干需求的组合，需要进一步细化为可由编码实现的若干低层的需求和衍生的低层需求。低层需求在软件设计文档中描述，它可用于编码实现，它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。

高层需求和低层需求之间的追溯性是为了保证高层需求及衍生的高层需求全部被转化成了低层需求。

1. 低层需求和源代码之间的可追溯性

对于已经转化为源代码的低层需求，标准同样要求保持两者之间的可追溯性，主要是为了方便代码的复核和检验。

DO178B/C标准所要求可追溯性关系如图8所示。



图 8 DO-178B/C中的可追溯性关系

除此之外，DO-178C还在第11章显示的要求建立测试用例和测试规程，测试规程和测试结果间的可追踪性。这些追溯性关系数据需要提供给适航审定机构，它们被称为可追踪性数据（Trace Data）。这些数据直接的追踪示意图如图9所示。可以看到这些软件数据之间存在明显的偏序关系。在机载软件研制的实践过程中，软件生命周期数据通常是条目化的，它是最小粒度且具有独立语义的数据，我们可以称它为数据元(Data Element)。由于在软件研发流程的定义中，并不能确定各种数据元（如系统需求）的条数，它们随着项目活动的进行才会被确定。软件需求数据的可追踪性应该建立在数据元的粒度之上。使用粒度得当的数据元对数据进行追踪对软件验证活动有十分重要的作用。



图 9 数据追踪示意图

## 2.2 文本相似度技术

文本相似度是指通过量化的方法计算两个词语或者文本间的相似程度，广泛应用在信息检索、文本分类和智能问答等领域。文本相似度计算是一项复杂的任务，目前有很多学者它进行了研究。它的计算主要分为两个部分：一是适当的对文本进行表示，二是选择适当的相似度计算方法。

### 2.2.1 向量空间模型

向量空间模型（VSM）在上世纪在上世纪60年代由Salton等人[33]提出，主要思想是将文档映射到向量空间，从而向量化文档。随着计算机技术的不断发展，VSM算法被广泛应用于信息检索领和文本分类域。VSM算法的一个特点是只关心文档中词或短语出现的次数，即词频有关，而与该词条在该文档中出现的位置和顺序无关。VSM算法把的核心在于将文档数据的处理简化为向量空间中的向量运算，从而快速计算出文档间的相似度。使用VSM算法将文本数据转换成向量之后，可以通过计算各向量之间的相似性来度量各文本之间的相关程度。

VSM算法的基本原理可表示为：在VSM中，每一篇文档都可以表示成公式(2.1)的形式。

 (2.1)

在这里，表示文档第个特征词条，则表示第个特征词所对应的权重。

可以看出，如何计算特征词的权重的计算是VSM算法的核心内容。在计算文档中每个特征权重时，应当考虑文档中词语代表文档特征信息的能力，能力越强，应赋予的权重值越大。特征项的权重计算方法[34]有词频算法和TF-IDF算法。常用的是TF-IDF算法。

词频（Term Frequency, TF）算法，顾名思义，即用文本特征词在文档中出现的次数作为该特征词的权重。显然文档中出现次数最多的词具有最大的词频，即拥有最大的权重值。这并没有很好的结合文本特征，文本中的很多高频词汇并不能很好区分文档，如常用的形容词，名词，代词等，它们出现频率很多，但却不能很好的区分文本类型，相反有些低频词对文本的区分度却更强。词频算法只考虑到高频词语忽视低频词语在文本检索，分类中并不实用。所以在实际的应用中单独使用词频算法构建权重的实例并不多。可用公式(2.2)计算得到。

 (2.2)

其中，表示第个词汇在第个文本中出现的次数。

通过对只考虑词频构造特征权重的词频算法进行改进，在构建特征权重时，用TF-IDF（Term Frequency-Inverse Document Frequency，TF-IDF）算法用来衡量一个特征项对一个文档的重要程度，其权重计算方法是将文本特征项的词频和逆文档频率相乘所得的结果作为文档特征项的权重。词频（TF）认为在一篇文档中出现次数越多的词或短语代表这篇文档的能力越强。逆文档频率（IDF）则认为如果包含某个词或短语的文档越少，则表明该词条代表这篇文档的能力越强，就越重要。TF-IDF将两者的思想结合起来，某个词或短语在某一篇文档中出现的次数越多，并且在其他文档中出现次数较少，TF-IDF则认为该词或短语具有很高的类别区分能力。TF-IDF的计算公式如(2.3)：

 (2.3)

其中，代表在第篇文档中第个特征词出现的次数，代表训练语料库中文档的总数， 则训练语料库中含有特征词的文档数。

TF-IDF算法的思想并不复杂，它充分结合词频和文档逆频特征，认为在所有文档中出现频数高的词而在具体一篇文档中出现频数低的词条重要，在一定程度上抑制了高频词的作用。在实际的应用中，TF-IDF也较TF算法表现出了更好的性能。

用VSM算法表示文档时，计算两篇文档 和 的相似度有以下三种计算方法：欧式距离、向量夹角余弦和向量内积。

首先是欧式距离，可用公式(2.4)表示：

 (2.4)

其中，代表文档特征向量的维数或者是文档的特征个数，即不同单词的数量，代表文档中第个特征的权重，代表文档中第个特征的权重。

使用欧式距离度量文本间相似度时，欧式距离越小，即两文本之间相距越近则它们越相似。反之，若它们的距离越大，则它们的相似度就越小。

然后是用向量夹角余弦(cosine)计算相似度，可用公式(2.5)表示：

 (2.5)

其中，代表文档特征向量的维数或者是文档的特征个数，代表文档中第个特征的权重，代表文档中第个特征的权重。

使用向量夹角余弦度量文本相似度时，余弦夹角值越小，则文本间的夹角越大，它们的相似度就越小。反之，当夹角余弦越大时，则它们的相似度就越大。

最后介绍使用向量内积来计算相似度，可用公式(2.6)表示：

 (2.6)

其中，代表文档特征向量的维数或者是文档的特征个数，代表文档中第个特征的权重，代表文档中第个特征的权重。

使用向量内积度量文本间相似度时，若两个文档特征向量的内积越大则它们的相似度越大；反之，它们的相似度越低。

VSM的一个特点是不考虑文档词或短语的位置与顺序关系，这极大简化了在计算文档特征权重时的计算任务，但也失去了许多关于文档结构和语义的相关信息。在实际应用中，向量空间模型是目前使用最广泛的文档表示模型之一。

### 2.2.2 潜在主题模型

潜在主题模型(Latent Topic Model)基于具有相似含义的词汇总是出现在相似的语境中的假设，利用文本中具有相似语境的词汇间的联系，将文本空间映射到具有更低维度的向量空间中。对于语义关系相近的词来说，它们的向量表示在向量空间中距离较接近，可以说它们拥有相同的主题，且可以用余弦距离来衡量它们的相似度。

LSI是主题模型的一种，首先，可以将文档集表示为一个稀疏矩阵，的列向量表示一个文档，矩阵的元素可以使词频或者TF-IDF表示。构建好矩阵后，LSI首先如公式(2.7)对矩阵进行奇异值分解(SVD)，其中为特征值矩阵。只保留其中个最大的特征值，将矩阵进行降维，公式(2.8)中的就是降维后得到的新矩阵。降维不仅能减小矩阵的规模，提高计算效率，还可以一定程度去除原文档中的噪声数据。最重要的是通过降维可以将文档矩阵中语义相关的部分合并，将原矩阵变换到一个潜在的低维主题空间。文档中的每个单词或者文档都可以用该空间的一组权值向量表示。这些权值反映了与对应潜在主题的关联程度的强弱。得到单词或文档的向量表示之后，就可以用余弦距离或者其他方法来衡量它们之间的相似度。

(2.7)

(2.8)

## 2.3 词嵌入简介

统计语言模型(Statistical Language Model，SLM)是表示语言基本单位(通常为句子)的概率分布函数，即该语言的生成模型，它被广泛的应用于自然语言处理领域的语，是目前语言模型的基础。为了更形象的说明，假设句子由个词顺序构成，即，则句子的概率可以由的联合概率表示为公式(2.9)：

(2.9)

利用Bayes公式，上式可以分解为各个词语条件概率的乘积，如公式(2.10)：

(2.10)

其中为词语的上下文，这些条件概率就是语言模型的参数，可以根据这些条件概率的值，计算出句子的概率。

Bengio等人[35]提出的神经概率语言模型(Neural Probabilistic Language Model，NPLM)，这种方法利用神经网络算法来构建语言模型，将词语嵌入固定维数的向量(Distributed representations)。



图 10 Skip-gram模型示意图



图 11 CBOW模型示意图

词向量的概念最早由Hinton[36]在1986年提出，是一种把词语映射到实数空间固定维数向量的技术，即词嵌入，这种方法的理论基础是具有相似上下文的词语具有相近的意思，因此，具有相似词语的向量在向量空间中距离很近，而这可以看作词语的语义的特征。

Thomas Mikolov等人[37,38]对传统的NPLM做了两个改进，一是将RCNN引入NPLM，使之可以处理变长序列，另一个是在输出层加入softmax函数，通过开启从而加快了训练速度，使这项技术被广泛应用于自然语言处理任务中。在文献[39]中，他还给出了两个模型，Skip-gram和CBOW，来帮助使用者更好的训练词向量，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9和图10所示。

## 2.4 学习排序

学习排序(Learning to Rank, LtR)是一类自动学习将信息检索系统返回的结果更好的进行排序的机器学习方法的统称[44]，它主要研究的问题是如何构建查询和候选文档之间相关程度的函数或模型。它在自然语言处理、信息检索和推荐等领域有重要的作用。在软件工程领域，主要用于故障定位，测试用例和代码间可追踪性等等问题中。

学习排序在信息检索中的所要解决的问题可以描述为：给定训练集和测试集。 中文档用三元组表示为，其中为查询文档和为候选文档集的组合，表示两者的相关性，相关的记为正相关，否则记为负相关。其中作为机器学习算法输入的和是从文档中提取的特征，通过排序算法得到可以用公式(2.11)表示。常用的学习排序特征有语义相似度，文档逆频率和等。

(2.11)

其中，函数将查询-文档对映射为特征向量，是表示特征向量的权重矩阵，是排序函数。

按照训练样本类型的不同，现有的学习排序方法可以分为三种类型单文档方法（Pointwise）、文档对方法（Pairwise）和文档列表方法（Listwise）。Pointwise方法的训练对象是单个文档，通过将文档转化为特征向量的方式，把排序问题转化为了分类和回归问题。Pairwise则是将任意一对文档(,)当作算法的输入，最后输出结果是文档， 的相对位置关系，即和哪一个应该排在前面，此时的排序问题被转化为了了二分类问题。Listwise方法的训练样本则是对应每个查询的候选文档列表，其将整个文档列表作为一个整体，在数据足够大的情况下，往往有最好的效果。

表2 常见的排序算法

|  |  |
| --- | --- |
| **排序方法分类** | **常见算法** |
| 单文档方法(Pointeise) | Discriminative model for IR、SVM、McRank |
| 文档对方法(Pairwise) | Ranking SVM、RankBoost、IR SVM、RankNet |
| 文档列表方法(Listwise) | RankCosine、ListNet、ListMLE、SVM-Map |

在学习排序中，排序特征的选取同样会对算法的结果产生较大影响。在软件工程中常使用有词语或者文本相似度、上下文相似度还有词频，文本长度等特征。排序学习中的特征根据是否与查询语句相关可以分为两类：依赖查询特征和不依赖查询特征[45]。其中相似度特征就属于查询依赖特征，文本长度则是不依赖查询特征，它与查询文本无关。合理的选择特征的种类和数量有助于提升结果的精度。

## 2.5 本章小结

本章首先对适航领域的相关内容进行了研究，包括目前国际适航审定的体系，针对软件适航审定的DO178B/C标准，及DO-178B/C中所规定的软件开发过程，软件生命周期数据和对于可追溯性的目标要求，然后介绍了一些在需求追踪领域常用的文本相似度算法，接着介绍了词嵌入的语言模型，包括基于统计的语言模型基础，及基于神经网络的模型，最后介绍了机器学习排序算法相关理论，这些是本文后续研究的理论和技术基础。

# 第三章 适航领域软件需求追踪算法模型

## 3.1 模型框架概述

在介绍需求追踪模型框架之前，本节首先分析适航领域软件生命周期数据的特点，一方面，对于航空发动机领域的适航审定文档数据，需求追踪相关数据涉及功能性需求，非功能性需求和安全性需求等不同类别的元素，它们的表达需要准确的描述所实现的功能和约束，无二义性。例如一个规范的低层需求应该包括：需求标识，需求标题，需求类别，需求优先级，输入，输出及描述等条目。可以看到它是具有独立语义的最小单位，即数据元。在实际计算需求间的追踪关系时，需要对这些可追踪性文本结构内容进行分析，提取出需求合理的描述，以便于动态追踪链的构建。另一方面，由于适航领域追踪数据种类的多样性和领域相关性，导致文本内容构成非常复杂，其中不仅包含众多的航空和计算机软件等技术领域的专业词汇，代码等，还包括一些项目中的自造词。这些自造词无法从外部语料获得。因此如何恰当的处理这些词语对于词向量的应用非常重要。

除此之外，符合DO-178B/C标准的软件开发过程有其特别之处。从图6可以看出，满足DO-178B/C的软件开发过程和软件综合过程中的软件验证过程并行执行，软件验证是软件适航审查的主线，对于软件需求的可追溯性验证也在该阶段。为了保证子过程的验证过程，需要引入它所依赖的其他阶段补充数据。

在第二章中提到，数据元是具有独立语义并且不能继续拆分的数据，而数据元按照一定原则可以组成较为粗粒度的数据项。在满足DO-178B/C标准的软件开发过程中，软件开发过程（如图6所示）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。软件验证过程是软件审查的主线，在验证过程中，软件需求、设计和编码过程中需要和软件验证过程需要进行数据项的交互，其内容交互如表3所示。

表3 开发过程子过程与软件验证过程交互数据

|  |  |
| --- | --- |
| **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 | 可执行源代码、软件开发计划、软件验证计划 |

在符合DO-178B/C标准的软件开发过程中会产生以上各种生命周期数据，软件需求间的可追溯性需要对各种生命周期数据进行处理，计算和分析，最终得到一个初步的需求间可追踪关系来辅助发动机控制软件的适航审定过程。

算法的整体流程如图11所示，包括数据预处理及训练词向量，文本相似度计算和排序模型训练三个阶段。



图12 需求追踪算法模型流程图

首先是预处理阶段，这个阶段主要包含对适航领域软件生命周期数据的处理、词向量的训练和辅助向量的计算。由于适航领域文档的通常是保密文档，在处理之前应先脱密处理。一种常用的脱密方法是将重要的名词使用其汉语拼音的首字母表示关键词。如“发动机控制系统”可以替换为“FDJKZXT”，对于在本文所提算法中可以适当使用下划线和数字，来增加该名词代表文档的能力。对于软件制品，其预处理过程要根据输入的不同具体进行处理。对于前面提到的规范的低层需求来说，可以将其需求标题和需求描述提取出来作为一条低级需求的描述。

词向量的训练需要一个权威并且足够大的语料库。在本文中选取了维基百科作为训练的语料库。然后可以根据词向量库和经过预处理的软件制品数据共同构建缺失词库，最后可以得到软件制品的辅助向量。这个过程将在第四章详细说明。

然后是文本相似度计算阶段，通过预处理阶段，可以得到已经处理好格式的软件文档、训练好的词向量和辅助向量等数据，而计算的目标是获得软件文档根据相似度排列的排序列表。在这个阶段，通过利用词向量和辅助向量来计算查询语句和候选文档间的相似度，从而得到一个按照文本相似度排序的列表。排序列表中相似度越大的排名越靠前，说明与查询间的相关度越高，即越可能具有可追踪性关系，这里将本文提出的相似度计算方法记为CWI。

最后，引入学习排序算法，以得到的排序列表作为输入，通过训练机器学习算法模型，对候选列表进行处理，得到最终可追踪性链。

## 3.2 文本相似度算法

本节提出了一种基于词向量技术的改进文本相似度计算方法，利用适航领域软件文本数据特性，在计算相似度过程中动态处理文本中的缺失词，并将其用来辅助文本相似度的计算。

### 3.2.1 面向领域的缺失词处理方法

使用词向量计算文本相似度时，需要先用外部的大规模语料库中进行训练来获得词向量，因此，在这种方法常常会面词汇无法表示的情况，对于领域软件文本中这种现象更是普遍。这是目前使用词向量解决问题的一个常见问题，也是词向量应用的一个研究方向，Dhingra等人[46]在文章中总结并指出，目前处理这种缺失词的主要方法有3种：第一，直接丢掉缺失词，即在计算过程中，遇到丢失词则直接跳过这个单词，忽略并不予处理，这种方法最简单，然而往往丢失了最多的文本信息。第二，将所有缺失词均映射到某一特定的向量。即所有的缺失词映射为同一未使用向量。第三，将缺失的词向量重新映射为自由向量并记录这些向量。在他们的文章进行了多组对比实验证明合理处理词向量能有效提升各种文本检索，理解的任务的精度。这三种方法在使用词向量表示文本时均有一定的应用[39,40]。但是，针对适航领域文本，这些方法并没有考虑到缺失词的特性，丢失了重要的信息，下面给出一种针对适航领域的缺失词处理方法。

在对适航领域软件文本处理过程中，无法通过事先训练好的词向量表示的缺失词往往是命名实体，技术词汇或者符合词等，如图13所示是某项目中的高层需求描述，“DPU-CCM”为该项目特有词汇，它无法用预先训练好的词向量表示。这些无法表示为词向量的缺失词往往在软件的文本相似度计算过程中起着重要作用。具有相同缺失词的文本倾向于有更相关的文本追踪关系，以上三种常用的处理方式并不能很好的处理这些缺失词。针对这个特点，本文提出了针对适航领域文本领域的缺失词处理方法，具体流程如图14所示。

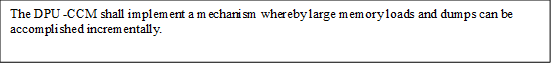


图13 某项目中高层需求的描述



图14 辅助向量计算流程

该流程可以描述为首先从预处理过程中得到可容易便于使用的软件文本文档，还有从维基百科领域数据训练而来的词向量。对于软件文本中的每个单词，可以通过查询是否可以由训练好的词向量表示来做相应处理。若可以表示则暂不处理，直接跳过单词，若不可以表示则将其加入缺失词汇表，可以用公式(3.1)形式化表示。

(3.1)

其中是指缺失词汇表集，代表软件文本中的单词，代表预先训练好的词向量词汇表。

在得到缺失词汇表后，即可以对软件文本中的缺失词用向量表示。在文本中如果一个词语在一个文档中出现频率较高，而在其他文档中出现频率较低时，则可以认为这个词语有较强的区别能力，即该词语对当前文本的表示有更大影响，基于此，本节选择第二章介绍的向量空间模型对软件文本中的缺失词进行表示，并使用TF-IDF构建特征向量，这样即可得到每个文本中缺失词向量，这个向量可以作为辅助向量应用于文本相似度的计算。

值得一提的是，对于不同的待审定项目，对应独立的缺失词汇表，并且可以通过人工审核和筛选，将缺失词汇表降低到一个对于用机器进行计算来说可以接受的维度。

### 3.2.2 改进的文本相似度算法

在使用词向量计算文本相似度前，需要先介绍使用词向量计算词与词之间的相似度的计算方法，它可以表示为如公式(3.2)。

(3.2)

其中，和是给定任意两个单词的词向量表示。从公式(3.2)可以看到，词语的相似度可以用向量点乘的结果与他们模的积之间的比值，即两个词向量的值。使用词向量可以很轻松的计算出词与词之间的相似度。

与词与词之间相似度的计算类似，在文献[39]中提到，在使用词向量计算文本相似度时，可以通过文本向量间的值来计算，首先，需要将文本表示为向量，本文使用将文本中所有词向量加和平均值的方式对文本进行表示，如公式(3.3)。

(3.3)

其中，是文本中词语的词向量了表示，即文本的向量表示。有两点需要注意：第一是所代表的是只由词向量来表示的文档向量形式，并未对缺失词进行处理。第二，用词向量的加和的平均值表示文本的原因是，在不同文本文件中句子的长度是不同的，为避免文本长度不同带来的影响，这里采用了求向量和的平均值的方法。

文本相似度则可以通过文本向量间的*cosine*值来求出，如公式(3.4)：

(3.4)

其中和分别代表需求计算相似度的两个文本向量。

在此基础上，本文结合上节给出的适航领域文本缺失词处理方法，将缺失词考虑加入来计算文本相似度，公式(3-4)可重写为公式(3.5):

(3.5)

在这个公式中，代表两个软件文本的相似度，代表文本使用词向量表示时的相似度，代表由文本辅助向量所计算出的相似度。两者之间依靠参数来确定。当越接近0的时候说明缺失词在文本相似度计算过程中越重要。在实际计算过程中可以将初始化为1，以步长为0.01递减，最终确定的值。因此，在不同的数据集中往往具有不同的值。可以看到这种计算文本相似的方式综合了词语语义和领域词汇的两种特征，算法的具体效果将在下一章中设置对比实验验证。

## 3.3 学习排序算法

在以往的需求可追踪链接研究中，通常只使用文本相似度一个特征作为判断软件文档间追踪关系的依据。而在目前的文本检索，自然语言处理等领域，使用机器学习方法结合文本中的多个特征，如词频、文档逆频率，文本长度等来对文本进行检索、分类已经在得到验证。因此，本文将学习排序应用到需求追踪任务当中。在上一节提到，通过计算每一条查询语句与候选文本间的相似度，可以产生一个排序列表，这个排序列表可以作为学习排序的输入。下面小节中将介绍学习排序算法和选取的特征。

### 3.3.1 排序算法选择

在第二章中提到，按照训练样本的不同，学习排序算法可以分为单文档排序、文档对排序和文档列表排序三种方法。其中文档列表排序每次训练对象是整个候选列表，在数据量足够的情况下，拥有最好的效果；而单文档方法由于只考虑了单个文本的特征，效果不如文档对方法。因此，综合效果和实验数据集的规模，本文选择了相对合适的文档对排序算法。 本文根据任务特点选择了针对信息检索领域的排序算法IR SVM，该算法是Ranking SVM在信息检索领域的改进。

#### 3.3.1.1 Ranking SVM

Ranking SVM排序算法主要分为将文档对的排序问题转化为分类问题和使用SVM算法分类模型训练并求解两个步骤。

假设给定查询-候选文档列表，并用一个维的特征向量来表示候选文档，然后输出空间用与查询的相关级别用 来表示，表示文档相关级别的数量，并且相关性级别按照递增排列。本节使用第二章中提到的排序函数将特征计算排序分数， 可以表示为公式(3.6) ，“”表示文档将相对位置关系。

 (3.6)

Herbich等人[51]在2000年首次提出可以将文档对排序问题转化为分类问题。其转化的推导可以假设排序函数为线性函数(可以为任意函数)。表示为公式(3.7)：

 (3.7)

其中是权重向量，该公式表示排序函数为权重向量和特征向量的内积。结合公式(3.8)，可以得到：

 (3.8)

可以由公式(3.8)看到，对于任一文档对，候选文档中 表示文档在文档的靠前位置。可以通过将文本特征向量重新组合为，结合公式(3.7)则可以得到两者相对位置的判别函数。根据线性函数的特性，可以得到每一对文档对相对位置的标签。接下来，可以依次求出候选文档列表中任意一组文档对组合的向量，并为其重新打标签。本文用和分别代表文档对中的第一个和第二个文档特征，两者间先后关系的标签用来表示，形式化的表示如公式(3.9)。

 (3.9)

这样就可以把给定的排序列表作为训练集，构建一个新的带有文档对顺序标签的数据集，见公式(3.10)。

 (3.10)



图15 原始排序问题

这里可以举例说明，假设候选文档与查询文档中共有三个相关等级，现在有两个查询分别对应两个候选文档列表，根据公式(3.7)，每个查询会计算得到一个权重向量，可以用图15形象表示。而本文使用文档对排序方法 ，根据公式(3.10)将文本特征向量两两重新组合为：、和，并给他们打上相应的标签，然后就可以将排序问题转化为了分类问题，如图16所示。



图16 排序问题转化为分类问题

本节使用作为分类算法的训练数据集，使用支持向量机(Support Vector Machine, SVM)作为分类算法，+1表示，为正样本，反之，-1为负样本。这样排序问题转化为0-1分类问题，根据SVM求解过程可以得(3.11)：

(3.11)

在这里是松弛变量，是训练集中实例数量，是第二范式，把松弛变量代入公式(3.12)中，可以得到：

(3.12)

在这里，公式加和的第一项为合页损失函数，第二项是防止过拟合而加入的正则项。

#### 3.3.1.2 IR SVM

Ranking SVM算法作为学习排序模型的排序算法应用于信息检索领域时[43]，有两点不足：

(1) 在上一小节提到，Ranking SVM排序算法的主要思想是将排序问题转化为分类问题。它在训练的时候把不同级别差值文档组，如和，其中 表示与查询相关的等级，同等对待，假如在训练实例中，越多，那么最终得到的结果也会越偏向于。而在信息检索领域的问题中，最终的结果列表中，文档排名越靠前，即r值越高，它对最终检索的结果影响越大，显然Ranking SVM在这里有待改进。

(2) Ranking SVM还有一点没有考虑的因素是，不同查询语句所对应的查询语句对所得的候选文档同等对待，这显示是不合理的。在这里可以举例说明：仍然假设文件的相关等级分为三个等级级别，现有两个查询，每个查询所对应的候选文档如表4所示，从中可以看到，对于结果中可提供正样本数量为14，多于可提供的8个，而由于Ranking SVM对不同查询语句所对应的查询语句对所得的候选文档同等对待，这也会影响最终检索结果。应该调整算法，使文档在相关等级相同的情况下，正样本多的在训练过程中应该对最终结果影响大于。

表4 排序列表实例

|  |  |
| --- | --- |
| **查询** | **结果** |
|  | 文档相关等级分别为：r3, r2, r2, r1, r1 |
|  | 文档相关等级分别为：r3, r2, r2, r1, r1, r1, r1 |

由于Ranking SVM算法的以上两个不足点，本文使用了它在信息检索领域的改进算法IR SVM，通过对它的优化Ranking SVM的损失函数，将0-1分类问题转化为代价敏感的二分类问题。具体来说就是，对于同一查询下的不同文档对，相关级别越高在最终排序结果中的重要性越大，这时候，对该文档对赋予较大的损失权重；对不同查询语句下的候选文档列表，为列表条目越少的文档对数赋予较大的损失权重。基于以上IR SVM改进损失函数可以表示为公式(3.13)。

(3.13)

在这里表示第个文档对的相关等级，表示等级下的权重值，为第个文档对所对应的查询，表示查询对应的相关文档对的参数。和作为惩罚因子，分别用来弥补上述Ranking SVM两个方面的不足。

对于参数值的确定，使用的是一种启发式算法：首先选定一种评价指标，如MAP, NDCG等。对于每条查询语句或者文档，首先找到它的最佳排序列表，然后随机挑选其中一对文档，交换两者的位置，记录下指标的变化值，重复该过程，直到所有文档对交换完成，最后使用评价指标变化的平均值作为参数的值。对于参数，它的主要作用是平衡候选文档列表条目数目所带来的影响，可以用最大候选文档条目和当前查询候选文档条目表示，可由公式(3.14)表示。

(3.14)

其中，代每条查询所对应的候选文档数量。于是可以简单的计算出来。

#### 3.3.1.3 求解IR SVM

令时，公式(3.13)可以等价变化为二次规划问题，如公式(3.15)。

(3.15)

|  |
| --- |
| **算法 1**: 坐标下降法 |
| 初始化为0,其中，令  迭代直至算法收敛： |

本文使用坐标下降法(Dual coordinate descent method)求解对偶问题来间接求解公式(3.15)的对偶问题从而得到它的最优解，首先引入拉格朗日引入拉格朗日乘子，然后对其求极大极小问题后，可以得到它的对偶形式，如公式(3.16)所示。

(3.16)

坐标下降法每次选择一个进行优化，记为迭代指定次数或目标函数值变化量达到指定精度时停止，其过程如算法1所示。

### 3.3.2 特征选择

在学习排序中，影响最后排序结果的另一个因素是文本特征的选择，在以往的研究工作中已经证明一些特征在信息检索领域的有效性。本文从常用的特征中选取五个特征作为排序算法输入。这五个特征中的前四个是依赖查询特征，这些特征强调了查询和候选文档间的关系，最后一个是不依赖查询特征，体现了候选文档本身的特点，主要作用是可以用来加快查询速度。

(1)文本相似度：表示查询查询文档与候选文档间语义的相似程度，可以由本文提出的相似度算法CWI计算得到。

(2)IDF之和：查询中词语在候选文档列表中的IDF值之和，它体现了查询在其对应的结果集的重要程度，可用公式(3.17)表示。

(3.17)

在这里代表查询文档中的单词，代表查询所对应的候选文档列表。

(3)广义Jaccard系数：通过文档向量表示查询文档和任一候选文档的上下文相似度，其计算方法由公式(3.18)所示。

(3.18)

这里代表查询文档的向量表示，是候选文档的向量表示，它们可以由公式(3.3)计算得出。

(4)缺失词数量：即在使用词向量表示文本时，缺失词的数量，体现缺失词语在查询中的重要程度。

(5)文本长度：表示查询候选文档列表的长度，学习排序中常用的不依赖查询特征，可以一定程度体现结果的有效程度。

## 3.4 本章小结

本章首先介绍了面向发动机适航的软件需求追踪算法的模型框架，然后介绍了基于词向量的文本相似度算法，算法包括了语义相似度的计算和缺失词语的动态处理，最后，介绍了机器学习排序将排序问题转化为0-1分类问题的原理，并给出了损失函数的优化和推导过程，结合当下的流行的机器学习技术，通过组合文本中的多个特征从而提升结果精度。

# 第四章 适航领域软件需求追踪算法模型验证

## 4.1 数据准备及预处理

软件生命周期数据的获取的过程如图17所示，其主要流程可描述为：



图17 软件生命周期数据获取

(1)数据收集和标注，由相关专业人员对软件生命周期数据的收集和标注，其中用到的软件生命周期数据包括软件需求文档，设计文档，代码文件等。

(2)脱密处理，对于航空发动机软件相关文档，有必要的话需进行脱密处理，表5列举了一些常用的脱密技术。其中为了更好的配合本文中提出的相似度计算方法，关键词可以表示字母加数字。

(3)文本处理，对于软件数据，可以使用一些需求抽取工具如信息抽取工具如SuperTracePlus[14]等来辅助相关人员行抽取，也可以编写脚本根据word文档的目录结构和标题对文本进行抽取，自动解析为结构化数据。

(4)最后需要专业人员对数据分类情况进行检查和修改，确认无误后，存入数据库。

表5 常用脱密方法

|  |  |
| --- | --- |
| **脱密方法** | **详情** |
| 彻底删除法 | 对一些工艺技术和关键数据，采用彻底删除的方法 |
| 模糊处理法 | 对于特定的装备，省去装备具体的名称，使用“xx”等或其他模糊名称；可使用虚数表示具体数字 |
| 替换取代法 | 将保密的某个装备表示为特定名词或符号 |
| 数据示意法 | 使用合理取值范围内其他数字表示装备性能参数或分数 |

这种数据获取方式可以应用于领域文本数据的获取，如航空发动机软件开发领域文档。但由于发动机控制系统软件的各种软件数据有很强的保密性，并不容易获得，同时软件需求间的可追溯性关系需要专业人员根据经验确定，在操作时有很多不便。因此，本文选取在需求追踪任务中常用的五个公开数据集来进行实验。选取这些数据集的原因除了克服上述适航领域数据的缺点之外，这些数据集中，每个单独的文件都是一个不能再进一步分解的数据。这些数据的类型与DO-178B/C所对应的高层需求、低层需求和源代码等所要求的类型相近，它满足实验模型的输入要求。其次这些数据集广泛在需求追踪任务中使用，有利于对比实验的设置，对比结果更加具有说服力，最后选择多个不同数据集的目的是为了说明本文所提出的算法可应用于不同的数据集。

本文选取的五个数据集分别是MODIS，CM1，eTOUR，EasyClinic和iTrust。这五个数据集有两个来源：MODIS是NASA公开项目中需求数据的一个子集，它由Hayes[14]等人整理，包含高层需求和低层需求两种文档，多次被用于需求间的可追溯性的研究当中。CM1同样来自于NASA的公开项目CM1，它也包含软件的高层需求和低层需求两类文档。eTOUR来自于一个专用于旅游的电子导航软件； EasyClinic来自于医疗管理的软件项目，两个项目均由Salerno大学开发，开发语言为Java；iTrust来自于一个用于记录医药信息的开源项目，开发语言同样是Java。这四个数据集均可从CoEST网站上得到。CoEST是一些研究学者为了鼓励和促进软件的可追溯性研究而建立的网站，在该网站上提供了多种已经整理好的软件数据集，这些数据集从实际的项目中整理而来，并经过多位专家学者验证，被广泛使用于软件可追溯性的研究当中。其中，DO-178B/C要求的软件需求的可追溯性所需数据，均可在这些数据集中找到。有一点需要指出的是，这些数据集中不仅包含软件的各种生命周期数据，还包括已经验证的可追踪链，这给本文的实验带来了极大的便利。实验数据集的描述见表6。

可以看到，在五个数据集中除了包括高层需求、低层需求、测试用例等不同类型文本文件，还有源代码等代码文件，除此之外，每个数据集中都提供了经由验证的可追踪关系结果集，并在单独文件中给出了软件制品间正确的追踪关系。

表6 实验数据集描述

|  |  |  |  |
| --- | --- | --- | --- |
| **数据集** | **描述** | **追踪链数量** | **文件总数** |
| MODIS | 19个高层需求，49个低层需求 | 41 | 68 |
| CM-1 | 2个高层需求，53个低层需求 | 45 | 75 |
| eTOUR | 58个用例，116个代码文件 | 308 | 174 |
| EasyClinic | 30个用例，20个UML交互图文件，  63个测试用例，47个代码文件 | 47 | 160 |
| iTrust | 131个用例，367个代码文件 | 534 | 498 |

选好实验数据集后，需要对数据集进行预处理以方面本文后面的实验使用。对于数据集的预处理使用传统的数据处理方法，对于每一个数据集：把数据集中所有文件作为输入，通过空格文本分割为单个单词，去除其中除了下划线和连字符之外所有的非数字、非字母符号，将所有单词转化为小写，并去除停用词，对于文本文件停用词是NLTK[48]所提供的停用词加上数据集中的冗余词，而代码文件中停用词是相应代码中的关键字(keywords)。

除了实验所需的软件相关数据集，由于计算文本相似度需要使用事先训练好的词向量，因此，词向量的训练需要一个大型并且权威的语料库，并且包含大量无标记的语料。对于语料库的选择，Siwei L.等人[49]研究并指出，训练词向量的最终质量与语料库的领域和大小相关，用领域相关语料训练效果比使用领域无关的训练效果要好；在同领域的情况下，训练所使用的语料库越大越好。因此，本文在选取语料时，选择了文本质量较高的维基百科词条作为训练语料。通过编写爬虫获得维基百科中技术分类下相关词条，解析维基百科网页，对网页信息进行清洗，从而获得的所有词条及其描述来获得领域相关的语料。其流程如图18所示。最终得到的语料库大小为326M。需要注意的一点是在处理维基百科词条时要尽量保证每个词条的完整性。Gensim是一个开源工具，它由python实现，集成了词向量模型，是一款很强大的工具。本实验中使用它来训练词向量，在训练过程中词向量的训练模型选择的Skip-gram，预测窗口的值设置为5，并将最终的向量的维度设置为200；在训练过程中选择开启softmax来加速训练速度。



图18 维基百科数据获取流程

## 4.2 实验设置

在第三章中，本文提出的面向航空发动机适航领域需求追踪算法可以分为两个阶段：文本相似度计算和学习排序模型训练。因此，本节针对每个阶段设置了对比实验，对于CWI算法，除了与计算文本相似度过程中所用到基于统计的VSM算法和只考虑词向量的WE算法，还和目前需求追踪领域较为成熟的LSI算法进行了对比实验，设置这组对比实验的目的是验证CWI算法在计算领域文本相似时的有效性。对于IR SVM，本文通过对比了使用排序算法前后的实验结果，最后对于整个算法模型，本文选择了与需求追踪领域国际领先的ENRL算法作为对比实验。下面详细介绍每组实验设置，实验对比方法如表7所示。

表7 设计的对比实验

|  |  |
| --- | --- |
| **实验** | **对比实验方法** |
| 实验一 | VSM、WE、LSI |
| 实验二 | IR SVM使用前后对比 |
| 实验三 | ENRL |

由于在训练好的词向量中，已经包含了单词的各种词形，因此不需词干化，但在使用VSM、LSI算法时对软件文本的处理略有不同，还需要对单词进行词干化。VSM和LSI具体算法已经在第二章介绍。这两个算法可以用来验证在的算法在考虑语义之后对最终结果的影响。

WE方法即利用公式(3.4)所计算出的语义相似度，这种方法直接忽略计算过程中的缺失词，使用这种方法是为了验证软件制品中缺失词对相似度计算的影响。

对于第二个实验，本节选择使用学习排序前后的实验结果进行对比，用以验证学习排序算法对需求追踪任务的影响。在上一章已经详细介绍了学习排序的原理及推导。实验中可以将查询所对应的候选文档列表中的每个文档当作一个实例，同时按照两两关系对文档打标签，两者相对位置关系正确记为+1，否则记为-1。然后对比实验前后的结果。

在第三个实验中，ENRL（Estimation of the Number of Remaining Links）方法是在文献21中算法的模型，它也是一种组合方法，首先它用不同的信息检索的方法计算出相似度，然后用机器学习分类方法。在这篇文章中，作者通过尝试12种信息检索方法和4种机器学习算法的不同组合并进行实验，最终对比所有实验结果从而得到了最优组合方法。他们论文中使用了两组与本文相同的数据集，eTOUR和EasyClinec，因此，可以直接与本文所提出模型进行对比，这也是本文选取这两组数据集的原因之一。进行该实验的主要目的是通过与目前较为领先的算法对比，验证本文中的算法在需求追踪的任务中具有一定的实用性。

本论文的实验所用机器为Intel i7八核处理器，操作系统为CentOS环境，运行内存为8G。

## 4.3 模型评价指标

为了评价不同算法的性能，本文中选取了精确率(precision)，召回率(recall)，F1分数(F1 score)，平均准确率均值(Mean Average Precision, MAP)，平均排序倒数(Mean Reciprocal Rank, MRR)和平均错误率(Mean Relative Error, MRE)多个评价指标。这些指标应用于不同算法中。首先，这里先定义一些变量：

TP(True Positive): 对于每个查询，算法所检索出的相关文档数目，通常可以表示可与查询建立追踪关系文档的数量。

FP(False Positive): 对于每个查询，算法所检索出的无关文档数目，即候选文档与查询无追踪关系，却被错误检索。

TN(True Negative):表示文档与查询不相关，也没有被算法所检索得到的文档数目。

FN(False Negative):表示文档与查询相关，却没有被算法检测出来的文档数目。

然后，给出在信息检索和学习排序中常用的评价指标定义：

精确率：检索算法得到的候选文档列表中与查询具有追踪关系的文档数量与候选文档总数的比值，计算方法见公式(4.1)。

(4.1)

召回率：检索算法得到的候选列表中与查询具有追踪关系的文档数量与该查询在数据集中有追踪关系的文档数量比值，计算公式如(4.2)。

(4.2)

F1分数：F1分数是对精确率和召回率的调和函数，它赋予精确率和召回率相等的权重，这时候F1值取得较好结果时，另两个指标数值也相对较好。

(4.3)

以上三个指标常用于软件需求可追踪性的文本相似度计算中。下面将介绍几个机器学习排序过程中常用的评价指标。

平均准确率(Average Precision): 单个查询所返回的结果列表是有序的，文档排名越靠前表示越相关，公式中表示文档列表中第个文档的排序级别：

(4.4)

MAP则是对于该集合所有查询的平均准确率均值。

(4.5)

MRR：是另一个常用的指标，与MAP不同的是MRR只关心第一个正确文档的位置。对于单个查询所对应的文档列表，找到其中第一个正确文档，并取倒数，最后对集合所有倒数取平均值。

(4.6)

相对误差(Relative Error)：可以用来预测评价预测模型的准确性，对于每条查询，其计算公式如下(4.7)：

(4.7)

MRE：对于数据集中所有查询，所有相对误差的均值，如公式(4.8)。

(4.8)

## 4.4 实验结果与分析

在第二小节中设置了三组对比实验来对算法模型进行评估，本节给出了实验结果，并对不同算法下的结果及其差异进行了分析。

### 4.4.1 文本相似度对比实验

本组实验选用了第二小节介绍的VSM、WE、LSI和本文所提出的文本相似度算法进行了对比实验，此次实验所使用的数据集为第一节所介绍的全部五组数据集，详细描述如表6所示。为了简便表示，可以将所使用数据集中数据名称的首字母大写代替，即HL表示高层需求文档(high-level requirements)，LL表示低层需求文档(low-level requirements)，UC表示用例文档(user cases)，CC表示代码文档(code classes)，ID表示UML交互图文档(interaction diagrams)，TC表示测试用例文档(test cases)。实验结果见表8， “->”表示可追溯性链，PRE代表precision，REC代表recall。为了更好的展示最终结果，本文计算了不同算法下的F1值，其直方图如图19所示。

表8 四种算法的precison和recall值

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| DataSets\Metrics | |  | **VSM** |  | **WE** |  | **CWI** |  | **LSI** |  |
|  | |  | **PRE** | **REC** | **PRE** | **REC** | **PRE** | **REC** | **PRE** | **REC** |
| MODIS | HL->LL | | 0.215 | 0.288 | 0.319 | 0.428 | 0.315 | 0.563 | 0.279 | 0.357 |
| CM1 | HL->LL | | 0.158 | 0.187 | 0.249 | 0.226 | 0.413 | 0.396 | 0.127 | 0.41 |
| eTOUR | UC->CC | | 0.07 | 0.301 | 0.104 | 0.302 | 0.135 | 0.563 | 0.077 | 0.221 |
| iTrust | UC->CC | | 0.006 | 0.245 | 0.227 | 0.404 | 0.243 | 0.422 | 0.009 | 0.45 |
| EasyClinic | UC->ID | | 0.185 | 0.764 | 0.324 | 0.732 | 0.342 | 0.806 | 0.259 | 0.833 |
| UC->TC | | 0.36 | 0.685 | 0.542 | 0.832 | 0.553 | 0.865 | 0.45 | 0.755 |
| UC->CC | | 0.208 | 0.487 | 0.215 | 0.677 | 0.247 | 0.764 | 0.317 | 0.503 |

这里将实验结果分为两组进行对比，第一组是VSM、WE和CWI方法间的对比，可以从表8及图19中看到，CWI对比其他两个算法精确率，召回率及F1值均有一定提升；由第三章可以知道CWI是在计算文本相似度时，既考虑了语义相似度，又考虑了在计算 语义相似度过程中的缺失词，通过CWI与这两个算法单独作用的对比。可以充分说明，本文提出的CWI算法由于考虑了文本词语的特征。

第二组的对比是WE、CWI算法和目前需求追踪领域成熟的LSI算法进行比较发现CWI算法总是比直接使用词向量的WE效果要好，这是因为当为1时，两者等价。即WE算法是CWI的下限。可以看WE、CWI算法在各个指标上对比LSI均有所提高。产生这种结果的一个主要原因是LSI算法在计算文本相似度时，将文本映射到一个较低维的空间，具有相同主题的词语被映射到同一维度，而这种变幻是通过数学方法得到，并没有考虑词语的语义特性，忽略了词语之间的关系，CWI方法恰好弥补了这一缺点。

图19 不同算法下的F1值

综合以上两组对比，可以看到CWI算法在计算文本相似度时，既考虑了语义的相似度，同时结合领域文本的特性，在运行过程有效的处理了文本数据中缺失词，在计算了文本相似度时具有一定优势。

### 4.4.2 学习排序算法效果对比实验

在本文算法的法第二阶段，应用了学习排序算法来提升最终的实验结果，以获得更多正确的可追踪性链接。因此，本组实验通过对在应用学习排序(IR SVM)算法前后的结果进行对比。在这组实验中采用了搜索排序中常用的两个评价指标MAP和MRR。在本次实验中同样采用全部所选用的五组数据集，考虑到实验数据集的相对规模，本次实验选择八折交叉验证方法，最终实验结果如图20、21所示。

由直方图可以直观的看出，相比于使用学习排序算法前，在使用学习排序算法后，MAP和MRR两个评价指标均有一定的提升。MAP关注的对于数据集中的候选文档的平均错误率，MAP的提升说明了使用IR SVM后候选文档的出现错误的概率降低；而MRR关注的是排序列表中第一个正确文档出现的位置，MRR越高说明正确文档出现的位置越靠前，也能说明使用学习排序算法后，排序列表中正确文档第一次出现位置更靠前，也表明最终生成的排序列表的顺序更合理。

图20 使用IR SVM前后MAP对比

图21 使用IR SVM前后MRR对比

这组实验说明，尽管目前主流的需求追踪算法大部分基于文档间的文本相似度，但是这仅仅考虑其中的一个特征，随着近年来机器学习算法的发展，将更多的文本特征与机器学习算法相结合可以一定程度帮助在可追踪性任务中找到更多，更准确的需求追踪链。

### 4.4.3 与ENRL方法对比

在上一小节已经提到，ENRL算法也是一种组合算法，它通过将12种不同的信息检索方法(VSM, LSI等)和4种不同的机器学习方法进行组合，最终给出了针对每个数据集的最优组合。为了与其实验设置保持一致，本次实验采用了eTOUR和EasyClinic两组数据集进行实验，并采用MRE作为衡量指标。表9为ENRL算法给出的在eTOUR和EasyClinic取得最优值的算法组合。

表9中，stop-stem表示单词的词干化，TF即词频，LSI100的意思是使用LSI算法，并将主题数量设置为100，文本间相似度通过cosine来计算。IBK是一个基于k-邻近算法的分类器。

表9 ENRL算法在eTOUR和EasyClinic上的最优组合方法

|  |  |  |
| --- | --- | --- |
| **数据集** | **NLP方法** | **机器学习分类算法** |
| eTOUR | stop-stem, TF, LSI100, cosine | Ibk |
| EasyClinic | stop-stem, TF, VSM, cosine | Ibk |

最后的实验结果如表10所示。从表中可以看出，在使用相同的数据集和相同的评价指标的情况下，本文提出的方法取得了更小的平均相对误差，这说明该方法在最终的结果比ENRL方法出现了更少的错误，即在需求追踪任务的最终结果有一定程度的提升。通过和ENRL进行对比，说明了本文所提出的需求追踪算法与目前较为先进的需求追踪算法，仍能取得较好的结果。

表10 与ENRL算法的MRE值对比

|  |  |  |
| --- | --- | --- |
| **方法**  **数据集** | **ENRL** | **Our method** |
| eTour | 0.03 | 0.022 |
| EasyClinic | 0.01 | 0.0052 |

## 4.5本章小节

在本章中，本章设置了三组对比实验来验证本文所提出的模型，首先介绍了软件生命周期数据的获取方法和实验所用数据集来源及预处理。然后介绍了对实验的设置和所选的评价指标等，对比实验从面向航空发动机的软件需求可追踪性模型算入手，为模型的每个阶段设计了对比实验，首先是文本相似度计算阶段，然后是学习排序阶段的两组对比实验，来验证每个阶段对实验结果的提升；最后，本文提出的算法和国际领先的ENRL算法进行了对比，说明该算法在目前的需求追踪任务中可以取得比较好的效果。

# 第五章 需求追踪算法原型系统的设计与实现

第三章介绍了软件需求追踪算法的整体框架和关键步骤的算法实现，本章基于本文第三章所提出的算法模型，结合航空发动机领域数据的实际情况，实现了面向航空发动机控制系统的软件需求追踪算法原型系统。其中主要包括文档录入，数据存储和管理，数据预处理模块，中文文本处理，文本相似度计算和学习排序算法模型模块,同时提供了需求追踪目标的满足性结果导出等模块。本章按照系统需求分析，总体设计和具体模块的设计与实现的顺序对进行了介绍，并在最后进行了系统界面展示。

## 5.1 需求追踪算法原型系统的需求分析

首先，面向航空发动机控制系统的软件需求追踪算法原型系统是一个B/S架构的系统，它是一个可以根据适航审定人员所指定的软件生命周期相关数据，自动计算其中的软件需求间的追踪关系，并展示需求可追溯性适航符合性检查结果的辅助系统。该系统的主要功能需求包括：文档数据的输入，数据预处理，用户权限管理和需求可追踪性模型计算模块，最后提供结果的下载。

1. 数据输入

数据输入提供两种方式：系统录入和word数据录入。系统录入由用户手动输入，录入时分为软件项目数据，生命周期数据类型，软件制品数据；word数据导入则是直接将有软件生命周期数据导入系统，系统对word文件进行处理，抽取文档内容，最后需要用户手动确认信息。用户录入的信息将保存到数据库。其用例图如图22所示。



图22 数据输入用例图

1. 数据预处理

对于已经存好的数据类型，首先要对数据预处理，包括对于中文文档进行分词和翻译。对于已处理好的词汇做相关处理，包括去除停用词，去除非法字符，单词转化为小写等步骤，同时，在预处理时同时进行计算词语的TF-IDF值，确定每个数据集的缺失词库等操作。

1. 文本相似度的计算

文本相似度的计算是需求追踪算法的基础，它可以通过word embeddings和缺失词共同来计算数据集中任意两个文本的相似度，这是判断文本间是否具有追踪关系的基础。

1. 需求间的可追溯关系可视化表示及结果下载

系统需要能够给出DO-178B/C所要求的可追溯性结果进行可视化展示，一个可选的展示方式是给出系统需求、高层需求、低层需求间三个需求间的追踪矩阵，可以自动判断需求间的追踪关系，并提供最后结果的下载。用例图如23所示。



图23 功能模块用例图

1. 权限控制

对于该应用系统实现权限管理是必要的，系统通过为用户分配的不同角色，使用户具有不同权限，从而保证数据的安全访问。

## 5.2 系统总体设计

### 5.2.1 系统总体架构设计

本节主要介绍面向航空发动机控制软件的需求追踪算法原型系统总体架构，通过第一小节中对系统需求的分析可以知道该系统首先需要对数据进行存储并进行管理，然后，数据的预处理模块，可追溯性计算模块两个模块作为服务层，最后需要提供一个web界面来进行数据交互界面和结果展示。因此，该系统的总体架构可以划分为数据存储层，服务层，数据交互层三层，其整体架构如图24所示。

从图中可以看出在数据存储层为整个系统提供了数据支持，本地数据库中存储的数据包括软件生命周期数据，从维基百科语料训练而来的词向量数据，还有经过处理得到的缺失词汇数据。服务层为数据的预处理和计算等模块，它是整个需求追踪系统的核心功能。数据交互层则提供与用户交互页面和结果展示页面。其中，数据交互层的业务逻辑使用PHP实现，服务层数据预处理和计算部分使用python实现，两者间的交互是通过PHP通过exec函数调用相应的python脚本实现。



图24 系统架构总体设计图

### 5.2.2 系统数据库设计

根据系统数据库中数据的来源不同，可以将数据表分为三类：预置表、软件制品相关表和实验中间表。其中预置表包括自定义词典表和预置词向量表，自定义词典表存储词典的英汉数据，可以用来辅助中文的翻译过程，而词向量表则存储由自己爬取的维基百科词条所获得语料训练而来的软件领域词向量，避免每次从二进制文件读取数据，加快了检索和计算速度。软件制品相关表则包括软件项目数据、软件生命周期分类数据和软件制品数据，用来存储软件相关的数据。实验中间表则包括中文软件文档分词、翻译后的存储表，缺失词向量表，文件特征表，计算结果的临时储存表等。数据库表间关系如图25所示。



图25 系统数据库设计图

其中，预置表包括word\_vector、dictionary表；软件制品相关表包括project、lifecycle和software\_artifact表；实验中间表包括trans、feature、oov、vsm和similarity表。对于各个数据表的详细介绍如下：

（1）word\_vector：词向量表，存储预先训练好的词向量，包括单词编号，单词及单词所映射成的词向量，其中词向量维数为200；

（2）dictionary：用户自定义词典表，存储事的事先从发动机词典上通过有道词典OCR接口获取的领域词典，其中zh\_word是中文单词，en\_word是对应的英文单词；

（3）project：项目信息表，用来存储软件项目基本信息，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、高层需求、低层需求和源代码等等；

（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）trans：翻译表，用来存储中文的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）feature：文本特征表，用来存储文本中各种特征，比如各个词的权重值，该值可以用TFIDF表示；

（8）oov：缺失词汇表，存储通过软件制品数据中无法从词向量表中的缺失词；

（9）auxVector：辅助向量表，用来记录缺失词汇向量空间向量表，存储每个软件制品中缺失词的用向量空间模型所表示的文本向量；

（10）similarity：相似度表，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 主要功能模块介绍

系统整体模块结构图如图26所示，系统主要分为数据管理模块，计算模块和用户管理模块三个模块。



图26 系统功能模块图

### 5.3.1 数据管理模块

后台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领域词典数据管理四部分，主要功能是实现每个模块对数据的增删改四个基本管理功能。

软件制品数据管理模块中数据的录入包括用户直接在web页面录入和将word文档导入系统两种形式。与用户直接录入数据不同的是，从word文档导入的数据需要使用脚本对其解析，该系统中使用python的docx库对本文进行解析。在这里，需要对文档的格式做一些要求，包括标题命名及标题的层级关系等。通过解析不同层级标题映射到相应的需求等级，正文则作为具体描述，可以是代码或纯文本。

在确认上传之前，通常需要用户（通常是有航空软件研发背景的专业人员）对所要上传数据进行补充和修改和确认，当没有问题后选择确认，完成上传。在这里，与数据库交互的业务逻辑由ThinkPHP提供的库实现。通过word导入文档的界面如图27所示。

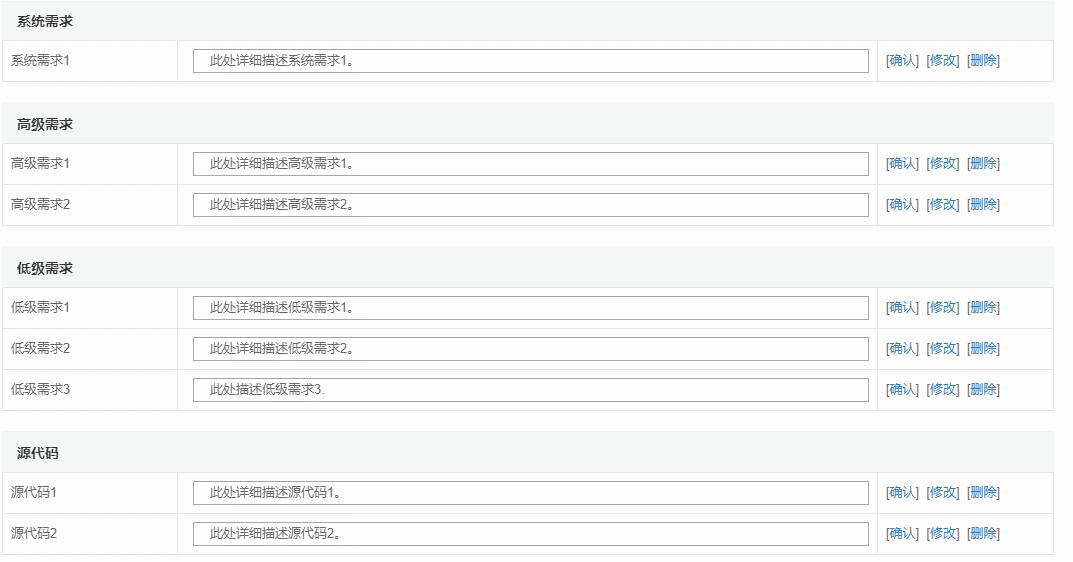


图27 用户对录入的word确认界面

领域词典数据管理模块主要是对数据词典进行管理，主要的功能是领域数据字典的录入。对于领域数据字典的录入，可以分为以下几步：(1)首先，通过有道词典提供的OCR开放接口API识别出《汉英航空发动机工程技术词典》电子版本字符，并临时保存；(2)格式化输出第(1)步识别出的字符，按照中文词汇—英文单词对的形式输出；(3)去除非法字符，并将词汇对存入数据库的dictionary表。

### 5.3.2 计算模块

计算模块包括对中文文本的分词及翻译、数据预处理、文本相似度计算、需求可追踪关系计算和可追溯性目标符合性判断五个部分，算法由python实现，类图如28所示。



图28 计算模块UML类图

从图18中可以看到，涉及需求追踪算法主要在Process类实现，其中包括了文本语义相似度计算算法、学习排序算法IR SVM，它是计算模块实现的核心，Calculate类、Audit类使用Process类实现文本相似度计算功能和需求追踪符合性审查功能，ProcessTuple类和ProcessLayer类和都继承自Process类，分别用来完成文档语义相似度计算和追踪关系判定和需求追踪关系的计算，而Translation类、RemoveStopwords类、TFIDFCALC类、OOVCALC类和VSMCALC类分别用于分词并翻译、去停用词、计算tfidf值、缺失词处理，文本的向量空间模型表示等过程，它们主要用于该系统的文本预处理过程。

考虑到系统的数据规模和程序性能，该系统中将计算过程中将几个关键步骤独立封装成单独的类，包括去停用词、分词和翻译、计算权重、缺失词处理、文本的向量空间模型表示和计算文本相似度等步骤，这样做一方面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，另一方面各个类模块相互独立，便于系统以后的维护和修改。

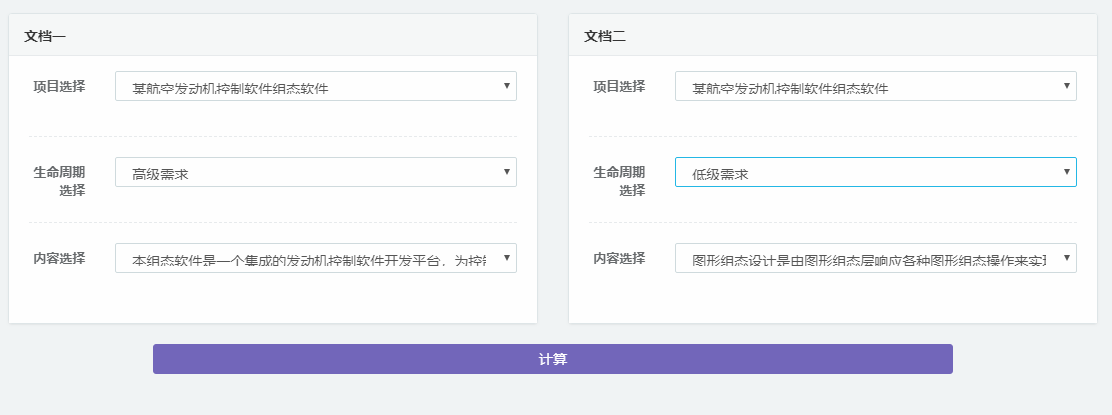


图29 文本相似度界面

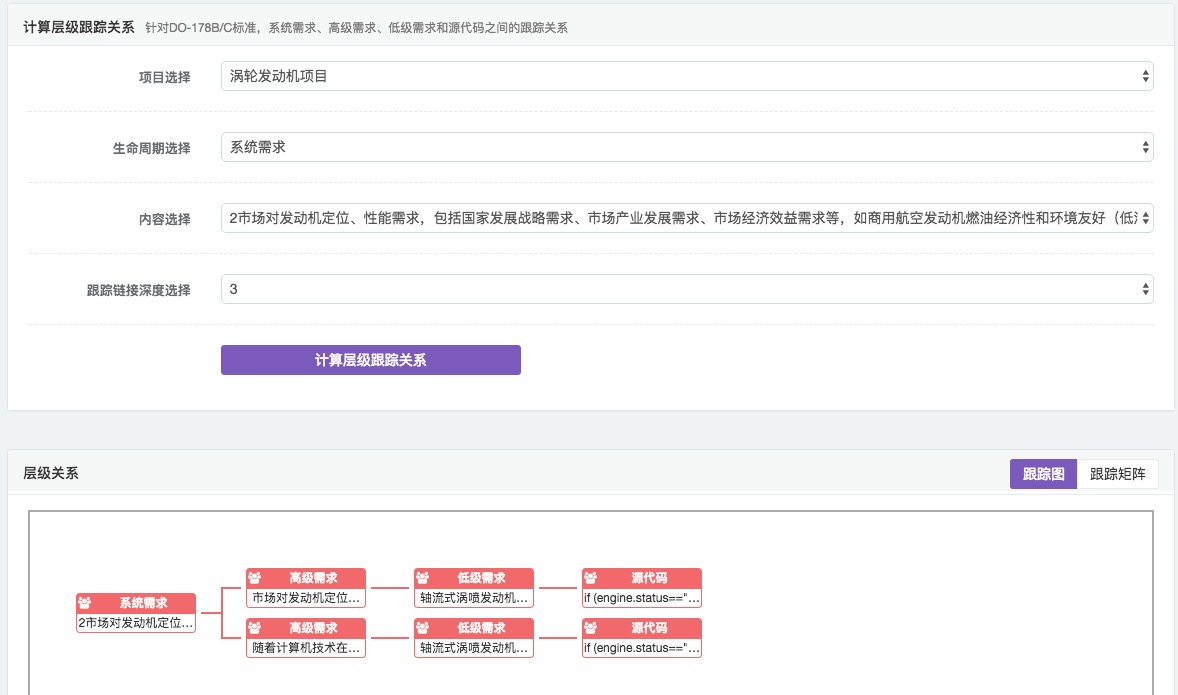


图30 追踪关系结果界面

事先计算好的词向量保存在数据库中供文本相似度的计算和缺失词处理时使用，使用numpy库对向量进行初始化，算法实现语言为python。其中系统业务逻辑部分使用PHP语言通过其exec函数执行已经写好的python脚本，并将结果返回给用户界面。在计算时，使用ajax异步提交数据，刷新表单，减少了页面刷新次数，使用更加流畅。在计算模块中涉及到的文本相似度计算、追踪关系建立和生成的适航审定检查单分别如图29、图30、图31所示。文本相似度计算给出了两个文档需求追踪关系的参考结果，追踪关系建立给出了追踪图和追踪矩阵两种形式结果，可追溯性目标符合性目标判定给出了统计结果和检查单下载。

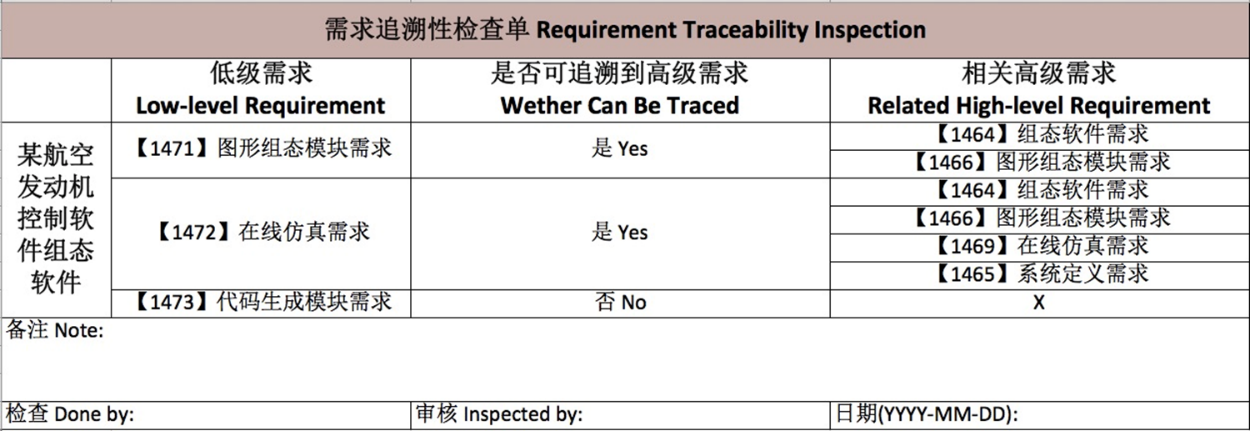


图31 生成的检查单

### 5.3.3 用户管理模块

用户管理模块包括两个部分：用户基本信息管理和权限管理两个部分。用户基本信息管理主要是对用户基本信息的管理，权限管理则是对不同级别用户分配不同权限，其中系统超级用户可以管理其他用户，并为普通用户角色分配权限。



图32 RBAC类图

ThinkPHP为使用者提供了两种权限控制方式：AUTH和RBAC。在本系统中，我们使用RABC模块实现，系统中用户被分为超级用户和普通用户两种角色，系统通过为用户分配不同的角色来管理用户所获得的权限。可以看到，角色是用户和权限相互之间关联的桥梁。三者间的关系如图32所示。为了避免用户的多次登陆，系统中使用了ThinkPHP所提供的session管理框架对浏览器session进行管理。当用户登陆时，使用ThinPHP提供的session框架会保存当前用户登陆信息，直到session失效，在session有效期间，用户角色的权限判断时通过session中权限表进行判断的，这样省去了查询数据库时间，加快了访问速度。用户管理界面如图33所示。



图33 用户管理界面

## 5.4 开发环境

系统的后台逻辑代码使用php实现，而文本数据处理和计算模块使用强大的python实现，该系统的运行在CentOS环境。系统开发环境在表11中详细列举。

表11 开发环境详细信息

|  |  |  |
| --- | --- | --- |
| **语言及工具** | **版本** | **说明** |
| PHP | 5.4.16 | 业务逻辑开发语言 |
| ThinkPHP | 3.2.3 | 后台开发框架 |
| Python | 2.7.5 | 文本处理和计算部分语言 |
| Nltk | 3.2.5 | 自然语言处理工具 |
| Gensim | 3.2.0 | 自然语言处理工具 |
| Mysql | 5.6.38 | 数据库 |
| Youdao ocr服务接口 | 无 | OCR API |
| Youdao 自然语言翻译接口 | 无 | 中英文短语翻译API |

## 5.5 本章小结

本章主要介绍了需求的可追踪性算法原型系统的设计与实现，结合实际情况给出了该系统的需求分析，系统总体架构设计，数据库设计和子模块的具体实现并给出了系统界面的效果展示，主要包括数据管理，相似度计算界面，可追踪关系建立界面，可追溯性的适航目标检查单及用户管理几个界面。本章的最后给出了该系统开发的详细环境。

# 总结与展望

## 总结

在我国航空事业的不断发展的背景下，对航空发动机的要求也越来越高，其控制系统也是越来越复杂，如何开发半自动或自动化的软件适航审查工具辅助软件系统的适航审定过程是一项很有意义的课题。本文以DO178B/C所要求的软件制品间可追溯性目标为切入点，提出了基于词向量和机器学习排序的需求追踪算法。本文通过分析以前基于信息检索方法的不足，主要在于未考虑词语的语义信息，因而从语义角度出发，结合适航领域文本的专业词汇较多特点来计算文本相似度，并通过使用文本文档其他特征作为机器学习算法的输入，使得最终精确率的提高。在适航审定过程中，使用本文提供的算法，可以一定程度减轻审查人员在构建或恢复软件制品的可追踪链接时的工作量也更不容易出错。在本文中，主要完成的工作包括：

（1）首先是对当前国内外适航体系的研究，并分析了基于DO178B/C的软件与普通软件开发过程的不同，软件生命周期数据及适航符合性目标等。这些是需求可追溯性验证的轮依据。

（2）分析了适航领域需求数据的特点，在计算可追踪性时，为了避免以往在计算文本相似度的时将单词当作为独立的单元，而造成词汇鸿沟的现象，引入词向量来计算文本相似度，同时结合适航领域文本特点，改进了文本相似度算法，使其可以在计算过程中处理缺失词，方便高效。并通过多组对比实验验证了该算法。

（3）考虑到文本文件，除了文本相似度，还有多个其它特征，将信息检索领域流行的机器学习排序算法IR SVM引入，并通过坐标下降算法加速损失函数求解过程。该算法通过对比实验前后的结果，验证机器学习排序的有效性。

（4）结合适航需求的特点，用开源数据集代替适航数据，并阐述原因，设置了三组对比实验，首先对于算法模型的不同阶段进行了验证，并通过与目前新颖的ENRL在相同实验条件的对比，从整个算法模型局部到整体的一步一步验证，从而整个需求追踪模型的有效性和实用性。

（5）基于所提出的算法，实现了适航领域的需求可追踪算法原型系统。主要包括数据的交互，数据的处理，需求间相似度的计算，需求间追踪关系的确定和可追溯性目标满足性的判断。同时作为应用系统实现了基于角色的用户权限管理功能和基本的数据管理功能。

## 展望

本文所提出的适航审定需求追踪算法仍有一些待改进和完善的空间：

首先，本文在使用词向量计算语义相似度时，尽管考虑到了语义特性，但仍是基于一种“词袋”模型，在计算时，并未考虑词语间的顺序关系，这种算法有一定的局限性，如何改进使用词向量计算文本相似度的方法有待改进。

其次，由于适航领域文本内容比较复杂，对于一些特定类型文档，如源代码文档。找到这些文档合适的预处理工作，可以有效的对于文本进行建模。

然后，在使用机器学习排序过程中，我们使用了一些普遍有效的特征作为训练特征。挖掘软件工程领域文档中更多的特征组合，从未提升最终的模型效果。

最后，目前国内的相关研究较少，对需求可追溯性研究所提供的数据更少。因此从相关部门获得并整理软件数据，进一步结合软件适航审定标准，也是这一研究重要的进一步深入方向。

# 参考文献

1. 中国民用航空总局航空器适航司. 中国民用航空器适航管理[M]. 中国民航出版社, 1994.
2. 樊思齐. 航空发动机控制[M]. 西北工业大学出版社 [等], 2008.
3. 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.
4. RTCA (Firm). SC 167. Software considerations in airborne systems and equipment certification [M]. RTCA, Incorporated, 2012.
5. Cleland-Huang, Jane, et al. "Software traceability: trends and future directions." on Future of Software Engineering, Fose 2014, Hyderabad, India, May 31 - June 2014:55-69
6. 钟佳涛, 张茂林. 基于静态与动态技术的需求追踪模型的研究[C]// 2008全国软件与应用学术会议. 2009.
7. 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.
8. 石斌. 航空发动机高可靠性FADEC软件系统技术研究[D]. 西北工业大学, 2004.
9. 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.
10. Settimi R, Cleland-Huang J, Khadra O B, et al. Supporting Software Evolution through Dynamically Retrieving Traces to UML Artifacts[C]// Principles of Software Evolution, International Workshop. IEEE Computer Society, 2004:49-54.
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. Dag J N O, Regnell B, Carlshamre P, et al. A Feasibility Study of Automated Natural Language Requirements Analysis in Market-Driven Development[J]. Requirements Engineering, 2002, 7(1):20-33.
13. Antoniol G, Canfora G, Casazza G, et al. Recovering traceability links between code and documentation[C]// International Conference on Software Maintenance. IEEE Computer Society, 2000:40.
14. 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.
15. Lormans M, Van Deursen A. Can LSI help reconstructing requirements traceability in design and test?[C]//CSMR. 2006, 6: 47-56.
16. Asuncion H U, Asuncion A U, Taylor R N. Software traceability with topic modeling[C]// ACM/IEEE International Conference on Software Engineering. ACM, 2010:95-104.
17. 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.
18. Sultanov H, Hayes J H, Kong W K. Application of swarm techniques to requirements tracing.[M]// Bildverarbeitung für die Medizin 2003. Springer Berlin Heidelberg, 2003:1287–1292.
19. Lucia A D, Penta M D, Oliveto R, et al. Applying a smoothing filter to improve IR-based traceability recovery processes: An empirical investigation ☆[J]. Information & Software Technology, 2013, 55(4):741-754.
20. 邓志成. 基于本体的动态需求跟踪[D]. 湖南大学, 2015.
21. 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.
22. Hayes J H, Dekhtyar A, Sundaram S K, et al. REquirements TRacing On target (RETRO): improving software maintenance through traceability recovery[J]. Innovations in Systems & Software Engineering, 2007, 3(3):193-202.
23. Ali N, Antoniol G. Trust-Based Requirements Traceability[C]// IEEE, International Conference on Program Comprehension. IEEE Computer Society, 2011:111-120.
24. Keenan E, Czauderna A, Leach G, et al. TraceLab: An experimental workbench for equipping researchers to innovate, synthesize, and comparatively evaluate traceability solutions[C]// International Conference on Software Engineering. IEEE Press, 2012:1375-1378.
25. Mahmoud A, Niu N. TraCter: A tool for candidate traceability link clustering[C]// Requirements Engineering Conference. IEEE, 2011:335-336.
26. 黄志球，徐丙凤，阚双龙等. 嵌入式机载软件安全性分析标准、方法及工具研究综述[J]. 软件学报. 2014. 25（2）: 200-218
27. AC 20-115B, RTCA,Inc Document RTCA/DO-178B[S],FAA. 1993
28. AC 20-115C,AirBorne Software Assurance[S], FAA. 2013
29. DO-330, Software Tool Qualification Considerations[S]. Washington, DC: RTCA，2011
30. DO-331, Model-Based Development and Verification Supplement to DO-178C and DO-278A[S]. Washington, DC: RTCA，2011
31. DO-332, Object-Oriented Technology and Related Techniques Supplement to DO-178C and DO-278A[S]. Washington, DC: RTCA，2011
32. DO-333, Formal Methods Supplement to DO-178C and DO-278A[S]. Washington, DC: RTCA，2011
33. Salton G, Wong A, Yang C S. A vector space model for automatic indexing[M]. ACM, 1974.
34. 陆玉昌，鲁明羽，李凡等．向量空间中单词权重函数的分析和构造[J]．计算机研究和发展，2002，39(10)：1205~1210
35. Bengio Y, Ducharme R, jean, et al. A neural probabiistic language model[J]. Journal of Machine Learning Research, 2003,3(6):1137-1155
36. Rumerlhar D E. Learning representation by back-propagating errors[J]. Nature, 1986, 323(3):533-536.
37. Mikolov T, Karafiát M, Burget L, et al. Recurrent neural network based language model[C]// INTERSPEECH 2010, Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September. DBLP, 2010:1045-1048.
38. Mikolov T, Corrado G, Chen K, et al. Efficient Estimation of Word Representations in Vector Space[C]// International Conference on Learning Representations. 2013:1-12.
39. Mikolov T, Sutskever I, Chen K, et al. Distributed representations of words and phrases and their compositionality[C]// International Conference on Neural Information Processing Systems. Curran Associates Inc. 2013:3111-3119.
40. Kusner, Matt J., et al. "From word embeddings to document distances." International Conference on International Conference on Machine Learning JMLR.org, 2015:957-966.
41. Kenter, Tom, and M. D. Rijke. "Short Text Similarity with Word Embeddings." ACM International on Conference on Information and Knowledge Management ACM, 2015:1411-1420.
42. Ye X, Shen H, Ma X, et al. From Word Embeddings to Document Similarities for Improved Information Retrieval in Software Engineering[J]. 2016:404-415.
43. 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.
44. Levitt, Jeremy I. "Learning to Rank Improves IR in SE." IEEE International Conference on Software Maintenance and Evolution IEEE Computer Society, 2014:441-445.
45. Niu H, Keivanloo I, Zou Y. Learning to rank code examples for code search engines[J]. Empirical Software Engineering, 2017, 22(1): 259-291.
46. Dhingra B, Liu H, Salakhutdinov R, et al. A Comparative Study of Word Embeddings for Reading Comprehension[J]. 2017.
47. Pinter Y, Guthrie R, Eisenstein J. Mimicking Word Embeddings using Subword RNNs[J]. 2017.
48. Xue N, Bird E. Natural Language Processing with Python[J]. Natural Language Engineering, 2011, 17(3): 419.
49. Lai S, Liu K, He S, et al. How to Generate a Good Word Embedding[J]. IEEE Intelligent Systems, 2015, 31(6):5-14.
50. 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.
51. Herbrich R. Large margin rank boundaries for ordinal regression[J]. Advances in Large Margin Classifiers, 2000, 88.
52. Yuan, Tian, et al. "Learning to rank for bug report assignee recommendation." IEEE, International Conference on Program Comprehension IEEE, 2016:1-10.
53. Ling, Wang, et al. "Two/Too Simple Adaptations of Word2Vec for Syntax Problems." Conference of the North American Chapter of the Association for Computational Linguistics – Human Language Technologies 2015.
54. Eder, Sebastian, et al. "Configuring Latent Semantic Indexing for Requirements Tracing." International Workshop on Requirements Engineering and Testing IEEE Press, 2015:27-33.
55. Hsieh C J, Chang K W, Lin C J, et al. A dual coordinate descent method for large-scale linear SVM[C]// Proc. International Conference on Machine Learning. ACM, 2008:408-415.
56. Le, Tien Duy B., et al. "A learning-to-rank based fault localization approach using likely invariants." International Symposium on Software Testing and Analysis ACM, 2016:177-188.
57. Kong, Wei Keat, et al. "How do we trace requirements: an initial study of analyst behavior in trace validation tasks." International Workshop on Cooperative and Human Aspects of Software Engineering ACM, 2011:32-39.
58. Antoniol G, Cleland-Huang J, Hayes J H, et al. Grand Challenges of Traceability: The Next Ten Years[J]. 2017.
59. 沈小明. 机载软件研制流程最佳实践[M]. 上海交通大学出版社, 2013

# 攻读硕士学位期间取得的学术成果

1. Qingsong Tian, Qinghua Cao, Qing Sun. Adapting Word Embeddings to Traceability Recovery[C]//2018 3rd International Conference on Communication Systems, Computing and IT Application IEEE.

# 致 谢

本文的研究工作是在我的导师曹庆华教授的悉心指导下完成的。为了学生能顺利完稿，曹老师总是在繁忙的科研工作中抽出宝贵的时间，给予及时细致的指点。从刚拜入曹老师门下直到现在即将完成学位攻读之际，老师广袤渊博的视野和知识，严谨求是的科研素养，兴趣自由的学术态度，始终感染着我们。曹老师指引我的不仅有创新的思维和思想，还有科学有效的方法论、积极平和的人生态度和实实在在的为人风范，虽言语不足以表达万一，感谢曹老师在硕士三年间的谆谆教诲和深切影响！

衷心感谢孙青老师在这三年来对我科研与生活的关心和帮助，孙老师为人热心，专业知识扎实，工作认真负责，在研究内容尚未清晰时，孙老师耐心与我探讨，帮我找到思路，再到论文投递前，孙老师耐心指出我论文中的不足之处，让我受益匪浅，牢牢铭记。再次感谢孙老师的无私帮助，祝孙老师身体健康、家庭美满、工作顺利。

特别感谢实验室刘云师兄、庞静雯师姐在校期间对学习生活的不断鼓励和帮助及在论文写作过程中为我牺牲的时间和精力；感谢一起学习的赵腾、王鑫冶、苑铎同学在实验室营造良好的学习生活氛围，祝愿大家学业有成、工作顺利、前程似锦；感谢谭特、吕梦雷和黄亮同学在论文写作过程的无私帮助。

感恩我辛劳的父母，在大学以及人生每个阶段对我提供的关心和帮助。

学业有期，学海无涯。我将用所学到的知识和方法，积极面对将来道路上的各种问题。