南京大學

研 究 生 毕 业 论 文
（申请工程硕士学位）

论　文　题　目　基于句法树的关键词文本扩增技术

作　者　姓　名　王擎宇

学科、专业名称　工程硕士（软件工程领域）

研　究　方　向　软件工程

指　导　教　师　陈振宇 教授

2022 年 **5** 月 **25** 日

学　　号：**MF20320161**

论文答辩日期：**2022** 年 **5** 月 **20** 日

指 导 教 师：　　（签字）

Keyword Text Data Augmentation Technology Based on Syntax Tree

by

Wang Qingyu

Supervised by

Professor Zhenyu Chen

A dissertation submitted to
the graduate school of Nanjing University
in partial fulfilment of the requirements for the degree of
Master of Engineering
in
Software Engineering

Software Institute
Nanjing University

May 25, 2022

南京大学研究生毕业论文中文摘要首页用纸

毕业论文题目：基于句法树的关键词文本扩增技术

工程硕士（软件工程领域）专业 2020 级硕士生姓名：王擎宇
指导教师（姓名、职称）：陈振宇 教授

摘　要

随着互联网上文本数据的爆炸增长,海量信息给当前的科技公司和社会带来了很大的挑战。如何准确地分辨所需要的信息数据以及如何对信息进行有效利用已经成为当今社会信息化的重要问题。因为关键词能够反映整个文本的主要内容，帮助人们快速了解文本概况，提高信息利用率，所以关键词提取技术逐渐承担了重要的作用。基于监督学习的关键词提取算法识别文档中的关键词短语，需要已经标注好的关键词文本数据集，利用相关数据集数据训练关键词提取模型，通过模型对文本进行关键词提取任务。为了更好的关键词提取效果，通常情况关键词提取模型需要高质量的训练数据。但关键词文本数据的标注因为人的主观差异导致不同结果、时间空间成本高的问题存在，导致关键词文本数据质量不高，标注好的关键词文本种类丰富程度不足等问题。

本文面向关键词模型文本进行文本数据扩增，旨在提供高质量的关键词文本数据，增加关键词文本的数据集样本多样性，从而提高关键词模型提取关键词的准确程度，帮助用户对文本进行分析工作，为后续对文本进行其他处理和提高用户体验提供坚实基础。本文对项目背景和技术现状进行了全面分析后，在集成了常见的 EDA 同义词替换扩增和回译扩增的通用扩增方法基础上，通过句法分析技术，分析文本语义上的特点，基于句法分析结果对关键词文本语句进行从复杂句到简单句、从并列句到普通独立语句的无损变换。在将语句分解成简单句的基本句式前提下，对语句进行依存句法树构建，对句子每个分词与关键词的相似度比较，将包含关键词信息的句子成分设置为句子的重要成分。通过人为设计的句法树变换规则，对句子的非重要成分进行删除或者 BERT 填词，完成对文本每一个语句的扩增。最后对语句的组合拼接构建完整的关键词文本数据，实现对于关键词文本的扩增。

本文技术基于 Python 实现，通过 Django 提供后端服务，获取扩增关键词文本结果。经过测试，本技术实现了基本设计目标，测试结果与预期相符合。本文使用第三方关键词抽取模型 BERT-KPE 进行实验评估，实验结果表明本文设计开发的基于句法树的关键词模型文本扩增技术真实有效，且符合语义的基本逻辑，与未使用扩增方法相比获得显著的精度提升，与使用其他扩增方法相比也更有效。本文开发的关键词文本扩增技术很大程度上节约了文本的标注成本，丰富了关键词文本的数据集多样性，从而提高模型的鲁棒性以及泛化能力。

关键词： 数据扩增，关键词文本，句法树，自然语言处理

南京大学研究生毕业论文英文摘要首页用纸

THESIS: Keyword Text Data Augmentation Technology Based on Syntax Tree

SPECIALIZATION: Software Engineering

POSTGRADUATE: Wang Qingyu

MENTOR: Professor Zhenyu Chen

Abstract

With the explosion of text data on the Internet, the vast amount of information available to technology companies and society today poses a great challenge. How to accurately distinguish the information data needed and how to make effective use of it has become an important problem in today's information society. Because keywords reflect the main content of the entire text, help people get a quick overview of the text and improve information utilisation, keyword extraction techniques are gradually assuming an important role. Keyphrase extraction methods based on supervised learning identify keyphrases in documents, which require an already annotated keyword text dataset, train a keyphrase extraction model using the data from the relevant dataset and execute the keyphrase extraction task on the text through the model. In order to achieve better keyphrase extraction results, the keyphrase extraction model usually requires high quality training data. However, the annotation of keyword text data suffers from different results due to subjective differences in people, high time and space costs, leading to problems such as poor quality of keyword text insufficient richness of the annotated keyword text variety.

This thesis is oriented towards keyphrase extraction model text for text data augmentation, aiming to provide keyword data with high quality, increase the diversity of the dataset samples of keyword text, thus improving the accuracy of keyphrase extraction model, helping to analyse the text and providing a solid foundation for other subsequent processing of the text and improving the user experience. After a thorough analysis of the project background and the current state of the art, this thesis analyses the semantic features of the text through syntactic analysis techniques based on common

EDA data augmentation and back-translation data augmentation methods, and execute lossless transformations of the text from complex to simple sentences and from parallel to ordinary independent sentences based on the syntactic analysis results. On the premise of decomposing the sentence into the basic syntax of simple sentences, the sentence is constructed as a dependent syntactic tree, and for each token of the sentence the similarity with the keyphrase is compared. the sentence components containing keyword information are set as the important components of the sentence, and the non-important components of the sentence are deleted or BERT filled in through artificially designed syntactic tree transformation rules to complete the augmentation of each sentence of the text. Through combining and splicing the sentences to form the complete keyword text data, the augmentation of the keyword text is completed.

The implementation of the technology in this thesis is based on Python , with Django providing the back-end service to obtain the results of the augmented keyword text. After testing, the technology achieved the basic design goals and the test results were in line with expectations. The experimental evaluation using a third-party keyphrase extraction model, BERT-KPE, shows that the text augmentation technology based on syntax tree designed in this thesis is realistic and effective, and conforms to the basic logic of semantics. Significant accuracy gains were obtained compared to unused amplification methods and were also more efficient compared to the use of other amplification methods. The keyword text data augmentation technology developed in this thesis largely saves the cost of text annotation and enriches the diversity of the keyword text dataset, thus improving the robustness of the model and its generalisation ability.

keywords: Data Augmentation, Keyword, Syntax tree, Natural language processing

目　　录

目　　录 ····· v
插图清单 ····· ix
附表清单 ····· xi
第一章　引言 ····· 1
1.1　选题的背景和意义 ····· 1
1.2　国内外研究现状及分析 ····· 3
1.3　本文研究内容及工作 ····· 5
1.4　本文的组织结构 ····· 6
第二章　技术综述 ····· 9
2.1　词向量 ····· 9
2.2　BERT 模型 ····· 11
2.3　句法分析 ····· 13
2.3.1　依存句法分析 ····· 13
2.3.2　成分句法分析 ····· 15
2.4　本章小结 ····· 16
第三章　基于句法树的关键词文本扩增技术 ····· 17
3.1　技术路线与研究过程 ····· 17
3.2　关键词模型构建过程分析 ····· 17
3.3　文本数据扩增方法 ····· 19
3.3.1　基于词向量的关键词提取 ····· 19
3.3.2　基于语义的语句简化 ····· 20
3.3.3　基于句法树的扩增方法 ····· 30
3.4　本章小结 ····· 34
第四章　需求分析与系统设计 ····· 37
4.1　系统整体概述 ····· 37
4.2　需求分析 ····· 38
4.2.1　功能性需求分析 ····· 38

4.2.2 非功能性需求分析 …… 39
4.2.3 用例分析 …… 40
4.2.4 用例描述 …… 40
4.3 系统总体设计 …… 46
4.3.1 系统总体架构设计 …… 46
4.3.2 4+1 视图 …… 47
4.3.3 系统基本功能模块 …… 52
4.4 数据交互模块设计 …… 53
4.5 数据管理模块设计 …… 56
4.6 数据扩增模块设计 …… 57
4.7 文本分析模块设计 …… 59
4.8 本章小结 …… 60
第五章 系统实现与测试 …… **61**
5.1 数据交互模块实现 …… 61
5.2 数据管理模块实现 …… 62
5.3 文本分析模块实现 …… 63
5.4 扩增模块实现 …… 64
5.4.1 通用扩增模块 …… 65
5.4.2 句法树扩增模块 …… 66
5.5 实验分析 …… 70
5.5.1 实验数据集 …… 70
5.5.2 实验内容 …… 70
5.6 系统测试 …… 73
5.6.1 测试目标 …… 73
5.6.2 测试环境 …… 74
5.6.3 功能测试 …… 74
5.6.4 性能测试 …… 79
5.7 本章小结 …… 80
第六章 总结与展望 …… **81**
6.1 总结 …… 81
6.2 未来工作的展望 …… 82
参考文献 …… **83**

简历与科研成果 ····· **89**
《学位论文出版授权书》 ····· **91**

插图清单

2-1 CBOW 和 Skip-gram 模型结构示意图 …… 10
2-2 Transformer 结构图 …… 12
2-3 BERT 模型结构图 …… 13
2-4 依存句法树图 …… 14
2-5 成分句法树图 …… 15

3-1 技术路线与研究过程示意图 …… 18
3-2 语句示例 1 …… 25
3-3 主语并列示例 …… 26
3-4 宾语并列示例 …… 27
3-5 语句示例 5 …… 28
3-6 补语并列示例 …… 29
3-7 依存分析结果示例 …… 32
3-8 句法树剪枝规则扩增方法示例 …… 32
3-9 句法树 BERT 填词扩增方法示例 …… 34

4-1 系统用例图 …… 41
4-2 系统架构图 …… 47
4-3 4+1 视图 …… 48
4-4 系统逻辑视图 …… 49
4-5 系统开发视图 …… 50
4-6 系统进程视图 …… 51
4-7 系统物理视图 …… 52
4-8 系统基本功能模块图 …… 53
4-9 数据交互模块核心类图 …… 54
4-10 文件存储流程 …… 55
4-11 数据管理模块核心类图 …… 56
4-12 数据扩增模块功能图 …… 57

4-13 数据扩增核心类图 ····· 58
4-14 文本分析模块核心类图 ····· 59

5-1 信息预览 ····· 61
5-2 计算 Sha1 关键代码 ····· 62
5-3 项目管理界面 ····· 63
5-4 构建依存句法树关键代码 ····· 64
5-5 训练词向量模型关键代码 ····· 65
5-6 句法树剪枝扩增流程 ····· 66
5-7 句法树从句识别部分代码 ····· 68
5-8 句法树剪枝流程伪代码 ····· 69
5-9 句法树 BERT 填词关键代码 ····· 69
5-10 数据示例 ····· 71

附表清单

3-1 语句 5 大基本类型 …… 21
3-2 主语从句 …… 22
3-3 同位语从句，表语从句 …… 22
3-4 定语从句 …… 23
3-5 状语从句 …… 23
3-6 语句并列示例 …… 24
3-7 语句级并列 …… 25
3-8 主语并列 …… 26
3-9 宾语并列 …… 27
3-10 谓语并列 …… 28
3-11 补语并列 …… 29
3-12 其他并列 …… 30
3-13 依存关系识别表 …… 31

4-1 功能性需求列表 …… 39
4-2 非功能性需求列表 …… 40
4-3 系统用例列表 …… 41
4-4 系统信息预览件用例描述 …… 42
4-5 上传下载关键词文本用例描述 …… 43
4-6 通用文本扩增用例描述 …… 44
4-7 关键词文本句法分析扩增用例描述 …… 44
4-8 关键词文本文件管理用例描述 …… 45
4-9 关键词文本扩增任务管理用例描述 …… 45
4-10 关键词文本扩增项目管理用例描述 …… 46

5-1 关键词提取实验数据集相关信息 …… 71
5-2 扩增方法效果对比 …… 72
5-3 扩增工具效果对比 …… 73

5-4 实验数据集相关信息 ····· 74
5-5 信息预览测试用例 ····· 75
5-6 关键词文本上传与下载测试用例 ····· 75
5-7 关键词文本通用扩增测试用例 ····· 76
5-8 基于句法树的关键词文本扩增测试用例 ····· 77
5-9 关键词文本扩增文件管理测试用例 ····· 77
5-10 关键词文本扩增任务管理测试用例 ····· 78
5-11 关键词文本扩增项目管理测试用例 ····· 78
5-12 用例测试结果 ····· 79
5-13 数据扩增性能测试 ····· 79

第一章　引言

1.1　选题的背景和意义

随着互联网上文本数据的爆炸增长, 海量信息给当前的科技公司和社会带来了很大的挑战。如何准确的分辨所需要的信息数据以及如何对信息进行有效利用已经成为当今社会信息化的重要挑战。关键词是最能概括文档的单字或多字词汇单位 [1]，它们对于索引、文本分类和浏览数字图书馆非常重要 [2]。然而，很少有文档指定了关键词。针对这个问题，关键词自动提取技术逐渐承担了重要的作用，其利用关键词反应文本的主要信息，能够提高信息的使用效率。关键词提取是自然语言处理（NLP）中一个非常热门的研究方向，它在信息检索等任务中发挥着重要作用，最常见的是在搜索引擎中，输入关键词就可以搜索到与之相关的页面。然而，大量的数据文本没有标记关键词，逐一手动标记是耗时和低效的。

当前主流的关键词提取方法主要是无监督学习方法和有监督学习方法这两个类别。无监督方法的原理是基于统计学中的统计信息。比如常见的 TF-IDF [3]，该算法简单高效，按照权重和阈值，选取前 k 个候选词语作为选取的关键词对象。相较于 TF-IDF 算法，TextRank [4] 算法的提出来源于谷歌的 PageRank 的思想，不仅考虑候选词的词频信息，还考虑了词语的相对位置信息。Wan [5] 等提出了 SingleRank 方法改进了传统的 TextRank 没有考虑到单词出现频率不同应该具有不同权重的问题，在 TextRank 中加入了单词共现次数的统计信息，他们计算各单词节点间边的权重通过单词的共现次数度量实现。虽然无监督学习方法不需要人工标注训练集，简单易实现，但提取关键词的准确率较低，仅有 20% 左右。在关键词提取模型发展过程中,LDA 主题模型 [6] 被提出来替代 TextRank 方法，通过利用文本关键词中隐含的主题进行建模和分析，理论上能够获得更好的结果。Pu [7] 利用 LDA 主题模型提出了 TDCS 模型取得了较好的效果。由于这类方法没有标注数据集，对于文本中的语句结构和语义内容不能合理挖掘，难以应用于实际生产活动中。

有监督学习方法把提取关键词问题化为分类问题和序列标注问题，最基础

的监督学习方法就是将关键词提取任务视为通过判断文本中每个词语是否为一个关键词短语进行词语的分类。分类方向的学界研究主要是基于大规模过往数据统计的基础上对关键词文本的特殊特征建立是否为关键词的分类器来提取关键词。不同于无监督学习方法，这类方法需要在标注好的数据集上，训练分类器来区分什么是关键词。常见的分类器有贝叶斯算法、SVM 算法、决策树等。基于决策树的分类器方法，有 Turney [8] 实现的一个关键词提取方法 GenEx,GenEx 方法通过使用单词的词性信息等特征设计了一个有效的决策树模型，大幅度提高了关键词文本的提取效率; 基于朴素贝叶斯方法，Frank [9] 构造了一个关键词文本分类器，实现了关键词文本提取方法 KEA, 相较于无监督方法，他们的方法可以利用更多的相关信息。通常来说，提取关键词的准确度会更高。但关键词提取问题看做分类问题，常见的一些分类器只考虑到单独的词语，没有利用上下文信息，这会导致性能难以继续提高。最近，神经网络技术已被应用于关键词任务。Meng [10] 等人制定了一个 seq2seq 学习任务，该任务从文档序列中提取和生成关键词序列；他们在 seq2seq RNN 中加入了一个复制机制，在生成过程中提取短语 (CopyRNN)，大幅度提高了模型的准确性。

由于监督学习方法通过标签衡量文本的标签是否有正确判断训练的效果，标签的正确性影响最终训练效果。与无监督学习方法相比，监督学习方法可以对模型进行有效的评价。但是，基于监督学习方法的关键词提取模型，需要使用大规模、高质量的数据来训练。对模型性能的评估也依赖于广泛的、有代表性的数据。而数据的质量和数量决定了是否在实际中可以投入使用，所以这些要求需要大量的标注数据。为了达到更好的关键词提取效果，基于深度学习的关键词提取模型成为了当前研究的重心。

在训练关键词提取模型的过程中，如何获得高质量的关键词文本数据成为关键，但由于现实中数据标注的困难，标注关键词需要较高的成本，需要大量人员参与标注，通过交叉比对确定关键词的标注结果。在典型的关键词提取任务中，数据问题的存在会使基于深度学习的关键词提取模型出现过拟合现象，严重影响关键词提取效果，不能提取到用户所需要的关键词。基于以上遇到的问题，如何利用有限的标注数据生成更多的数据来减少基于深度学习的关键词提取模型出现的过拟合现象，进而能够训练出能够在生产环境让用户放心使用的关键词提取模型已经成为一个迫切的问题。

数据扩增技术能够通过扩增方法扩增出大量的关键词文本数据。以前的关键词提取工作侧重于现成的科学摘要，并使用作者指定的关键词代表专家注

释。然而，这带来了两个主要问题：1）关键词生成的神经模型不能很好地跨领域推广，从而限制了它们在实践中的应用；2）作者指定的关键词表现出严重的不一致问题，对模型的性能产生负面影响。因此，非常需要来自不同来源的注释数据，这些数据既足够大，以支持基于神经网络模型的训练，又包含专家提供的真实标记标签。

数据扩增是通过人为设计的扩增方法解决数据较少的问题，通过这些扩增方法能生成大量近乎等价的相关数据。虽然现在国内外已经出现了一些文本数据扩增方法，包括基于回译扩增，简单数据扩增方法 EDA 之类的文本扩增方法。但这些方法主要集中于对单个语句进行扩增，通过对文本的加噪实现扩增。在进行关键词提取中，文本数据不再是简单的一个语句，而是由多个语句组成的短文本或者长文本，修改了体现文本关键词特征信息以及关键词文本的语义结构信息，导致生成的关键词文本不能很好地被人和机器理解，缺乏逻辑性，扩增文本质量较低，对关键词模型性能的提升效果不明显。因此，需要提出对关键词模型训练数据文本进行高效扩增的方法。

当前暂时没有针对关键词文本的扩增方法，而基于深度学习的关键词提取模型对数据的质量，数量有较高的要求，因此本文研究一种针对关键词文本数据的扩增技术，使用句法分析、bert 模型等技术充分挖掘数据集语义特征，帮助用户获取大量高质量的关键词文本，提高关键词提取模型的适用性和精度。

1.2 国内外研究现状及分析

与传统的无监督学习方法相比，基于深度神经网络的关键词提取模型的训练需要大量数据。虽然通过调参可以提高模型的表现，但模型的开发者使用小样本数据集的时候经常会遇到一个两难的问题，如果训练足够多的迭代次数，虽然在该训练数据集上拥有较好的表现，但在其他数据集上会产生严重的不适用情况，这被称之为过拟合现象。如果通过提前停止 Early stopping[11] 减少模型训练的迭代次数，仅仅训练几个迭代。虽然不会产生过拟合，但模型性能也难以接受。使用 Dropout [12] 随机将神经网络单元暂时从网络中丢弃，通过正则化项 [13] 降低模型复杂度的方式可以减轻过拟合情况的发生。但高质量的数据才是一切的根本，如果数据扩增技术能够生成的多领域高质量的关键词文本数据，关键词提取模型的泛化能力将大幅度提高。

数据扩增，将原有数据通过人为设定的变异方法生成用户所需要数据的过

程。如果有相应的数据扩增方法，机器学习中训练数据不足的问题将大幅度得到缓解。使用数据扩增的好处，不仅是数量上的变多，而且让数据分布均匀。数据扩增可以分为图像数据扩增方法，文本数据扩增方法和其他数据扩增方法。文本数据，是通过字符组合而成，无法使用广泛应用的图像扩增方法对文本进行扩增。因为图像扩增相较于文本扩增相对容易，数据扩增方法先在 CV 领域大幅应用，研究者们可以通过传统 CV 领域中常见的基于各种矩阵算子实现图片模糊，基于颜色通道数值的修改实现对图片颜色的修改，通过来自于数学方法的无损几何变换方法实现对于图片的几何变换。通过诸如 imgaug 这种封装好的 Python 库可以快速实现图像的批量化扩增。而自然语言在计算机中是通过符号进行表示的，很难通过数学上的一些简单算式进行生成或者通过其他方法对这些符号进行简单的修改，任何符号上的变化都可能大幅度改变文本本来的意思。因此，文本扩增相较于其他两种扩增，难度更高 [14]。

回译方法来源于语言学，在语言学中通过回译方法学习其他语言，验证语言翻译的准确性。而在数据扩增领域的回译扩增方法是将关键词文本数据通过神经网络翻译系统翻译为其他语言，在通过该神经网络翻译系统翻译回原语言文本。回译方法因为神经网络翻译系统的翻译不准确，可以生成大量相似的文本数据。通过回译扩增方法，能够保留文本的基本语义信息。J Ma [15] 对回译扩增进行了验证，实验结果表明回译扩增能够提高文本分类模型的性能。

加噪是指通过对文本数据的进行替换里面的单词、删除部分单词或者字符等操作产生一些噪声，生成一些原有文本数据相似的新数据。Jason W. Wei 等 [16] 提出了 EDA 扩增方法，包括了在文本中随机对语句插入一些单词、使用同义词进行替换、对文本中的单词随机交换他们的顺序、在文本中随机删除一些字符或者单词。该方法在文本训练数据较少的情况下取得了较好的效果。

Zhang Xiang [17] 等人在实验中因为实验数据不足，想利用已有的同义词词库，把单词替换为同义词来生成新的数据，实验结果表明该方法能够提高他们的文本分类模型性能。

S Kobayashi [18] 提出一种新型的标记句子的数据扩增方法，称为上下文增强。随机地用一个双向语言模型在单词位置预测的其他单词来替换单词。对语言模型进行了标签条件架构的改造，这使得模型可以在不破坏标签的情况下对句子进行扩增。这样可以用于对于文本分类的数据扩增。

HOU Y 等人 [19] 提出了基于序列的数据扩增框架，利用训练数据中一个语词的相同语义替代词来扩增一个语词，而不考虑它与其他词语的关系。首次将

多样性等级纳入到语料表征中，使模型产生多样化的语料。这些经过多样性增强的语料有助于改善语言理解模块，但该方法是否存在一定的限制性，是否具有广泛的适用性，仍需进一步探讨 [20]。

回译和加噪两种文本数据扩增路线，通过产生所需要的数据，能够帮助深度学习模型提高性能和泛化能力。但目前来看，没有针对关键词文本数据的文本扩增方法，通用的文本扩增方法可能在关键词文本扩增的时候，不能够产生所需要的扩增关键词文本数据。

1.3 本文研究内容及工作

基于深度学习的关键词提取模型需要数以万计的数据，并且这些数据质量越高越好，数据数量越多越好。在现实的关键词提取模型，通常仅仅针对某个领域的文本进行训练。纵观关键词提取模型的历史，最早的关键词提取语料库就是来自科学领域，这些通常包括了技术报告和科学文献。这是因为科学领域关键词文本语料库很容易获取，报告和文献的作者已经标注了相应的关键词。如果想要不同领域的关键词文本数据，就会遇到数据语料库不足的问题。对不同领域的文本进行关键词标注成本高，而且标注质量也很差，阻碍了关键词提取模型的发展。

目前，数据扩增领域已经有一些文本数据扩增方法，而这些简单的扩增方法易抹去体现关键词文本特征的关键词语以及修改关键词文本的语义结构信息，导致扩增后的文本会改变文本的关键词标签，导致关键词提取模型精度的提升不大，甚至提取出大量没有用的短语。本文的主要研究内容为：针对关键词模型文本，使用依存句法分析，通过 BERT 模型等技术充分挖掘数据集特征，扩增出质量更高的关键词模型文本数据，用于相关关键词提取模型的训练。

为了保障关键词模型有足够的高质量关键词文本语料，本文研究了基于句法树的关键词文本扩增技术。不仅通过对通用的文本数据扩增方法的研究，提供 EDA 同义词替换，回译二种通用的文本扩增方法的设计和实现，而且针对关键词文本数据，本文研究了基于句法树的扩增技术的设计和实现。通过对语句依存句法树进行操作，使用基于句法树剪枝规则和基于句法树 bert 填词方法对关键词文本进行扩增。

本文的工作主要分为基于句法树的关键词文本扩增技术的研究和设计，基

于句法树的关键词文本扩增系统的实现。

对于基于句法树的关键词文本扩增技术的研究和设计，本文借鉴了已有的文本扩增方法，从关键词模型构建角度出发，研究了关键词文本数据的特点，通过句法分析技术，分析文本语义上的特点，实现基于句法树关键词文本扩增。首先将文本从复合句到简单句、从并列句到普通独立语句的核心语义保留变换。在基于简单句句子的基本句式前提下，对语句进行依存句法树构建，对句子每个 token 与关键词的相似度比较，将包含关键词信息的句子 token 成分设置为句子的重要成分。通过人为设计的句法树变换规则，对句子的非重要成分进行删除或者 BERT 填词，完成对文本每一个语句的扩增。最后拼接这些文本，构建成完整的关键词文本数据，完成对于关键词文本的扩增。

而关键词文本扩增系统的实现，主要包括了基于 Web 的关键词文本扩增服务的开发。该扩增服务包括了基于 Angular 框架开发的前端页面，基于 Django 开发的后端服务器，独立不同的子服务模块提供后端服务器所需要的一些数据处理能力。该系统的关键词文本扩增模块在集成了基于句法树的关键词文本扩增方法以外，集成了常见的 EDA 同义词替换扩增方法和回译扩增方法。关键词文本数据相关信息通过 MySQL 进行数据的持久化，而关键词文本和其他扩增数据都存储在系统的本地文件夹中。

1.4 本文的组织结构

本文基于已有的文本扩增方法设计了一种基于句法树的关键词文本扩增方法，并实现了基于该技术的关键词文本扩增系统，最后通过测试验证该扩增系统的可用性，通过实验验证了该方法的有效性。本章的组织结构如下：

第一章引言，介绍了关键词文本扩增技术的选题背景，研究了关键词提取模型和数据扩增方法的研究现状，开发关键词文本扩增技术意义和本文的主要工作。

第二章技术综述，简略介绍了实现基于句法树的关键词文本扩增技术所需要的相关知识和技术，包括了核心的句法分析技术、BERT 语言模型、词向量相关知识。

第三章基于关键词文本的扩增方法，通过分析关键词模型构建，挖掘关键词模型特征，提出基于句法树的关键词文本扩增技术，并对这种技术进行阐述。

第四章需求分析与系统设计，通过需求分析明确系统的目标和内容。通过概要设计，将之前需求分析的结果，转换为系统物理和逻辑上的设计。通过4+1 视图，阐述系统的总体设计。接着介绍了各个功能模块的详细设计。

第五章系统实现与实验测试，基于需求分析和系统设计结果，对相关的功能模块的实现进行阐述。对基于句法树的关键词文本扩增技术进行关键词文本扩增实验，并进行相关的实验分析。通过测试用例的设计，进行对系统功能和性能的测试。

第六章总结与展望，对基于句法树的关键词文本扩增技术进行总结，并对该技术未来的发展进行展望，提出当前技术不足之处的改进方向。

第二章　技术综述

本章介绍了在研究基于关键词的文本扩增技术时所用到的一些技术，包括词向量、BERT 语言模型、句法分析技术。这些技术对于分析关键词文本数据，扩增关键词文本数据具有重要作用。

2.1　词向量

由于在 NLP 中，文本是字符数据构成的，不具有结构化的性质。计算机不能直接计算字符数据，需要先将文本转化为有意义的数值数据。一般情况，首先通过分词技术将文本分成由单词构成的列表，将单词作为文本的最小单位。然后将文本分词结果列表里的单词转化为数值数据。词语的数值化表示主要有离散型和分布式两种表示方法。

为了在向量中体现词与词之间的关系，Hinton [21] 提出了词语的分布式表达形式，每个词对应的分布式表达是一个低维度的实值向量，其中每一个维度均可以被认为是一个词的可能人并不知晓的特征。其中单词词表的所包含的单词数量大小作为向量长度大小，通过 onehot 方法对每个词进行编号。每一个词的向量均为一个 n 维数值向量，有且仅有一个位置上的数值是 1。

$$w^{abandon}=\begin{bmatrix}1\\0\\0\\\vdots\\0\end{bmatrix}, w^{a}=\begin{bmatrix}0\\1\\0\\\vdots\\0\end{bmatrix}, w^{at}=\begin{bmatrix}0\\0\\1\\\vdots\\0\end{bmatrix}, \ldots w^{zebra}=\begin{bmatrix}0\\0\\0\\\vdots\\1\end{bmatrix}$$

这种表示方法具有简单，健壮的优势。然而单词表示之间没有关联性，存在语义上的理解难度，无法通过向量得知不同单词之间的关系，因为无论是哪两个单词之间的他们向量的距离都是相等的。其次它不能反应词语的顺序性，语言不是每个单词都完全独立的序列。因为某些单词之后只能出现在特定的单词之后，它们组合起来才能构成一个有特定意义的短语，有特殊顺序的多个词

按照相应的序列才能构成一个可以被理解有意义的句子。最后存在维数灾难的问题，高维数的情况下，数据样本稀疏，模型计算量急剧增加。

因为以上的一些缺点，词向量逐渐成为主流。与简单的分布式表示方法相比，词向量是一个维度相对而言较低的稠密向量，词向量将原始稀疏的高维向量强制降到小维空间。将独热编码的向量从整数转换为浮点型，它的每一个维度都是实数，而不是大多是 0 的稀疏向量。

$$w^{zebra} = \begin{bmatrix} 0.33 \\ 0.41 \\ 0.62 \\ \vdots \\ 0.21 \end{bmatrix}$$

Word2Vec 是在 2013 年由 Mikolov 等 [22, 23] 提出并实现的一种词向量生成技术，用于快速获得低维词向量，其基本思想是基于一个简单的神经网络生成相应的维度的数值数据。Word2Vec 其模型仅仅包括三层神经网络，通过输入层、隐藏层和输出层构成了模型。模型框架主要包括 CBOW 和 Skip-gram 模型 [23]。CBOW 的方式是在知道词 w_t 的上下文 $w_{t-2}, w_{t-1}, w_{t+1}, w_{t+2}$ 的情况下预测当前词 w_t. 而 Skip-gram 是在知道了词 w_t 的情况下, 对词 w_t 的上下文 $w_{t-2}, w_{t-1}, w_{t+1}, w_{t+2}$ 进行预测, 如下图所示:

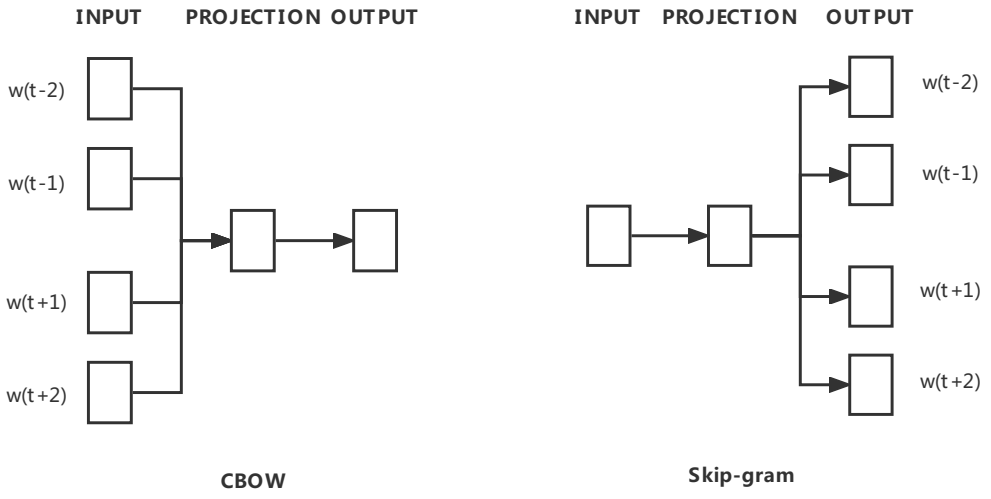

图 2-1: CBOW 和 Skip-gram 模型结构示意图

通过使用 Word2Vec 训练可以获得词语的低维向量，并且存在相似语义的

词汇在词向量空间中距离也会更近。在本文中，使用词向量技术中的 Word2Vec 用于词语之间的语义相似度的计算，综合 Word2Vec 技术获取到每个词语的词向量，与关键词的词向量进行比较，用于提取关键词文本中所出现与关键词相似的词语。

2.2 BERT 模型

BERT 语言模型 [24](Bidirectional Encoder Representations from Transformers) 是 Google 2018 年提出的一种语言模型，在多个不同的 NLP 任务中取得了 SOTA 的表现。BERT 与过往的预训练模型相比，预训练以后不需要修改 BERT 结构，只需要添加一个额外的输出层，通过 fine-tune 就能应对不同的下游任务，并且 Google 的开发者们发现了掩码语言模型可以在预训练模型上取得非常好的效果。虽然掩码语言模型 Taylor [25] 早已提出，但没有人将其应用于预训练语言模型中。

BERT 语言模型通过 Mask LM（MLM）和 Next Sentence Prediction（NSP）对 BERT 进行预训练，MLM 训练数据生成器按照 15% 的比例通过随机替换语句中的一些子词为 [mask] 符号对应的 tokenid，预测被 [mask] 符号遮蔽的结果。由于 BERT 模型并不知道需要预测的内容，也不知道多少内容被替换，所以需要学习上下文内容，抽取 token 的所有表征信息，理解文本深层次的特征来进行预测。由于 MLM 任务倾向于抽取 token 层面的表征信息，所以需要一种任务能够获取句子之间的表征信息，通过 NSP 让 BERT 模型猜测两个句子是否为上下句，来让 BERT 语言模型捕捉句子间信息。由于 MLM 和 NSP 都不需要从有标签的文本中进行学习，实际上是进行自监督性质的学习任务，可以使用更广泛的数据，获得更好的预训练效果。

BERT 语言模型不像以前的预训练模型结构会受到单向语言模型的限制，只能从左到右或者从右到左，最多使用浅层拼接两个单向语言模型。BERT 基于 Transformer encoder 的双向编码器，比 LSTM 更高效，不仅能够捕捉文本的依赖关系，还能够获得复杂的上下文关系，挖掘深层次的文本特征。BERT 能够解决一词多译的问题，比如一句话“你喜欢苹果，我喜欢苹果手机。”，如果使用 Word2Vec 生成词向量，句子中两个苹果是无法进行区分的，他们具有相同的词向量，但如果使用 BERT，就能够区分前后者。

Transformer [26] 抛弃了传统的 CNN，RNN，是 2017 年谷歌 AI 实验室提

出的基于注意力的神经网络模型，它是一种序列到序列的模型，包括 Encoder 和 Decoder 机制。Transformer 中的 Encoder 用来读取文本输入将语言序列映射成隐藏层，Decoder 用于生成任务预测将隐藏层映射为自然语言序列。每个 Transformer 模型通过将 Encoder 和 Decoder 组合，处理接收到的文本数据，它的输入是一组序列数据，模型输出经过处理的序列数据。Transformer 的结构如图 2-2所示。

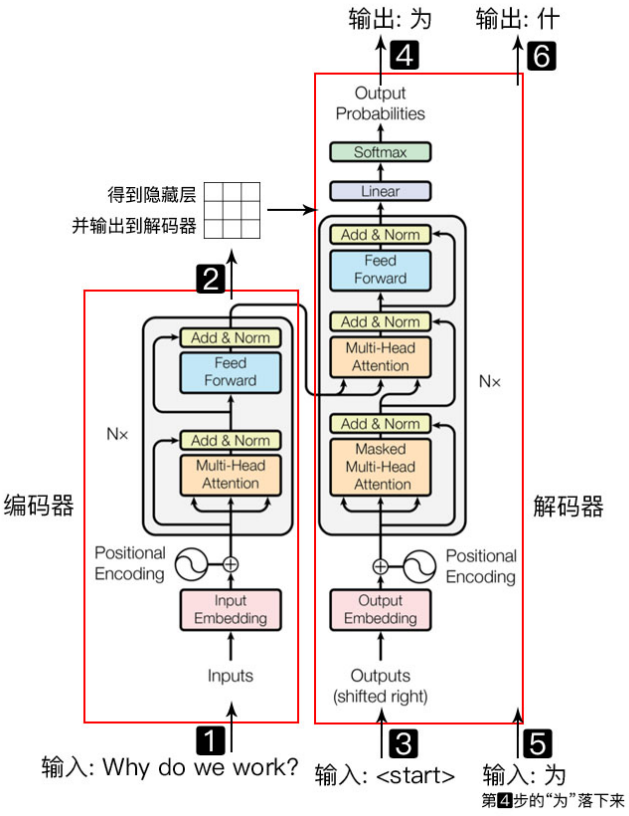

图 2-2: Transformer 结构图

BERT 模型的结构如图 2-3所示，图中每个 Trm 被称为一个 Transformer Block。可以看出 BERT 语言模型利用了 Transformer Block, 构造了一个多层双向的 encoder 网络。

本文通过 BERT 模型执行填词任务来对关键词文本数据进行填词，由于 BERT 模型预训练本身就使用了 Masked LM 作为预训练任务，实现填词相较于其他方法更加简单和精准有效。因为 BERT 填词会考虑到上下文的语义信息，所以基于 BERT 的关键词文本填词，可以产生高质量的生成数据。

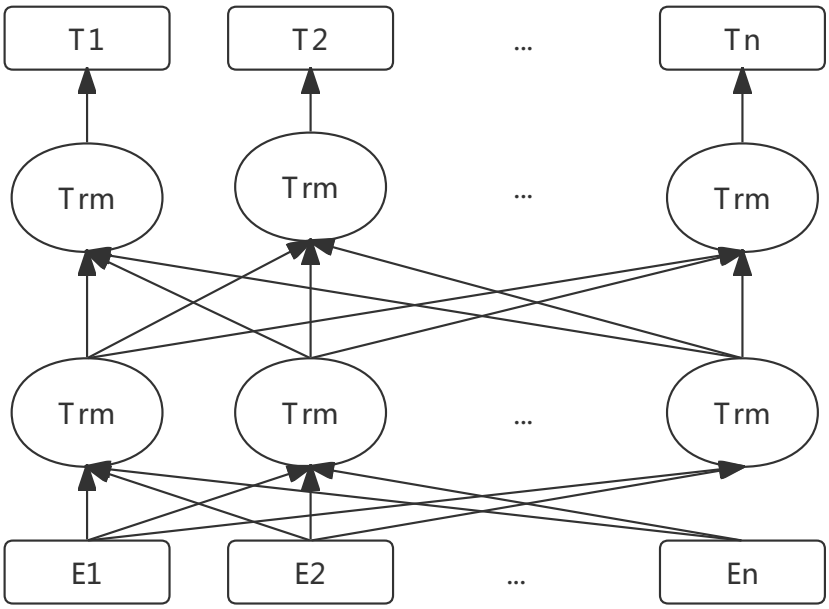

图 2-3: BERT 模型结构图

2.3 句法分析

随着文本等语言上的数据的不断增加，自然语言处理技术作为人工智能领域的一项关键技术越来越重要。自然语言处理主要包括了词法分析、句法分析和语义分析。句法分析 [27] 起到了将词法分析结果转换相关语义特征，为语义分析提供基础。因此句法分析的重要性不言而喻。

目前存在两种主流的句法标注体系：依存句法分析和成分句法分析。成分句法分析主要是识别语句中的各个短语之间的关系；而依存句法分析主要是识别语句中所有词语之间的依赖关系。这两种句法标注体系，在自然语言处理中起到了重要的作用，并被广泛使用。

2.3.1 依存句法分析

依存句法分析（dependency syntactic parsing），又称依存关系分析，简称依存分析，作用是识别句子中词汇与词汇之间的相互依存关系。依存句法分析理论由 Tesniere [28] 于 1953 年在法国首次提出，他通过语句内不同的单词成分的依赖关系，分析语句的句法结构。在 Tesniere 提出依存句法理论以后，人们

在他的基础上不断研究更合理的依存句法分析方法，其中包括最著名的 Prague School 的 Functional[29]。通过共同的假设，句法结构

不同的依存句法结构的建立以“句法结构本质上包含词和词对之间的依存或者修饰关系”的共同的假设为基础。依存句法结构中的依存关系是一个（核心词，修饰词，依存关系）三元组构成。目前利用语料库，研究者提出众多的自然语言句法分析法。较成功的句法分析技术都是利用语料库来获取语法和概率数据信息。自 2014 年 Sutskever 等人 [30] 提出深度学习框架以来，越来越多的研究者基于深度学习方法来进行依存句法分析。当前依存句法分析方法主要分为基于深度学习的方法 [31] 和非深度学习方法。非深度学习方法主要包括了基于人为设计规则的方法 [32, 33] 和基于数学统计的方法 [34, 35] 这两种方法。

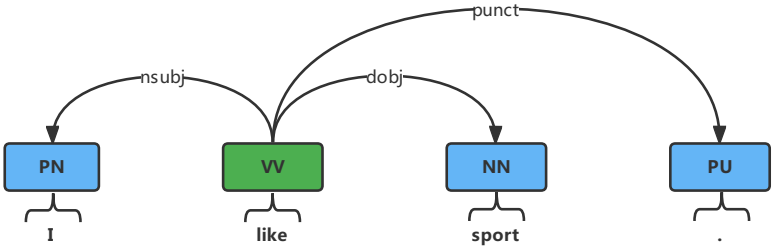

图 2-4: 依存句法树图

图 2-4 是“I like Sport.”的依存句法分析结果, 因为使用 Stanford CoreNLP 工具，其中句法结构和词性都遵循 Universal Dependencies (UD) [36] 中的标注规则。箭头上面的文字代表了词与词之间的依赖关系，箭头指向了修饰词所依赖的核心词成分，而箭头的标签则表示了相应的依存关系。如例句中的核心词“like”，它就是根结点。“I”、“sport”和“.”与“like”建立了依存关系，其中“I”的句法标签是“nsubj”与核心词构成了主语谓语关系，“sport”的句法标签是“dobj”与核心词 like 构成了谓语宾语关系。通过依存句法树可以快速了解表示句子的依存关系，通过对依存句法树按照特定的规则进行操作，可以进行对主句语义影响不大的词或者词对的删除。

2.3.2 成分句法分析

成分句法分析 (Constituent parsing)，它的作用是获取语句中的短语之间的层次关系。当前，成分句法分析在自然语言处理中得到了广泛的使用。例如在情感分析任务中，开发者们通常会利用成分句法分析结果，来分析语句的情感状态。也有开发者使用训练好的成分句法树直接转为依存句法树，从而提升依存句法分析的准确率。

正如在依存句法分析领域深度学习方法的火爆一样，如何将深度学习方法引入成分句法分析也成为了研究热点。当前，研究者们基于深度学习的编解码器结构来实现文本的成分句法分析。

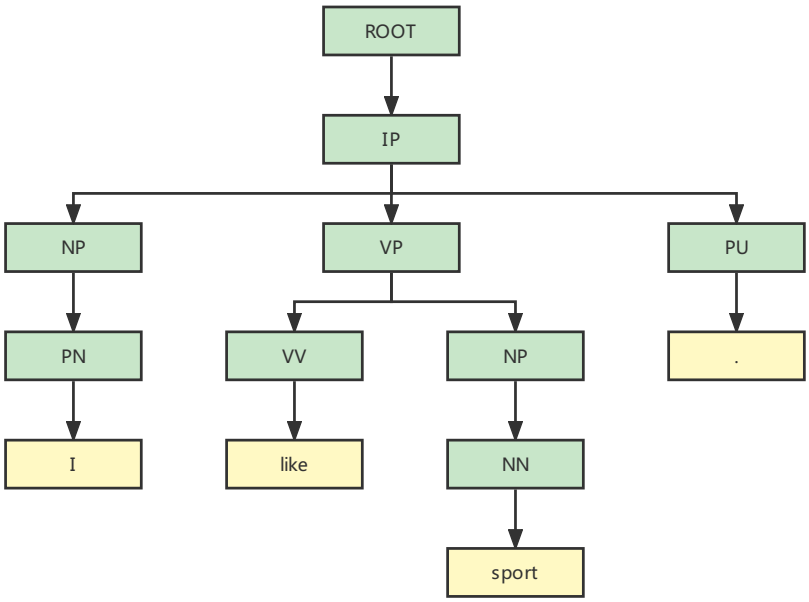

图 2-5: 成分句法树图

图 2-5 给出了一个成分句法分析结果，其中 IP 表示“I like sport.”是一个短句，NP 表示“I”和“sport”是个名词短语，VP 表示“like sport”是一个动词短语，NN 表示“sport”是一个名词，VV 表示“like”是一个动词,PU 表示“.”是标点符号。成分句法树中最终的的叶子节点是输入句子的每一个词，如：“I”、“like”和“sport”。

在本文中，综合使用成分句法分析结果和依存句法分析结果，通过这两种方法保留关键词文本的核心语义成分，简化语句形式，为关键词文本数据扩增

提供了所需要的信息。

2.4 本章小结

本章主要介绍了自然语言处理的相关技术，这是进行基于句法树的关键词文本扩增技术的关键技术，包括词向量、句法分析、BERT 语言模型。这些内容为接下来对基于句法树的关键词文本扩增技术的研究提供技术支撑。

第三章　基于句法树的关键词文本扩增技术

3.1　技术路线与研究过程

本文主要研究以下内容：（1）基于关键词模型数据集的文本扩增方法，用于缓解关键词提取模型文本数据样本较少导致的过拟合问题；（2）将此扩增方法用于关键词提取模型；（3）对扩增结果数据集和关键词提取模型进行有效性评估。

本文的技术路线与研究过程如图 3.1 所示，首先对关键词文本进行特征分析，分析需要获得什么样的扩增结果。根据对于关键词文本的特征分析，本文设计了基于句法树的关键词文本扩增技术，它包括了基于句法树的剪枝规则扩增方法和基于句法树的 BERT 填词扩增方法。如何进行句法树的剪枝，需要经过实验，得到合适的句法树剪枝规则，并通过训练相关的 BERT 模型，结合句法分析结果获取高质量的 BERT 填词结果。通过对于句法树的变换，保留文本的核心语义和关键词内容，提供高质量的关键词文本扩增数据。最后将扩增结果放到关键词提取模型进行训练中，与未扩增数据训练结果进行对比。

3.2　关键词模型构建过程分析

首先，我们对关键词模型的特征进行分析，其中包括关键词模型构建方法，关键词模型的文本特征分析。

无监督方法和有监督方法是现在两种主流的关键词提取技术。无监督关键词提取方法主要基于统计信息以及基于图的排序算法。有监督的关键词提取方法方法通常将关键短语提取描述为短语分类以及序列学习。由于传统的无监督算法在实际关键词提取方法，大多利用文本的特征，未将文本的语义信息作为筛选关键词的依据，而被忽略的语义信息往往对关键词提取的准确度有着较大影响，最终导致了不佳的性能。有监督的关键词提取方法, 尤其是使用神经

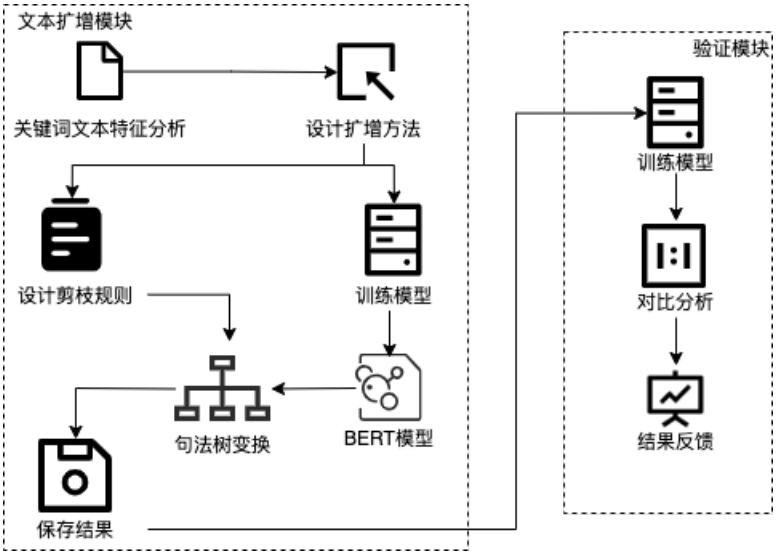

图 3-1: 技术路线与研究过程示意图

网络的关键词提取方法, 在许多关键词提取基准测试中都比最先进的无监督方法具有显著的性能优势。最早的神经关键词提取方法通常将关键词提取视为 sequence-to-sequence learning [10] 和序列标注 [37] 任务, 都采用基于 RNN 的编码器框架进行关键词提取。这些方法的性能受到文本数据的浅层表示的限制，性能还有继续提升的空间。最近, Xiong [38] 等人将关键词提取描述为 n-gram 级别的关键词组块任务, 并提出 BLINGKPE。它将深度预训练的表示形式合并到卷积 Transformer 体系结构中, 以模拟 n-gram 表示形式。该方法结合全文语义信息，将候选词输入隐藏层获取动态词向量，考虑 Transformer 模型隐藏层对语义表示效果的影响 [39]，增强了对文本语义信息的利用。与以前的方法相比,BLING-KPE 取得了很大的改进。因为传统的文本扩增方法没有考虑到文本语义的影响，因此需要设计一种能够保留文本大体语义的数据扩增方法。

关键词文本数据中的每一个文本都是词语的集合，而关键词是最能代表整个文本的几个核心的词语。对关键词文本进行分析，由于英文单词之间一般采用空格隔开，不需要特殊的分词方法。在英文关键词文本中，停用词（介词、连词、数量词）通常没有特殊含义，在关键词提取中不具备关键词短语的特征 [40]，即在英文关键词文本的关键词短语提取过程中停用词是没有意义的，对其进行加噪对关键词文本的关键词没有影响。如果某个词语在文本出现大量情态上和时态上的变化在英文关键词文本中，通常不会被视为关键词，没有特殊的语义内容。而情态或者时态上的变化通常是动词。因此，在英文的关键词文本处理过程中动词作为关键词的概率并不高。

对普通的关键词文本进行关键词提取任务，从概率学上发现名词或名词性短语的具有较高的概率是关键词内容 [41]。文本关键词多以 4—6 字名词性组合词出现。因此词性标注为名词的词语有更高的概率为关键词。名词具有更高的权重优先级，在扩增过程中，如果不想改变关键词内容，对于名词的修改需要由合适的规则进行处理。

3.3 文本数据扩增方法

依照对于关键词模型和关键词文本进行了特征分析，本文提出了基于句法树的关键词文本扩增技术，主要包含了基于词向量的关键词提取，核心语义保留的的句子简化。基于句法树的扩增方法包含了句法树剪枝规则扩增方法和句法树 BERT 填词扩增方法。

对于每个未扩增的语句，通过基于句法树的剪枝规则扩增方法，或者 BERT 填词扩增方法生成一个扩增句子列表，具体而言，该方法包括以下步骤。

1）识别关键词文本中的关键词内容，对包含关键词的词语进行提取保留。

2）将复合句转换为简单句，进行核心语义保留的的句子简化。

3）保证核心语义和关键词的不变性的前提下，对非重要语句成分的修改来扩增语句。

3.3.1 基于词向量的关键词提取

关键词模型的训练文本，与关键词相似的词语通常会出现在整个文本中，而这些词语对于关键词的提取起到了重要作用，关键词提取模型偏向于文本中出现过的词语作为提取出来的关键词。因此在进行数据扩增之前，需要识别关键词文本中与关键词标签相似的词语，将他们提取出来并将他设置为不可修改成分。

如何获取词语的关键词是一个重要的步骤。因为文本是一种非结构化的数据信息，所以它是不可以直接被计算的。文本表示的作用就是将这些非结构化的信息转化为结构化的信息，词嵌入技术 (Word Embedding)，是一种常用的词语计算机表示方法，将文本中的词语转换成可以计算的数字向量，为了在实际

使用中来对单词序列进行分析，就需要把向量序列以数字形式作为输入。词嵌入技术的计算过程就是把使用 one-hot 编码的包含了所有词数量维数的高维空间降维到一个维数低得多的连续向量空间中，生成结果就是词语序列对应的词向量。

词向量空间中有一个特征可以被用来进行文本相似度计算，在词向量空间语义相近的词语会被映射到空间中相近的位置。根据这一特性，使用夹角余弦值可以计算词语间的相似度。夹角余弦的计算方法如公式所示，通过对两个单词的词向量进行夹角余弦的计算，结果越接近 1，他们相似度越高。

$$\mathrm{sim}\,(W_1, W_2) = \frac{\sum_{i=1}^{n} W_{1i} * W_{2i}}{\sqrt{\sum_{i=1}^{n} (W_{1i})^2}\sqrt{\sum_{i=1}^{n} (W_{2i})^2}}$$

其中: W_1, W_2 为词向量, W_{1i}, W_{2i} 分别表示 W_1, W_2 的各分量。

如果两个词语的余弦相似度高于 0.9，那么将视它们为相似的文本。在数据扩增的过程中，不应该改变这些词语，并将它们保留下来。

3.3.2 基于语义的语句简化

核心语义保留的语句简化，目的是提高句法分析的精度，提高句法分析的效果，为基于句法树的数据扩增方法提供支撑。使用来自 Stanford 大学的 Stanford CoreNLP 工具，对文本的句子进行成分分析，依存句法分析。通过成分分析树和依存句法树，如果不含有并列成分和从句成分称之为简单句，否则称之为复合句。若判断为复合句，如果存在从句将从句独立成一个新的句子。如果存在并列成分，将并列句转换为非并列句。依据成分分析结果，将复合句拆分为简单句，来规避出现多次 S，V，O，C 的情况。

3.3.2.1 语句 5 大基本句式

核心语义，是构成语句的最重要的成分。经过语言学的研究，句子总共有五大基本类型，包括 SV，SVC，SVO，SVOO，SVOC 五种, 通过这些基本的结构能够组合成绝大多数语句，修改这些成分会导致句子语义产生巨大变化，所以核心语义部分不应该改变。下表 3-1就是五大基本类型的句子样例。

表 3-1: 语句 5 大基本类型

ID	Type	Example
$T1$	SV	$[Albert\ Einstein]_S\ [died]_V$.
$T2$	SVO	$[Albert\ Einstein]_S\ [has\ won]_V\ [the\ Nobel\ Prize]_O$..
$T3$	SVC	$[Albert\ Einstein]S\ [is]_V\ [smart]_C$.
$T4$	SVOO	$[The\ Royal\ Swedish\ Academy\ of\ Sciences]_S\ [gave]_V[Albert\ Einstein]_O\ [the\ Nobel\ Prize]_O$.
$T5$	SVOC	$[Albert\ Einstein]_S\ [declared]_V\ [the\ meeting]_O\ [open]_C$.

S:Subject, V:Verb, C:Complement, O:Object

S+V，即 (主语)+Verb(谓语)。主谓结构，顾名思义就是主语加谓语组成的句式。主语可以由一个或者多个谓语构成，谓语也可以由一个或者多个构成。

S+V+C，即 Subject(主语)+Link.V(系动词)+predicate(表语)。主系表结构是指英语句子中的主要成分是主语、系动词和表语。主语是一句话的中心，系词本身有一定的意义，不能单独使用，表语是用来修饰的主语的。

S+V+O，即 Subject(主语)+Verb(谓语)+Object(宾语)。主谓宾，句法顺序为主语—谓语—宾语的结构。

S+V+O+O，即 Subject(主语)+Verb(谓语)+Indirect object(间接宾语)+Direct object (直接宾语) 句子中有两个宾语时，其中指物或指事的就是直接宾语。指人（或动物）的就是间接宾语。间接宾语指受影响的事或人，直接宾语为动作的承受者，如 He passes me the ball. 中，me 为间接宾语，the ball 为直接宾语。

S+V+O+C，即 Subject(主语)+Verb(动词)+Object(宾语)+Complement(补语)。宾语补足语是指句子中的一些及物动词，虽然已经含有相应的宾语，但它仍然是不完整的，还有额外的句子成分来补充说明宾语的意义、状态等，并与其补足语构成复合宾语。复合宾语通常由两个部分构成，首先包含了名词和代词构成的简单宾语，其次包含了表示第一个部分简单宾语所发出的动作或者身份等特征，这个被称之为宾语补足语。

3.3.2.2 从句剔除方法

对于从句而言，完整的从句会影响依存句法树的精度，从句主要包括以下几种类型，主语从句，同位语从句，宾语从句，表语从句，定语从句，状语从句几种。我们使用成分分析树，来对从句进行剔除操作。我们使用 Stanford

CoreNLP 关于成分句法树正则匹配的规则，来进行从句的发现和剔除。

（1）主语从句

表 3-2: 主语从句

Transformation Pattern	Extracted Sentence
ROOT «: (S <(SBAR <,(WHNP)),,VP)	(SBAR <,(WHNP)
ROOT «: (S <(SBAR <,(IN)),,VP)	(SBAR <,(IN))
ROOT «: (S <(SBAR),,VP)	(SBAR)
Example: Who will be our monitor hasn’t been decided yet.	
Extract: Who will be our monitor.	
Result: he will be our monitor. he hasn’t been decided yet.	

主语从句的模式识别正如 3-2所示，主语从句的模式主要分为三种：第一种是提取到 (SBAR <,(WHNP)) 形式的主语从句，对 WHNP 成分进行识别，如果是 who，在主句中将主语从句替换为 he, 对提取出来的主语从句替换 WHNP 部分为 he；如果是非 who 类 WHNP 在主句中将主语从句替换为 it，对提取出来的主语从句替换 WHNP 部分为 it。第二种是提取到 (SBAR <,(IN) 形式的主语从句，对提取出来的主语从句 IN 部分进行删除（例如 Whether），在主句中将主语从句替换为 it。第三种是提取到 (SBAR) 在主句中将主语从句替换为 it, 保持主语从句内容部分不变。

（2）同位语从句表语从句

表 3-3: 同位语从句，表语从句

Transformation Pattern	Extracted Sentence
ROOT «: (S <(VP «VB$,,(SBAR))) and {}<ccomp	(SBAR)
Example: Who will be our monitor hasn’t been decided yet.	
Extract: I heard the news that our team had won.	
Result: I heard the news it. that our team had won.	

表语从句是说明主语是什么的从句，而同位语从句即重复说明同一个称谓

或事件的从句。表语从句和同位语从句的模式识别正如 3-3所示，同位语从句和表语从句的明确识别需要在这个句式的基础上使用依存句法分析，在依存句法树中从句的根结点是与主句是 ccomp 的关系。提取出来的从句在主句中原句移除相应的 SBAR，用 it 代替。在提取出来的从句中将这个 SBAR 独立成新的语句。

（3）定语从句

表 3-4: 定语从句

Transformation Pattern	Extracted Sentence
ROOT «: (S <(NP«NP,, (SBAR))) and {}<acl	(SBAR)
ROOT «: (S <(NP«NP.. (SBAR))) and {}<acl	(SBAR)
Example: The government which promises to cut taxes will be popular.	
Extract: which promises to cut taxes.	
Result: The government will be popular. which promises to cut taxes.	

定语从句是修饰名词或代词，对它进行限制、描绘和说明。定语从句的模式识别是 3-4, 定语从句的明确识别需要在这个句式的基础上使用依存句法分析，在依存句法树中从句的根节点是和主句是 acl 的关系，提取出来的从句在主句中删除相应的 SBAR，定语从句的删除并不会影响主句的完整性。所以只需要在提取出来的从句中将这个 SBAR 独立成新的句子。

（4）状语从句

表 3-5: 状语从句

Transformation Pattern	Extracted Sentence
S«:(SBAR) and {}<advcl	(SBAR)
Example: I know how to light a camp fire because I had done it before.	
Extract: because I had done it before.	
Result: I know how to light a camp fire. because I had done it before.	

状语从句通常用来说明主句中某一动词，形容词，副词，对其起状语作用

的从句。状语从句的模式识别是 3-5, 在依存句法树中，从句的根结点和主句是 advcl 的关系，提取出来的从句需要在主句中删除相应的 Sbar，状语从句的删除不会影响主句的完整性和语义的完备。所以只需要在提取出来的从句中将这个 SBAR 独立成一个新的句子。

3.3.2.3 并列识别方法

并列，是指内容排名顺序，不分高下。并列的合理拆分，直接提高了依存句法分析的效果，能够更好的构建依存句法树，并帮助更好的进行句型的识别。不同的并列有着不同的并列结构和方式，需要对每一种并列进行细致的分析。

(1) 语句级并列

两个及以上的简单句用并列连词或者逗号连接在一起，这样构成的句子叫做并列句，下表 3-6就是语句级并列的示例。语句级并列，会影响依存句法分析。语句的合理拆分，直接提高了依存句法分析的效果，能够更好的构建依存句法树，并帮助更好的进行句型的识别。不同的并列语句有着不同的并列结构和方式，需要对每一种并列进行细致的分析。

表 3-6: 语句并列示例

并列类型	示例
语句并列	You and I like fish,he like cat.

对于示例的句子进行成分句法分析，”You and I like fish,he like cat.”，如 3-2所示。ROOT 结点的子节点 S 代表着句子，是最高级的结构，一个 S 结点拥有多个 S 子节点，可以看作为一种语句级并列结构，识别语句方式如下表 3-7：

通过分析成分句法分析树，语句的 ROOT 结点是每个语句都存在的根结点。ROOT 结点的子节点 IP 代表着简单从句，一个 S 结点拥有多个 S 子节点，可以看作为一个并列句结构。如果遇到这种结构的语句，我们将这些子 S 节点的句子进行拆分，拆分成多个子句，消除语句级别的并列。如”You and I like fish,he like cat.”，变为”You and I like fish.”，”he like cat.” 两个句子。

(2) 字段级并列成分

表 3-7: 语句级并列

Transformation Pattern	Extracted Sentence
ROOT «:(IP < (IP $.. IP))	(IP$.. IP) 每个 IP
ROOT «:(S < (S $.. S))	(S$.. S) 的每一个 S
Example: You and I like fish,he like cat.	
Extract: S1:You and I like fish S2:he like cat	
Result: You and I like fish. he like cat.	

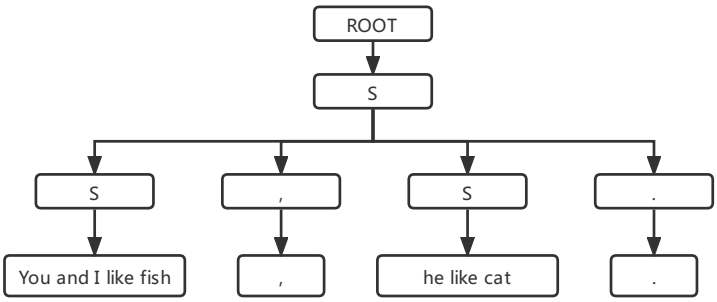

图 3-2: 语句示例 1

在语句级并列识别完成以后，仍存在字段级并列的内容，包括了主语字段并列，谓语字段并列，宾语字段并列，状语字段并列，表语字段并列等。由于他们的存在可能会影响对依存句法树的剪枝操作，并为了扩增更多有价值的语句，我们需要对其进行拆分操作。对于这些并列的识别，需要使用到依存句法分析的结果，我们使用 Stanford CoreNLP 的依存句法匹配规则去匹配并列成分。

（1）主语并列

对于主语并列而言，我们可以分析他的依存句法内容，比如有一个句子：”You and I like fish.”，我们可以看到他存在主语成分的并列，而在 Stanford-CoreNLP 的依存句法关系是一个通用依赖三元组，通用依赖 (UD) 是：关系名称、调控器和依赖所构成的。经过转换，我们把它修改为树状结构以便更好的操作。现在句子的依存句法树如下图 3-3所示：

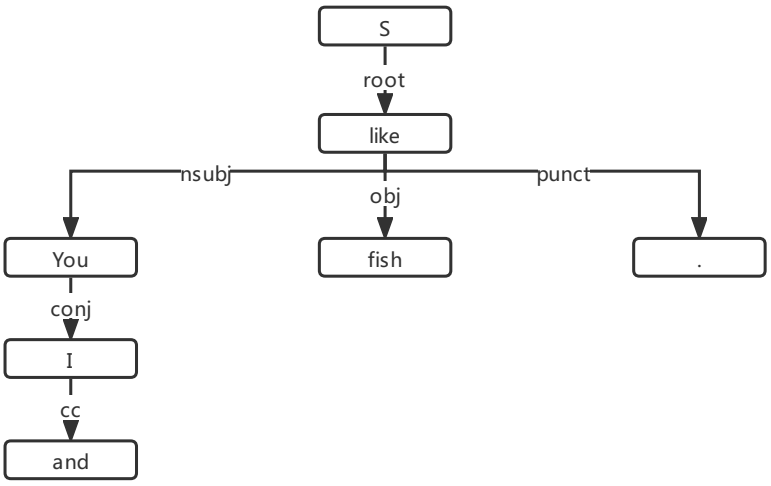

图 3-3: 主语并列示例

可以看到的是，S 作为一个句子的代指，他的根关系 root 是和动词相关联，而其他关系与这个动词关联，动词是这个句子的核心。nsubj 代表他所指向的对象是主语成分，而 conj 关系代表，I 与 YOU 是并列的关系，即我们可以通过 nsubj 和 conj 这几个关系去识别语句中的主语并列关系，显然我们可以得到一个匹配方法，匹配方式如表 3-8所示：

表 3-8: 主语并列

Transformation Pattern	Extracted Sentence
{} <conj ({} <nsubj {})	{} <conj 左部 {} 内容
Example: You and I like fish.	
Extract: and I	
Result: You like fish. I like fish.	

对获取到的主语并列成分，删除可能存在代表并列词 cc 的叶子结点 and，并将其设置为新的主语成分，这样我们就能够获得两个新的生成的句子，”You like fish.”，”I like fish.”，在大体保持语句意思不变的基础上，生成了两个新的语句。

（2）宾语并列

对于宾语并列而言，我们可以分析他的依存句法内容，比如有一个句子：

”You like fish and meat.”，我们可以看到他存在宾语成分的并列，现在句子的依存句法树如下图 3-4所示：

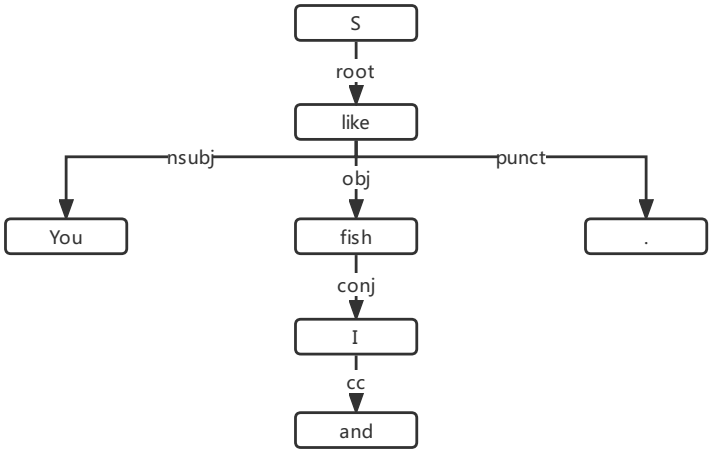

图 3-4: 宾语并列示例

可以看到的是，S 作为一个句子的代指，他的根关系 root 是和动词相关联，而其他关系与这个动词关联，动词是这个句子的核心。iobj/obj 代表他所指向的对象是宾语成分，而 conj 关系代表，fish 与 meat 是并列的关系，即我们可以通过 iobj，obj 和 conj 这几个关系去识别语句中的宾语并列关系，显然我们可以得到一个匹配方法，匹配方式如下 3-9：

表 3-9: 宾语并列

Transformation Pattern	Extracted Sentence
{} <conj ({} <iobj/obj {})	{} <conj 左部 {} 内容
Example: You like fish and meat.	
Extract: and meat	
Result: You like fish. You like meat.	

对获取到的宾语并列成分，删除可能存在代表并列词 cc 的叶子结点 and，并将其设置为新的宾语成分，这样我们就能够获得两个新的生成的句子，”You like fish.”，”You like meat.”，在大体保持语句意思不变的基础上，生成了两个新的语句。

（3）谓语并列

对于谓语并列而言，我们可以分析他的依存句法内容，比如有一个句子：”You often go shopping and sometimes swim.”，我们可以看到他存在宾语成分的并列，现在句子的依存句法树如下图 3-5所示：

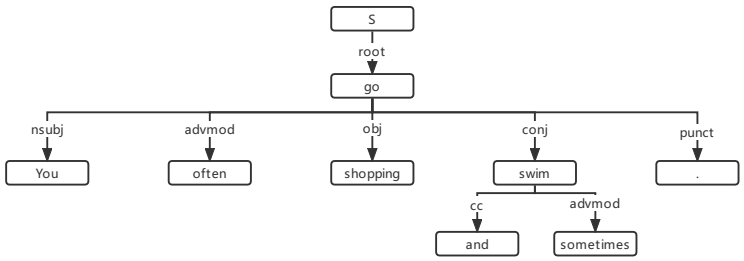

图 3-5: 语句示例 5

可以看到的是，S 作为一个句子的代指，他的根关系 root 是和动词相关联，而其他关系与这个动词关联，动词是这个句子的核心。root 代表他所指向的对象是谓语成分，而 conj 关系代表，often go shopping 和 sometimes swim 是并列的关系，即我们可以通过 root 和 conj 这几个关系去识别语句中的宾语并列关系，显然我们可以得到一个匹配方法，匹配方式如表 3-10：

表 3-10: 谓语并列

Transformation Pattern	Extracted Sentence
{} <conj ({} <root {})	{} <conj 左部 {} 内容
Example: You often go shopping and sometimes swim.	
Extract: and sometimes swim	
Result: You often go shopping. You sometimes swim.	

对获取到的谓语并列成分，删除可能存在代表并列词 cc 的叶子结点 and，并将其设置为新的宾语成分，这样我们就能够获得两个新的生成的句子，但这仍存在一些问题，比如 often 这个 advmod 成分不应该被复制到 sometimes 之前，即当 conj 存在于主语相同结构的左子树内容，就应该剔除，这样才能维持语句的合法性，最终生成了”You often go shopping”，”You sometimes swim.”，在大体保持语句意思不变的基础上，生成了两个新的语句。

（4）补语并列

对于补语并列而言，我们可以分析他的依存句法内容，比如有一个句子：”We believe him to be clever and wise.”，我们可以看到他存在补语成分的并列，现在句子的依存句法树如下图 3-6所示：

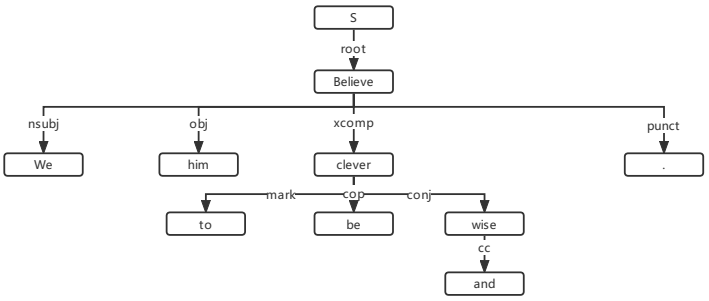

图 3-6: 补语并列示例

可以看到的是，S 作为一个句子的代指，他的根关系 root 是和动词相关联，而其他关系与这个动词关联，动词是这个句子的核心。xcomp/comp 代表他所指向的对象是补语成分，而 conj 关系代表，wise 与 clever 是并列的关系，即我们可以通过 xcomp/comp 和 conj 这几个关系去识别语句中的宾语并列关系，显然我们可以得到一个匹配方法，匹配方式如下 3-11：

表 3-11: 补语并列

Transformation Pattern	Extracted Sentence
{} <conj ({} <root {})	{} <conj 左部 {} 内容
Example: We believe him to be clever and wise.	
Extract: and wise	
Result: We believe him to be clever. We believe him to be wise.	

对获取到的补语并列成分，删除可能存在代表并列词 cc 的叶子结点 and，并将其设置为新的补语成分，这样我们就能够获得两个新的生成的句子，”We believe him to be clever.”，”We believe him to be wise.”，在大体保持语句意思不变的基础上，生成了两个新的语句。

（5）其他并列

当然，这不是所有的并列成分，仍然存在其他并列成分的内容，我们可以根据现有的结构，进行推广，即其匹配方式如下表 3-12：

表 3-12: 其他并列

Transformation Pattern	Extracted Sentence
{} <conj ({} <other relations {})	{}<conj 左部 {} 内容
Example: I sing in the classroom and at home.	
Extract: and at home.	
Result: I sing in the classroom. I sing at home.	

对获取到的其他并列成分，删除可能存在代表并列词 cc 的叶子结点 and，并将其设置为新的其他成分，这样我们就能够获得两个新的生成的句子，”I sing in the classroom.”，”I sing at home.”，在大体保持语句意思不变的基础上，生成了两个新的语句。

3.3.3 基于句法树的扩增方法

在完成从复合句到简单句的化简以后，现在依存句法树中，不再存在并列成分，完整的从句，我们需要在这里保留 SV，SVC，SVO，SVOO，SVOC 这些句子的基本结构，并保留依存关系中与关键词标签相似度较高的文本，通过对依存句法树的关系类型进行分析，保证关键词文本标签的不变性的前提下，对非重要成分的变换来扩增语句。

3.3.3.1 句法树剪枝规则扩增方法

字符级的删除是一种常用的加噪方式，可以用于文本数据的扩增。而简单的删除语句中非关键成分，发现会生成非法的语句，导致文本的歧义。所以需要研究一种能够不产生大量非法语句的词语删除方法。

本文提出了基于依存关系构建依存句法树，通过在依存句法树上对关键词文本非关键成分的操作实现合法的语句删除。通常而言，依存句法树层级越

深，节点对核心语义的影响越小，越属于不重要的成分。可以按照依存句法树的深度逐步删除不重要的成分，实现对于关键词文本的扩增。

在实验中，我们发现例如:With something 这种短语，当我们直接删除 something 的时候，会出现 With() 这种情况，导致语句并不合法，破坏了语义的可理解性，所以这种短语结构需要一起删除，我们称之为级联删除。如果一个成分的删除不会影响到其他语句成分，我们认为他是可以直接删除的成分。

最终通过实验分析，确定了 3 种需要处理的类型包括依存关系中不可删除成分，可以独立删除成分，需要与其他成分一起级联删除的成分，如下表 3-13所示, 其中类型-1, 0，1，2，3 分别代表-1 暂时不处理标点符号，0 可以直接删除，1 需要级联删除，2 存在并列相关内容，3 需要保留成分。

表 3-13: 依存关系识别表

关系	类型	关系	类型	关系	类型
punct	-1	aux	1	cc	2
amod	0	auxpass	1	conj	2
predet	0	ccomp	1	appos	2
preconj	0	xcomp	1	ref	2
advmod	0	expl	1	cop	3
neg	0	det	1	obj	3
tmod	0	mwe	1	dobj	3
nmod	0	mark	1	iobj	3
poss	0	nummod	1	nsubj	3
compound	0	prt	1	xsubj	3
other relationship	0	goeswith	1	ROOT	3
dep	1	case	1	key word	3

下面就是一个剪枝的示例，当从文章提取一个存在关键词成分的语句” Comparison with manual program repair has yielded unsurprising conclusions ”，关键词是 program repair, 使用之前我们所制定的剪枝方式。他的依存句法分析结

果如下图 3-7所示。

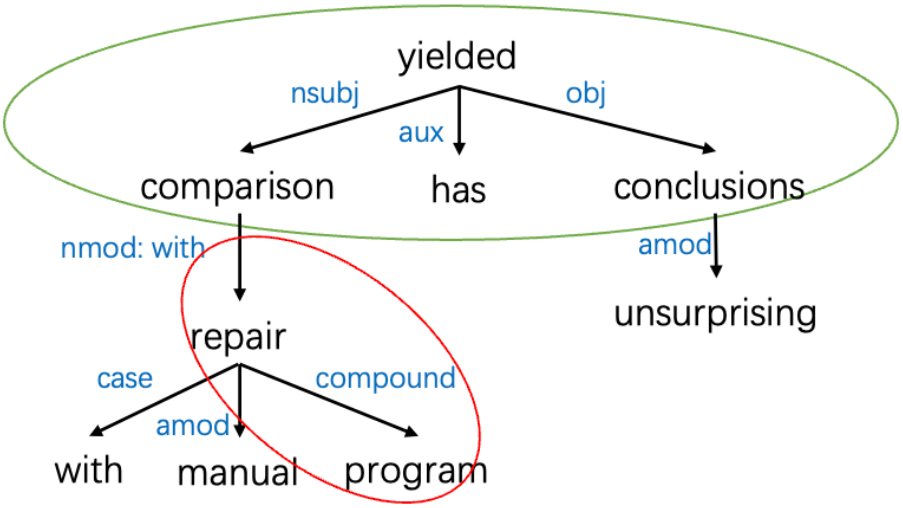

图 3-7: 依存分析结果示例

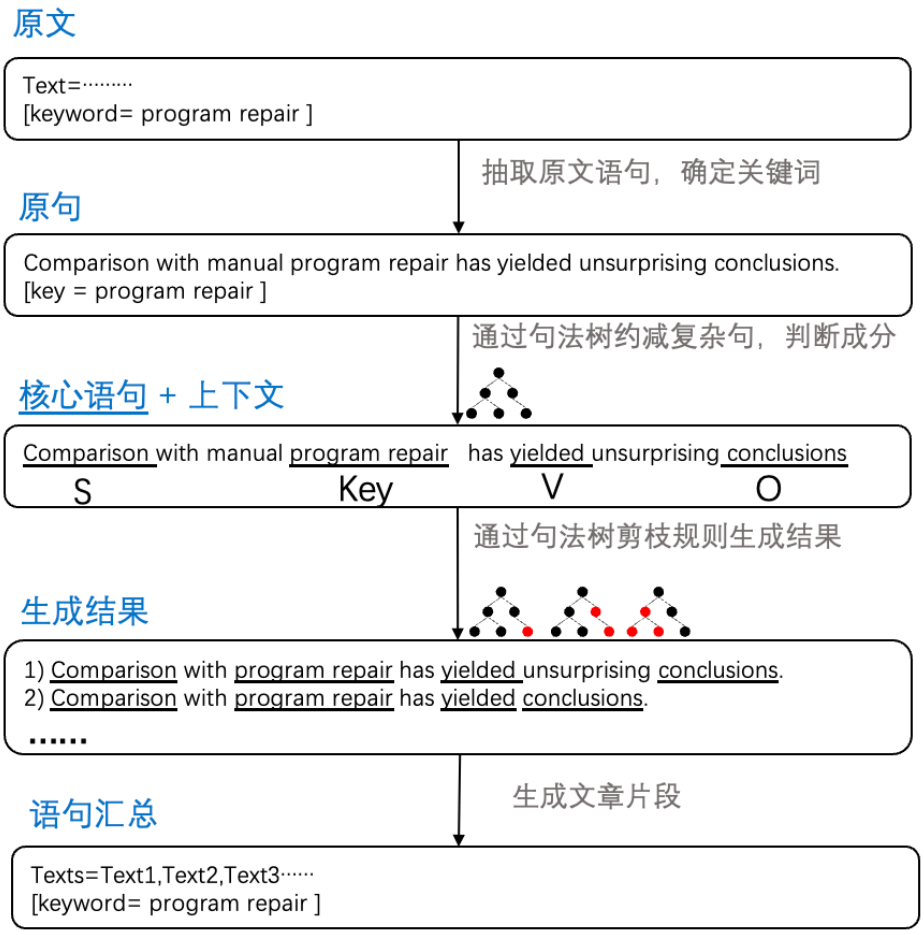

图 3-8: 句法树剪枝规则扩增方法示例

如图 3-8，首先在文本中提取出每个独立语句，确定文本的关键词成分。在识别出关键词以后，进行复杂语句约减，判断语句成分如果是核心语句，就需要进行保留，不能修改。我们可以看到主语成分 Comparison 关键词 program repair，动词成分 yield，宾语 conclusions 被标注了出来，在依存句法树剪枝的过程中依存句法树根据依存关系识别表，由依存句法树的从最高深度的节点开始，从树的左子树到树的右子树，对于每个成分查询，获得节点在关系表中是否可以直接删除，然后删除语句对应成分，记录新的扩增语句。

首先依次检查 with,manual,program 三个单词，发现 with 需要级联删除，所以他的标注是对号，需要暂时保留，而 manual，program 在依存句法分析结果是 compound 成分，但 program 属于 keyword 的组成部分，所以只能删除 manual 节点。然后查询他们的父节点，发现 comparison 是 nmod 类型，keyword 成分，不能删除。然后检查 unsurprising 节点，发现他是 amod 类型的节点，可以直接删除。最后发现树已经没有可以剪枝的成分，现在整个句子已经达到了最简单语句结构。图中第二个生成结果就是剪枝的部分过程最终的结果，最后不可删除的主干成分以及构成合法语句所需要的成分全部得到了保留。

3.3.3.2 句法树 BERT 填词扩增方法

最后是句法树 BERT 填词扩增方法，这种方法能够在结合句法树剪枝规则扩增方法的前提下使用 BERT 进行填词，提供更多扩增样本。普通的关键词文本和司法文书不同，用语比较固定。单纯使用 BERT 模型进行填词，在不考虑句子结构的前提下，仅仅填入少数几个词语的情况下，填词的质量和速度不如 EDA 中的同义词替换，通常同义词替换能够提供更好的填词效果和文本可理解性。并且随着填词的长度越长，BERT 模型填词的质量会有非常明显的下降，扩增文本失去语义上的逻辑和价值，最后导致文本质量下滑，对于关键词模型，失去可理解的意义和价值。所以本方法结合了句法树剪枝规则的扩增方法，在进行句法树剪枝之前，对要删除的叶子节点所处的语句位置进行遮蔽，让 BERT 模型填词，这样能够生成大量高质量的 BERT 填词结果。与单纯的 BERT 填词任务相比，强调了核心语义的重要性，仅对非核心语义成分进行了修改。

通过这个示例 3-9展示了句法树 BERT 填词扩增方法的大致流程。与句法树剪枝规则扩增方法类似，我们通过依存句法分析，识别文本的非重要成分，对非关键语义部分进行剪枝。与句法树剪枝规则扩增方法不同的是，该方法是

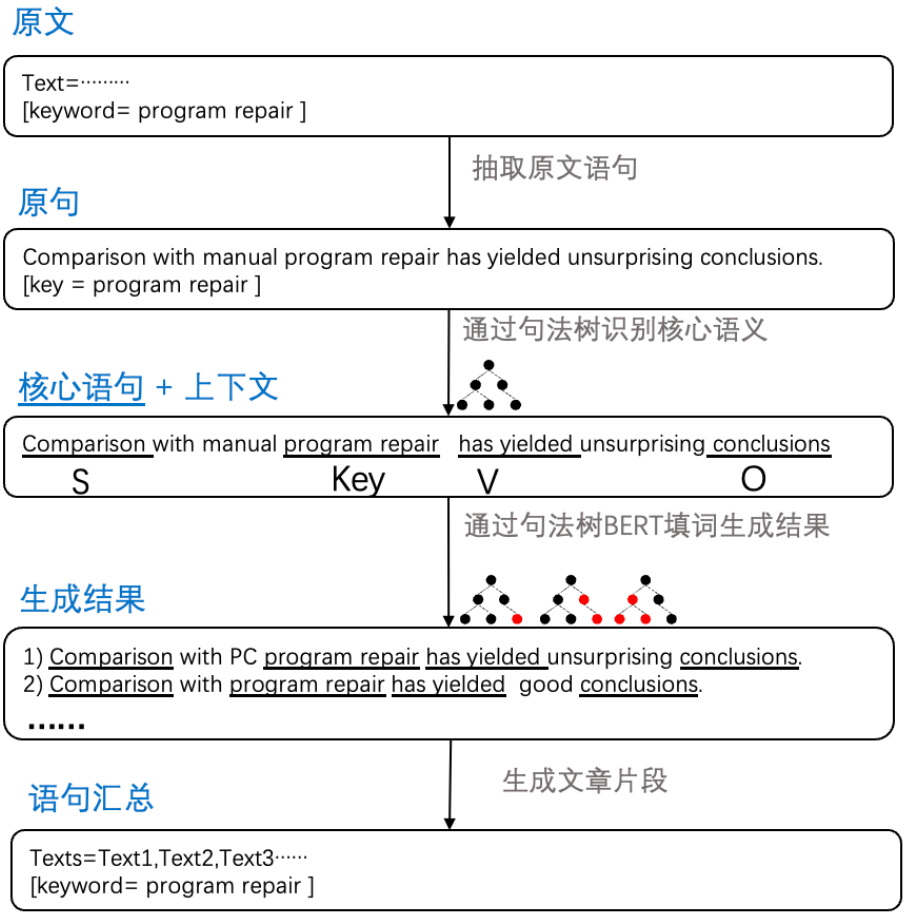

图 3-9: 句法树 BERT 填词扩增方法示例

对于要删除的成分进行遮蔽，通过进行 BERT Mask LM 填词任务，选择评分较高的几个返回结果，替换要剪枝的部分，生成了新的语句，并保存到扩增结果表，接下来将这个 Masked 部分剪枝。对比句法树剪枝规则扩增方法，实际上就是对剪枝成分的填词替换来生成新的语句，并且这次填词结果不会影响到下一次的剪枝填词过程。通过多次这样的遮蔽填词，BERT 模型填词结果对文本语义的影响被缩小，能获得大量高质量的扩增关键词文本数据。

3.4 本章小结

本章从关键词模型构建角度出发，研究了关键词文本数据的特点，介绍了我们的关键词模型文本的扩增技术。首先，对于句子每个 token 与关键词的相似度比较，将包含关键词信息的句子 token 成分设置为句子的重要成分，其次通过句法分析技术，分析文本语义上的特点，实现基于句法树，对文本进行从

复合句到简单句、从并列句到普通独立语句的变换。在基于简单句句子的基本句式前提下，对语句进行依存句法树构建，通过人为设计的句法树剪枝规则，对句子的非重要成分进行删除或者 BERT 填词，完成对文本每一个语句的扩增，最后随机拼接这些文本，实现有效扩增关键词文本数据。

第四章　需求分析与系统设计

需求分析与系统设计这章主要介绍了如何设计基于句法树的关键词文本扩增技术的关键词文本扩增系统。主要包括了两个部分通过对功能性需求和非功能性需求进行需求分析确定系统的内容和目的，通过概要设计，对系统的物理和逻辑实现进行界定。通过对概要设计划分的各个模块进行详细设计，阐述模块的类图和整体设计。

4.1　系统整体概述

基于监督学习方法的关键词提取模型逐渐成为研究与应用热点，使用深度学习技术的关键词提取模型如雨后春笋般出现。但仔细查看他们的实验，除了科研领域的关键词文本数据集较多以外，他们使用的数据集大多集中在少数两个数据集 OpenKP，KP20k。因为不同领域的关键词文本数据语料数据集较少，使用数据挖掘技术进行采集到的关键词数据集大多质量也难以保证，最终导致训练出来的关键词提取模型的泛化能力不高，精度不高等问题。

针对关键词模型文本特点，引入了基于句法树对文本进行变换的方法，将词向量，BERT 填词等技术加入了扩增方法中，最终本文设计了基于句法树的关键词模型文本扩增技术，并在本章进行关键词文本扩增系统设计。该系统在已经包括了基于句法树的关键词模型文本扩增技术的基础上，集成通用文本扩增方法中基于同义词替换的扩增方法和基于回译思想的回译扩增方法。该系统针对了关键词提取模型所使用的的关键词文本特征，能够生成高质量的关键词模型文本数据。

基于句法树的关键词文本扩增系统的目标是能够让用户轻松使用该系统扩增出他们所需要的关键词文本数据。本系统对关键词模型文本扩增流程如下：总体分为通用文本扩增和基于句法树的关键词模型文本的扩增。首先解析用户上传的关键词文本，通用文本扩增方法直接使用神经网络机器翻译实现回译扩增，通过预先设定的同义词字典实现同义词替换扩增；基于句法树的关键词模型文本的扩增通过计算词语的相似度，将输入文本中的关键词词语识别出来并

保留下来，结合基于句法树剪枝扩增分和 BERT 填词扩增方法对关键词文本进行扩增。以上部分共同组成了关键词模型文本数据扩增全过程。

基于关键词模型文本扩增流程，系统应该为用户能够提供关键词文本数据扩增的能力，管理扩增相关数据的能力和为了达到前两个能力所需要具备的其他能力。根据这些要求，基于句法树的关键词文本扩增系统可以将模块简单划分为关键词文本扩增模块、相关数据管理模块、其他支持模块这三个大模块，通过对这三个大模块的详细划分，能够提供用户所需要的所有功能。

4.2 需求分析

4.2.1 功能性需求分析

基于句法树的关键词文本扩增系统的功能性需求通过之前大概的业务流程确定了最基本的功能性需求，包括查看当前相关信息、对关键词文本进行数据扩增，对相关数据进行管理。根据这些基本的用户想要的内容，阐述具体的需求如下 4–1所示。

系统信息预览，为用户提供查看当前系统运行状态的基本信息，用户可以快速得知当前关键词文本扩增系统的运行状态和系统资源的概要信息。文本的上传下载是使用该系统的基础性前提，扩增需要提前上传好的关键词文本数据，用户使用扩增好的关键词文本数据也需要下载这些数据。数据扩增是系统建立的条件和目的，是核心功能性需求，整个扩增过程分为通用扩增和句法树扩增，通过 EDA 同义词替换，回译，基于句法树剪枝规则剪枝，结合句法树剪枝规则的 Bert 填词扩增，完成扩增任务。数据扩增项目管理是连接用户和系统的桥梁。具体来说用户针对不同的待扩增文件，可以创建不同的扩增项目。数据扩增任务管理是系统调用数据扩增模块的基础。具体来说用户针对当前的扩增项目，可以选择创建不同的参数的扩增任务，然后调用系统模块进行数据扩增。通过关键词文本文件管理模块，对关键词文本数据文件进行管理，具体包括文件的使用关系说明、文件创建时间的展示、文件的删除这几个部分。

表 4-1: 功能性需求列表

功能性需求	需求名称	需求描述
需求 1	系统信息预览	用户可以通过系统相应的界面查看当前系统的运行状态和任务完成情况
需求 2	关键词文本上传下载	用户使用关键词文本扩增系统的时候，可以上传和下载关键词文本数据。
需求 3	通用文本扩增	系统对项目需要进行扩增的关键词文本使用通用扩增方法进行扩增。
需求 4	句法分析文本扩增	系统对项目需要进行扩增的关键词文本使用基于句法分析的扩增方法进行扩增。
需求 5	数据扩增文件管理	用户能查看和管理系统历史的扩增文件。
需求 6	扩增任务管理	用户能查看和管理系统历史的扩增任务。
需求 7	扩增项目管理	用户能够创建扩增项目，并且每个扩增项目可以根据实际需求拥有多个扩增任务。

4.2.2 非功能性需求分析

如表 4-2所示，为系统的非功能性需求。非功能性需求是对功能性需求的补充，虽然不是系统工作流程的描述，但是系统不可或缺的一部分。非功能性需求是确保系统的健壮性和鲁棒性的方式，它对软件系统的规格提出了要求。本系统的非功能性需求包括了可靠性，易用性，可扩展性，可移植性。

可靠性，要求关键词文本扩增系统能够在绝大多数情况下都能够正确运行，如果用户没有按照关键词文本扩增系统预期的行为操作，需要正确处理这些异常情况，并保持系统的正常运行。

易用性，关键词文本扩增系统需要面向真实的用户，因此在系统的界面的设计上需要简洁高效，人机交互合理，符合人的直觉，无论是新接触系统的人还是熟练使用系统的人员都能够快速高效的上手该系统。

可扩展性，表示关键词文本扩增系统的功能之间通过合理的设计，将功能

表 4-2: 非功能性需求列表

非功能性需求	具体内容
可靠性	关键词文本扩增系统应该在用户使用的时候能够做出符合用户预期的行为，在用户使用的过程中，关键词文本扩增系统不应该出现不可用情况。
易用性	关键词文本扩增系统应该做到界面简洁，该系统操作逻辑符合人的直觉。
可扩展性	关键词文本扩增系统当前还处于初始状态，为了应对未来可能发生的变更等其他情况，系统应该能够较为简单的对其中的功能进行扩展或者修改。
可移植性	随着各种国产系统的发展壮大，本系统应当能适应不同操作系统服务器，并能够较快速的的部署。

分离成合理的模块，达到了低耦合的状态，系统可以较为方便的修改和迭代。

可移植性，要求系统能够在不同操作系统上运行，可以通过使用跨平台语言，不涉及对操作系统底层的修改和操作，构建关键词文本扩增系统。

4.2.3 用例分析

在完成对系统功能性需求的分析和描述之后，通过系统用例图 4-1了解系统的用户，需要使用什么样的服务。关键词文本扩增系统的使用者首先需要创建关键词文本扩增项目，上传关键词文本。然后在扩增任务管理中设定扩增参数创建关键词扩增任务，进行关键词文本扩增任务。关键词文本经过文本数据扩增之后，扩增的结果存储在系统的文件夹中，用户通过浏览扩增任务的详细信息下载该扩增任务的扩增关键词文本。

4.2.4 用例描述

通过对关键词文本扩增系统的用户进行用例分析之后，虽然已经有简单的系统用例。但还需要对这些用例进行详细的阐述，基于句法树的关键词模型文本扩增系统的用例如表 4-3所示，包括了系统信息预览、上传下载关键词文本、通用文本扩增、基于句法树的句法分析文本扩增、关键词文本扩增文件管理、关键词文本扩增任务管理和关键词文本扩增项目管理。通过对这些用例的

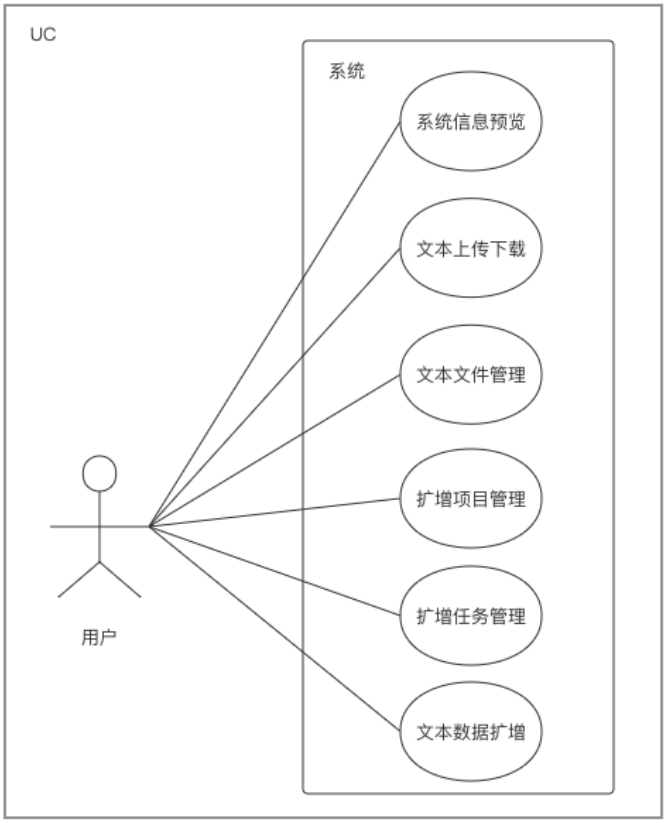

图 4-1: 系统用例图

描述，可以了解用户到底需要什么样的功能。

表 4-3: 系统用例列表

用例 ID	用例名称	需求 ID
用例 1	系统信息预览	需求 1
用例 2	上传下载关键词文本	需求 2
用例 3	通用文本扩增	需求 3
用例 4	关键词文本句法树分析扩增	需求 4
用例 5	关键词文本文件管理	需求 5
用例 6	关键词文本扩增任务管理	需求 6
用例 7	关键词文本扩增项目管理	需求 7

系统信息预览用例描述如表 4-4所示，系统信息预览界面用户能够快速了

解当前系统的运行状态，获得当前运行的任务，了解当前关键词文本扩增系统存储的文件数量。

表 4-4: 系统信息预览件用例描述

ID	用例 1
名称	系统信息预览
参与者	关键词文本扩增系统的使用者
描述	数据扩增系统用户使用信息预览描述
前置条件	打开关键词文本扩增系统前端页面
后置条件	无
优先级	低
正常流程	1. 用户从首页点击信息预览标签，进入信息预览页面。 2. 用户查看相关信息。
异常流程	无

上传下载关键词文本用例如表 4-5所示，在使用关键词文本扩增系统时，用户需要上传关键词文本。用户上传的关键词文本会存储到服务器本地文件夹中，存储前系统会计算文件的 sha1 哈希值，此举是为了避免存储相同的文件内容，如果系统已经存储了同样哈希值的文件，系统将不会保存该文件，而是将其文件地址改为数据库相同哈希值已存在的文件地址。如果系统数据库中不存在这个文件，存储时系统拼接当前系统的时间和随机的 5 位整数将其作为存储时文件名，文件的信息将会存储到系统数据库中，其中包括了文件的 ID，文件的哈希值，文件的系统存储地址等星系，这些信息的存储有助于文件的查找和管理。用户在关键词文本扩增任务完成以后，可以点击关键词文本扩增任务详情中下载按钮下载扩增后的关键词文本。在下载时，系统会自动生成相应的下载链接，让用户可以通过浏览器下载到用户本地文件夹中。

通用文本扩增模块用例如表 4-6所示。通用文本扩增模块是使用通用的文本扩增方法对关键词文本数据进行扩增，和基于句法分析的文本扩增模块一起构成了本系统的所有关键词文本扩增方式。这一部分这部分包括了 EDA 扩增和回译扩增。模块可以进行 EDA 中的同义词替换扩增方法，并可以使用回译这种基于神经网络机器翻译的技术，回译扩增的质量取决于机器翻译模型的质量。

句法分析文本扩增模块用例描述如表 4-7所示。基于句法分析的文本扩增会考虑到文本语义上的特征，首先比对文本内容和关键词内容设置为不可删除部分，然后基于依存句法树和成分句法树的变换规则，在不修改文本的核心语

表 4-5: 上传下载关键词文本用例描述

ID	用例 2
名称	上传下载关键词文本
参与者	关键词文本扩增系统的使用者
描述	上传下载关键词文本流程
前置条件	关键词文本上传： 1. 用户按照系统要求的格式准备好关键词文本数据 2. 用户处于项目创建页面 关键词文本下载： 1. 用户已经在系统中完成该扩增任务 2. 用户在扩增任务详情里面点击下载按钮
后置条件	关键词文本上传下载成功
优先级	高
正常流程	关键词文本上传： 1. 用户点击上传按钮，选择上传文件 2. 用户点击保存，系统存储该关键词文本 关键词文本下载： 1. 用户进入扩增任务详情页面 2. 用户点击下载按钮 3. 系统根据数据库文件信息从系统中下载文件至本地
异常流程	关键词文本上传： 1. 上传关键词文本格式非法，不是 json 格式 1a. angular 前端提示上传文件格式错误 1b. 使用者重新上传关键词文本 关键词文本下载： 1. 关键词文本扩增任务未完成 1a. 前端页面扩增任务详情中隐藏关键词文本下载按钮 1b. 用户等待扩增完成，再点击扩增任务详情界面进行下载

义的情况下，选择通过对文本非关键叶子节点的删除或者通过 BERT 填词任务填充其他词语，最后达到在文本大体语义不变的前提下，能够扩增出较多的文本，可以保证文本扩增的质量和语义上的可理解性。

关键词文本文件管理模块用例描述如表 4-8所示。关键词文本文件管理用于管理用户的关键词文本文件。用户进入文件管理页面之后，可以查看文件的相关信息。在文件的列表页面，可以快速查看使用该文件的扩增项目或者扩增任务。鉴于多个扩增项目可能会使用相同的待扩增关键词文本，关键词文本文件的删除前，会搜索数据库使用该文件的项目，如果有且仅有一个项目使用该文件，那么系统可以直接删除该文件内容，否则不会删除。

表 4-6: 通用文本扩增用例描述

ID	用例 3
名称	通用文本扩增
参与者	关键词文本扩增系统的使用者
描述	使用通用扩增方法，对文本进行扩增
前置条件	系统中存在扩增项目，并选择了扩增模式为通用文本扩增
后置条件	无
优先级	高
正常流程	1. 用户系统中创建了扩增任务，并设置扩增方式为通用文本扩增，并选择了具体扩增模式 2. 用户点击创建按钮以后，系统调用通用扩增模块，进行通用文本扩增 3. 扩增任务结束，系统存储扩增结果，并将信息存储到数据库中

表 4-7: 关键词文本句法分析扩增用例描述

ID	用例 4
名称	关键词文本句法树分析扩增
参与者	关键词文本扩增系统的使用者
描述	使用句法分析扩增方法，对文本进行扩增
前置条件	系统中存在扩增项目，并选择了扩增模式为句法分析扩增
后置条件	无
优先级	高
正常流程	1. 用户系统中创建了扩增任务，并设置扩增方式为句法分析文本扩增，并选择了具体扩增模式句法树剪枝规则扩增或者使用句法树 BERT 填词扩增 2. 用户点击创建按钮以后，系统调用句法分析扩增模块，进行句法分析文本扩增 3. 扩增任务结束，系统存储扩增结果，并将信息存储到数据库中。

扩增任务管理模块用例描述如表 4-9所示。扩增任务管理用于管理用户的创建的扩增任务。用户进入扩增任务列表以后，可以查看扩增信息，而且可以创建基于扩增项目的扩增任务。至于扩增任务的删除，系统删除扩增前，会查看扩增任务的状态，如果扩增任务已经完成，那么系统可以直接删除该扩增任务以及扩增文件内容，否则不会删除。

扩增项目管理模块用例描述如表 4-10所示。扩增任务管理用于管理用户的创建的扩增项目。用户进入扩增项目列表以后，可以查看扩增项目信息，而且可以创建 = 扩增项目。至于扩增项目的删除，系统删除扩增前，会查看扩增任务的状态，如果扩增任务已经完成，那么系统可以直接删除扩增项目以及相关的扩增文件，扩增任务，否则不会删除。

表 4-8: 关键词文本文件管理用例描述

ID	用例 5
名称	关键词文本文件管理
参与者	关键词文本扩增系统的使用者
描述	使用数据扩增文件管理模块，对系统内的文件进行管理
前置条件	用户点击进入文件管理页面
后置条件	无
优先级	中
正常流程	1. 用户进入关键词文本文件管理页面 2. 用户查看系统存储的关键词文本文件的相关信息 3. 如果不需要使用该文件，可以通过删除按钮删除文件
异常流程	1. 系统不存在任何关键词文本文件 1a. 系统返回空的文件列表 2. 系统删除多个项目使用的关键词文本文件 2a. 系统查看当前文件的使用情况 2b. 系统拒绝用户删除该关键词文本文件，并返回错误消息

表 4-9: 关键词文本扩增任务管理用例描述

ID	用例 6
名称	关键词文本扩增任务管理
参与者	关键词文本扩增系统的使用者
描述	使用任务管理模块，实现对关键词文本扩增任务的管理
前置条件	用户进入扩增任务管理界面
后置条件	如果需要删除的扩增任务不处于运行状态，则可以删除
优先级	高
正常流程	1. 用户进入扩增任务管理页面 2. 用户点击创建扩增任务 3. 用户通过下拉按钮选择该扩增任务归属的扩增项目 4. 用户输入合适的关键词文本扩增参数 5. 用户点击保存，创建关键词文本扩增任务 6. 用户在扩增任务列表中点击删除按钮删除关键词文本扩增任务
异常流程	1a. 用户关键词文本扩增任务创建失败 1. 用户创建信息非法，系统拒绝请求 2. 系统返回提示信息，用户重新创建任务 2a. 用户关键词文本扩增任务未完成 1. 用户无法删除关键词文本扩增任务

表 4-10: 关键词文本扩增项目管理用例描述

ID	用例 7
名称	关键词文本扩增项目管理
参与者	关键词文本扩增系统的使用者
描述	使用数据扩增项目管理模块，对系统的扩增项目进行管理
前置条件	用户点击进入关键词文本扩增项目管理页面
后置条件	扩增项目删除时，不能存在正在执行的扩增任务
优先级	中
正常流程	1. 用户点击进入扩增项目列表页面 2. 用户创建相关扩增关键词文本项目，填写相关信息，上传待扩增关键词文本 3. 用户点击删除按钮，删除扩增项目，并删除与之相关的扩增任务和扩增文件
异常流程	1. 用户扩增项目创建失败 1a. 用户创建信息非法或者未上传扩增文件，系统拒绝请求 1b. 系统返回提示信息，用户重新创建任务 2. 删除扩增项目时，该项目的扩增任务未完成 2a. 用户无法删除扩增项目 2b. 系统返回错误信息

4.3 系统总体设计

4.3.1 系统总体架构设计

基于句法树的关键词文本扩增系统架构设计如下 4-2，该系统基于前后端分离的思想，数据接口采用 RESTful 风格进行通讯。前端页面使用 Angular Web 开发框架进行开发，后端服务器使用基于 Python3 的 Django 框架进行开发。数据扩增部分也是使用 Python 语言进行开发的，通过 Django 封装成独立的服务模块。基于句法树的关键词文本扩增系统的数据持久化使用了 MySQL 这个数据库，该数据库易用上手方便，而且免费。BERT 模型，Stanford CoreNLP 也通过 Django 封装成外部的服务。

前端页面的开发使用了基于 TypeScript 的 Angular 框架，前端 UI 组件库使用的 ng-nest, 这是一个基于 Angular 的页面展示框架，有上手方便，开发简单以及样式美观的优点。通过 ng-nest 的 UI 组件，本文实现了前端的快速开发。在前后端交互方面全部使用 RESTful 风格的接口，系统采用了 RxJS 进行异步请求，系统通过 observable 序列高效进行基于事件的页面数据的更新。通过封装好的 HTTP 服务，前端的逻辑处理代码能够快速获得所需要的数据。

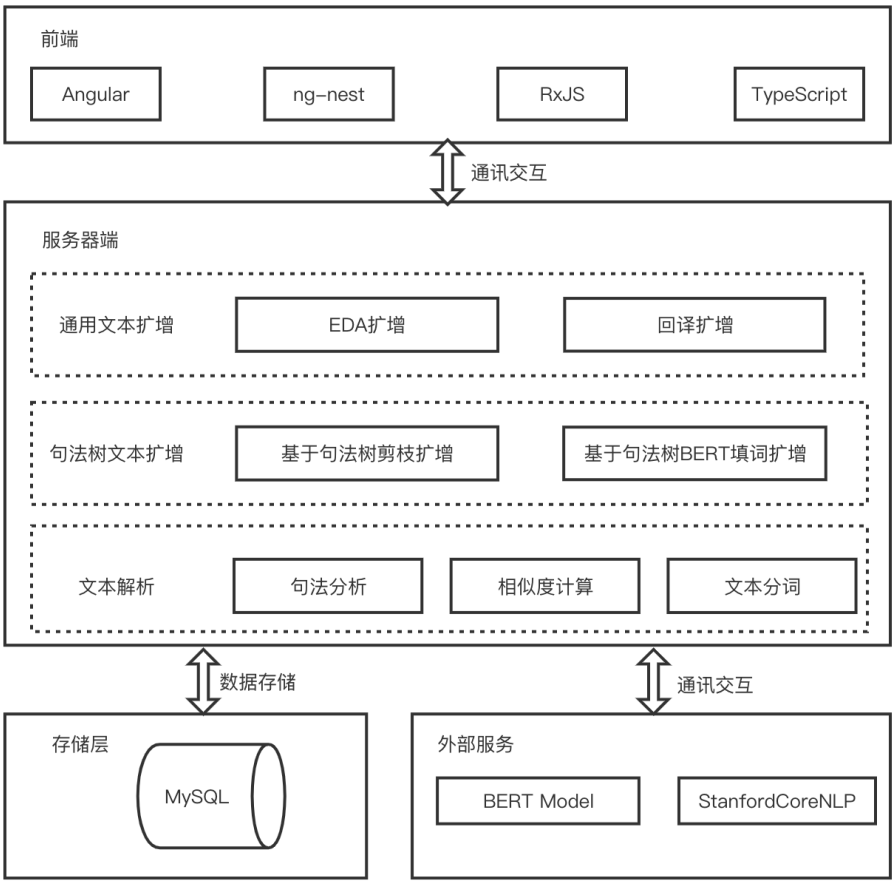

图 4-2: 系统架构图

基于句法树的关键词文本扩增系统服务器后端以 Django 后端框架为基础进行开发，对模型，控制器，服务进行了分离，通过分层的设计让每个部分都关注于逻辑代码本身。在数据存储方面，系统使用 Mysql 来进行数据的持久化。BERT 模型和 Stanford CoreNLP 工具作为单独的服务与系统通过 RESTful 风格的 HTTP 请求进行交互。

4.3.2 4+1 视图

本文通过 4+1 视图对基于句法树的关键词文本扩增系统进行逻辑上的描述，场景视图已经在用户的用例描述中详细的阐述了用户的使用场景。而其他视图通过不同参与者的不同视角来综合描述基于句法树的关键词文本扩增系统的逻辑设计。

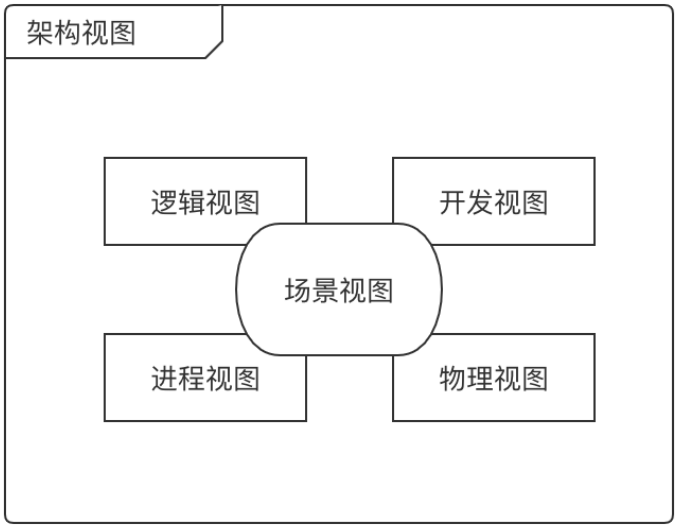

图 4-3: 4+1 视图

逻辑视图从关键词文本扩增业务上的角度来介绍关键词文本扩增系统，逻辑视图是面向系统用户的，本系统中的用户是基于句法树的关键词文本扩增系统的使用者，描述的是系统提供给用户的服务，这里通过核心类图展示，具体如图所示 4-4。在逻辑视图中，本文所有服务都继承于 HttpResponse 来实现网络通讯的基本能力。DataInteraction 是负责数据交互的类，提供了系统所需要的概要信息，并提供了关键词文本扩增相关信息的存储，读取。通过抽象 DataInteraction 这个类，提供抽象数据库的能力，现在是 MySQL 数据为系统提供数据库数据读写和查询服务，但通过合理的接口设计，能够隔离数据库的影响。TextAnalysis 在系统中负责通过服务，解析文本数据进行句法分析，文本的相似度判断功能，文本的分词功能。DataAugmentation 是系统数据扩增的接口，他被 UniversalAugmentation 和 ParseAugmentation 两个类实现。其中 UniversalAugmentation 是系统进行通用关键词文本扩增所使用的基类，其子类 BackTranslateAugmentation 和 Easy Augmentation，分别实现了回译扩增和 easy data Augmentation 的功能，ParseAugmentation 是句法分析关键词文本扩增的基类，其子类包括了 ParseRuleAugmentation 和 ParseBERTAugmentation。其中 ParseRuleAugmentation 是单纯的基于依存句法树的剪枝规则设计的扩增方法，而 ParseBERTAugmentation 结合了依存句法树的剪枝规则和 BERT mask LM 填词任务进行填词设计的扩增方法。

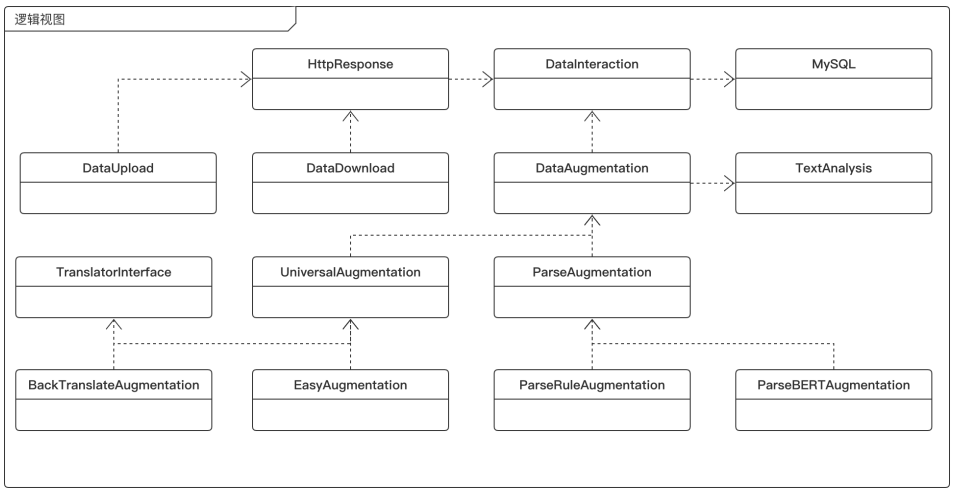

图 4-4: 系统逻辑视图

在阐述了系统的逻辑视图之后，我们通过开发视图 4-5，从系统开发的角度来描绘系统。开发视图展现了基于句法树的关键词文本扩增系统代码组织结构，通过开发视图，开发者能够快速了解本文所开发系统的大概状况。开发视图通过分层展示了系统由 UI 部分，服务部分和其他外部依赖部分构成。

UI 层是由 Angular、Ng—Nest、RxJS 和 TypeScript 的构成，通过 UI 层用户控制基于句法树的关键词文本扩增系统的业务流程。在 UI 层，用户能够对相关功能进行管理，上传和下载关键词文本。

服务部分主要描述了基于句法树的关键词文本扩增系统的业务功能的实现。通过扩增方法，本关键词文本扩增系统提供了 4 种关键词文本扩增方法，包括了基于句法树剪枝的关键词文本扩增方法，基于句法树 BERT 填词的关键词文本扩增方法，基于回译的关键词文本扩增方法和基于同义词替换的关键词文本扩增方法。在数据交互服务中，提供了对相关数据存储的能力，并提供了前端返回系统当前状态的能力。而在管理服务中，包括了扩增任务管理，扩增文件管理和扩增项目管理，通过这些管理服务提供了对整个数据扩增的控制。文本分析服务是分离了较为复杂的一些逻辑，从业务逻辑中分离出来，为后续的变更提供相关的准备。文本分析服务包括了文本相似度的计算服务，句法分析服务，分词服务。

最后是技术服务层，是使用的底层技术支持服务，这些服务通常需要消耗大量资源，通过分离这些服务能够为用户提供分开部署提高系统性能的能力，包括了 MySQL 的数据存储，BERT 模型和 Stanford CoreNLP 文本分析工具。

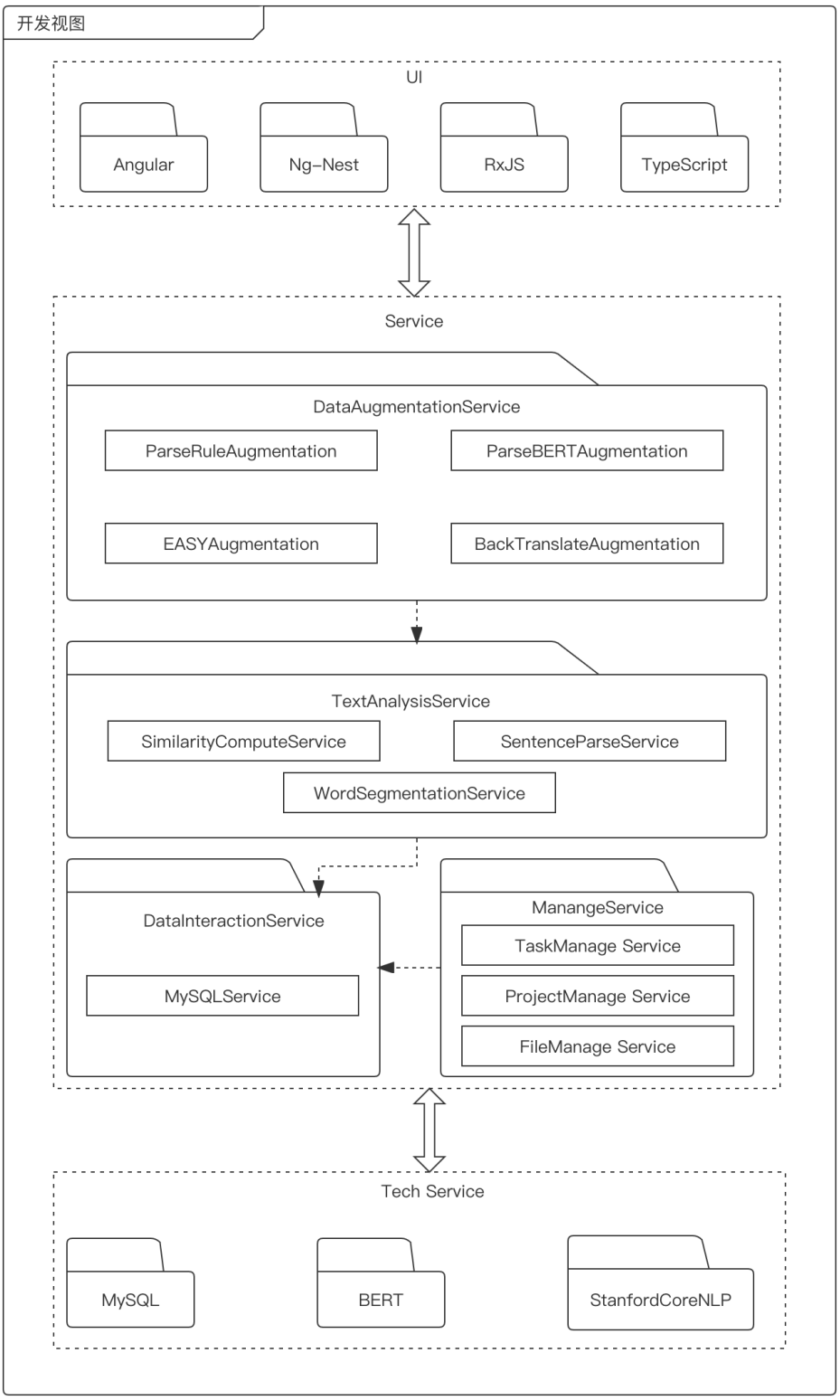
开发视图
UI
Angular
Ng-Nest
RxJS
TypeScript
Service
DataAugmentationService
ParseRuleAugmentation
ParseBERTAugmentation
EASYAugmentation
BackTranslateAugmentation
TextAnalysisService
SimilarityComputeService
SentenceParseService
WordSegmentationService
DataInteractionService
MySQLService
ManangeService
TaskManage Service
ProjectManage Service
FileManage Service
Tech Service
MySQL
BERT
StanfordCoreNLP

图 4-5: 系统开发视图

而系统的进程视图 4-6，主要描述了系统核心进程如何与其他进程进行交互，满足需求分析中的非功能性需求。为了响应用户的请求，基于句法树的关键词文本扩增系统的主逻辑进程需要一直存在。当用户进入系统前端页面，访问关键词文本扩增项目管理页面，创建相关项目的时候，进程就开始运行。首先在创建关键词文本扩增项目时，主进程会向数据库上传相关的信息。接着用户去关键词扩增任务管理界面创建该项目的关键词文本扩增任务，系统会依照扩增任务所填写的相关扩增信息，向关键词文本扩增进程提交有相关扩增参数的扩增任务。关键词文本扩增进程调用文本解析进程对关键词文本进行分析，文本解析进程调用外部服务进程进行关键词文本的句法分析，分词，关键词相似度判断等任务。然后数据扩增进程在进行基于句法树 BERT 填词扩增的时候会调用外部服务进程，使用 BERT 模型提供的服务，进行文本填词任务。最后将扩增好的关键词文本和相关信息存储到数据库中，完成关键词文本的扩增任务，文本会存储在电脑内文件夹中。最后当需要下载相关的扩增后的关键词文本时，主进程调用 dataDownload 获取文件相关信息，用户通过浏览器下载扩增后的关键词文文本数据。GetInfo 函数将前端界面所需要的全部信息返回到前端中，用于信息的展示和查看。

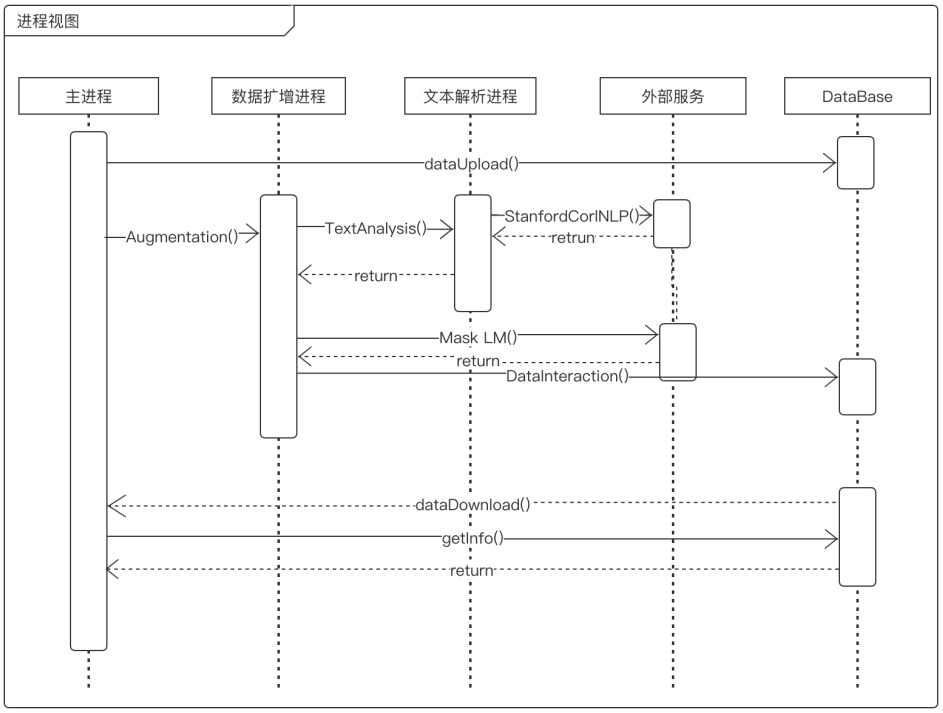

图 4-6: 系统进程视图

系统的物理视图 4-7，主要描述了系统软件和物理硬件的对应关系。用户在客户端使用浏览器访问基于句法树的关键词文本扩增系统。前端发送请求到 APP Server 服务器进行相应的逻辑处理，APP Server 服务器与数据库服务器进行通讯，存储关键词文本扩增中产生的相关信息。数据扩增 Server 与 APP Server 进行通讯，确认关键词文本扩增任务，并进行关键词文本的扩增，然后存储相关数据到数据库服务器中。数据扩增 Server 会调用外部服务 Server 中 CoreNLP 节点进行句法分析，调用相似度计算节点对关键词相似度进行计算，调用 BERT 模型进行关键词文本填词任务。

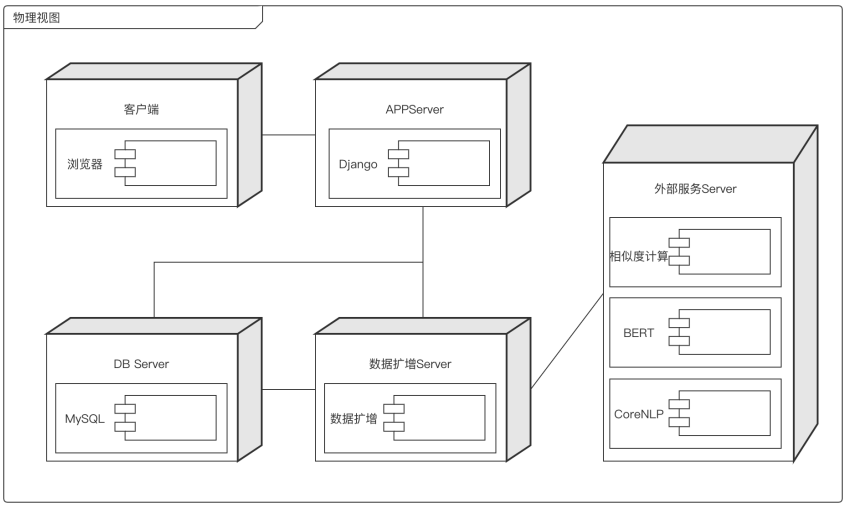

图 4-7: 系统物理视图

4.3.3 系统基本功能模块

在讨论了系统 4+1 视图之后，对基于句法树的关键词文本扩增系统进行功能模块的划分，确定功能模块的设计。如表所示 4-8，系统的基本功能模块包括了关键词文本扩增系统的数据管理模块、针对关键词文本数据扩增模块、负责数据流转的数据交互模块、还有为数据扩增模块提供服务的文本解析模块。

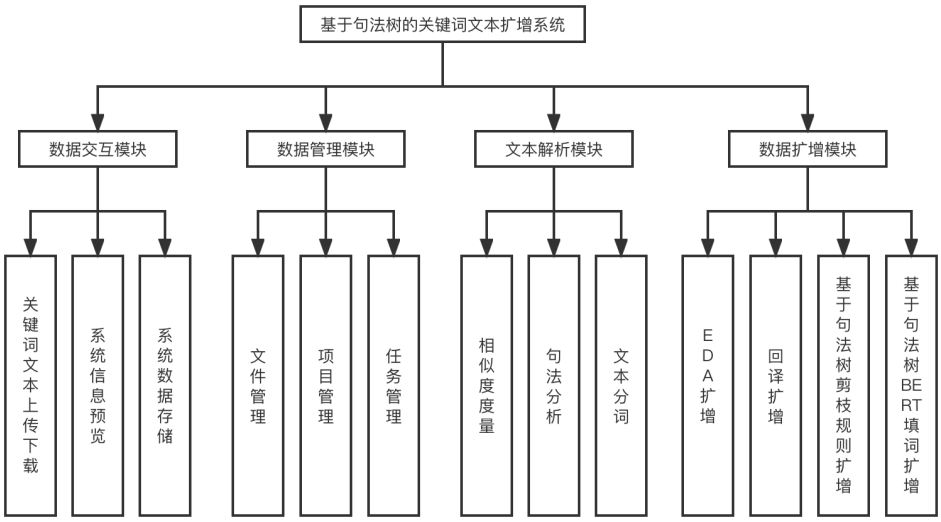

图 4-8: 系统基本功能模块图

对系统基本的功能模块继续进行拆分: 数据交互模块提供了系统相关数据存储，系统信息预览和关键词文本上传下载能力；数据管理模块主要是由关键词文本扩增项目的管理，关键词文本扩增文件的管理和关键词文本扩增任务的管理组成；文本解析模块提供了相似度度量，句法分析，文本分词这三个功能；数据扩增模块提供了 EDA 扩增，回译扩增，基于句法树剪枝规则扩增，基于句法树 BERT 填词扩增。

4.4 数据交互模块设计

在进行了系统的基本功能模块划分之后，对数据交互模块进行详细的设计。数据交互模块主要用于存储关键词文本扩增系统在运行过程中需要持久化存储的相关信息，为跨域的关键词文本文件上传下载提供支持，向系统的用户提供关键词文本扩增系统信息预览的能力。

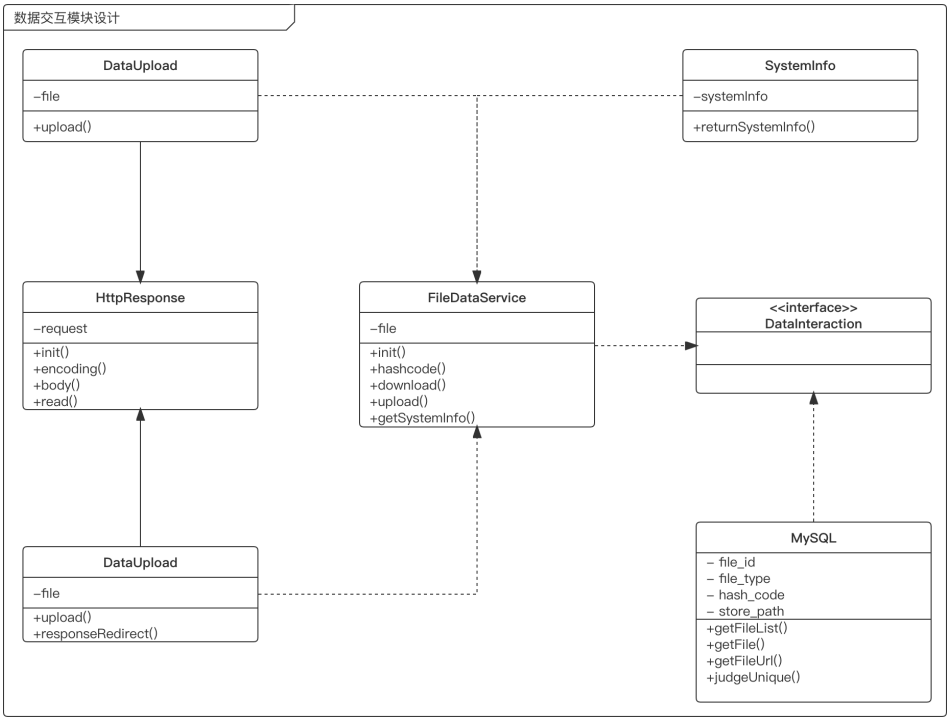

图 4-9: 数据交互模块核心类图

通过类图 4-9，我们可以快速了解数据交互模块是由什么类构成的。该模块主要有 DataInteraction 抽象类用于对具体数据库的抽象分离，DataUpload、DataDownload 实现跨域的文件上传和下载，FileDataService 提供对相关文件与系统交互的管理。通过 SystemInfoService 获得当前系统运行的概要信息。

显然，在使用过程中，用户可能会重复上传同一个文件多次，为了节省存储资源，降低系统的资源占用，应避免同一个文件的重复上传。本系统在存储文件时引入了 SHA-1 安全散列算法，SHA-1 算法与 SHA256 相比有速度快很多的特点，虽然 SHA256 有更高的安全性，但在本系统中不需要这些特性。该算法通过对文件的进行哈希运算，获得独特的 160 位特征码哈希值，特征码高度离散而且不可逆，在统计学中每个关键词文本数据应该是独特的。

利用 SHA-1 算法的特点，本系统会对上传的 Json 文本数据进行哈希计算，得到对应的哈希值，然后存储到系统数据库 MySQL 中，如果用户再次上传了同样内容的关键词文本数据，那么这个文本会与数据库记录的文本哈希值相重复。系统判断已有相应内容的关键词文本数据，新传入的文件将不会进行存储。但考虑到虽然文件内容相同但文件名不同的情况，系统在处理用户上传的时候也会存储用户上传时的文件名，能够帮用户更好的区分文件信息，而不是

需要下载下来查看具体的内容。对于文件的删除的管理就比较简单，系统数据库 MySQL 会比对系统内扩增项目，扩增任务的相关信息，如果存在多于一个项目，任务引用这个文件，系统会检查并保持，而不是直接删除这个文件。

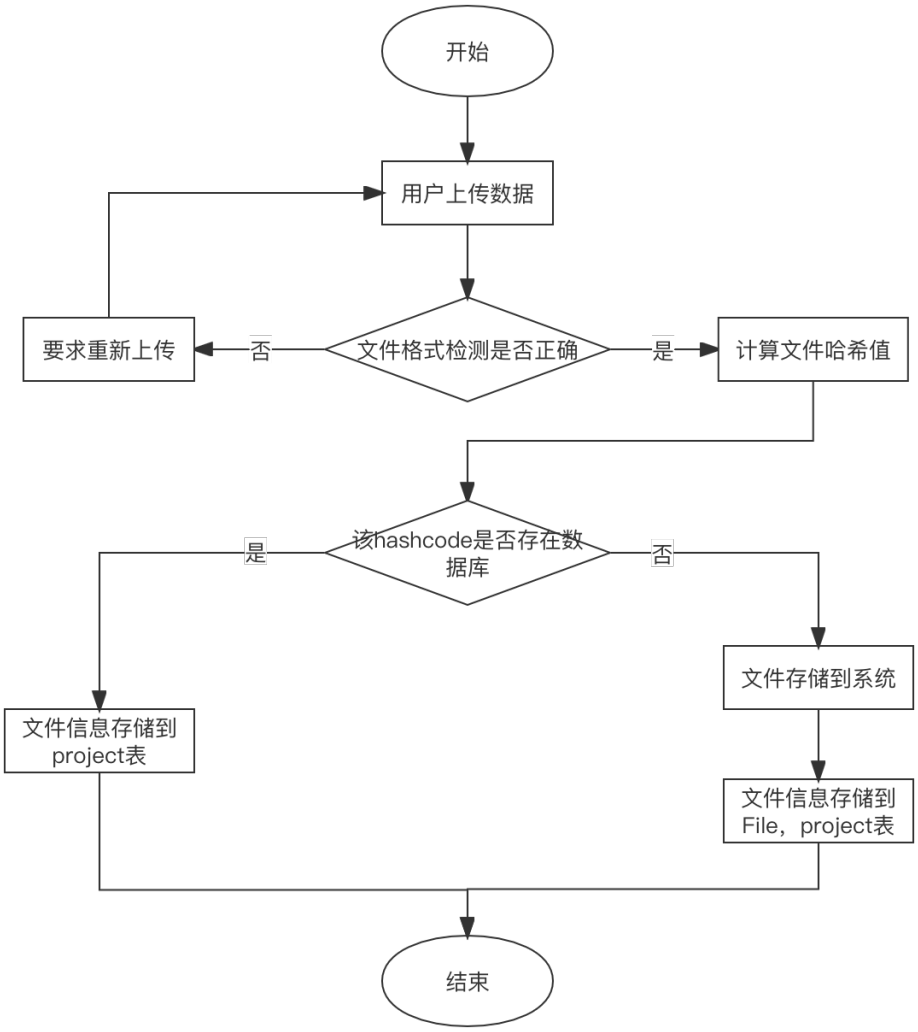

图 4-10: 文件存储流程

系统存储流程如图 4-10所示，当新的关键词文本文件被上传时，系统前端界面会检测文件格式，如果文件格式非法，会给用户发出反馈，要求重新上传文件。如果文件格式合法，系统会计算关键词文本文件内容的 SHA-1 哈希值，并与数据库中的 file 表中的 hashcode 进行比较，如果新上传的文件不是重复的，则系统会按照时间加随机数的名称存储文件，并将文件相关信息存储到 MySQL 数据库中，并修改 Project 表中的相关信息，能够为用户保存独特的文

件名称等信息，并将文件存储到系统中。如果新上传的文件是重复的，则系统会直接修改 project 表中关于文件的相关信息。

4.5 数据管理模块设计

数据管理模块是传统的系统实现所必须的内容，它承担了数据的管理工作，通过对数据管理模块的设计，能够达到高内聚低耦合的特性。通过数据管理模块实现对数据库的操作，能够让前端获取到所需要的数据，对页面进行渲染。

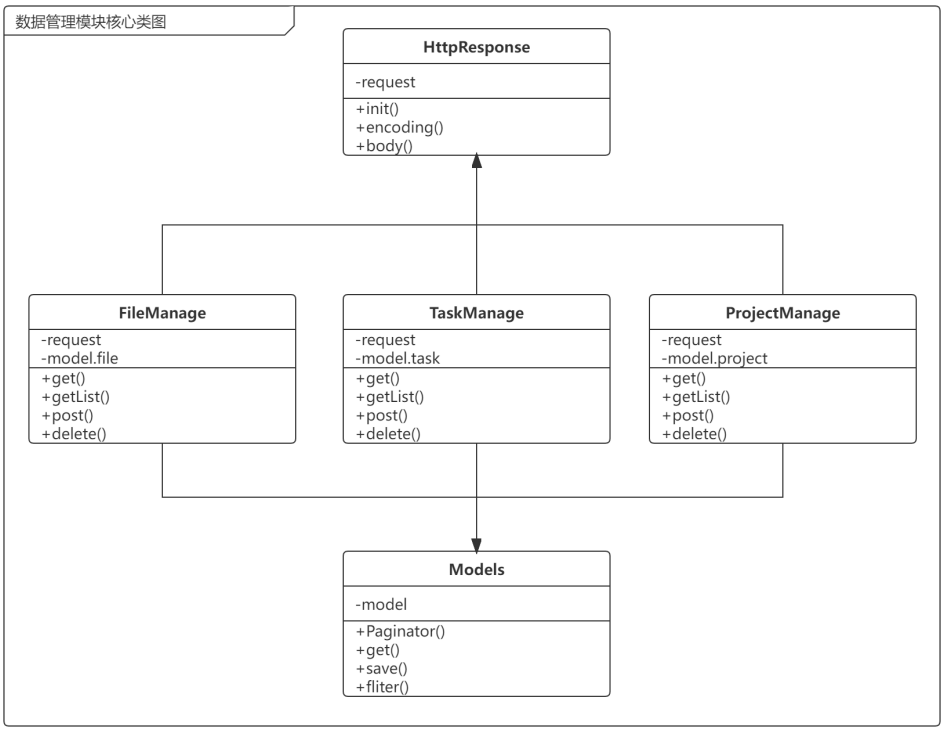

图 4-11: 数据管理模块核心类图

通过数据管理模块的核心类图 4-11，我们可以快速了解该模块功能和类的对应关系。该类图主要包括了 HttpResponse、Model、FileManage、TaskManage、ProjectManage 构成，FileManage、TaskManage、ProjectManage 继承于 HttpResponse，承担了与前端交互的功能，通过这些类实现了前端所需要的所有信息内容的提供和操作的能力。对于这些模块的操作，需要考虑到相互之间的影响。比如：删除一个 Project，就需要删除与这个 project 相应的数据扩

增 task，以及所涉及到的 File。如果这个 File 被其他项目所使用的，那就不需要删除这个 File，仅需要删除相应的 project 和 task。

4.6 数据扩增模块设计

对于基于句法树的关键词文本扩增系统，最重要的模块设计就是数据扩增模块的设计。关键词文本扩增模块决定了该系统的使用价值。如图 4-12所示，关键词文本扩增模块主要包括了通用文本扩增方法和基于句法树的关键词文本扩增方法。其中通用文本扩增模块包括了 EDA 扩增和回译扩增。EDA 扩增用于用户验证和作为基线的数据扩增方法，回译扩增用于关键词文本扩增。而句法树文本扩增方法，提供了基于句法树剪枝规则的扩增方法和基于句法树 BERT 填词的扩增方法。

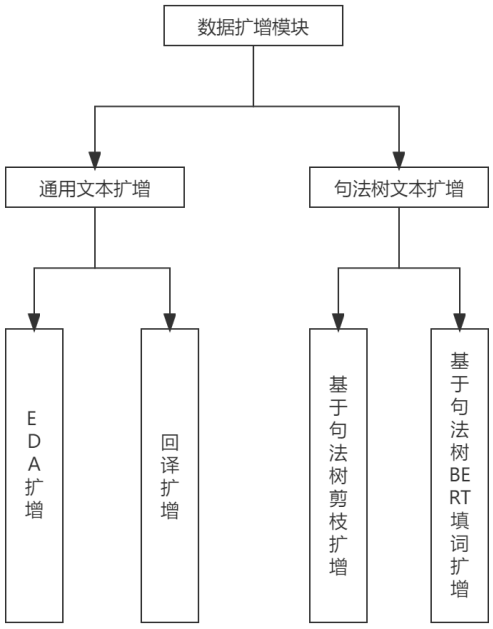

图 4-12: 数据扩增模块功能图

通过关键词文本扩增系统中数据扩增模块的核心类图 4-13, 可以了解扩增模块是如何实现低耦合设计的,DataAugmentation 是该模块的基类，该类包括了其组件部分的一些通用接口，例如存取读取文件，保存相关信息的功能等。UniAugmentation 和 ParseAugmentation 类是 DataAugmentation 的一部分，两者分别是通用文本扩增和句法树分布扩增的组件，它们包含了数据扩增所需要的

基础函数。

EDAAugmentation 和 BackTranslationAugmentation 是通用文本扩增的组件，功能是进行通用文本扩增，EDA 扩增作为 baseline，可以用于验证数据的有效性，BackTranslationAugmentation 使用回译扩增技术，扩增的效果由神经机器翻译 NMT 模型决定扩增的质量和效果，理论上而言单纯的回译扩增不会改变数据的语义。BackTranslationAugmentation 通过选择不同在线翻译的接口，可以获得不同神经机器翻译的结果。

BertParseAugmentation 和 ParsePruneAugmentation 是和 ParseAugmentation 的组件，提供基于基于句法树裁剪的方法和基于句法树 BERT 填词的方法。基于句法树裁剪的方法，只是对句法树非关键部分按照依存句法树进行删除，扩增数据。而基于句法树 BERT 填词，是针对与句法树非关键部分进行 BERT 模型填词，实现数据扩增理论上，不会大幅度改变文本的语义，也能提供相较于同义词替换更丰富的扩增结果。

在完成数据扩增之后，扩增结果会以 Json 文本的形式保存到文件系统中，并调用其他模块，对 Project，Task 状态等信息进行修改，并为用户提供下载接口。

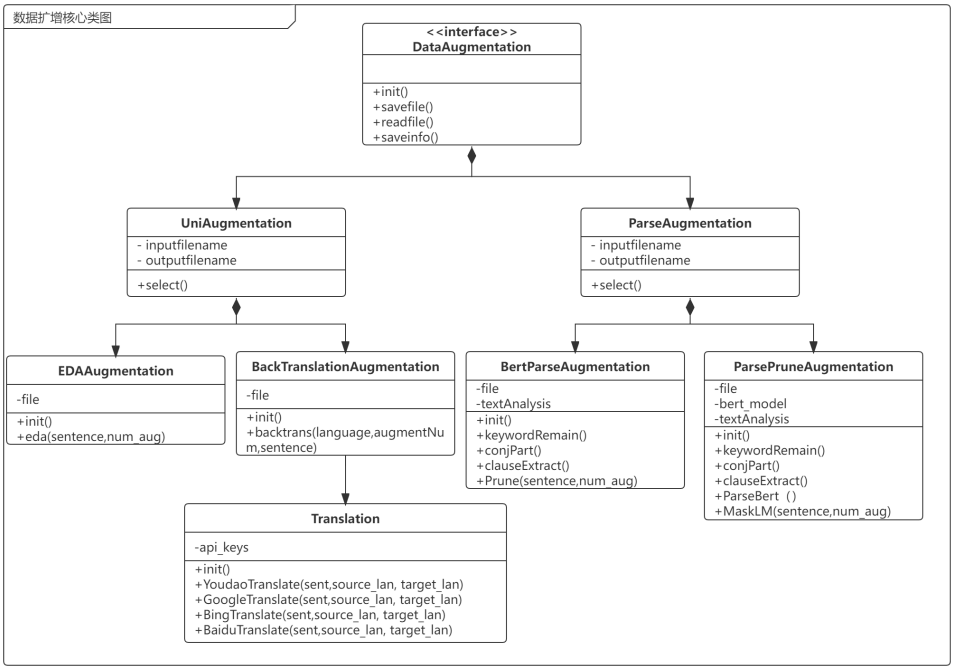

图 4-13: 数据扩增核心类图

4.7 文本分析模块设计

文本分析模块相当于一个工具模块，为基于句法树的数据扩增模块提供工具上的帮助。独立文本分析模块的原因是文本分析的功能需要需要占用大量的系统资源，会降低关键词文本扩增系统的运行速度，这些功能如果出现故障，会影响系统的运行。所以需要将这些功能进行独立，提高关键词文本扩增系统的鲁棒性，降低系统的耦合程度。

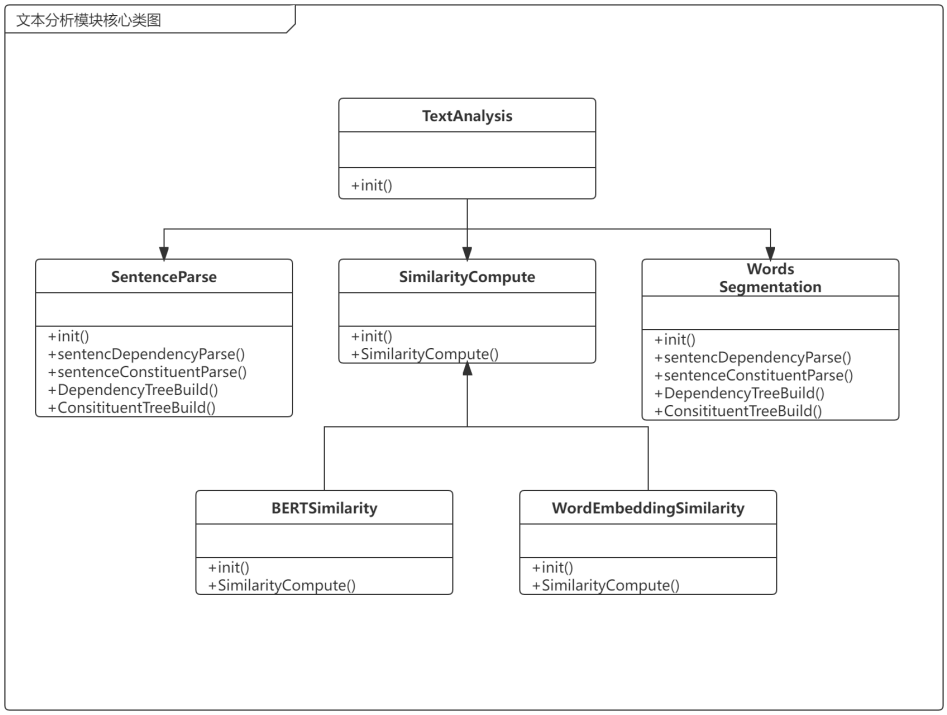

图 4-14: 文本分析模块核心类图

文本分析模块的核心类图主要如图 4-14所示，模块内主要有 WordsSegmentation、SimilarityCompute、SentenceParse，TextAnalysis 四个类构成。其中其他三个类组成 TextAnalysis 类。WordSegmentation 定义了用于分词的函数，提供分词的能力。SimilarityCompute 类是一个基类，提供了计算文本相似度的能力，WordEmbeddingSimilarity，BERTSimilarity 继承于这个基类，提供了不同的相似度计算能力，相较而言 BERT 准确率会更高，但 BERT 所需要的资源远超于 WordEmbedding 计算相似度，用户可以在配置文件中修改所需要的相似度计算方法，通常而言使用基于 Word2Vec 的文本相似度已经足够使用，并且该方法运行速度会高于 BERT 方法，所以系统中默认启用 WordEmbedded-

ingSimilarity 计算方法。而 SentenceParse 类是本模块的核心类，整个系统的构建都依赖于句法分析能力。这个类提供了依存句法分析和成分句法分析能力，为了方便对依存句法分析，成分句法分析结果进行操作，还需要提供构建依存句法树和成分句法树的能力。

4.8 本章小结

本章研究了基于句法树的关键词文本扩增系统的相关背景和意义。通过需求分析，明确了要开发什么样的关键词文本扩增系统。通过系统的总体设计，确定了系统功能模块的划分。使用低耦合、高内聚的设计思想，对系统数据交互模块、关键词文本扩增模块、关键词文本扩增系统所需要的相关管理模块、文本解析模块进行详细的设计。

第五章　系统实现与测试

本章主要介绍了基于句法树的关键词文本扩增系统的实现，对该系统的相关功能进行测试并对基于句法树的关键词文本扩增技术的有效性和实用性进行相关的实验。系统的实现包括了关键词文本扩增系统数据交互模块的实现，文本分析模块的实现，数据管理模块的实现和数据扩增模块的实现。

5.1　数据交互模块实现

数据交互模块负责存储关键词文本扩增系统在运行中需要持久化的数据，并为需要跨域访问的关键词文本上传下载提供支持，而且提供了对系统运行状况预览的相关能力。

通过信息预览界面 5-1，能够查看到当前基于句法树的关键词文本扩增系统的项目数量，正在运行的关键词文本扩增任务的数量。

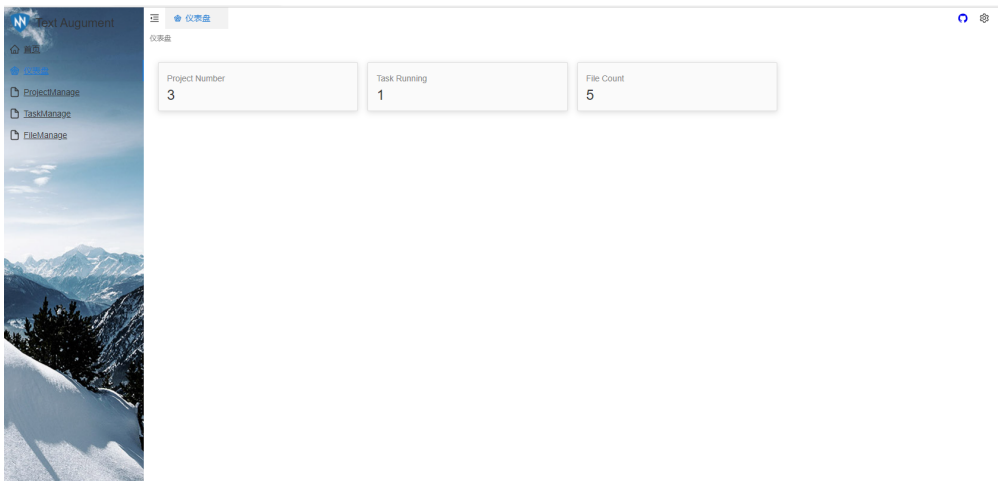

图 5-1: 信息预览

正如在第四章对数据交互模块进行的类图描述一样，系统中关键词文本文件上传通过 Data Upload 组件进行负责，由于在前端中进行了上传文件格式的校验，所以后端不再需要对文件格式进行二次校验，只需要对文本内容进行合法性校验，即 json 文件中是否对关键词文本数据是否包括了标题、摘要与关键

词内容。DataInteractionService 组件存储相关数据到 MySQL 中，保存该文件到系统文件夹中。用户点击已完成的关键词扩增任务详情页面，可查看到下载按钮。DataDownload 组件负责将跨域的关键词文本传输到前端页面，浏览器通过前端页面提供的链接下载扩增以后的关键词文本数据。

对于可能遇到的大量相同文件存储，本系统通过计算 SHA-1 哈希值保证结果的唯一性。如果输入信息有任何的不同，输出的对应摘要不同的几率非常高。考虑到如果读取较大的文件会存在内存不够的情况，所以每次读取 10240 字节的信息，通过分段读取降低内存紧张的可能性，这样能够提高系统的鲁棒性。

```
def CalcSha1(filepath):
    with open(filepath, 'rb') as f:
        sha1obj = hashlib.sha1()
        while True:
            data = f.read(10240)
            if not data:
                break
            sha1obj.update(data)
        hash = sha1obj.hexdigest()
    return hash
```

图 5-2: 计算 Sha1 关键代码

5.2 数据管理模块实现

数据管理模块，用于管理系统的相关的信息，包括了扩增项目的管理，扩增任务的管理和扩增文件的管理。虽然业务内容不同，但本质他们是几乎相同类型的业务，也具有极高的相似性。本系统的扩增项目管理列表如图所示 5-3。

项目列表展示了当前存在的项目，每一行都是一个项目。对于每个项目，UI 会显示其项目名称，创建时间等关键性信息。系统会在用户点击进入相关界面的时候，在后端中从 MySQL 数据库中检索相关的信息，按照前端页面的格式进行展示。

用户可以方便的在关键词文本扩增系统的对应管理模块中对自己的关键词文本扩增项目，任务，文件进行管理操作，包括查看详细信息，删除等常用操作。例如在关键词文本扩增项目管理页面，编辑按钮提供用户修改，编辑相关

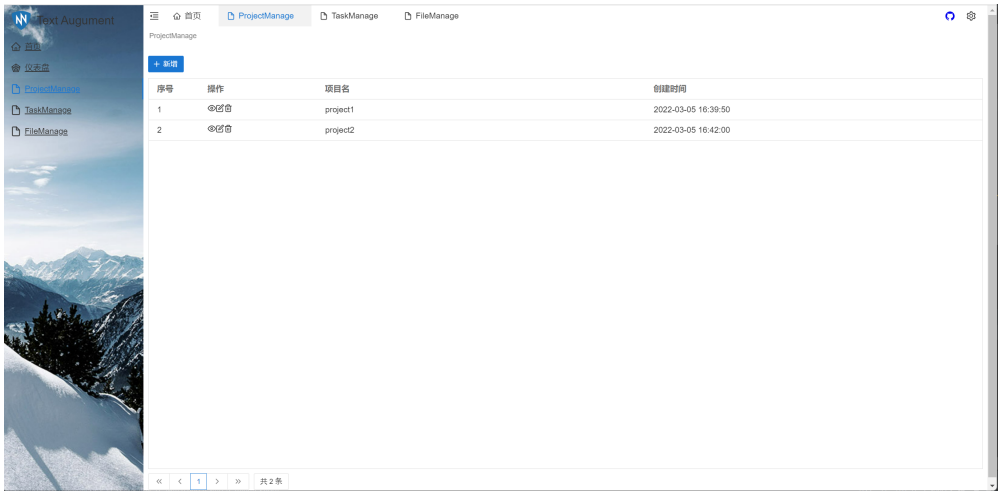

图 5-3: 项目管理界面

项目的能力。删除按钮提供用户删除对应关键词文本扩增项目的能力，项目连同他相关的扩增任务以及仅有他使用的文件会直接删除，在文件夹中删除这些文件。点击创建按钮，会进入关键词文本扩增项目创建页面，通过填写相关信息，可以创建关键词文本扩增项目。

5.3 文本分析模块实现

文本分析模块是为了帮助数据扩增模块更好的运行，是一个工具模块。文本分析模块将一些可以从数据扩增模块分离出来的功能分离出来，提供更高的模块化能力。而这些功能需要占用相当多的资源，比如相似度度量需要提前将词向量模型加载到内存中。而使用 StanfordCoreNLP 进行文本分词，句法分析也需要使用大量资源，一个中文模型需要 3GB 左右的内存，英文模型也需要使用 2GB 以上的内存。将这个模块独立出来，为以后多端部署提供可能性。这样可以节约单个节点的负载状况，提高系统的运行效率。

在成分句法分析的过程中，NLTK 已经实现了将成分分析结果转换为树状结构的形式，我们可以很方便的对其进行树的操作。而在进行依存句法分析以后，StanfordCoreNLP 返回的是代表关系的四元组列表数据，没有较为简便的方法，并不方便我们未来对其进行操作分析。所以本文在文本分析模块实现将依存句法分析结果转换为树状数据结构存储的方法。下图 5-4即为依存句法树构建的关键代码。

通过 Treelib 库提供的相关函数，我们可以较为轻松的构建依存句法树，并

```
def dependencyTree(dependency, token):
    """
    构建 dependencytree
    :param dependency: 依存关系元组
    :param token: 分词结果列表
    :return: 依存关系树
    """
    tree = []
    dependencyParse = copy.deepcopy(dependency)
    token = copy.deepcopy(token)
    tree = Tr()
    root = -1
    # 构建依存树
    dependencyParse.sort(key=lambda x: x[1])
    while (len(dependencyParse) != 0):
        dependencyParseItem = dependencyParse.pop(0)
        i, begin, end = dependencyParseItem
        if begin == 0:
            root = end
            tree.create_node(token[end - 1], end, data=node(i, 0, token[end - 1]))
            continue
        elif tree.contains(begin):
            # print(token[end-1],end,begin)
            tree.create_node(token[end - 1], end, parent=begin, data=node(i, 0,
token[end - 1]))
        elif len(dependencyParse) >= 1:
            dependencyParse.append(dependencyParseItem)
    return tree, root
```

图 5-4: 构建依存句法树关键代码

对依存句法树进行剪枝。这个函数读取 StanfordCoreNLP 解析出来的依存句法树和分词结果，基于这些信息构建依存句法树，最后返回整个树状数据结构和根结点索引。

5.4 扩增模块实现

关键词文本数据扩增模块将未扩增关键词文本数据经过本文设计的关键词文本扩增技术，扩增出新的高质量关键词文本数据。关于关键词文本数据扩增模块的实现主要包括了基于通用扩增方法的实现和基于基于句法树扩增方法的实现，具体的系统实现会在接下来的内容里介绍。

5.4.1 通用扩增模块

本系统的通用扩增方法共有两种，分别为 EDA 扩增中的同义词扩增和基于神经网络翻译的回译扩增方法。

EDA 扩增的具体实现是：根据具体需要扩增的关键词文本所涉及的领域，获取到一批文本数据，对文本数据进行 word2vec 生成他们的词向量，相似的词向量被视为同义词，我们在每一句中进行同义词的随机替换，就可以生成大量的扩增语句，由于只是替换同义词，通常情况并不会大幅度修改文本的语义上的意思。

本系统提供了相应的词向量训练接口，通过训练文本的词向量，能够获得更高精度的词语的同义词信息，获得更高的扩增质量。如图 5-5是训练词向量的关键代码，可以通过调用相应的接口，设置词向量的训练参数，对词向量模型进行训练。通过 synonyms.nearby(word)[0] 这个代码能够获得当前单词的近义词，通过随机替换非停止词的近义词，本系统实现了基于 eda 的关键词文本扩增技术。

```
@csrf_exempt
@api_view(http_method_names=['post'])  # 只允许 post
@permission_classes((permissions.AllowAny,))
def train_model(request):
    ……#语料文件预处理
    file= open(seg_corpus_path, 'w', encoding='utf-8')
    for s in seg_list:
        file.write(s + ' ')
    tmp_model = Word2Vec(LineSentence(seg_corpus_path), window=window,
min_count=min_count, vector_size=dim,
                         workers=multiprocessing.cpu_count())
    tmp_model.save(model_dir + "word2vec.model")
    tmp_model.wv.save_word2vec_format(model_dir + "word2vec.vector", binary=True)
    return Response({"code": 200, "msg": "模型训练完成！", "data": ""})
```

图 5-5: 训练词向量模型关键代码

回译扩增的具体实现是：根据扩增关键词文本的语言，翻译到其他语言以后生成翻译结果，再将翻译结果翻译回原始关键词文本的语言。考虑到当前神经网络翻译 NMT 所支持的语言类型存在限制，如果想要生成更多的结果，可以重复这一个流程，直到生成足够多的数据，随着会议次数的增加，语义改变的可能性会逐步增加，翻译结果的可理解性会逐渐下降。

本系统提供了多种翻译的接口，包括 Bing,Google,Baidu,Youdao 这几种常用的神经网络翻译器，他们提供了可以快速使用的 API，在本系统中抽象翻译接口，可以随意设置想要的翻译接口，获得更好的效果。

5.4.2 句法树扩增模块

句法扩增模块是相较于使用通用扩增方法设计的通用扩增模块之外的基于句法分析的扩增方式。句法扩增模块包括了句法树 BERT 扩增方法，句法树剪枝扩增方法。

5.4.2.1 句法树剪枝扩增模块实现

句法树剪枝模块会对语句进行句法分析之后的所构建的依存句法树进行剪枝，生成新的句子，实现扩增的技术。基于有效的句法树剪枝规则，能够生成合法有效的剪枝结果。

如下图 5-6所示，整个句法树剪枝扩增流程，首先读取待扩增关键词文本数据，进行句法树分析，在通过并列识别，删除并列成分以后。再通过从句识别，删除多余的从句。当现在的语句都是简单句以后，生成相应的依存句法树。通过设计的剪枝规则，判断依存句法树的节点是否可以直接删除或需要连同其他节点删除。在删除的过程中保存句法树剪枝的每一个中间状态，通过这些中间状态最后生成潜在语句列表，组合这些潜在语句可以生成大量的句法树剪枝扩增结果。

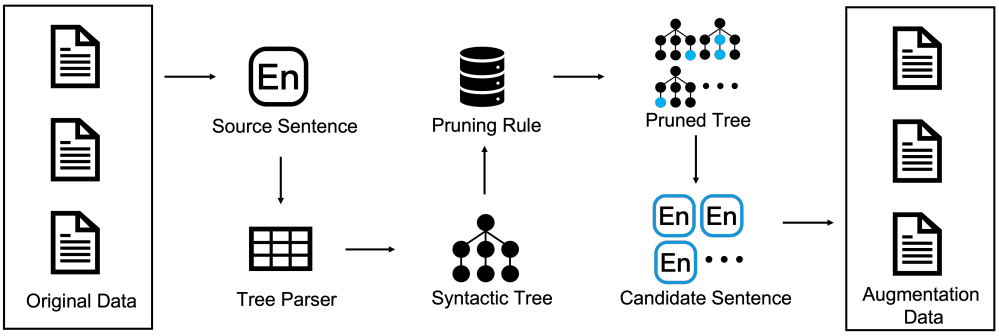

图 5-6: 句法树剪枝扩增流程

依存句法分析随着语句长度的增加，依存句法分析的精度会逐渐下降，甚至产生完全错误的依存句法分析结果。如果一个语句长度非常长，通常情况下为了提高文本的可理解性，文章的作者们会使用多种从句来构成长语句。为了

提高依存句法分析效果，需要将复杂的语句拆分为简单的语句。下图 5-7是句法从句识别的部分代码，正如在本文第三章所提到主语从句识别模式。本文通过对于我们预设的主语从句的判别方式，能够正确识别主语从句，本代码提取到 (SBAR <,(WHNP)) 形式的主语从句，对 WHNP 成分进行识别，如果是 who，在主句中将主语从句替换为 he, 对提取出来的主语从句替换 WHNP 部分为 he；如果是非 who 类 WHNP 在主句中将主语从句替换为 it，对提取出来的主语从句替换 WHNP 部分为 it。其他从句识别方法虽然实现的细节略有差异，但每一种从句识别的方法逻辑是一样的。

下图 5-8是基于句法树剪枝流程伪代码, 考虑到源代码的距离较长，简单概述一下主要的代码的实现逻辑。关于句法树的剪枝流程，首先按照深度优先的方式将当前节点的子节点加入队列。然后读取剪枝规则表，如果存在与关键词一致的成分，将不可剪枝成分设置为保留，可以直接删除的关系设置为直接删除，如果需要和其他词语一起删除的部分，设置为级联删除关系，并设置为相应的字典便于后续的读取。接着对于每个加入队列的节点，如果没有子节点，通过 REMOVE() 函数判断是否可以直接删除，可以剪枝就直接删除，如果不能剪枝就暂时保留。如果有子节点就递归剪枝他的子节点内容。最后如果当前节点可以直接删除，那么当前节点连同他的子节点就一起删除。而在这中间，减去每一个节点的过程中，都生成相应的句子，加入生成列表中。随着语句的剪枝过程，能够生成大量的中间语句。

最后生成的数据，是按照需要扩增语句的数量对经过 nltk 分句后的语句列表，对其中每个语句基于句法树剪枝规则扩增后的组合随机挑选。与常规方法相比，该方法能够生成大量的扩增结果，并不改变语句的核心语义。

5.4.2.2 句法树 BERT 扩增模块实现

而基于句法树 BERT 扩增方法的核心思路大体与句法树剪枝规则扩增方法一致，只不过在进行剪枝的同时，对剪枝减去的部分使用 BERT 填词，这样能够达到更多样性的扩增结果。由于是对于语句非核心部分的内容进行修改，不会大幅度修改文本的核心语义。

基于句法树的 BERT 填词扩增，其大体流程与基于句法树剪枝规则的扩增是一致的，主要区别在于 Pruning 函数中的 Remove 函数，在句法树剪枝规则中，他主要是直接删除相应的子节点，并生成删除这个子节点以后的语句。而在基于句法树 BERT 填词中，在删除相应的子节点以后，再根据相应子节点的

```
def sub_sentence(sent):
    #其他从句识别
    #主语从句:
    queue=[]
    queue.append(ptree)
    s_subsentence=""
    s_add_sentence=""
    s_sentence= ""
    while len(queue)>0:
        current = queue.pop(0)
        if isinstance( current,tree.Tree):
            flagS = False
            if current.label() == "SBAR":
                if len(queue)==0:
                    break
                currentnext = queue[0]
                if isinstance(currentnext,tree.Tree) /
                  and current.parent() ==currentnext.parent()   /
                  and currentnext.label()=="VP":
                    flagS = True
                    s_sentence=current.leaves()#用于主句删除
                    son = current[0]
                    if isinstance(son, tree.Tree):
                        if son.label()=="WHNP":
                            if not isinstance(son.leaves(),list) and son.leaves().lower() == "who":
                                s_add_sentence="he"
                            else:
                                s_add_sentence="it"
                            current.remove(son)
                        elif son.label()=="IN":
                            current.remove(son)
                        s_subsentence=current.leaves()
                    break
            for i in range(len(current)):
                queue.append(current[i])
```

图 5-7: 句法树从句识别部分代码

单词数量，进行 BERT 填词任务，进行 BERT 填词的时候不能选择与原始单词一致的结果，只能选择其他结果。

下图 5-9就是 BERT 填词任务的关键性代码，首先使用 tokenizer 和词汇表，使用 encode 函数将一个字符串转换为一连串的 id。将我们需要遮蔽的词汇起始位置和长度使用 mask_id 进行遮蔽，接着使用 MLM 模型去预测被 mask 掉的部分，最后使用 decode 函数将结果转换成对应的词语列表，返回结果用于接

function PRUNING($source\_sent$, $dependency\_tree$, $head$)
if $len(head.children())$ > 0 then
$nodes \leftarrow head.children()$
$dep\_nodes \leftarrow DEP\_PRIOR(dependency\_tree)$ ⊲ prioritize the
nodes in the dependency tree
$relation\_set \leftarrow RELATION$ () ⊲ predefined operation set
for each $node \in dep\_nodes$ do
$source\_sent \leftarrow REMOVE$(source_sent, relation_set, node)
if $len(node.children())$ > 0 then
PRUNING($source\_sent$, $dependency\_tree$, $node$)
end if
if node.is_removed() then
$gen\_sents.add$ ($source\_sent$)
end if
end for
else
$source\_sent \leftarrow REMOVE$(source_sent, relation_set, head) ⊲
determine whether to delete the node according to the relation dictionary
if head.is_removed() then
$gen\_sents.add$ ($source\_sent$)
end if
end if
return gen_sents
end function

图 5-8: 句法树剪枝流程伪代码

下来的选择合适的词语进行填词。

```
# 完形填空，预测 MASK 的字符
def get_mask_character(self,language,string, mask_index, mask_len,top_k):
    if language == 'en':
        token_ids, segment_ids = self.tokenizer_en.encode(string)
        num = len(token_ids)
        # mask 掉“”
        for i in range(mask_len):
            token_ids[mask_index+i] = self.tokenizer_en._token_mask_id
        token_ids, segment_ids = to_array([token_ids], [segment_ids])
        # 用 mlm 模型预测被 mask 掉的部分
        probas = self.model_en.predict([token_ids, segment_ids])[0]
        top_k_index_list = []
        for i in range(top_k):

        top_k_index_list.append(self.tokenizer_en.decode(probas[mask_index:mask_index+mask_len].argmax(axis=1)))
        for j in range(mask_len):
                probas[mask_index+j][probas[mask_index:mask_index+mask_len].argmax(axis=1)[j]] = -float('inf')
        return top_k_index_list
```

图 5-9: 句法树 BERT 填词关键代码

5.5 实验分析

在对基于句法树的关键词文本扩增技术进行实验之前，首先需要提出两个基本的问题。

RQ1：基于句法树的关键词文本扩增技术是否能够对关键词文本进行有效扩增？

RQ2：基于句法树的关键词文本扩增技术与业内主流通用文本扩增技术相比有什么优势？

基于对这些问题的思考，对关键词文本扩增方法进行实验，验证基于句法树的关键词文本扩增技术的方法有效性，比较不同扩增工具的扩增效果，得出本扩增技术的价值和意义。

5.5.1 实验数据集

过去的关键词文本数据集，主要集中在科研领域。导致关键词提取模型不能很好地跨领域推广，因此限制了它们在实践中的使用。其次作者指定的关键词表现出的一致性问题，对模型的性能有负面影响，非常需要来自不同来源的注释数据。

KPTimes[42] 关键词文本数据集通过专家标注和大量来源于不同领域的新闻大幅度缓解了上述问题，与过往的关键词文本数据集相比能够提供更高质量的数据。KPTimes 包括了 279923 篇新闻文章组成的数据集，含有编辑人工分配的高质量关键词。如表 5-1所示是 KPTimes 的具体信息，KPTimes 数据集每个数据包括了新闻的类比信息、新闻的发布时间、新闻的标题、新闻的摘要和新闻的关键词等字段。虽然只是新闻摘要的关键词，但 KPTimes 的新闻摘要长达 300-1500 个单词，使用该数据集能够有效的验证基于句法树的关键词文本扩增系统的有效性。

下图 5-10就是数据示例，选取的关键词通过彩色的标注出来，能够显示当前和不存在的关键词短语，以及关键短词变体。

5.5.2 实验内容

针对之前所提出的两个 RQ，进行关键词文本扩增方法效果实验和文本扩增工具对比实验。使用 BERT-KPE[43] 作为使用的的关键词提取模型, 具体参

表 5-1: 关键词提取实验数据集相关信息

关键词文本数据集	KPTimes
关键词文本数量	279923
数据格式	json
文本包含字段	新闻分类、时间、标题，摘要，关键词，id
单个摘要文本长度	约 300-1500 个单词

Muslim Women in Hijab Break Barriers: 'Take the Good With the Bad'

When Ginella Massa, a Toronto-based TV reporter, recently accepted a request to host an evening newscast, she was not planning or expecting to make history for wearing a hijab. She was just covering for a colleague who wanted to go to a hockey game. And that's how Ms. Massa, who works at CityNews in Toronto, became the first Canadian woman to host a newscast from a large media company while wearing the head scarf. [...] This new trend of inclusion occurs amid a more sinister one, as reported hate crimes against Muslims are on the rise in the United States and Canada. The F.B.I. says that a surge in hate crimes against Muslims has led to an overall increase in hate crimes in the United States; Muslims have borne the brunt of the increase with 257 recorded attacks. [...] In Canada, where Ms. Massa has lived since she was a year old, the number of reported hate crimes has dropped slightly overall, but the number of recorded attacks against Muslims has grown: 99 attacks were reported in 2014, according to an analysis by the news site Global News of data from Statistics Canada, a government agency. [...]

keywords: US; Islam; Fashion; Muslim Veiling; Women and Girls; (News media, journalism); Hate crime; Canada

图 5-10: 数据示例

数上基于谷歌官方 BERT 模型的初始化参数，选用效果最好的 Bert2Joint 方法（一个多任务模型在分块任务和排名任务上被联合训练，平衡了对关键词质量和显著性的估计）。为了便于使用，本文将 KPTimes 的 JSON 数据格式转换为 OpenKP 的格式，在 VDOM 域中其 id、引用的文本、start_idx、end_idx 通过数据预处理，其他特征值都为 0。

（1）关键词文本扩增效果实验

随机抽取 KPTimes 10000 条数据作为训练集，1000 条数据作为验证集，1000 条数据作为测试集。分别对训练集进行五倍扩增，随机从各个方法中挑选 50000 条扩增结果数据，分别选择无扩增方法、EDA 同义词替换、回译扩增、基于句法树剪枝规则扩增和基于句法树 BERT 填词扩增。使用 BERT-KPE 关键词提取模型进行训练和实验，最终通过 Precision、Recall、F1 这三个指标评估实验效果。

实验结果于表 5-2所示, 通过表中数据可以得知，在 BERT 这种大参数预训

表 5-2: 扩增方法效果对比

实验扩增方法	Precision@1	Precision@3	Recall@3	F1@3
无扩增工具	0.441	0.281	0.423	0.338
EDA 同义词替换扩增方法	0.449	0.289	0.431	0.346
回译扩增方法	0.456	0.297	0.429	0.351
句法树剪枝规则扩增方法	0.453	0.299	0.433	0.354
句法树 BERT 填词扩增方法	**0.479**	**0.309**	**0.457**	**0.369**

练关键词提取模型上，数据样本不足导致精度不高的问题极为突出。通过对 1 万个关键词文本使用不同的文本扩增方法进行扩增，都能够提高该关键词提取模型的性能表现，证明了本系统集成的关键词文本扩增方法都是有效的。而对于不同方法内部的比较来看，基于句法树 BERT 填词扩增方法取得了最好的效果。句法树 BERT 填词方法在返回的第一个结果是关键词的可能性是 0.479，与其他方法相比取得了最好的成绩, 在前三个返回结果的精度也取得了 0.309 的最高表现，在 Recall 和 F1@3 也都取得了不错的结果，证明了基于句法树的扩增方法对关键词文本扩增的有效性。

（2）文本扩增工具对比实验

在文本扩增工具对比实验上，本文选择了主流的文本扩增工具 Python 库 textaug 的扩增数据结果作为对比实验，他集成了 EDA 扩增方法，能够快速生成大量扩增关键词文本。在这也加入了回译扩增方法作为对比，这也是一种常用的文本扩增工具。因为基于句法树 BERT 填词扩增方法在实验中取得了最好的效果，本实验中选择此方法作为对比方法。在本实验中需要验证基于句法树的关键词文本扩增技术是否能够比其他两种通用的文本扩增方法取得更好的表现。

随机取 KPTimes 10000 条数据作为训练集，500 条数据作为验证集，500 条数据作为测试集。分别对训练集进行五倍扩增，随机从各个工具中挑选 50000 条扩增结果数据。表 5-3中记录了实验结果。从表中数据可以看出，相较于 textaug 工具和 NLPaug 工具，实现了基于句法树 BERT 填词扩增方法的关键词文本扩增系统在每个指标上都有最好的表现，与通用文本扩增工具相比具有更好的性能表现。textaug 扩增工具，相较于上个实验中进行简单的同义词替换，

表 5-3: 扩增工具效果对比

实验扩增工具	Precision@1	Precision@3	Recall@3	F1@3
无扩增工具	0.441	0.281	0.423	0.338
Textaug	0.463	0.301	0.445	0.359
NLPaug	0.456	0.304	0.441	0.360
基于句法树 BERT 填词扩增	**0.479**	**0.309**	**0.457**	**0.369**

取得了更好的效果。而 NLPaug 也与 Textaug 工具取得了类似的结果。而我们的方法中基于句法树 BERT 填词扩增，通过分析关键词文本特征，保留关键词和核心语义内容，比未使用扩增工具提高了 3.8% 的精度，比使用其他扩增方法也提高了 1% 以上的精度，充分展现了方法的优越性，并且在各项指标上都没有其他损失。

通过上述两个实验，很好的回答的之前 RQ1，RQ2 所提出的问题。实验表明，基于句法树的关键词文本扩增技术与不使用关键词文本扩增相比能够提高关键词提取模型的精度，该技术是有效的。与其他通用文本扩增方法相比，基于句法树的关键词文本扩增技术的确针对关键词文本进行有效的扩增，该扩增系统与其他文本扩增工具相比提供了更好的扩增关键词文本数据，生成的数据具有更高的质量。

5.6 系统测试

5.6.1 测试目标

在 KPTimes 数据集上验证基于句法树的关键词文本扩增技术能够对于关键词文本数据进行有效扩增之后，需要对基于该技术实现的关键词文本扩增系统进行测试，通过测试用例的执行对关键词文本扩增系统的功能实现的正确性进行验证，对关键词文本扩增系统是否达到用户的相关功能性需求和非功能性需求进行验证。

5.6.2 测试环境

对基于句法树的关键词文本扩增系统的测试，需要搭建实际的测试环境。本次系统测试环境如下表 5-13所示。用户使用 Windows 11 作为操作系统，Chrome 浏览器作为访问扩增系统前端的工具。服务器部署在 Linux 环境之上。用户在测试环境上环境上使用关键词文本扩增系统，通过对测试用例的测试验证是否达到用户预期要求。

表 5-4: 实验数据集相关信息

服务器操作系统	Ubuntu 20.04 LTS
服务器规格	CPU: 14C20T MEM: 32GB GPU: RTX3060 12GB
用户操作系统	Windows11
用户浏览器	Chrome
编程语言	Python 3.7.3
数据持久化	MySQL 8.0

5.6.3 功能测试

通过对基于句法树的关键词文本扩增系统的功能进行测试，验证该关键词文本扩增系统达到了扩增系统使用者的预期。为了测试该关键词文本扩增系统的功能，需要通过第四章进行的系统用例流程验证。测试主要包括了对信息预览界面的查看、关键词文本的上传与下载、通用文本扩增、基于句法树的关键词文本扩增、关键词文本扩增项目管理、关键词文本扩增任务管理、关键词文本文件管理这些功能进行测试。

表 5-5是对系统信息预览界面的测试。主要测试是否无论何时，系统信息都能够正确的显示。用户在其他界面点击仪表盘，能够查看到相关的系统信息。如果当前系统正在执行较多的关键词文本扩增任务，用户可以暂时放弃本次关键词文本扩增任务的创建，为系统保留更多资源。

表 5-6是关键词文本上传和下载测试用例，主要用于测试关键词文本扩增系统能否正常上传或者下载关键词文本。用户在创建关键词文本扩增项目的过程中，需要通过点击上传按钮，选择上传的待扩增关键词文本文件。待扩增的

关键词文本文件应该是 Json 格式，而不是其他格式。如果为其他格式，在前端界面就会对文本的格式正确性检查。如果是正确格式，系统会在后端中对数据是否包括了关键词和正文内容进行检查，并会返回相关信息。创建的关键词文本扩增任务执行完毕之后，用户可以点击任务详情按钮，进入详情界面下载扩增关键词文本。

表 5-5: 信息预览测试用例

测试用例 ID	测试用例 1
测试用例名称	系统信息预览
测试功能	系统能够正确的展示当前运行情况
测试步骤	1. 用户在首页点击仪表盘 2. 用户在关键词文本任务管理界面点击仪表盘 3. 用户在关键词文本项目管理界面点击仪表盘 4. 用户在关键词文本文件管理界面点击仪表盘
预期结果	无论从何种页面进行跳转，都能够正确显示当前系统的运行状态

表 5-6: 关键词文本上传与下载测试用例

测试用例 ID	测试用例 2
测试用例名称	关键词文本上传与下载
测试功能	在关键词文本扩增项目创建时，用户能够正确上传关键词文本, 在完成的关键词文本扩增任务详情，用户能正确的下载关键词文本
测试步骤	关键词文本上传流程： 1. 进入关键词文本扩增项目创建页面 2. 选择 ZIP 文件进行上传 3. 选择正确的 Json 关键词文本上传 关键词文本下载流程： 1. 用户进入已完成的关键词文本扩增任务详情 2. 用户点击下载，下载扩增好的关键词文本数据
预期结果	关键词文本上传中： 1. 系统拒绝 ZIP 格式的关键词文本数据 2. 系统完成关键词文本的上传 关键词文本下载流程： 1. 前端创建下载链接，浏览器下载扩增好的关键词文本

表 5-7是对关键词文本执行通用文本扩增测试用例，系统不仅仅继承基于

句法树的关键词文本扩增技术，也提供了基于回译扩增方法和基于 EDA 同义词替换的文本扩增方法。需要对系统同义词替换方法进行功能验证，是否能够有效生成同义词替换的关键词文本。需要验证在联网环境下，能够正确调用神经网络翻译的 API 接口，进行神经网络翻译任务，系统能够正确执行回译扩增方法。系统能够将两种扩增方法生成的关键词文本保存在相应位置，将关键词文本扩增过程中产生的相关信息变动存储到数据库中。

表 5-7: 关键词文本通用扩增测试用例

测试用例 ID	测试用例 3
测试用例名称	关键词文本通用扩增测试用例
测试功能	能够正确对关键词文本数据进行 EDA 同义词替换扩增和基于回译的扩增
测试步骤	1. 用户进入关键词文本扩增任务创建界面 2. 用户对一个项目选择 EDA 扩增方法，扩增数量 10 3. 用户再次进入关键词文本扩增任务创建界面 4. 用户对一个项目选择回译扩增方法，扩增数量 10
预期结果	1. 步骤 2, 关键词文本扩增系统正确进行 EDA 扩增任务，生成了 10 倍关键词文本数据。 2. 步骤 4，关键词文本扩增系统正确进行回译扩增任务，生成了 10 倍关键词文本数据。

表 5-8是基于句法树的关键词文本扩增测试用例，测试系统能否将原始关键词文本数据，使用基于句法树剪枝规则的文本扩增方法，通过剪枝能够生成设定扩增数量的关键词文本数据，以及使用句法树 BERT 填词的文本扩增方法，通过剪枝规则结合 BERT 填词方法能够生成指定数量扩增关键词文本数据，以及能否把扩增结果存入系统中。

表 5-9是关键词文本文件管理测试用例，测试关键词文本文件管理界面是否能够正确显示关键词文本的相关信息，以及它的使用情况，并且能够删除特定的关键词文本。

表 5-10是关键词文本扩增任务管理测试用例，主要测试关键词文本扩增任务管理界面能否完整展示用户创建过的任务信息，能否进行用户所需任务的创建，并对创建进行格式上的要求，能否对创建的关键词文本扩增任务的参数等信息进行修改，以及能否顺利删除指定的关键词文本扩增任务。

表 5-11是关键词文本扩增项目管理测试用例，用于了解关键词文本扩增项目管理是否能够正常展示所有关键词文本扩增项目，用户是否能够查看指定关

表 5-8: 基于句法树的关键词文本扩增测试用例

测试用例 ID	测试用例 4
测试用例名称	基于句法树的关键词文本扩增测试用例
测试功能	能够正确对关键词文本数据进行句法树剪枝扩增和句法树 BERT 填词扩增
测试步骤	1. 用户进入关键词文本扩增任务创建界面 2. 用户对一个项目选择句法树剪枝扩增方法，扩增数量 10 3. 用户保存关键词文本扩增任务 4. 用户再次进入关键词文本扩增任务创建界面 5. 用户对一个项目选择句法树 BERT 填词扩增方法，扩增数量 10 6. 用户保存关键词文本扩增任务
预期结果	1. 步骤 3，关键词文本扩增系统正确进行句法树剪枝扩增任务，生成了 10 倍关键词文本数据。 2. 步骤 6，关键词文本扩增系统正确进行句法树 BERT 填词扩增任务，生成了 10 倍关键词文本数据。

表 5-9: 关键词文本扩增文件管理测试用例

测试用例 ID	测试用例 5
测试用例名称	关键词文本文件管理
测试功能	能够正确查看系统当前文件信息，并可以进行文件的删除
测试步骤	1. 进入关键词文本文件管理页面 2. 点击删除按钮删除关键词文本
预期结果	1. 对于步骤 1，用户能查看当前系统中所有关键词文本，并且能够看到相关信息和正在使用该文本的项目和任务 2. 对于步骤 2，系统因根据具体情况决定是否删除关键词文本 2a. 如果有多个项目正在使用该文本，拒绝删除并返回错误 2b. 如果有任务正在使用该文本，拒绝删除并返回错误 2c. 如果没有任何项目或者任务使用该文件，直接在系统文件夹中删除相应的关键词文本，并更新相关信息到数据库中。

键词文本扩增项目的详细信息，能否修改特定关键词文本扩增项目的待扩增关键词文本，能够放创建用户所需要的关键词文本项目，以及能否顺利删除指定关键词文本扩增项目。

表 5-10: 关键词文本扩增任务管理测试用例

测试用例 ID	测试用例 6
测试用例名称	关键词文本扩增任务管理
测试功能	关键词文本扩增任务管理界面可以创建新的关键词文本扩增任务，查看相关信息，对需要修改的关键词文本扩增任务进行修改, 删除
测试步骤	1. 进入关键词文本扩增任务管理界面 2. 进入特定关键词文本扩增任务，查看详细信息 3. 点击删除，删除特定关键词文本扩增任务 4. 点击修改，修改特定关键词文本扩增任务的信息 5. 进入关键词文本扩增任务创建页面，填写参数，创建新的扩增任务
预期结果	1. 步骤 1，正确展示当前系统所有关键词文本扩增任务 2. 步骤 2，正确展示当前关键词文本扩增任务相关信息 3. 步骤 3，如果特定关键词文本扩增任务没有处于运行状态，删除该条数据 4. 步骤 4，修改特定关键词文本扩增任务信息，查看详情，修改已完成 5. 步骤 5，对于正确填写参数的关键词文本扩增任务，系统在后台执行该任务

表 5-11: 关键词文本扩增项目管理测试用例

测试用例 ID	测试用例 7
测试用例名称	关键词文本扩增项目管理
测试功能	关键词文本扩增项目管理界面可以创建新的关键词文本扩增项目，查看相关信息，对需要修改的关键词文本扩增项目进行修改, 删除
测试步骤	1. 进入关键词文本项目管理界面 2. 进入特定关键词文本扩增项目详情，查看当前待扩增关键词文本等信息 3. 点击删除，删除特定关键词文本扩增项目 4. 点击修改，修改特定关键词文本扩增项目的待扩增关键词文本 5. 进入关键词文本扩增项目创建页面，填写名称，插入关键词文本，创建新的扩增任务
预期结果	1. 步骤 1，正确展示当前系统所有关键词文本扩增项目 2. 步骤 2，正确展示特定关键词文本扩增项目的详细信息 3. 步骤 3，如果没有正在执行的关键词文本扩增任务，将连同该项目及其归属的关键词文本扩增任务一起删除。 4. 步骤 4，修改特定关键词文本扩增项目信息，查看详情，修改已完成 5. 步骤 5，成功创建关键词文本扩增项目，并在关键词文本扩增项目列表中显示

经过对以上测试用例的实际执行，获得了如表 5-12所示的结果。在测试

过程中，基于句法树的关键词文本扩增系统，按照测试用例预期的执行结果执行。经过功能测试，验证了系统功能实现的正确性。

表 5-12: 用例测试结果

用例 ID	测试用例 ID	测试结果
用例 1	测试用例 1	测试通过
用例 2	测试用例 2	测试通过
用例 3	测试用例 3	测试通过
用例 4	测试用例 4	测试通过
用例 5	测试用例 5	测试通过
用例 6	测试用例 6	测试通过
用例 7	测试用例 7	测试通过

5.6.4 性能测试

基于句法树的关键词文本扩增系统的性能测试，主要需要对关键词文本扩增方法速度的测试。通过对扩增速度的测试，了解关键词文本扩增系统的执行速度。

表 5-13: 数据扩增性能测试

扩增方法	时间
EDA 同义词替换扩增方法	1.96 秒 50 条生成结果
回译扩增方法	15 秒 50 条生成结果
基于句法树剪枝规则扩增方法	40 秒 50 条生成结果
基于句法树 BERT 填词扩增方法	57 秒 50 条生成结果

通过创建关键词文本扩增任务，统计各个不同扩增方法的执行时间。获得 EDA 同义词替换扩增方法、回译扩增方法、基于句法树剪枝规则扩增方法的执行效率。通过实验结果表 5-13得知，EDA 同义词替换方法具有最快的执行时间，回译扩增方法受限于神经网络翻译 API 接口的请求次数限制，取得了第二

快的速度。而基于句法树剪枝规则扩增方法和基于句法树 BERT 填词扩增方法因为需要使用 Stanford CoreNLP 和 BERT 模型当前运行速度较慢。

5.7 本章小结

本章介绍基于句法树的关键词文本扩增系统的实现，对基于句法树的关键词文本扩增技术进行了有效性的验证，以及对系统进行了功能测试，性能测试。在实现方面详细介绍了基于句法树的关键词文本扩增技术的实现，文本分析模块的实现，简略介绍了数据交互模块和关键词文本相关管理模块的实现。通过在数据集 KPTimes 上的实验回答两个 RQ，表明了基于句法树的关键词文本扩增系统的有效性。通过设计测试用例，进行性能测试，展示了基于句法树的关键词文本扩增系统的可用性。

第六章　总结与展望

6.1　总结

随着互联网上的文本数据的爆炸性增长，如何处理海量信息已经成当今社会的一大挑战。关键词能够高效概括文档的主要思想，关键词提取模型具有广阔的应用前景。而要想训练一个具有泛化能力的关键词提取模型，离不开不同领域的关键词文本数据集。

而现有的关键词生成数据集主要适用于学术领域，大多是非专家注释。如果想要使用其他领域的关键词文本数据，不可避免的会遇到数据集数据量少，数据集关键词标注质量低，数据集所涉及的文本领域不够广泛等问题。使用少量高质量的关键词文本数据会导致关键词提取模型出现过拟合的现象，如何通过少量高质量关键词文本数据生成大量有效的关键词文本具有重要价值。

基于以上的问题，本文通过使用句法分析和 BERT 填词的方法，针对关键词文本特点，充分考虑语义特征，提出了一种基于句法树的关键词文本扩增技术。该技术能够有效提高关键词提取模型在小样本情况下的性能，具有实用价值。基于句法树的关键词文本扩增技术的设计，借鉴了已有的文本扩增方法，从关键词提取模型构建角度出发，研究了关键词文本数据的特点，通过句法分析技术，分析文本语义上的特点，实现基于句法树关键词文本扩增。首先将文本从复杂句到简单句、从并列句到普通独立语句的变换。在基于简单句句子的基本句式前提下，对语句进行依存句法树构建，对句子每个 token 与关键词进行相似度比较，将包含关键词信息的句子 token 成分设置为句子的重要成分。通过人为设计的句法树剪枝规则，对句子的非重要成分进行删除或者 BERT 填词，完成对文本每一个语句的扩增。最后通过拼接这些文本，构建成完整的关键词文本数据，实现对于关键词文本的扩增。

本文首先介绍了项目的背景和意义，包括了文本数据扩增的研究现状，受到现有工作的启发，提出了基于句法树的关键词文本数据扩增技术。其次，对于本文使用的科学理论，自然语言处理技术进行了介绍。接着介绍了技术的研究过程，对基于句法树的关键词文本扩增技术进行研究。然后对系统进行了需

求分析和总体设计，基于低耦合，高内聚的设计思想，对关键词文本扩增系统进行了功能模块上的划分，设计。接着详细介绍了基于句法树的关键词文本扩增技术的实现，文本分析模块的实现，简略介绍了数据交互模块和关键词文本扩增的相关管理模块。最后通过测试，验证该系统满足了用户的需求。

6.2 未来工作的展望

本文在已有数据扩增技术的基础上，挖掘关键词提取模型的特征，提出基于句法树的关键词文本扩增技术。通过在 KPTimes 数据上进行了相关实验，证明了基于句法树的关键词文本扩增技术相较于其他扩增技术，的确针对了关键词文本的特点进行设计，提供了更高质量的扩增关键词文本。

1）关键词文本的扩增速度较慢。基于句法树的关键词文本扩增技术使用句法树 BERT 填词方法的时候，在 BERT 填词过程中需要调用预训练的模型进行填词任务。可以使用 Albert 这种较小的 BERT 模型代替当前使用的谷歌官方的 BERT 模型，虽然会降低一些填词的质量，但会大幅度提高 BERT 填词速度。

2）句法树剪枝规则、并列识别方法仍存在不完善的缺点，所设计的句法树剪枝规则仍会存在歧义的问题需要随着实验的进行进一步扩增句法树剪枝规则。当前的并列识别方法，对于并列句的处理仍然较为简单，会导致出现非法语句的产生，需要进一步的思考如何实现并列句的处理。

3）句法分析模型质量受限。StanfordCoreNLP 的模型质量，虽然在测试数据集取得了高达 90% 以上的精度表现，但在实际使用中随着语句长度变化和特殊字符的出现，句法分析效果会逐步下降，降低了本系统的扩增效果。

参考文献

[1] EVANS D A, ZHAI C. Noun-Phrase Analysis in Unrestricted Text for Information Retrieval[J]. CoRR, 1996, cmp-lg/9605019.

[2] WITTEN L H, BAINBRIDGE D. 5 - Markup and metadata: Elements of organization[G] // WITTEN L H, BAINBRIDGE D. The Morgan Kaufmann Series in Multimedia Information and Systems : How to Build a Digital Library. San Francisco : Morgan Kaufmann, 2003 : 221 – 282.

[3] JONES K S. A statistical interpretation of term specificity and its application in retrieval[J]. J. Documentation, 2004, 60(5) : 493 – 502.

[4] MIHALCEA R, TARAU P. TextRank: Bringing Order into Text[C] // Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing , EMNLP 2004, A meeting of SIGDAT, a Special Interest Group of the ACL, held in conjunction with ACL 2004, 25-26 July 2004, Barcelona, Spain. [S.l.] : ACL, 2004 : 404 – 411.

[5] WAN X, XIAO J. Single Document Keyphrase Extraction Using Neighborhood Knowledge[C] // FOX D, GOMES C P. Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI 2008, Chicago, Illinois, USA, July 13-17, 2008. [S.l.] : AAAI Press, 2008 : 855 – 860.

[6] BLEI D M, NG A Y, JORDAN M I. Latent Dirichlet Allocation[J]. J. Mach. Learn. Res., 2003, 3 : 993 – 1022.

[7] PU X, JIN R, WU G, et al. Topic Modeling in Semantic Space with Keywords[C] // BAILEY J, MOFFAT A, AGGARWAL C C, et al. Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015. [S.l.] : ACM, 2015 : 1141 – 1150.

[8] TURNEY P D. Learning Algorithms for Keyphrase Extraction[J]. CoRR, 2002, cs.LG/0212020.

[9] FRANK E, PAYNTER G W, WITTEN I H, et al. Domain-Specific Keyphrase Extraction[C] // DEAN T. Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, IJCAI 99, Stockholm, Sweden, July 31 - August 6, 1999. 2 Volumes, 1450 pages. [S.l.] : Morgan Kaufmann, 1999 : 668 – 673.

[10] MENG R, ZHAO S, HAN S, et al. Deep Keyphrase Generation[C] // BARZILAY R, KAN M. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers. [S.l.] : Association for Computational Linguistics, 2017 : 582 – 592.

[11] TUITE C, AGAPITOS A, O'NEILL M, et al. Early stopping criteria to counteract overfitting in genetic programming[C] // KRASNOGOR N, LANZI P L. 13th Annual Genetic and Evolutionary Computation Conference, GECCO 2011, Companion Material Proceedings, Dublin, Ireland, July 12-16, 2011. [S.l.] : ACM, 2011 : 203 – 204.

[12] SRIVASTAVA N, HINTON G E, KRIZHEVSKY A, et al. Dropout: a simple way to prevent neural networks from overfitting[J]. J. Mach. Learn. Res., 2014, 15(1) : 1929 – 1958.

[13] SAHA B N, KUNAPULI G, RAY N, et al. AR-Boost: Reducing Overfitting by a Robust Data-Driven Regularization Strategy[C] // BLOCKEEL H, KERSTING K, NIJSSEN S, et al. Machine Learning and Knowledge Discovery in Databases. Berlin, Heidelberg : Springer Berlin Heidelberg, 2013 : 1 – 16.

[14] LU X, ZHENG B, VELIVELLI A, et al. Research Paper: Enhancing Text Categorization with Semantic-enriched Representation and Training Data Augmentation[J]. J. Am. Medical Informatics Assoc., 2006, 13(5) : 526 – 535.

[15] MA J, LI L. Data Augmentation For Chinese Text Classification Using Back-Translation[J]. Journal of Physics: Conference Series, 2020, 1651(1) : 012039.

[16] WEI J W, ZOU K. EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks[C] // INUI K, JIANG J, NG V, et al. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019. [S.l.] : Association for Computational Linguistics, 2019 : 6381 – 6387.

[17] ZHANG X, ZHAO J J, LECUN Y. Character-level Convolutional Networks for Text Classification[J]. CoRR, 2015, abs/1509.01626.

[18] KOBAYASHI S. Contextual Augmentation: Data Augmentation by Words with Paradigmatic Relations[C] // WALKER M A, JI H, STENT A. Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers). [S.l.] : Association for Computational Linguistics, 2018 : 452 – 457.

[19] HOU Y, LIU Y, CHE W, et al. Sequence-to-Sequence Data Augmentation for Dialogue Language Understanding[C] // Proceedings of the 27th International Conference on Computational Linguistics. Santa Fe, New Mexico, USA : Association for Computational Linguistics, 2018 : 1234 – 1245.

[20] 魏小娜, 李英豪, 王振宇, et al. 医学影像人工智能辅助诊断的样本增广方法[J]. 计算机应用, 2019, 39(09) : 2558 – 2567.

[21] PACCANARO A, HINTON G E. Learning Distributed Representations of Concepts Using Linear Relational Embedding[J]. IEEE Trans. Knowl. Data Eng., 2001, 13(2) : 232 – 244.

[22] MIKOLOV T, CHEN K, CORRADO G, et al. Efficient Estimation of Word Representations in Vector Space[C] // BENGIO Y, LECUN Y. 1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings. 2013.

[23] MIKOLOV T, SUTSKEVER I, CHEN K, et al. Distributed Representations of Words and Phrases and their Compositionality[C] // BURGES C J C, BOTTOU

L, GHAHRAMANI Z, et al. Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States. 2013 : 3111 – 3119.

[24] DEVLIN J, CHANG M, LEE K, et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding[J]. CoRR, 2018, abs/1810.04805.

[25] TAYLOR W L. “Cloze Procedure”: A New Tool for Measuring Readability[J]. Journalism Quarterly, 1953, 30(4) : 415 – 433.

[26] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is All you Need[C] // GUYON I, von LUXBURG U, BENGIO S, et al. Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA. 2017 : 5998 – 6008.

[27] 冯志伟. 自然语言处理的形式模型 [M]. [S.l.] : 中国科学技术大学出版社, 2010.

[28] ARRIVÉ M. Les Éléments de syntaxe structurale, de L. Tesnière[J]. Langue française, 1969, 1(1) : 36 – 40.

[29] KÜBLER S, MCDONALD R T, NIVRE J. Synthesis Lectures on Human Language Technologies : Dependency Parsing[M]. [S.l.] : Morgan & Claypool Publishers, 2009.

[30] SUTSKEVER I, VINYALS O, LE Q V. Sequence to Sequence Learning with Neural Networks[C] // GHAHRAMANI Z, WELLING M, CORTES C, et al. Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada. 2014 : 3104 – 3112.

[31] CHEN D, MANNING C D. A Fast and Accurate Dependency Parser using Neural Networks[C] // MOSCHITTI A, PANG B, DAELEMANS W. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP

2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL. [S.l.] : ACL, 2014 : 740 – 750.

[32] NEPIL M. Learning to Parse from a Treebank: Combining TBL and ILP[C] // . 2001.

[33] BRILL E D. A Corpus-Based Approach to Language Learning[D]. USA : [s.n.], 1993.

[34] COLLINS M. A New Statistical Parser Based on Bigram Lexical Dependencies[C] // JOSHI A K, PALMER M. 34th Annual Meeting of the Association for Computational Linguistics, 24-27 June 1996, University of California, Santa Cruz, California, USA, Proceedings. [S.l.] : Morgan Kaufmann Publishers / ACL, 1996 : 184 – 191.

[35] COLLINS M. Three Generative, Lexicalised Models for Statistical Parsing[C] // COHEN P R, WAHLSTER W. 35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference, 7-12 July 1997, Universidad Nacional de Educación a Distancia (UNED), Madrid, Spain. [S.l.] : Morgan Kaufmann Publishers / ACL, 1997 : 16 – 23.

[36] MCDONALD R T, NIVRE J, QUIRMBACH-BRUNDAGE Y, et al. Universal Dependency Annotation for Multilingual Parsing[C] // Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 2: Short Papers. [S.l.] : The Association for Computer Linguistics, 2013 : 92 – 97.

[37] ZHANG Y, LI J, SONG Y, et al. Encoding Conversation Context for Neural Keyphrase Extraction from Microblog Posts[C] // WALKER M A, JI H, STENT A. Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers). [S.l.] : Association for Computational Linguistics, 2018 : 1676 – 1686.

[38] XIONG L, HU C, XIONG C, et al. Open Domain Web Keyphrase Extraction Beyond Language Modeling[J]. CoRR, 2019, abs/1911.02671.

[39] AINSLIE J, ONTAÑÓN S, ALBERTI C, et al. ETC: Encoding Long and Structured Inputs in Transformers[C] // WEBBER B, COHN T, HE Y, et al. Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020. [S.l.]: Association for Computational Linguistics, 2020: 268 – 284.

[40] 谢湘宁. 浅谈数据挖掘技术在专利信息分析中的应用 [J]. 中国发明与专利, 2015(01): 59 – 62.

[41] 胡少虎, 张颖怡, 章成志. 关键词提取研究综述 [J]. 数据分析与知识发现, 2021, 5(03): 45 – 59.

[42] GALLINA Y, BOUDIN F, DAILLE B. KPTimes: A Large-Scale Dataset for Keyphrase Generation on News Documents[C] // van DEEMTER K, LIN C, TAKAMURA H. Proceedings of the 12th International Conference on Natural Language Generation, INLG 2019, Tokyo, Japan, October 29 - November 1, 2019. [S.l.]: Association for Computational Linguistics, 2019: 130 – 135.

[43] SUN S, XIONG C, LIU Z, et al. Joint Keyphrase Chunking and Salience Ranking with BERT[J]. ArXiv, 2020, abs/2004.13639.

简历与科研成果

基本信息

王擎宇，男，汉族，1998 年 5 月出生，江苏常州人。

教育背景

2020 年 9 月 — 2022 年 6 月 南京大学软件学院 硕士

2016 年 9 月 — 2020 年 6 月 扬州大学信息工程学院 本科

《学位论文出版授权书》

本人完全同意《中国优秀博硕士学位论文全文数据库出版章程》（以下简称“章程”），愿意将本人的学位论文提交“中国学术期刊（光盘版）电子杂志社”在《中国博士学位论文全文数据库》、《中国优秀硕士学位论文全文数据库》中全文发表。《中国博士学位论文全文数据库》、《中国优秀硕士学位论文全文数据库》可以以电子、网络及其他数字媒体形式公开出版，并同意编入《中国知识资源总库》，在《中国博硕士学位论文评价数据库》中使用和在互联网上传播，同意按“章程”规定享受相关权益。

作者签名：_____

_____年____月____日

论文题名	基于句法树的关键词文本扩增技术				
研究生学号	MF20320161	所在院系	软件学院	学位年度	2022
论文级别	☐ 硕士　☐ 硕士专业学位 ☐ 博士　☐ 博士专业学位 （请在方框内画勾）				
作者 Email	MF20320161@smail.nju.edu.cn				
导师姓名	陈振宇 教授				

论文涉密情况：

☐ 不保密

☐ 保密，保密期 (_____年_____月_____日 至 _____年_____月_____日)

注：请将该授权书填写后装订在学位论文最后一页（南大封面）。