计算机科学
COMPUTER SCIENCE

一种结合代码片段和混合主题模型的软件数据聚类方法

魏林林, 沈国华, 黄志球, 蔡梦男, 郭菲菲

引用本文

魏林林, 沈国华, 黄志球, 蔡梦男, 郭菲菲. 一种结合代码片段和混合主题模型的软件数据聚类方法[J]. 计算机科学, 2024, 51(6): 44-51.

WEI Linlin, SHEN Guohua, HUANG Zhiqiu, CAI Mengnan, GUO Feifei. Software Data Clustering Method Combining Code Snippets and Hybrid Topic Models [J]. Computer Science, 2024, 51(6): 44-51.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.230300091

一种结合代码片段和混合主题模型的软件数据聚类方法

魏林林[1,2]　沈国华[1,2,3]　黄志球[1,2,3]　蔡梦男[1,2]　郭菲菲[1,2]
1 南京航空航天大学计算机科学与技术学院　南京 211106
2 南京航空航天大学高安全系统的软件开发与验证技术工业和信息化部重点实验室　南京 211106
3 软件新技术与产业化协同创新中心　南京 210093
(linlinwei@nuaa.edu.cn)

摘　要　使用主题模型进行文档聚类是众多文本挖掘任务中一种常见的做法。许多研究针对软件问答网站的数据,利用主题模型进行聚类来分析不同领域在社区的发展情况。然而,这些软件相关数据往往包含代码片段且文本长度分布不均,使用传统单一的主题模型对文本数据建模,易得到不稳定的聚类结果。文中提出了一种结合代码片段和混合主题模型的聚类方法,并使用 Stack Overflow 作为数据源,构造了在该平台上被提问数量排名前 60 的 Python 第三方库数据集,经过建模,该数据集最终划分为以下 6 个不同的领域:网络安全、数据分析、人工智能、文本处理、软件开发和系统终端。实验结果表明,在自动评估和人工评估的指标上,使用代码片段结合文本进行主题建模,在聚类结果划分的质量上表现良好,而联合多个模型进行实验,一定程度上提高了聚类结果的稳定性和准确性。

关键词:代码片段;主题模型;Stack Overflow;Python;聚类
中图分类号　TP311

Software Data Clustering Method Combining Code Snippets and Hybrid Topic Models

WEI Linlin[1,2],SHEN Guohua[1,2,3],HUANG Zhiqiu[1,2,3],CAI Mengnan[1,2] and GUO Feifei[1,2]
1 College of Computer Science and Technology,Nanjing University of Aeronautics and Astronautics,Nanjing 211106,China
2 Key Laboratory of Safety-Critical Software,Ministry of Industry and Information Technology,Nanjing University of Aeronautics and Astronautics,Nanjing 211106,China
3 Collaborative Innovation Center of Novel Software Technology and Industrialization,Nanjing 210093,China

Abstract　Using topic model to cluster documents is a common practice in many text mining tasks. Many studies use topic models to cluster data from software websites to analyze the development of communities in different fields. However,due to the fact that these software-related data often contain code snippets and the uneven distribution of text length,it is easy to get unstable clustering results by using traditional single topic model to handle this text data. This paper proposes a clustering method combining code snippets and hybrid topic models,and uses Stack Overflow as the data source to construct a Python third-party libraries dataset with the top 60 questions on the platform. After analyzing,it is finally divided into the following six different areas:network security,data analysis,artificial intelligence,text processing,software development and system terminal. Experimental results show that in terms of automatic evaluation and manual evaluation indicators,using code snippets combined with text for topic modeling,the quality of clustering results division performs well, while combining multiple models for experiments can improve the stability and accuracy of clustering results to a certain extent.

Keywords　Code snippets,Topic model,Stack Overflow,Python,Cluster

1　引言

主题模型已经被广泛应用于许多文本挖掘任务,很多学者针对自己关注的领域,例如区块链[1]、数据库[2],并发[3]和网络编程[4]等,使用该模型对这些领域的文本信息进行主题划分,以此来帮助其理解众多文档中潜在的语义信息和主题分布,从而为该领域的相关工作者提供一些有价值的反馈信息,帮助相关研究人员规避风险。然而,研究人员在使用主题模型对软件问答网站(SQA)进行分析时,由于 SQA 中的数据含有大量的代码信息,为了降低实验出现不可预期结果的

到稿日期:2023-03-11　返修日期:2023-07-13
基金项目:国家自然科学基金(61772270,U2241216);民航应急科学与技术重点实验室开放基金(NJ2022022)
This work was supported by the National Natural Science Foundation of China(61772270,U2241216) and Open Foundation of Civil Aviation Emergency Science and Technology Key Laboratory(NJ2022022).
通信作者:沈国华(ghshen@nuaa.edu.cn)

可能性，往往会对数据中的代码片段进行过滤，虽然这一定程度上降低了风险，但是也遗漏了代码片段中隐含的语义和主题信息。除此之外，大部分实验结果基于单一主题模型，如LDA(Latent Dirichlet Allocation)[5]，尤其对这些文本长度分布不均的大型数据集表现效果不佳，容易陷入局部最优[6]，造成主题分布不稳定和边界不清晰等问题。

本文基于传统的主题建模方法，将代码片段和文本信息转化为词向量，对传统方法的单模态输入进行扩展，从而降低遗漏代码片段中隐含关键信息的可能；并使用基于词频权重的TF-LDA主题模型，结合基于非负矩阵分解的NMF(Non-negative Matrix Factorization)文档聚类模型，来提升聚类结果的稳定性。为了验证该方法的可行性，本文用一个流行的软件问答网站(Stack Overflow，以下简称SO)作为数据源，进行了实证研究。SO是一个大型的问答社区，主要供计算机相关人员讨论问题和分享观点，截至2022年10月，该社区关于Python的提问帖子数量已经超过200万条。首先，提取了SO中2009－2021年之间与Python第三方库相关的数据，构建了提问数量排名前60的第三方库文档集作为模型的输入，经过主题建模最终总结归纳出了Python研究人员主要关注的6种不同的领域，即网络安全、数据分析、人工智能、文本处理、软件开发和系统终端，实验结果有助于Python相关工作者更好地理解不同主题在社区的发展状况。具体地，本文探索了以下4个研究问题。

问题1：哪些python第三方库在SO社区中被开发人员经常提问？

问题2：Python开发者在SO中主要讨论哪些主题？

问题3：这些主题随时间在社区的发展趋势如何？

问题4：代码片段与混合模型对提升模型聚类质量是否有帮助？

本文第2章介绍了一些相关研究工作；第3章介绍了本文提出的实验方法，包括文本数据和代码数据的预处理、转换为词向量、主题建模以及如何获取混合模型的分类结果；第4章介绍了实验结果，并回答了提出的研究问题，探究了文中提出的方法的有效性，具体做法是将代码和文本分别作为输入，探究代码片段是否可以增强模型的聚类效果，以及本文提出的混合模型是否比单个模型在聚类结果上的表现更加良好；最后简要地总结了本文，包括方法上的局限性和实验上的一些不足，并对下一步的研究进行了展望。

2 相关工作

主题模型经常被用来揭示由一组文本所跨越的主题。其被广泛应用于自然语言处理、信息检索、计算机视觉、相关性判断、情感分析和社交媒体分析等领域。下文主要介绍了主题建模在文本挖掘任务中的一些相关研究，和一些研究者以不同的角度对主题模型进行扩展的工作。

Ajam等开展了一项研究，揭示了SO帖子中关于API讨论背后的主题，如API定义、API使用方式和API文档等[7]；Ahasanuzzaman等使用机器学习算法来分类SO中关于API的讨论[8]。Zhao等使用LDA主题模型对SO中深度学习相关帖子进行建模分析，总结出了30个深度学习相关的主题并进行了分析[9]；Yang等使用遗传算法调优的LDA对不同的安全相关的问题进行了聚类，并对每一类进行了深入分析[10]，Rosen等使用LDA来总结SO中关于移动应用的问题，并进行了深入分析[11]。Chen等将LDA和知识图谱结合，利用SO中移动应用第三方库的相关数据，提出了一种新的方法，为安卓开发人员推荐相应的安卓第三方库[12]；Alexandre等对安卓应用中第三方库的使用模式进行了深入分析[13]；Allamanis等使用LDA将编程概念和标识符与SO上的特定类型的问题关联起来[14]。

出于不同的目标，研究人员已经开发出多种不同的主题模型。潜在语义索引(LSI)[15]使用奇异值分解对文档术语矩阵进行低秩逼近，目的是将文档和术语映射到低维语义空间。概率潜在语义索引(pLSI)[16]是对LSI的一种扩展，其中文档被视为一组主题上的概率分布。此外，文档单词可能由单个的主题生成，而不同的单词可能由不同的主题生成。潜在狄利克雷分配模型(LDA)避免了pLSI的固有限制，即在文档级别上缺乏概率模型，从而生成主题混合比例。LDA从几个角度进行了扩展。相关主题模型(CTM)[17]通过合并逻辑正态分布，解决了LDA无法捕获主题相关性的问题。文献[18]开发了动态主题模型来分析文档语料库中主题的时间演化。Bigram主题模型包含了一个词序[19]的概念。此外，单词嵌入[20]促进了增强的主题模型的发展，该模型包含了关于单词之间的句法和语义关系的外部知识。由于实验数据集的特点，本文基于传统的TF-LDA和NMF模型，通过代码片段来扩充输入信息，以及通过对模型输出的主题词按贡献度分配权重来提高聚类质量，最终完成对软件数据的聚类任务。

3 研究方法

本章主要介绍所提出的研究方法，首先获取原始数据并对其进行过滤和筛选，得到所要关注的Python领域数据集；其次在对数据进行一些常规的预处理后，分别提取数据中的文本和代码信息，并按照一定规则构造文本和代码文档集，统一格式后将其作为模型的输入；最后根据两个模型输出的主题词按照贡献度合并分类，将主题词映射到文档，完成聚类任务。图1给出了本文研究方法的整体框架。

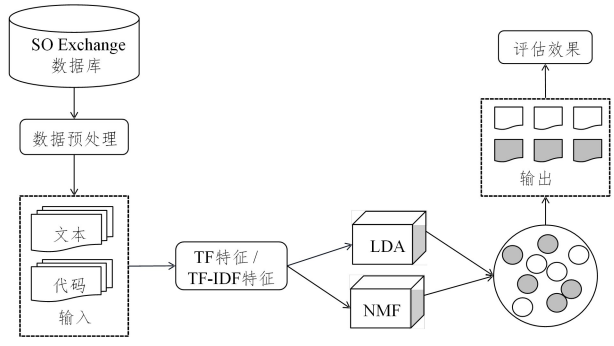

图1 本文研究框架

Fig. 1 Research framework of this paper

3.1 构造数据集

首先，从SO Exchange网站获取了原始数据集Post.

xml，其中包含 2008 年 6 月至 2022 年 3 月之间用户的问答数据。由于原始数据中包含其他话题，实验仅关注与 Python 第三方库相关的讨论帖子，因此需要制定一定的规则来获取相关数据。

3.1.1 筛选数据

SO 是一个软件问答网站，它为用户提供了发布问题、回答问题和发表观点等操作选项。当开发人员在 SO 中发布一个新问题时，该网站允许发布者给问题添加至多 5 个不同的标签，这些标签一般是涉及相关技术的术语[21]，例如用户为其创建的问题设定了两个标签，分别是“python”和“scipy”，这表明该问题和 Python 的第三方库“scipy”的使用有关。本文正是基于这样的标签机制[22]来筛选数据。图 2 给出了一条数据（帖子）的基本组成结构，限于篇幅，仅列出与研究内容有关的字段，其中代码片段位于 Body 字段中。

```
<row
    Id="22545304"
    PostTypeId="1"
    CreationDate="2014-03-20T21:18:59.317"
    Score="1"
    ViewCount="116"
    Title="Black lambda Magic"
    Tags="<python><lambda><scipy>"
    Body="<p>In scipy.splev I found this piece of code:</p>
    <pre><code>
    def splev(x,tck,int der=0,int ext=0):
    t,c,k=tck
    try:
        c[0][0]
        parametric=True
    except:
        parametric=False
    if parametric:
        return list(map(lamda c,x=x,t=t,k=k,der=der:
        splev(x,[t,c,k],der,ext),c))
    else:
        return y.reshape(shape)</code></pre>
    <p>and I fail to understand the lamda,please enlighten me.</p>"
/>
```

图 2 单条数据结构的示例

Fig. 2 Example of a data structure

下文介绍筛选实验数据的具体过程。首先，根据数据中的 Tags 字段，筛选带 Python 标签的帖子。为了保证数据质量，在筛选数据时还添加了一条判断逻辑：选择 Score 字段大于 0 的帖子（用户浏览 SO 中的帖子时，可以选择赞成或不赞成该条帖子，Score 定义为赞成数减去不赞成数的值，当它大于 0 则说明该条帖子的信息是有价值的），然后对这些带有 Python 标签帖子的所有标签进行统计，过滤掉出现频次低于 10 的标签，因为这些标签数据几乎不包含实验所需信息，最后基于 Python 第三方库索引网站（pypi.org）和人工的方式，对这些标签进行判断，若是第三方库，则将其添加到第三方库集合中，最终得到一个标签集合 S。

筛选与 Python 第三方库相关数据的过程如下：在原始数据集 Dr 中遍历一条帖子 p 的标签字段，得到标签集合 T，其中 T 包含 1－5 个标签，若 T 中有第三方库集合 S 中的标签 t，则将该条帖子写入表格中，遍历完成后，得到了与 Python 第三方库相关的数据集 Dl。

经过对数据集 Dl 进行分析后，最终得到了这些第三方库在 SO 中出现频次的分布情况，经过分析发现，开发人员在社区中主要关注一些比较热门和技术已经比较成熟的第三方库，许多其他的第三方库出现频次极低甚至没有。因此，实验选择了在 SO 中出现频次最高的 60 个第三方库，后续基于这些数据进行分析。图 3 给出了实验中使用 WordCloud 技术统计出的若干常用的第三方库标签的分布情况。

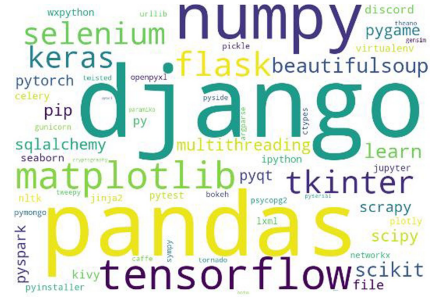

图 3 部分常用的第三方库标签

Fig. 3 Some examples of third-party library tags

3.1.2 构造文本和代码文档集

对于上文提到的常用第三方库标签，分别构造其对应的代码和文本文档集。以一个标签为例，若 Tags 字段包含该标签，则提取该条数据对应的 Body 字段。由于文本中代码片段存在于<pre><code>的标签中，因此可以使用正则表达式（pattern：‘<code>(.*?)</code>’）将文本与代码分离，并写入两个文件中，以同样的方式处理其他的第三方库，最后可以得到 60 个文本文档集与 60 个代码文档集。

因为文本和代码在格式上也未统一，并且各自包含着一些无效信息，所以需要对文档中的数据进行主题建模前的预处理，其中包括去除停用词和一些特殊符号，将大写字母转化为小写，删除无效链接以及乱码文本。此外，对于代码片段，需要删除空行、多余空格和特殊字符等，以此来保证代码片段所蕴含的主题信息不被遗漏并且可以被模型处理。

3.2 主题模型

考虑到数据集的特点，本文选取了目前比较成熟且被广泛使用的两种模型。第一种是基于非负矩阵分解 NMF 的聚类算法，NMF 可以生成大型数据集的基本部分的表示，从而产生可解释的模型。第二种方式是基于词向量的 TF-LDA 主题模型。本文将 NMF 与 TF-LDA 两种模型并联，并统一输入输出，形成混合模型来完成实验。下文将简要介绍两种模型的工作原理以及如何将其联合使用。

3.2.1 NMF

非负矩阵分解（NMF）可以有效地表示文本或图像中的主要成分，它假设数据和成分是非负的，在数据矩阵不包含负值的情况下，NMF 将样本 X 分解为非负矩阵 $\boldsymbol{W}$ 和 $\boldsymbol{H}$，通过优化距离 d 之间 X 和矩阵乘积 $\boldsymbol{WH}$，来更新目标函数直到

收敛。该方法适用于两个不同目标函数，即 Frobenius 范数和广义 Kullback-leibler 散度，前者是欧几里得范数对矩阵的扩展，而后者是基于概率的。

本文在使用 NMF 模型的过程中，将两种目标函数都考虑在内，进行多次实验，综合比较结果，最后选择了更适合文中数据集特点的广义 Kullback-leibler 散度作为模型的目标函数。下面是一些主要的参数设置，主题数 Topic_numbers 为 6，最大迭代次数 Max_iter 为 500，批处理大小 Batch_size 为 128。初始化方法 Init 为“nndsvda”，该属性决定了模型的初始化方法，对聚类效果性能的影响很大，它将分解的两个近似数据矩阵中的零元素都设置为所有元素的平均值，来降低这些“零点”对模型输出的不利影响。本文通过 Python 的第三方库 Scikit-learn 中的 Decomposition 模块训练 NMF 模型。

3.2.2 TF-LDA

基于词向量的 TF-LDA 模型是使用 TF-IDF 模型将语料中的单词转化为向量，再使用 LDA 主题模型进行拟合。具体来说，其首先对文本进行分词和去停用词，并构造词典；然后将其转化为 TF-IDF 特征，并作为 LDA 的输入。

LDA 是一种无监督机器学习技术，主要用于在文本集合中寻找潜在的主题，常用于从文档集合中发现抽象主题。它认为一篇文档是有多个主题的，而每个主题又对应着不同的词。LDA 中的主题是一序列相关的概念的抽象，模型使用语料训练并总结不同主题下的主题词，然后基于这些词，人为总结出一个概念，这个概念一般是这一系列概念的上位词或者抽象。

LDA 的图模型是一个三层贝叶斯模型，包含词、主题和文档 3 层结构，如图 4 所示。在图模型中，语料库 D 是一批文档 d 的集合，文档是一个单词序列，有 N 个词，K 个主题，方框代表重复抽样。$z_{d,n}$ 代表文档 d 中第 n 个单词的主题，$w_{d,n}$ 代表文档 d 中第 n 个词，β_k 代表第 k 个主题下的特征词分布，θ_d 表示文档 d 的主题分布。α 代表文档下主题的多项分布的 Dirichlet 先验参数，η 代表主题下特征词的多项分布的 Dirichlet 先验参数，一般根据经验给定。

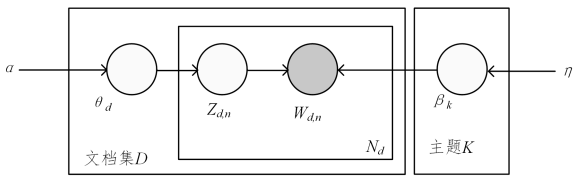

图 4 LDA 主题模型图示

Fig. 4 LDA topic model illustration

图模型中每个节点都是一个随机变量，在生成过程中发挥作用。阴影节点表示可以观察到的已知变量，无阴影点表示隐藏（潜在）变量。在这种情况下，语料库中的单词是我们观察到的唯一数据。潜在变量决定了语料库中主题的随机混合以及文档中单词的分布。LDA 的目标是使用观察到的词来推断隐藏的主题结构。参数设置上，主题数与 NMF 相同，最大迭代次数为 200，学习率 learning-offset 为 50.0，学习方式 learning-method 为 online。当前 LDA 主题模型有多种实现方式，例如 Python 的第三方库 scikit-learn 和 gensim[23]，它们被广泛应用于软件工程的数据挖掘和数据分析领域[24-25]。

3.2.3 混合模型

在数据处理部分，为了保证模型输入能够正常工作，本文将代码和文本信息进行了格式的统一处理，此时数据集是以多个文档的形式存在。在输入时先创建 3 个文档列表，text_code 列表保存文本结合代码信息的输入数据，text 和 code 列表分别存储文本和代码的输入数据，以便后续的对比实验。在统一输入格式之后，使用 TF-IDF 模型抽取文档特征用于拟合模型，其中参数 max_df 为 0.95，min-df 为 2，max-features 为 2000。

本文将两个模型对每个主题总结的主题词个数 n_topic_words 设置为 20，本文计算两个模型生成的簇之间的距离，根据簇之间共同主题词的个数以及其余主题词之间的语义相似度，可以判断是否将两个簇合并为一个主题，类似合并剩余主题。然后根据平均主题贡献度（输出越靠前的单词对主题贡献度越大）计算最终的主题词分布情况。在两个模型权重相同的基础上进行下面的计算，第一种情况，假设主题词 word1 在 LDA 中 Topic1 输出顺序中排名第一，在 NMF 中排名第三，那么 word1 对该主题的贡献度则排名第二；第二种情况，假设该单词只在一个簇中出现，那么其对主题的贡献度一定低于出现两次的单词，并按照这些单词排名顺序依次排在第一种情况之后。本文将迭代次数分别设置为 20，50，100，200 和 500 进行多次实验，并计算每个簇中主题词的平均贡献度，以得到更加稳定的主题词分布。本文构造的第三方库文档集是以第三方库名称为文件名称，例如 gensim 库的文档表示为 gensim.txt 文件，这个文件相当于代码和文本的混合文档，由于文中实验数据具有很强的领域特征，因此第三方库名称也会作为主题词出现在簇中。在对两个模型的簇进行调整后，根据文件名与不同簇内的关键词进行匹配，将这 60 个文档归为 6 个类别，以完成聚类。

4 实验结果与分析

基于本文方法，在 Stack Overflow 公开数据集上进行相关实证研究，下面给出实验结果，并回答上文提出的研究问题。

4.1 哪些 Python 第三方库在 SO 社区中被开发人员经常提问？

Python 有着大量的第三方库供开发人员直接使用，根据 Python 第三方库索引网站的数据，截至 2022 年 12 月，该网站已经发布超过 40 万个三方库，且它们在社区中的数量是不平衡的。

本文主要关注在 SO 中被经常讨论的第三方库，以使开发人员更深入地理解它。在对数据集的 Tags 字段进行分析后，得到了社区中讨论数量最多的 60 个常用库。表 1 列出了它们在社区中出现的频次。从表中可以看出，即使是出现频次最高的第三方库，讨论数量在社区中依然不平衡，例如针对 Python 中两种常用的开发框架 django 和 flask，前者似乎更受欢迎。而对于排名最多的前 10 个库，还涉及了一些大数据与科学计算、人工智能相关的技术，可以合理地推测开发人员在 Python 的使用领域上，主要集中在软件开发和人工智能。

表 1 提问数量排名前 60 的第三方库

Table 1 Top 60 third-party libraries by number of questions

标签	数量	标签	数量
django	281554	nltk	6468
pandas	228249	jinja2	5904
numpy	97664	plotly	5449
tensorflow	73463	pyinstaller	5287
matplotlib	62235	lxml	5061
flask	48021	jupyter	4921
tkinter	43234	networkx	4605
selenium	39340	pymongo	4351
keras	37686	pickle	4233
beautifulsoup	27658	openpyxl	4111
scikit-learn	24851	sympy	3787
multithreading	19921	urllib	3702
pip	19584	sphinx	3695
scipy	18981	bokeh	3686
sqlalchemy	17772	pycurl	3547
pygame	17740	gunicorn	3524
scrapy	16231	ctypes	3514
pyqt	15584	psycopg2	3380
pyspark	15469	argparse	3152
pytorch	15230	twisted	3070
file	13674	tornado	3069
kivy	10501	pyside	2900
discord.py	10179	caffe	2837
requests	9214	tweepy	2612
seaborn	7852	theano	2448
celery	7042	paramiko	2262
wxpython	7035	genism	2243
ipython	6869	pyserial	1871
pytest	6581	boto	1715
virtualenv	6459	cryptography	1552

4.2 Python 开发者在 SO 中主要讨论哪些主题？

从 4.1 节的实验结果可以看出，Python 涉及的领域比较广泛，但是开发人员对 Python 的应用更多的是集中在某些领域或主题下。基于之前构造的结合代码片段的数据，利用混合主题模型来揭示其内部蕴含的主题信息，并根据本文提出的映射规则将文档进行聚类，在模型参数的不断调整和训练之后，最终得到了 6 个主题。为了更加直观地观察主题词的分布情况，利用工具 pyLDAvis 对实验结果进行可视化。

图 5 给出了 2 号主题的部分主题词分布情况。图 5 中，横轴代表词频，排名越高代表该主题词对其所属主题的贡献度最大；灰色阴影部分代表该词在文档中的词频。以该主题为例，在去除停用词后，排名高位的主题词的词频一般比低位的高，这也符合文章中心思想一般与文章中多次出现的单词有关的这种规律。

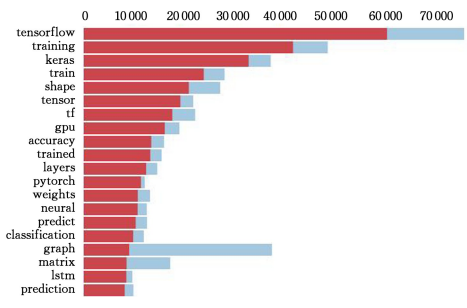

图 5 主题 2 的主题词分布情况

Fig. 5 Distribution of topic words in topic 2

为了更加深入了解这 6 个主题，在使用词干算法处理掉主题词集合中意义重复的词后，选择对主题贡献度最高的 20 个主题词，并在研究小组中请 5 名有着 Python 学习背景的人员对这些主题词进行了总结。表 2 列出了这些主题词的分布以及总结出的主题名称，不同于以往的挖掘主题，实验主要针对的是 Python 第三方库，有很强的领域特征，例如一些第三方库的名称往往作为该领域下的主题词，因此这也是本文按照名称来将文档映射到不同主题下的动机。

表 2 主题名称及主题词分布

Table 2 Topic names and topic words distribution

主题名称	主题词
网络安全	login button curl admin nginx urllib javascription docker password token gunicorn ajax cookieheroku proxi css email authent username
人工智能	tensorflow train keras shape tensor gpu accuraci layer pytorch weight neural predict classifi graph matrix lstm genism numpi Theano sklearn
数据分析	polt axi seaborn bokeh graph plotli matplotlib plot chart dash legend networkx scatter heatmap subplot labels panda td axe boxplot
文本处理	selenium xpath lxml scrape xml excel beautifulsoup td tagsoup openpyxl span webdriver spider chrom button bs4 tr href sheet
系统终端	virtualenv cell notebook argpars spark pyspark ctype ipynb openpyxl kdb anaconda3 markdown jupyterlab df kernel lab colab wmac venv paramiko
软件开发	button widget tkinter qt kivi pyseri pil serial qml pyqt arduino wxpython label menu dialog kv wx mouse click

4.3 这些主题各自有什么趋势？

趋势分析可以使社区开发人员对该领域的变化情况有着更深刻的认识。积极的趋势可能代表着 Python 在该领域的发展前景较好，这样可以使相关人员了解现在开始在这一主题下开展研究是否明智，以帮助他们制定合理的研究计划，规避风险。根据 4.2 节的实验结果，本文利用每条帖子的标签字段和创建日期字段，按照年份对每个主题下的帖子数量进行了趋势分析。这里选择 2009－2021 年的原因是这些年份的数据比较完整，可以体现每年变化的趋势。实验结果如图 6 所示。

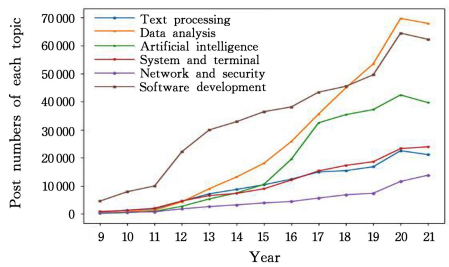

图 6 2009－2021 年不同主题下帖子数量的变化趋势

Fig. 6 Trends in the number of posts under different topics from 2009 to 2021

从图中可以看出，2012 年之前，Python 几乎只是应用在软件开发这个领域，随后几年便在各个主题下广泛应用，在数据分析、软件开发和人工智能中似乎更加流行。可以带来的一个启示是，Python 相关工作者未来从事这 3 个工作领域可能会更加容易。

4.4 代码片段与混合模型对提升模型聚类质量是否有帮助？

本节主要研究代码片段是否能为主题模型提供一定的信息，从而提高模型的建模质量，以及关注将文档聚类 NMF 与主题模型 LDA 结合，是否提高了模型的稳定性，从而避免局部最优解。最后给出主题数 topic_numbers 的选择原因。

4.4.1 代码片段的作用

在上文提到的模型参数的基础上，将文本文档和结合代码的文档分别作为模型的输入，并且根据困惑度来衡量聚类结果质量，以此来探究代码片段能否提供一定的主题信息。困惑度是在自然语言处理领域用来衡量主题模型的主流评测标准，它直观地表达了模型的聚类效果。困惑度越低，说明效果越好。困惑度的计算式如式(1)所示：

$$Perplexity=\exp\left\{\frac{-\sum\log(p(w))}{N}\right\} \tag{1}$$

其中，N 为测试集中出现的所有词个数，$p(w)$指测试集中出现每一个词的概率。图 7 给出了在不同主题数下两者的对比。从图中可以发现，无论主题数设置为多少，当使用结合代码的文档作为输入时，困惑度会明显下降，说明 SO 中用户提供的代码片段是有助于主题建模的。代码本质上也是一种特殊格式的文本，它也由单词组成，可以认为这些离散的单词中也包含了不同主题的语义信息。

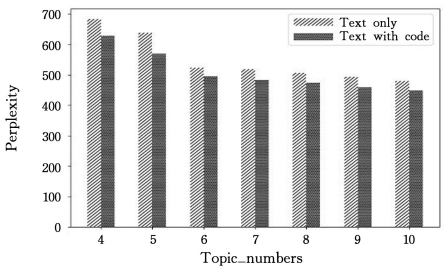

图 7 两种输入在不同主题数下的困惑度

Fig. 7 Perplexity of two inputs under different number of topics

4.4.2 混合模型的效果

一个理想主题模型每次生成的主题词应该分布相同或近似相同，且分类准确。然而，在实际实验中，通常随着参数的微小调整，每次得到的主题分布都有所差异。例如某个主题词在多个主题之间来回跳跃，或对同一主题的贡献度时高时低。下文将分别讨论混合模型聚类结果的稳定性和准确性。

为了验证混合文档聚类 NMF 与主题模型 TF-LDA 的作用，本文在单个模型和混合模型的基础上进行多次实验，提出了一种可以在不同迭代次数下判断模型聚类稳定程度的指标，并根据每一次实验结果中主题词的分布情况，来判断模型的稳定程度。假定同一模型在不同迭代次数下，生成的一个主题下不同的主题词个数为 N，每次都出现的主题词个数为 M，当 M/N 接近 1 时说明该模型在聚类上更加稳定，即有更高的簇内聚度，因为在 topic_numbers 确定的情况下，受迭代次数的波动幅度较小。本文还对每个主题 T 都计算其簇的稳定性，最终的结果如表 3 所列。实验结果表明，同一模型在不同迭代次数下训练的结果是有一定差异的，甚至在那些边界本身不够清晰的主题下，这种差异会被放大，例如 $T1$ 和 $T6$ 主题，在不同迭代次数下，模型得到的主题词分布差异明显，甚至大部分的主题词都不会重复出现。由于混合模型结合两者的输出重新计算分簇，因而一定程度上提高了主题词分布的稳定性。

表 3 混合模型在不同主题下的稳定性

Table 3 Stability of hybrid models under different topics

	*T*1	*T*2	*T*3	*T*4	*T*5	*T*6
TF-LDA	0.432	0.432	0.367	0.556	0.541	0.421
NMF	0.289	0.622	0.553	0.579	0.683	0.400
混合模型	**0.481**	**0.682**	**0.610**	**0.636**	**0.765**	**0.468**

基于 Python 第三方库索引网站分类标准以及专家知识，手工对其进行标注，对于不属于文中所提到分类的三方库，标准为“其他”。根据实验结果对第三方库的分类，对比标注的实际分类，以信息检索和统计学分类领域广泛采用的查全率(Recall)、查准率(Precision)和 F1 值作为衡量模型聚类准确性的指标。具体计算式如下：

$$Recall=\frac{TP}{TP+FP} \tag{2}$$

$$Precision=\frac{TP}{TP+FN} \tag{3}$$

$$F1=\frac{2\times TP}{2\times TP+FN+FP} \tag{4}$$

其中，TP 和 FP 分别是判断为真实分类和不属于真实分类的个数，FN 为实际不属于真实分类的个数。将混合模型和单一模型的实验结果与真实分类进行对比，结果如表 4 所列。

表 4 混合模型准确率的对比

Table 4 Comparison of accuracy on hybrid models

	Recall	*Precision*	*F*1
TF-LDA	0.604	0.563	0.583
NMF	0.623	0.689	0.654
混合模型	**0.742**	**0.712**	**0.727**

实验结果表明，提出的混合模型在聚类准确性上较单一模型有明显提升，这得益于联合单一模型输出，从而尽可能保留了那些多次出现的主题词，提升了聚类质量。本小节的实验结果表明，提出的混合模型在稳定性和准确性上都表现出较好的效果。

4.4.3 主题数 topic_numbers 的设置

本小节将探讨在主题数 topic_numbers 上的选择原因。由于困惑度 Perplexity 可以直观地展示主题数选择的合理性，因此根据在 Perplexity 上的表现来选取合适的主题数。实验针对 60 个第三方库的文档集进行主题分析，以达到聚类效果，因此在主题数选取上不易过多，否则会达不到聚类效果，例如将主题数设置为 60，那么在合适的训练参数下，最终得到的结果会是每一个文档自身占据一个类别，这将是毫无意义的，最后本文将主题数范围限制在 2～10 进行实验。图 8 给出了在该区间内困惑度的变化曲线。从图中可以看出，主题数小于 6 时，困惑度迅速下降，在主题数设置为 6 时，该值似乎达到一个“拐点”，随着再度细分主题，该值变化下降速度放缓。上文提到，主题数不易设置过多，否则会失去主题建模的意义，因此实验也不必追求困惑度的最小化，故认为设置主题数为 6 是最合适的。

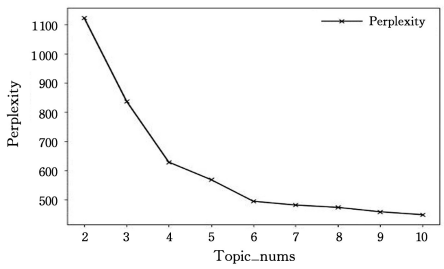

图 8 不同主题数对应的困惑度变化情况

Fig. 8 Changes of the perplexity corresponding to different topic numbers

结束语 虽然实验结果一定程度上表明了本文方法的有效性，但仍然有许多因素可能会对本文的研究构成潜在的威胁。

在数据的选取上，通过 Tags 字段来筛选所需数据，但在 SO 中仍然存在一些数据与实验相关，但不能通过标签筛选得到，因此在标签筛选中，将范围扩展到 Python 上，但这仍然会遗漏部分信息。为了提高数据集质量，在过滤时将字段 Score 考虑在内，也可能会过滤掉对实验有用的数据。在主题数的选取上，实验根据困惑度的变化情况以及聚类结果的额外分析，选定了一个认为合适的参数 topic_numbers，没有过度追求指标的最小化，因为这样带来的分类结果将是无意义的，但这样的选择也可能会对实验结果带来一定的偏差。此外，SO 是一个广泛使用和受欢迎的问答网站，但它并没有涵盖所有关于 Python 第三方库的讨论，以及用户在创建问题时，在分配标签的过程中不能保证该标签一定是准确的，这都会对实验的数据集产生一定的影响。

本文提出了一种结合代码片段和混合主题模型的方法，用于对软件问答网站的数据进行主题建模和聚类分析。首先，构造了与 Python 第三方库相关的数据，并且将代码和文本进行分离，形成文本和代码两种类型的文档；其次，对数据进行统一处理后，联合文本聚类模型 NMF 以及主题模型 TF-LDA 生成主题；最后，通过本文提出的方法，将构造的第三方库文档分为 6 个类别，分别是网络安全、数据分析、人工智能、文本处理、软件开发和系统终端。实验结果表明，使用代码结合文本的数据，有利于对编程领域相关的数据进行主题建模，并且联合多个模型可以一定程度上提高模型聚类结果的稳定性。在未来，本文将考虑引入更多的信息来扩展主题模型，并将其应用于其他领域的数据中。

参 考 文 献

[1] WANG Z Y, XIA X, AHMED E H, et al. What do programmers discuss about blockchain? a case study on the use of balanced lda and the reference architecture of a domain to capture online discussions about blockchain platforms across the stack exchange communities[J]. IEEE Transactions on Software Engineering, 2021, 47(7): 1331-1349.

[2] SUWONCHOOCHIT N, SENIVONGSE T. Classification of Database Technology Problems on Stack Overflow[C]// Proceedings of the 2021 IEEE/ACIS 19th International Conference on Software Engineering Research, Management and Applications, 2021: 21-26.

[3] SYED A, MEHDI B. What do concurrency developers ask about? A large-scale study using stack overflow[C]// Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. 2018: 1-10.

[4] BAJAJ K, PATTABIRAMAN K, MESBAH A, et al. Mining questions asked by web developers[C]// Proceedings of the 11th Working Conference on Mining Software Repositories. 2014: 112-121.

[5] BLEI M, NG Y, JORDAN I, et al. Latent Dirichlet Allocation [J]. Journal of machine Learning research, 2003, 2003(3): 993-1022.

[6] COSTA G, ORTALE R. Hierarchical Bayesian text modeling for the unsupervised joint analysis of latent topics and semantic clusters[J]. International Journal of Approximate Reasoning, 2022, 147: 23-39.

[7] AJAM G, CARLOS R, BENATALLAH B, et al. API Topics Issues in Stack Overflow Q&As Posts: An Empirical Study[C]// Proceedings of the 2020 XLVI Latin American Computing Conference. 2020: 147-155.

[8] AHASANUZZAMAN M, ASADUZZAMAN M, ROY K, et al. Classifying stack overflow posts on API issues[C]// Proceedings of the 2018 IEEE 25th International Conference on Software Analysis, Evolution and Reengineering. 2018: 244-254.

[9] ZHAO H H, LI Y H, LIU F W, et al. State and tendency: an empirical study of deep learning question and answer topics on Stack Overflow[J]. Science China Information Sciences, 2021, 64(11): 1-23.

[10] YANG X L, LO D, XIA X, et al. What security questions do developers ask? A large-scale study of stack overflow posts[J]. Journal of Computer Science and Technology, 2016, 31(5): 910-924.

[11] ROSEN C, SHIHAB E. What are mobile developers asking about? A large scale study using stack overflow[J]. Empirical Software Engineering, 2016, 21(3): 1192-1223.

[12] CHEN J, LI B, WANG J, et al. Knowledge Graph Enhanced Third-Party Library Recommendation for Mobile Application Development[J]. IEEE Access, 2020, 8: 42436-42446.

[13] ALEXANDRE R, OUNI A, SAIED A M, et al. On the Identification of Third-Party Library Usage Patterns for Android Applications[C]// Proceedings of the International Conference on Evaluation and Assessment in Software Engineering. 2022: 255-259.

[14] ALLAMANIS M, SUTTON C. Why, when, and what: Analyzing Stack Overflow questions by topic, type, and code[C]// Proceedings of the 10th Working Conference on Mining Software Repositories. 2013: 53-56.

[15] DEERWESTER S, DUMAIS S, LANDAUER T, et al. Indexing by latentsemantic analysis[J]. Journal of the American society for Information Science, 1990, 41(6): 391-407.

[16] HOFMANN T. Probabilistic latent semantic indexing[C]// Proceedings of International ACM SIGIR Conference on Research and Development in Information Retrieval. 1999: 50-57.

[17] BLEI D, LAFFERTY J. Correlated topic models [C]// Proce-

dings of Advances in Neural Information Processing Systems. 2005:147-154.

[18] BLEI D,LAFFERTY J. Dynamic topic models[C]// Proceedings of International Conference on Machine Learning. 2006:113-120.

[19] WALLACH H. Topic modeling:beyond bag-of-words[C]// Proceedings of International Conference on Machine Learning. 2006:977-984.

[20] BENGIO Y,DUCHARME R,VINCENT P,et al. A neural probabilistic language model[J]. Machine Learning, 2003, 3(2003):1137-1155.

[21] BARUA A,THOMAS W,HASSAN E,et al. What are developers talking about? an analysis of topics and trends in stack overflow[J]. Empirical Software Engineering,2014,19(3):619-654.

[22] PLETEA D,VASILESCU B,SEREBRENIK A,et al. Security and emotion: sentiment analysis of security discussions on GitHub[C]// Proceedings of the 11th Working Conference on Mining Software Repositories. 2014:348-351.

[23] PEDREGOSAF,VAROQUAUX G,VINCENT W,et al. Scikit-learn:machine learning in Python[J]. Journal of Machine Learning Research,2011,12(2011):2825-2830.

[24] CHEN Z F,MA W W Y,LIN W,et al. A study on the changes of dynamic feature code when fixing bugs:towards the benefits and costs of Python dynamic features[J]. Science China Information Sciences,2018,61(1):1-18.

[25] CHEN L,WU D,MA W W W Y,et al. How C++ templates are used for generic programming[J]. ACM Transactions on Software Engineering and Methodology,2020,29(1):1-49.

WEI Linlin, born in 1998, postgraduate. His main research interests include software engineering and text mining.

SHEN Guohua, born in 1976, Ph.D, is a senior member of CCF (No. E200018171S). His main research interests include software engineering, software metrics, semantic web and description logic.

(责任编辑:喻藜)