计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.201200031

面向中文 APP 用户评论数据的软件需求挖掘方法

王 莹 郑丽伟 张禹尧 张晓妘
北京信息科技大学计算机学院 北京 100101
(wy_2407556211@163.com)

摘 要 从 APP 用户反馈数据中挖掘用户需求是 APP 迭代更新和需求获取的一种重要方式,用户在 APP 应用市场中发表对 APP 不同维度的评价,其中蕴含着用户对 APP 软件的改善需求。但是,目前用户反馈数据存在数量大、质量良莠不齐的状况,如何从海量的用户评论数据中省时省力地挖掘出有价值的需求,具有重要的研究与现实意义。文中着眼于 APP 开发问题,选取 360 手机助手中的 APP 用户评论数据,旨在挖掘蕴含于用户评论数据中的软件需求。首先,从功能性需求与非功能性需求两个维度出发,将 APP 用户评论数据中蕴含的软件需求划分为功能待添加、功能待改进、性能、可用性、可靠性 5 个需求类别;其次,对用户评论进行数据采集、标注,构建 APP 评论需求挖掘数据集;最后,利用构建好的数据集进行模型训练与交叉验证,探究主流深度学习方法相较于统计机器学习模型在该任务上的表现。实验表明,采用的深度学习模型 TextCNN,TextRNN 和 Transformer 相比传统的统计机器学习模型在此任务上更具优势。

关键词:APP 用户评论;软件需求挖掘;机器学习;中文数据集
中图法分类号 TP311

Software Requirement Mining Method for Chinese APP User Review Data

WANG Ying,ZHENG Li-wei,ZHANG Yu-yao and ZHANG Xiao-yun
School of Computer Science,Beijing Information Science and Technology University,Beijing 100101,China

Abstract Mining requirements from APP user review data is an important way to obtain requirements,because users publish reviews of different dimensions of APP in the APP application market,which contain many requirements for APP. The APP user review data on the 360 mobile assistant is chosen in our experiments,aiming to discover the software requirements contained in these review data. Firstly,the software requirements contained in APP user review data are divided into five categories,which include functions to be added,functions to be improved,performance,availability,and reliability. Secondly,data collection,labeling of user comments and constructing app review requirements mining data set are carried on. Finally,the constructed data set is used for model training and testing to explore the performance of deep learning methods compared with statistical machine learning models on this task. The experiment results show that the deep learning models,TextCNN,TextRNN,and Transformer used in this paper,have more advantages in this task than traditional statistical machine learning models.

Keywords APP user reviews,Software requirements mining,Machine learning,Chinese data set

1 前言

随着当前 APP 数量的快速增长,基于用户视角的 APP 评论数据逐渐增多。用户评论数据中包含着丰富的潜在软件需求信息[1-4],挖掘用户维度的软件需求对辅助 APP 的设计、驱动 APP 的迭代更新具有一定的研究意义。Jiang 等[5]认为 APP 在线用户评论具有覆盖用户范围广、内容丰富、时效性强等优势,是软件使用反馈获取的重要资源。Schneider[6]首次发现 APP 用户反馈中包含着大量的需求种子,因此建议采用 APP 用户反馈数据来提取需求并改善软件产品。Iacob 等[7]认为,用户可以通过评论为应用程序建议新功能,或者表达对已经存在的功能进行重新设计的偏好。Chen 等[8]认为,对于开发人员修复错误、实现新功能以及敏捷改善用户体验而言,来自用户的及时、建设性的反馈至关重要。Khan 等[9]认为,作为需求工程师,聆听最终用户并了解他们的需求对软件系统设计和演进的方向至关重要。用户论坛和社交媒体的广泛使用,使得它们成为获取用户需求数据的天然信息源[10-11]。因此,从 APP 应用市场的用户评论数据中挖掘用户需求,为当前迭代式软件开发提供了一种良好的需求获取途径。然而,APP 用户评论数据的用户群体广泛、数据规模大、数据杂乱、数据质量良莠不齐且数据中包含很多混乱且无用的信息,如何清晰地从海量评论数据中过滤无用的数据并挖

到稿日期:2020-09-03 返修日期:2020-10-31 本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
基金项目:国家自然科学基金项目(61402043)
This work was supported by the National Natural Science Foundation of China(61402043).
通信作者:郑丽伟(zlw@bistu.edu.cn)

掘出包含用户需求的有用信息，是当前亟待解决的问题。

2　相关工作

软件需求的误解和缺失容易导致设计缺陷，同时诱发代价高昂的问题[12]。因此完整准确地找出用户真正的需求在软件开发中具有重要的意义。从海量用户评论数据中挖掘出有价值的知识和信息是有效抽取用户需求的重要途径。目前，国内外相关研究工作大多结合文本分析、情感分析、自然语言处理和词典等方法对原始数据进行特征定义与预处理，并运用机器学习等分类方法对预处理好的评论数据进行分类。这些工作主要从数据本身的结构特征出发，有助于过滤掉无用的评论，并且帮助利益相关者（如开发人员、分析师和其他用户）发现一些容易被忽略、其不感兴趣但包含用户需求的评论。Chen 等[8]提出了一种用于提取和挖掘信息评论的计算框架 AR-Miner。其采用半监督学习的朴素贝叶斯（Naive Bayes，NB）方法对来自 Google Play 的 4 个典型 APP 的用户评论数据进行有用信息与无用信息的分类。Maalej 等[13]使用评论元数据、文本分类、自然语言处理和情感分析技术将 APP 用户评论数据分为错误报告、功能请求、用户体验和评级 4 种类型。他们在分类工作中发现，仅使用元数据作为特征会导致较差的分类准确性，而当结合自然语言处理时，效果有明显提升，并且发现多个二元分类器优于单个多分类器。类似的工作还有 Panichella 等[14]首次结合了自然语言处理、文本分析和情感分析方法对软件维护和演化相关的用户评论数据进行分类。该工作从句子粒度分析用户评论数据集，认为句子是独立的，每个句子的特征都是单独构建的，并将用户评论数据分为功能请求、问题发现、信息搜索和信息提供 4 个类别；与此同时引入了丰富的功能集，包括语言功能、文本功能和情感功能，指出可以研究文本的 3 个方面——词典（评论中使用的特定单词）、结构（构成评论的语法框架）、情感（用户对所讨论主题的态度或情绪）。但是该设计会丢失用户评论（尤其是简短的句子）中重要的结构信息，从而导致提取的结构信息性能不理想。而在与传统方法 AR-Miner 的对比工作中，Villarroel 等[15]提出了 CLAP（Crowd Listener for Release Planning）方法，将用户评论自动分类为对新功能、错误报告和其他功能的建议。与此同时 Panichella 等[16]提出了 ARdoc 工具，其结合了自然语言解析、文本分析和情感分析等技术，能够从 APP 用户评论中提取结构、情感和词典特征，并通过决策树（Decision Tree，DT）方法对 APP 评论中的有用反馈进行自动分类，从而可为 APP 开发人员提供相关有用反馈数据。实验结果表明，基于 Ardoc 工具的分类结果具有很高的精确率和召回率。Suprayogi 等[17]使用文本挖掘、情感分析和主题建模的组合方法来提取有关应用程序的用户评论信息。其使用支持向量机（Support Vector Machine，SVM）将评论的内容分为 4 类：问题、改进、要求和其他。Buchan 等[18]运用 SVM 方法从用户的在线评论中自动识别用户对新功能存在需求的文本，将提取的文本分类为功能和非功能两种，极大地提高了分类的准确度。Chen 等[19]提出了 RASL（Review Analysis method based on SVM and LDA）方法，其针对移动应用的中、差评提取特征，并对评论进行多标签分类。Hu 等[20]构建了评价对象和评价观点抽取规则，来抽取一条用户评论的评价对象和评价观点。其借鉴半监督自学习的思想，并基于候选评论模式库扩充评论种子库，从而实现 APP 软件使用反馈的用户评论在“软件满足的需求”“软件存在的问题”和“软件未达到的期望”3 种类别上的循环挖掘，给潜在用户提供软件实际使用情况的参考。

以上对 APP 用户评论数据的分析挖掘工作主要从数据本身的结构特征出发对需求类别进行定义；同时，结合文本分析、情感分析等方法对原始数据的特征、规则进行处理，在相关研究领域内，在一定程度上提升了有价值的需求信息的挖掘效率和质量，但是在以下方面尚有不足。1）基于评论数据特征进行的多类别分类，其类别受主观影响较大。另外，软件需求通常被划分为功能性需求与非功能性需求，以上工作不能清晰地将这两种需求进行区分，往往将二者混合，类别定义较为模糊[17,20]。因此，本文提出一种针对用户评论数据的需求分类体系。该体系能够有效区分功能性需求与非功能性需求，更加清晰明了地展示出有价值的用户反馈信息，从而辅助 APP 的开发与迭代更新。2）在机器学习模型选取方面，目前相关研究工作多采用经典的统计机器学习方法而不采用较为热门的深度学习技术。造成这一现象的主要原因是，虽然经典的深度学习技术在图像、声音等模式识别领域具有明显优势，但在类似用户评论等中文自然语言数据上表现欠佳。近年来，随着机器学习技术在自然语言处理，尤其是中文自然语言处理领域的不断推进，我们认为，机器学习，尤其是新兴深度学习技术，将成为软件需求挖掘的有效手段。因此，本文从类别定义与分类模型出发，对软件需求类别从功能性需求与非功能性需求两个维度进行更加详细的定义，并基于此分类体系，选择 TextCNN，TextRNN，Transformer3 种深度学习方法，选择 SVM，NB，DT 和随机森林（Random Forests，RF）4 种传统统计机器学习方法，分别在选定的 APP 用户评论数据中进行软件需求挖掘，并通过实验比较这两类方法在本任务上的性能表现。

3　研究框架

本文的整体研究框架如图 1 所示。

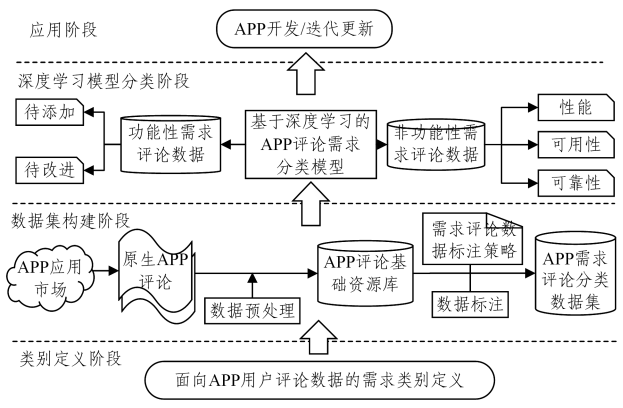

图 1　总体框架

Fig. 1　Overall framework

第一个阶段是面向 APP 用户评论数据的需求类别定义。本文基于对所采集 APP 用户评论数据的分析，定义了面向 APP 用户评论数据的功能性需求与非功能性需求，进而将功能性需求类型进一步划分为待添加、待改进，将非功能性需求类型划分为性能、可用性和可靠性。该分类体系能够基本覆盖当前所采集数据蕴含的 APP 开发需求。随着数据集的不断扩充与提炼，我们也将对此分类体系进行持续改进。

第二个阶段是面向 APP 用户评论数据的需求挖掘数据集的构建阶段。首先，调研已有应用市场，主要以用户评论数据的质量为参考标准，选取了 360 手机助手作为爬取对象，详细分析应用市场各个类别下的 APP 用户评论数据，以数据质量、下载排行榜以及领域覆盖为导向，选择系统安全类、通讯社交类、影音视听类、教育学习类下的共 12 个热门 APP，在爬取了数万条原生的 APP 用户评论数据后，构建了 APP 评论基础资源库。然后，对用户评论数据进行分析，制定需求评论数据标注策略，安排数名标注者进行数据标注从而构造 APP 评论需求分类的实验数据集。

第三个阶段是深度学习模型分类阶段。经调研，选择 TextCNN，TextRNN，Transformer 3 种典型深度学习方法从整体性能、数据类别与预训练 3 个方面对 APP 用户评论数据中的需求发现与挖掘进行探讨，同时对比相关研究工作中的 SVM，NB，DT，RF 4 种传统统计机器学习方法。

第四个阶段是应用阶段。将上一阶段获取到的分类结果，即各类别下的用户需求应用到 APP 的开发与迭代更新上，旨在填补需求缺失，让 APP 设计者更加方便快捷地获取到一线 APP 用户的高频需求，从而依据用户需求完成新 APP 的开发与已有 APP 的迭代更新。

4 面向 APP 用户评论数据的需求类别划分

在软件工程[21]中对软件需求的定义是：1)用户解决问题或达到目标所需的条件或权能；2)系统或系统部件要满足合同、标准、规范或其他正式文档应具有的条件或权能；3)一种反映定义 1)或 2)所述条件或权能的文档说明，主要包含功能性需求与非功能性需求。基于 APP 用户评论需求的定义立足于软件需求的定义，主要表达了用户在功能性与非功能性上对 APP 的需求，但是多数用户在表达需求的过程中并非直面地表达，往往较为隐性，例如微信的用户评论“内存占用太大了，就不能给我省省手机空间吗?”用户并没有直接表达出“建议压缩微信的占用空间”这一需求，而是将其以一种吐槽、抱怨的方式表现出来。我们认为此类隐性表达用户需求的 APP 用户评论也蕴含着真正的软件需求。由此，本文将 APP 用户评论中蕴含的需求类别划分为 2 大类，5 小类。以微信为例，不同类别需求的评论数据样例如表 1 所列。

表 1 不同类别数据样例

Table 1 Sample data for different requirement categories

类别		数据样例
功能性需求	待添加	想要添加特别关心功能
	待改进	为啥实名还要银行卡，这对我们很麻烦的。
非功能性需求	性能	占内存越来越大，太离谱！
	可用性	广告太多了，按钮位置简直反人类啊！
	可靠性	我微信打不开了。

4.1 面向 APP 用户评论数据的功能性需求定义

功能性需求通常被描述为系统要实现的功能或系统在特定条件下的行为，即软件具体要做的事情，可以用功能和行为点构成的需求集合来描述。借鉴功能性的定义对 APP 功能性需求进行定义：对于某个 APP 的系统实现的功能(静态，具有或缺失某一功能)、系统在特定条件下运行时的行为(动态，描述应用程序运行某功能时出现问题或意外行为)等方面，用户(意见发表人)对软件发表了带有一定需求的评论，称之为基于 APP 用户评论数据的功能性需求，可以形式化地表示为以下的三元组，其中功能性需求类型可以进一步划分为待添加和待改进。

面向 APP 用户评论数据的功能性需求：=〈软件，功能/行为方面描述，功能性需求类型〉

4.2 面向 APP 用户评论数据的非功能性需求定义

非功能性需求的准确获取对项目的成功至关重要[22-23]，其面向的是软件的整体属性，是指对系统功能或服务附加的质量约束，即软件产品为满足用户业务需求而必须具有除功能性需求以外的特性，通常可以用性能、可靠性、可用性、安全性和可维护性 5 大类别[24]对其进行识别与区分。APP 的可维护性与系统安全性多为运营方关注，相关的需求数据较少出现在用户评论信息中。实际上，在数据标注过程中我们也发现，目前采集的数据集中对用户安全性的反馈数据较少，不足以支撑后续的模型训练，因此本文仅考虑用户对软件在性能、可靠性和可用性 3 个方面的使用反馈。我们将从用户对软件发表的关于这 3 个属性的评论数据中发现的需求，称为基于 APP 用户评论数据的非功能性需求，其具体类别及描述如表 2 所列。

面向 APP 用户评论数据的非功能性需求：=〈软件，性能/可靠性/可用性相关描述，非功能性需求类型〉

表 2 非功能性需求类别描述与关键词

Table 2 Non-functional requirements category description and keywords

类别	类别描述	典型关键词
性能	对时间及时性及资源经济性要求的符合程度	响应时间、占用、内存、效率、及时、准确率等
可靠性	在特定时间段内无失效运行的概率	可靠、稳定、容错、故障、崩溃、易恢复等
可用性	对用户来说少错和令人满意的程度，即用户对软件的主观感受	易用、易学、界面、好记、人性化、视觉等

5 面向 APP 用户评论数据的需求挖掘数据集的构建

综合调研分析后，本文选取 360 手机助手的用户评论数据，以数据质量、下载排行榜以及领域覆盖为导向，最终选择系统安全类、通讯社交类、影音视听类、教育学习类的共 12 个热门 APP 进行用户评论数据爬取。在各 APP 的用户评论中，好评类别中包含需求性信息的数据很少，因此我们只对每个 APP 评论中的中评和差评数据进行爬取。

此外，由于 APP 评论数据存在格式不规范、表达多样等特点，在爬取过程中，需同时进行数据清洗与过滤等数据预处

理工作。首先，去除评论文本中的特殊符号，如“#”“*”等，删除这些符号可以使得文本更纯粹，便于机器理解；接下来根据评论文本长度对文本进行过滤，这里过滤掉文本长度小于 2 的数据；最后，对句子中出现的英文单词进行转换，统一转换为小写，方便后续处理。

各 APP 用户评论爬取信息如表 3 的 APP 评论基础资源库所列，4 类 12 款 APP 共计爬取数据 12 072 条，其中中评 6 045 条，差评 6 027 条，这些数据构成 APP 评论基础资源库。此外，考虑到 BERT[25] 等深度学习模型可以字为单位进行计算，且效果甚至比分词的性能更优，因此本文在存储评论数据时同时存储了未分词的评论数据，以支持在后续的实验中使用以字为单位的深度学习模型。

表 3　APP 评论基础资源库

Table 3　APP review basic resource library

APP 类别	APP 名称	中评数据	差评数据	数据总量
系统安全	360 手机卫士	513	510	1 023
	QQ 安全助手	502	510	1 012
	百度	503	518	1 021
	QQ 浏览器	502	502	1 004
通讯社交	微信	508	508	1 016
	手机 QQ	508	525	1 033
影音视听	网易云音乐	518	526	1 044
	QQ 音乐	502	501	1 003
	腾讯视频	509	510	1 019
	Bilibili 视频	524	555	1 079
教育学习	有道词典	507	506	1 013
	金山词霸	449	356	805
ALL		6 045	6 027	12 072

数据的标注体系按照前文中定义的类别设置。这里需要额外考虑一种特殊的类别，即不能为开发者提供有用信息的评论，只是纯粹的用户情感表达或模糊的意见，我们将这一类别定义为其他类别。最终，完整的标注体系中，类别设定为 3 大类、6 小类。3 大类分别为功能性需求类、非功能性需求类和其他类，其中功能性需求类又可划分为待添加与待改进，非功能性需求类则划分为性能、可用性和可靠性，其他类不进行更细粒度的划分。根据设定的标注体系，首先让多名标注者同时对 2600 多条评论数据进行标注，然后对标注结果进行一致性验证，针对单条评论数据，取一致性最高的结果作为最终标注结果。各类别评论数据数量及占整体数据的比例如表 4 所列。

表 4　APP 需求评论分类数据集

Table 4　APP requirements review classification data set

类别	子类别	评论数量/占比	合计数量/占比
功能性需求	待添加	122/4.52%	425/15.74%
	待改进	303/11.22%	
非功能性需求	性能	157/5.81%	847/31.37%
	可用性	390/14.44%	
	可靠性	300/11.11%	
其他	其他	1408/52.15%	1408/52.15%

6　基于深度学习分类模型的 APP 用户评论需求挖掘

6.1　基于 TextCNN 的挖掘

卷积神经网络(Convolutional Neural Networks，CNN)在图像领域应用广泛。TextCNN[26] 模型首次将 CNN 引入文本分类的任务中，其核心思想是利用不同尺度的卷积核对编码后的文本输入进行卷积，不同尺度的卷积核可以捕捉到不同个数的相邻词的相关性特征，其模型结构如图 2 所示。

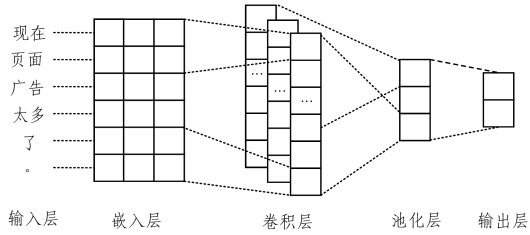

图 2　TextCNN 模型结构

Fig. 2　Model structure of TextCNN

基于 TextCNN 的挖掘过程如下。

设评论 $C=\{w_1,w_2,\cdots,w_t,\cdots,w_n\}$，$n$ 为文本长度，w_t 为评论 C 中的第 t 个词。首先，将 C 输入至嵌入层进行词编码，得到编码表示 $\boldsymbol{X}\in R^{n*d}$，如式(1)所示：

$$\boldsymbol{X}=Embedding(C) \tag{1}$$

其中，$Embedding$ 表示编码层，d 为嵌入层编码维度。

编码后，使用不同尺度的卷积核进行卷积，将卷积后的结果进行拼接得到 $\boldsymbol{ConvX}\in R^{n*3h}$，如式(2)－式(5)所示：

$$\boldsymbol{ConvX}_1=Conv1D_{k1}(\boldsymbol{X}) \tag{2}$$

$$\boldsymbol{ConvX}_2=Conv1D_{k2}(\boldsymbol{X}) \tag{3}$$

$$\boldsymbol{ConvX}_3=Conv1D_{k3}(\boldsymbol{X}) \tag{4}$$

$$\boldsymbol{ConvX}=[\boldsymbol{ConvX}_1,\boldsymbol{ConvX}_2,\boldsymbol{ConvX}_3] \tag{5}$$

其中，$Conv1D_k$ 为卷积核大小为 k 的一维卷积层，h 为单个卷积层的输出通道数。

经过卷积层编码后，需要进行一层池化，将全部的编码信息压缩到单维的向量空间，这里使用最大化池化作为池化层的池化策略，如式(6)所示。池化过后，得到评论的单维向量表示 $\boldsymbol{G}\in R^{3h}$。

$$\boldsymbol{G}=MaxPooling(\boldsymbol{ConvX}) \tag{6}$$

最后，将评论的完整表示 H 输入预测层进行预测输出，预测层由全连接网络与 Softmax 激活函数组成，如式(7)所示，得到 $\boldsymbol{O}\in R^L$。

$$\boldsymbol{O}=softmax(\boldsymbol{WG}+\boldsymbol{b}) \tag{7}$$

其中，L 为类别总数，$\boldsymbol{W}$ 和 $\boldsymbol{b}$ 为全连接网络中可学习的参数。

6.2　基于 TextRNN 的挖掘

循环神经网络(Recurrent Neural Network，RNN)可以处理序列类型的数据，当输入为序列信息时，RNN 会按照序列的顺序进行编码。考虑到数据中的序列关系，RNN 往往能取得比 CNN 更好的性能。基于 RNN 的文本分类模型 TextRNN[27] 的结构如图 3 所示。整体来看，TextRNN 模型与 TextCNN 模型的主要差异在编码层，TextRNN 模型将多个不同尺度的 CNN 替换为 RNN。然而，RNN 存在长时间依赖、梯度爆炸、梯度弥散等问题，因此研究者往往会使用其变体——长短期记忆[28](Long Short-Term Memory，LSTM)网络和门控循环单元[29](Gated Recurrent Unit，GRU)。

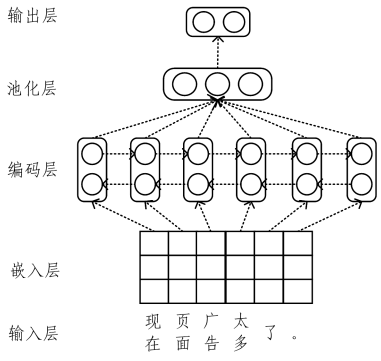

图 3 TextRNN 模型结构

Fig. 3 Model structre of TextRNN

本文选择使用 GRU。GRU 对 RNN 中的循环单元进行改进，加入了重置和更新两个门结构，并通过门结构控制信息的编码状态。在时间步 t 时刻，设输入为嵌入层编码后的向量表示 $\boldsymbol{x}_t$，$t-1$ 时刻的隐藏状态为 $\boldsymbol{h}_{t-1}$。

首先，t 时刻的输入 $\boldsymbol{x}_t$ 与 $t-1$ 时刻的隐藏状态 $\boldsymbol{h}_{t-1}$ 分别输入至重置门和更新门，分别得到重置门控状态 $\boldsymbol{r}_t$ 和更新门控状态 $\boldsymbol{z}_t$，如式(8)、式(9)所示：

$$\boldsymbol{z}_t=\sigma(\boldsymbol{W}_z\cdot[\boldsymbol{h}_{t-1},\boldsymbol{x}_t]) \tag{8}$$

$$\boldsymbol{r}_t=\sigma(\boldsymbol{W}_r\cdot[\boldsymbol{h}_{t-1},\boldsymbol{x}_t]) \tag{9}$$

其中，$\boldsymbol{W}_z$ 和 $\boldsymbol{W}_r$ 为可学习的参数，σ 表示 Sigmoid 激活函数。

接下来，利用重置门的计算结果 $\boldsymbol{r}_t$ 对 $t-1$ 时刻的隐藏状态 $\boldsymbol{h}_{t-1}$ 进行“重置”，得到 $\boldsymbol{h}'_{t-1}$，再将其与输入 x_t 进行拼接，计算候选隐藏状态 $\boldsymbol{h}_t'$，如式(10)、式(11)所示：

$$\boldsymbol{h}'_{t-1}=\boldsymbol{r}_t*\boldsymbol{h}_{t-1} \tag{10}$$

$$\boldsymbol{h}_t'=\tanh(\boldsymbol{W}_h\cdot[\boldsymbol{h}'_{t-1},\boldsymbol{x}_t]) \tag{11}$$

其中，$\boldsymbol{W}_h$ 为可学习的参数。

最后，利用更新门的计算结果 $\boldsymbol{z}_t$，结合隐藏状态 $\boldsymbol{h}_t'$ 与 $t-1$ 时刻的隐藏状态 $\boldsymbol{h}_{t-1}$，计算 t 时刻的隐藏状态 $\boldsymbol{h}_t$，如式(12)所示：

$$\boldsymbol{h}_t=(1-\boldsymbol{z}_t)*\boldsymbol{h}_{t-1}+\boldsymbol{z}_t*\boldsymbol{h}_t' \tag{12}$$

此外，在处理文本数据时，仅单向地考虑序列信息还不够，因为这相当于模型只看到了数据的上文，而忽略了下文，对于文本编码表示而言，其下文同样重要。因此，在 RNN 的使用中一般使用双向编码的版本。这里，同样以 $\boldsymbol{X}$ 表述嵌入层编码的结果，用 $\boldsymbol{G}$ 表示双向 GRU 编码后的结果，$\boldsymbol{G}$ 的计算过程如式(13)－式(15)所示：

$$\overrightarrow{\boldsymbol{G}}=\overrightarrow{GRU(\boldsymbol{X})} \tag{13}$$

$$\overleftarrow{\boldsymbol{G}}=\overleftarrow{GRU(\boldsymbol{X})} \tag{14}$$

$$\boldsymbol{G}=[\overrightarrow{\boldsymbol{G}},\overleftarrow{\boldsymbol{G}}] \tag{15}$$

6.3 基于 Transformer 的挖掘

Transformer[30] 结构在 2017 年被提出，用于机器翻译任务。该结构基于自注意力(Self-Attention)机制，后续的大量研究表明，该结构具有比 CNN 和 RNN 更强的信息编码能力。基于 Transformer 编码器的文本分类模型的结构如图 4 所示。与 TextCNN 和 TextRNN 不同的是，Transformer 的嵌入层不仅需要对输入文本中的词进行嵌入编码，还需要对文本中各个词的位置信息进行嵌入编码，这样做的原因在于 Transformer 结构全面采用了注意力机制，其自身结构无法编码输入序列的位置信息。

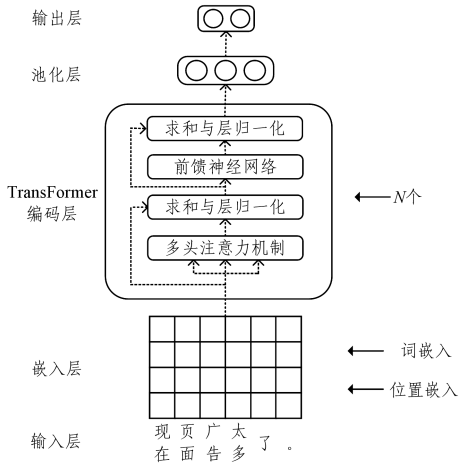

图 4 Transformer 文本分类模型结构

Fig. 4 Model structure of Transformer for text classification

采用 Transformer 的挖掘过程如下。

首先，词嵌入的部分与上文一致，不再赘述。

经过词嵌入后得到 $\boldsymbol{X}_{word}\in R^{n*d}$。关于位置嵌入，其计算公式如式(16)所示：

$$PE=\begin{cases}\sin(pos/10000^{2i/d}), & i\%2==0\\ \cos(pos/10000^{2i/d}), & i\%2==1\end{cases} \tag{16}$$

其中，pos 表示词在当前输入序列中的位置，i 表示 d 中的第 i 个维度。

完整的嵌入层编码结果为词嵌入与位置嵌入两者分别编码的和，如式(17)所示，最终得到 $\boldsymbol{X}\in R^{n*d}$。

$$\boldsymbol{X}=\boldsymbol{X}_{word}+PE(C) \tag{17}$$

Transformer 结构的编码器包含两个主要子层：多头注意力层(Multi-head Attention)和前馈神经网络层(Position-wise Feed-Forward network)，其中多头注意力层是 Transformer 的核心子层。多头注意力层由多个缩放点击注意力(Scaled dot-product attention)组成，这部分的计算方式如式(18)－式(20)所示：

$$MultiHead(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V})=Concat(\boldsymbol{head}_1,\cdots,\boldsymbol{head}_h)\boldsymbol{W}^O \tag{18}$$

$$\boldsymbol{head}_i=Attention(\boldsymbol{QW}_i^Q,\boldsymbol{KW}_i^K,\boldsymbol{VW}_i^V) \tag{19}$$

$$Attention(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V})=softmax\left(\frac{\boldsymbol{QK}^T}{\sqrt{d_k}}\right)\boldsymbol{V} \tag{20}$$

其中，$\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}$ 为输入，这里在计算时统一替换为 $\boldsymbol{X}$ 即可；h 为 Head 的数量，$\boldsymbol{W}_i^Q,\boldsymbol{W}_i^K,\boldsymbol{W}_i^V,\boldsymbol{W}^O$ 为可学习的参数，d_k 表示 $\boldsymbol{K}$ 的编码维度。相比多头注意力层，前馈神经网络层相对较为简单，仅由两层全连接神经网络组成，主要提供非线性变换，这里的公式不再进行展开。

除了两个主要子层外，Transformer 对子层与子层之间的连接方式同样进行了设置，使用了残差连接的方式将数据进行连接，同时对残差连接的结果进行层归一化，其计算方式如式(21)所示：

$$\boldsymbol{Layer}_{\text{output}}=LayerNorm(\boldsymbol{x}+SubLayer(\boldsymbol{x})) \tag{21}$$

其中，*SubLayer* 可以是多头注意力层或者前馈神经网络层。

Transformer 可以包含多个编码器，只需要将各个编码器叠加，即上一个编码器的输出作为下一个编码器的输入即可，多个编码器拼接在一起构成完整的 Transformer 结构。同样，使用 Transformer 进行文本分类时，将 TextCNN 中的编码层替换为 Transformer 即可，后续的结构则与 TextRNN 保持一致，只需要进行池化即可进行预测层的预测，这里不再展开讨论。

7　实验及结果分析

7.1　实验设置

(1)统计模型设置

目前，针对 APP 用户评论挖掘相关的工作所使用的模型主要集中在统计机器学习模型。我们选取相关工作中最常使用的 4 种模型作为基准模型，分别为朴素贝叶斯、决策树、支持向量机和随机森林。对于基准模型，本文使用 scikit-learn 框架进行实现，使用 TF-IDF 方法提取特征作为输入。

1)朴素贝叶斯[31]。朴素贝叶斯是基于贝叶斯原理的模型，该模型假设属性之间相互独立，求解各个属性的概率，并利用概率信息来进行决策。

2)决策树[32]。决策树是一种树形结构的机器学习方法，其子结点代表一个输出的结果，非子结点表示一个属性的判断，该模型具有较强的可解释性。

3)支持向量机[33]。支持向量机是一种典型的有监督学习模型，其学习策略是求解样本之间的最大超平面距离；同时该方法可以支持核函数，将原本线性不可分的数据映射到更高维度的空间，从而解决非线性可分问题，是常见的核学习方法之一。

4)随机森林[34]。随机森林是一种集成学习算法，根据训练样本训练多棵决策树，在输出时多棵决策树共同决策。该算法对样本类别不均衡的数据效果较好。

(2)深度学习模型设置

本文使用深度学习框架 Pytorch[35] 实现深度学习模型，不同模型的通用超参数(Hyperparameters)设置如表 5 所列。

表 5　参数设置

Table 5　Parameter settings

参数	设置
训练轮数(Epoch)	30
学习率(Learning Rate)	1.00×10^{-3}
权重衰减(Weight Decay)	1.00×10^{-4}
优化器(Optimizer)	Adam[36]
批大小(Batch Size)	32
最大文本长度(Max Length)	32

除上述通用参数外，TextCNN 模型中 3 个卷积核的大小分别为 3，4，5，输出通道数设置为 300，每个卷积层后面均连接一个丢弃层，丢弃率(Dropout Rate)为 0.3；TextRNN 模型中，RNN 的隐藏状态的维度为 300；Transformer 的隐层维度设置为 300，编码器个数设置为 3，Head 的数值设置为 4。

(3)数据集设置

实验所用的数据大小为 2 680 条，为保证结果的有效性，使用 K 折交叉验证(K Fold Validation)的方式对模型的性能进行评估，将数据整体划分为 K 份，进行 K 次验证。实验中，每次取 $K-1$ 份作为训练集，剩余 1 份作为测试集，最终采用每一折验证结果的平均值作为最终的结果，这里 K 值设置为 5。

7.2　评价指标

本文使用分类任务的指标进行评价，包括查准率(Precision)、查全率(Recall)、F1-value。事实上，在分类任务上还可以使用准确率(Accuracy)作为评价指标，由于数据不均衡，我们未采用准确率作为评价指标。

Precision 衡量的是模型对数据进行预测时预测的结果是否精准，即正确预测某类数据个数与预测为某类数据个数的比值，其公式如下：

$$Precision_C=\frac{P_C\cap R_C}{R_C} \tag{22}$$

其中，C 为具体的类别，P_C 为预测为 C 类的数据，R_C 为真实为 C 类数据。

Recall 衡量的是模型在进行预测时，能否将某类的数据全部预测出，即正确预测某类数据的个数与真实属于某类数据个数的比值，其公式如下：

$$Recall_C=\frac{P_C\cap R_C}{R_C} \tag{23}$$

从计算公式上看，Precision 和 Recall 可以看作一组相反的指标，查准率高，说明模型更加谨慎，而谨慎导致的结果就是预测的结果并不完全；查全率高，说明模型更加大胆，但同样可能导致预测的结果不准确。F1-value 则是对 Precision 和 Recall 的权衡，其综合考虑了 Precision 和 Recall 的结果，计算公式为：

$$F1_C=\frac{2*Precision_C*Recall_C}{Precision_C+Recall_C} \tag{24}$$

7.3　结果分析及讨论

首先，本文在完整的数据集上分别使用了深度学习模型和统计学习模型进行训练验证，表 6 给出了各模型在完整数据集上的表现。其中，Precision，Recall，F1 均是对数据 5 折交叉验证取平均的结果，最后一列 ALL 中的数值为每一行数值的平均值，代表模型的整体性能，其他列为不同类别上的结果。整体来看，深度学习方法较统计学习方法有较大优势，TextCNN 和 TextRNN 的 F1 值达到了 0.59，Transformer 的 F1 值达到 0.56，而统计机器学习方法中效果最好的随机森林的 F1 值仅达到了 0.44，朴素贝叶斯方法最差，其 F1 值仅有 0.22。深入到具体类别上，可以看到深度学习方法相比统计学习方法，优势更多地体现在样本类别少的数据上，其中最明显的是待添加类别，该类别下样本只有全部数据的 4.52%。在待添加类别上，3 种深度学习模型的 F1 值分别为 0.38，0.36 和 0.26，而作为统计学习方法中整体性能最优的决策树方法，其待添加类别的 F1 值仅有 0.13，朴素贝叶斯和支持向量在这一类别上的 F1 值甚至为 0。

表 6 深度学习模型和统计学习模型的实验结果

Table 6 Experiment results of deep learning models and statistical learning models

模型		类别						
		其他	待添加	待改进	性能	可用性	可靠性	ALL
朴素贝叶斯	Precision	0.55	0.0	0.1	0.90	0.86	0.83	0.54
	Recall	0.99	0.0	0.01	0.08	0.10	0.17	0.23
	F1	0.71	0.0	0.02	0.15	0.18	0.28	0.22
决策树	Precision	0.72	0.15	0.35	0.51	0.43	0.46	0.44
	Recall	0.77	0.12	0.25	0.47	0.43	0.48	0.42
	F1	0.74	0.13	0.29	0.48	0.42	0.47	0.42
支持向量机	Precision	0.62	0.00	0.31	0.88	0.78	0.72	0.55
	Recall	0.95	0.00	0.15	0.38	0.34	0.39	0.37
	F1	0.75	0.00	0.18	0.52	0.46	0.50	0.40
随机森林	Precision	0.65	0.24	0.57	0.78	0.67	0.58	0.58
	Recall	0.91	0.03	0.22	0.47	0.39	0.47	0.41
	F1	0.76	0.06	0.28	0.58	0.48	0.52	0.44
TextCNN	Precision	0.80	0.45	0.50	0.71	0.56	0.62	0.61
	Recall	0.81	0.35	0.44	0.69	0.53	0.68	0.58
	F1	0.80	0.38	0.46	0.69	0.53	0.64	0.59
TextRNN	Precision	0.79	0.41	0.53	0.71	0.59	0.68	0.62
	Recall	0.86	0.34	0.41	0.69	0.53	0.63	0.58
	F1	0.82	0.36	0.46	0.70	0.56	0.65	0.59
Transformer	Precision	0.79	0.29	0.45	0.76	0.62	0.63	0.59
	Recall	0.85	0.27	0.40	0.74	0.47	0.63	0.56
	F1	0.82	0.26	0.41	0.75	0.52	0.62	0.56

总之，无论是在整体性能还是在各个小的类别上，基于深度学习的方法在当前任务上都显著优于统计学习方法，性能上有较大提升。虽然在本文研究中，基于深度学习的方法显著优于统计学习方法，但仍然有一些方面需要讨论：1）我们的目标之一是探究深度学习模型在需求发现任务上的表现，对于如何得到最优异的性能尚有待研究。2）在统计模型设置上，使用 TF-IDF 作为模型的特征输入并不能完全发挥出统计模型的优势。在下一步工作中，我们将考虑加入更好的特征作为模型的输入，以对模型进行对比。3）本文未将采用启发式规则的挖掘方法列入对比，主要原因是简单的启发式规则目前尚无法在包含需求和非需求的混合自然语言文本中准确区分识别需求语句[37]。而本文所面向的用户评论数据相较于规格化程度较高的需求文档中的需求语句更缺乏严谨性，从中发现高质量的启发式规则更为困难。虽然如此，启发式方法作为一种更易在实际中应用的数据筛选挖掘手段，在本文目标任务的达成方面也可能存在特定的有效的启发式规则，因此未来工作也考虑将基于启发式规则的挖掘方法加入模型的对比实验中，以期找到更好的需求发现途径。

7.4 实验结果影响因素分析

（1）其他类别数据的影响

在前期采集的数据集中，其他类别的数据在整体数据中的占比超过 50%，如此大的比例会导致数据严重不均衡，同时也会对其他类别的识别效果产生影响。在实验结果中可以看到，深度学习方法对其他类别的数据识别效果是最好的，而对占比较小的数据识别效果较差。为了探究当前任务中其他类型数据对深度学习模型的影响程度，我们去除了其他类别数据，仅使用功能性需求类别数据和非功能性需求类别数据进行实验。去除其他类别数据后，各类型数据占比如图 5 所示，可以看到整体上各类别占比的差距缩小了，数据整体更为均衡。

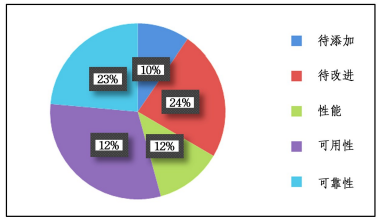

图 5 各类型数据占比

Fig. 5 Proportion of each type of data

模型部分，各参数设置保持不变，实验结果如图 6 所示，其中 * 表示去除其他类别数据后的实验结果。整体来看，在去除了其他类别的数据后，TextCNN，TextRNN 和 Transformer 模型均有不同程度的提升，Transformer 模型提升最大，整体的 F1 值从 0.56 提升至 0.61，提升了 0.05；TextCNN 则从 0.59 提升至 0.62，提升了 0.03；TextRNN 相对提升最小，提升了 0.02。其中，就待添加这一类别而言，Transformer 同样提升最大，从 0.26 提升至了 0.43，超越了 TextCNN 和 TextRNN 的成绩，而未去除其他类型的数据之前，Transformer 模型在这一类别上的结果相较 TextCNN 和 TextRNN 相差了 0.1 以上，说明 Transformer 模型对数据不平衡十分敏感。

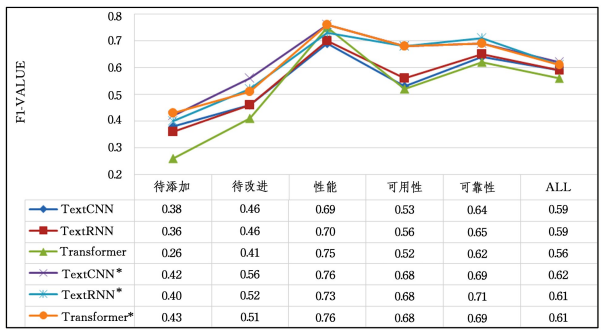

图 6 其他类别数据对深度学习模型结果的影响

Fig. 6 Influence of other types of data on results of deep learning models

实验表明，在本文任务中，其他类型数据的存在对深度学习方法有较大影响，去除该类别后模型性能会有较为明显的提升。

(2)嵌入层预训练对结果的影响

在本文采用的 3 个深度学习模型中，文本都要首先经过嵌入层进行编码，再采用不同的编码器编码，最后进行预测。嵌入层的编码信息随着训练的进行逐渐学习。事实上，在嵌入层编码阶段，可以采用预训练的方式对其进行初始化。本文使用 word2vec[37] 词向量训练方法进行初始词向量的训练，嵌入层利用训练好的结果进行初始化。为了保证词向量的效果，我们额外爬取了 1GB 的中文百科语料作为训练语料。测试数据同样为去掉其他类别的数据集。关于词嵌入对深度学习模型影响的实验结果如图 7 所示。其中，R/T 表示词嵌入层随机初始化，且随模型训练进行学习；W/N 表示词嵌入层使用 Word2vec 预训练的词向量进行初始化，但在模型训练中不再进行学习优化；W/T 表示词嵌入层使用 Word2vec 预训练的词向量进行初始化，且随模型训练进行学习。

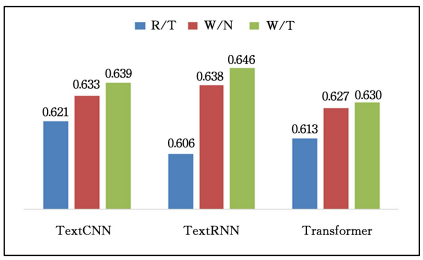

图 7　预训练词嵌入对深度学习模型的影响

Fig. 7　Influence of pre-trained word embedding on deep learning models

从图 7 可以看到，使用预训练词向量信息作为词嵌入层初始化参数的模型，明显要优于未预训练的版本。其中 TextRNN 从中获得的收益最大，在引入预训练词向量后，其在全部数据上的 F1 值从最低变成了最高，提升了将近 0.04，而 TextCNN 和 Transformer 分别提升了 0.018 和 0.017。此外，通过实验对比发现，预训练词向量如果跟随模型训练一起进行学习优化，会使得最终结果优于词向量冻结的模型，这在 3 个模型上均有体现。

7.5　实验总结

本文通过实验详细对比了 3 种常见的深度学习文本分类模型在本文任务上的表现。

从性能即评价指标的角度来看，深度学习模型显著优于统计学习模型，尤其在样本较少的类别上，深度学习方法更具有优势，深度学习模型成为了本文任务的首选。

此外，本文针对数据中样本不均衡的问题进行了实验，在去除其他类别数据的基础上对比了 3 种深度学习模型在本任务上的表现。总的来说，数据不均衡均对 3 种深度学习模型都会产生影响，只是影响程度不同，其对 Transformer 模型的影响程度最大，对 TextRNN 模型的影响程度最小。

最后，从预训练的角度对深度学习模型的嵌入层初始化以及是否训练进行了研究。实验证明，预训练的词嵌入会对深度学习模型产生较为明显的影响，引入预训练词向量为嵌入层初始化，是一种提升模型性能的非常有效的策略。

结束语　从 APP 用户反馈数据中挖掘用户需求是 APP 迭代更新和需求获取的一种重要方式，用户在 APP 应用市场中发表对 APP 不同维度的评价，其中蕴含着用户对 APP 软件的大量潜在需求。本文旨在从 APP 用户评论数据中有效地将潜在需求挖掘出来。本文的主要贡献包括：

1)将基于 APP 评论数据可能蕴含的用户需求划分为功能待添加、功能待改进、性能、可用性、可靠性 5 个子类别。

2)构建了 APP 用户需求评论挖掘数据集，主要包括 APP 评论基础资源库与 APP 需求评论分类数据集。

3)选择 TextCNN，TextRNN，Transformer 3 种典型的深度学习方法从整体性能、数据类别与预训练 3 个方面对 APP 用户评论数据中的需求发现与挖掘进行探讨，同时对比相关研究工作中的传统统计机器学习方法 SVM，NB，DT 和 RF。

在下一步工作中，考虑增加情感权重因素，评估用户在评论时的情感倾向及其强烈程度，从而判定用户更倾向于表达什么需求。有效的情感分析虽然难度较大，但确实能够帮助我们准确地判别用户评论中对某种软件的好恶以及渴望程度，进而能够对所挖掘需求的优先级进行判定。

参 考 文 献

[1] SARRO F，HARMNA M，JIA Y，et al. Customer rating reactions can be predicted purely using app features[C]// Proc of the 26th Requirements Engineering Conference. IEEE，2018.

[2] SHI L，CHEN C，WANG Q，et al. Understanding feature requests by leveraging fuzzy method and linguistic[C]// Proc of the 32th IEEE/ACM International Conference on Automated SoftwareEngineering (ASE). 2017：440-450.

[3] PALOMBA F，SALZA P，CIURUMELEA A，et al. Recommending and localizing change requests for mobile apps based on user reviews[C]// Proc of the 39th International Conference on Software Engineering. USA：IEEE，2017：106-117.

[4] SORBO A DI，PANICHELLA S，ALEXANDRU C V，et al. What would users change in my app? summarizing app reviews for recommending software changes[C]// Proc of the 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering. USA，ACM，2016：499-510.

[5] JIANG W，ZHANG L，DAI Y，et al. Analyzing Helpfulness of Online Reviews for User Requirements Elicitation[J]. Chinese Journal of Computers，2013，36(1)：119-131.

[6] SCHNEIDER K. Focusing spontaneous feedback to support system evolution[C]// Proc of the 11th Requirements Engineering Conference. IEEE，2011：165-174.

[7] IACOB C，HARRISON R. Retrieving and analyzing mobile apps feature requests from online reviews[C]// Proc of the 10th Working Conference on Mining Software Repositories (MSR). San Francisco，2013：41-44.

[8] CHEN N，LIN J，HOI S C H，et al. AR-miner：mining informative reviews for developers from mobile app marketplace[C]// International Conference on Software Engineering. ACM，2014.

[9] KHAN J A，XIE Y，LIU L，et al. Analysis of Requirements-Related Arguments in User Forums[C]// Proc of the 27th IEEE International Requirements Engineering Conference (RE). Jeju Island，Korea (South)，2019：63-74.

[10] KHAN J A，LIU L，JIA Y，et al. Linguistic Analysis of Crowd Requirements：An experimental study[C]// Proc of the RE

Workshop. Empri, 2018.
[11] MAALEJ W, NAYEBI M, JOHANN T, et al. Toward data-driven requirements engineering[J]. IEEE Software, 2016, 33(1): 48-54.
[12] HOUMB S H, ISLAM S, KNAUSS E, et al. Eliciting security requirements and tracing them to design: an integration of Common Criteria, heuristics, and UMLsec[J]. Requirements Engineering, 2010, 15(1): 63-93.
[13] MAALEJ W, NABIL H. Bug report, feature request, or simply praise? On automatically classifying app reviews[C]// Proc of the 23rd IEEE International Requirements Engineering Conference (RE). Ottawa, ON, 2015: 116-125.
[14] PANICHELLA S, SOEBO A D, GUZMAN E, et al. How Can I Improve My App? Classifying User Reviews for Software Maintenance and Evolution[C]// International Conference on Software Maintenance & Evolution. IEEE, 2015.
[15] VILLARROEL L, BAVOTA G, RUSSO B, et al. Release Planning of Mobile Apps Based on User Reviews[C]// Proc of the 38th IEEE/ACM International Conference on Software Engineering (ICSE). Austin, TX, 2016: 14-24.
[16] PANICHELLA S, SORBO DI A, GUZMAN E, et al. ARdoc: app reviews development oriented classifier[C]// Acm Sigsoft International Symposium on Foundations of Software Engineering. ACM, 2016: 1023-1027.
[17] SUPRAYOGI E, BUDI I, MAHENDRA R. Information Extraction for Mobile Application User Review[C]// Proc of International Conference on Advanced Computer Science and Information Systems (ICACSIS). Yogyakarta, 2018: 343-348.
[18] BUCHAN J, BANO M, ZOWGHI D, et al. Semi-Automated Extraction of New Requirements from Online Reviews for Software Product Evolution[C]// Proc of the 25th Australasian Software Engineering Conference (ASWEC). Adelaide, SA, 2018: 31-40.
[19] CHEN Q, ZHANG L, JIANG J, et al. Review Analysis Method Based on Support Vector Machine and Latent Dirichlet Allocation[J]. Journal of Software, 2019, 30(5): 349-362.
[20] HU T Y, JIANG Y. Mining of User's Comments Reflecting Usage Feedback for APP Software[J]. Journal of Software, 2019 (10): 3168-3185.
[21] ZHANG H F. Introduction to Software Engineering[M]. Beijing: Tsinghua University Press.
[22] CLELAND-HUANG J, SETTIMI R, ZOU X, et al. The Detection and Classification of Non-Functional Requirements with Application to Early Aspects[C]// Proc of the14th IEEE International Requirements Engineering Conference (RE'06). Minneapolis/St: Paul, MN, 2006: 39-48.
[23] GLINZ M. On Non-Functional Requirements[C]// Proc of IEEE International Requirements Engineering Conference. IEEE, 2005.
[24] JIA Y D, LIU L. Recognition and Classification of Non-Functional Requirements in Chinese[J]. Journal of Software, 2019, 30(10): 3115-3126.
[25] DEVLIN J, CHANG M W, LEE K, et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding [C]// Proc of NAACL-HLT (1). 2019.
[26] KIM Y. Convolutional Neural Networksfor Sentence Classification[C]// Proc of Conferenceon Empirical Methods in Natural Language Processing (EMNLP). 2014
[27] LAI S, XU L, LIU K, et al. Recurrent convolutional neural networks for text classification[C]// Proc of the 29th AAAI conference on artificial intelligence. 2015.
[28] HOCHREITER S, SCHMIDHUBER J. Long short-term memory[J]. Neural computation, 1997, 9(8): 1735-1780.
[29] CHO K, VAN MERRIENBOER B, GULCEHRE C, et al. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation[C]// Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2014: 1724-1734.
[30] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is All you Need[C]// Proc of Neural Information Processing Systems. 2017: 5998-6008.
[31] LEWIS D. Naive (Bayes) at Forty: The independence assumption in information retrieval[C]// Proc of European Conference on Machine Learning. Springer, Berlin, Heidelberg, 1998.
[32] QUINLAN J. C4: 5: programs for machine learning[M]. Elsevier, 2014.
[33] CORTES C, VAPNIK V. Support-vector networks[J]. Machine Learning, 1995, 20(3): 273-297.
[34] BREIMAN L. Random forests[J]. Machine Learning, 2001, 45(1): 5-32.
[35] Paszke A, GROSS S, MASSA F, et al. Pytorch: An imperative style, high-performance deep learning library[C]// Proc of Advances in Neural Information Processing Systems. 2019: 8026-8037.
[36] KINGMA D P, BA J. Adam: A method for stochastic optimization[C]// Proc of the 3rd International Conference on Learning Representations. 2015.
[37] ABUALHAIJA S, ARORA C, SABETZADEH M, et al. A Machine Learning-Based Approach for Demarcating Requirements in Textual Specifications[C]// Proc of 27th International Requirements Engineering Conference (RE). IEEE, 2019.
[38] MIKOLOV T, CHEN K, CORRADO G, et al. Efficient estimation of word representations in vector space [C]// Proc of the 3rd International Conference on Learning Representations. 2013.

WANG Ying, born in 1996, postgraduate. Her main research interests include requirement engineering and social networks.

ZHENG Li-wei, born in 1979, Ph.D, associate professor. His main research interests include requirement engineering, social networks and data quality enhancement.