基于系统模型的用户评论中非功能需求的自动分类

李雪莹,王田路,梁 鹏,王 翀
(武汉大学计算机学院,湖北武汉 430072)

摘 要: 移动应用程序中的用户评论是获取用户需求的重要来源.从用户评论中获取的用户需求,不仅可以帮助开发人员维护现有系统,还可以快速、准确地定位新的用户需求.本文主要关注移动应用用户评论中的非功能需求,并基于系统模型、采用机器学习和深度学习算法将其自动分类为行为型需求和表示型需求.在使用机器学习方法分类时,将2种特征提取技术与5种机器学习算法进行组合.在使用深度学习方法分类时,使用了2种基于词嵌入的深度学习算法和1种基于字符嵌入的深度学习算法.从性能和时间消耗2个维度比较了机器学习模型和深度学习模型,结果表明,机器学习模型比深度学习模型表现更好.此外,支持向量机(Support Vector Machine,SVM)与词频-逆文档频率(Term Frequency-Inverse Document Frequency,TF-IDF)组合获得了最好的分类性能,精确率为0.941,召回率为0.990,F1-score为0.965.

关键词: 用户评论;系统模型;非功能需求;自动分类;机器学习;深度学习
中图分类号: TP311.5 **文献标识码:** A **文章编号:** 0372-2112(2022)09-2079-11
电子学报URL:http://www.ejournal.org.cn **DOI:**10.12263/DZXB.20210454

Automatic Classification of Non-Functional Requirements in App User Reviews Based on System Model

LI Xue-ying, WANG Tian-lu, LIANG Peng, WANG Chong
(*School of Computer Science, Wuhan University, Wuhan, Hubei* 430072, *China*)

Abstract: App user reviews are an important source of user requirements. The requirements obtained from user reviews can not only help developers maintain the existing systems, but also quickly and accurately locate new user requirements. This work focuses on non-functional requirements in App user reviews, and further classifies them into behavioral requirements and representational requirements based on system model with machine learning and deep learning algorithms. When using machine learning to classify non-functional requirements, we combined two feature extraction techniques with five machine learning algorithms. When applying deep learning to classify non-functional requirements, we used two deep learning algorithms based on word embedding and one deep learning algorithm based on character embedding. We compared machine learning models and deep learning models from the performance and time consumption perspectives. The results show that, machine learning models perform better than the deep learning models. In addition, the combination of SVM (Support Vector Machine) and TF-IDF(Term Frequency-Inverse Document Frequency) achieves the best performance of classification, with a precision of 0.941, a recall of 0.990, and an F1-score of 0.965.

Key words: user review; system model; non-functional requirement; automatic classification; machine learning; deep learning

1 引言

随着移动设备的迅速普及,移动应用程序和应用商店在人们日常生活中的应用越来越广泛.移动应用库中包含了大量的用户评论,这些用户评论包含用户需求、系统故障以及使用体验等信息,被认为是获取需求的重要来源.抽取、识别和分类存在于用户评论中的需求,不仅可以帮助开发人员维护现有系统,还能快速、准确地定位新的用户需求,从而添加现有系统缺乏的功能.然而,移动应用商店每天都有海量的用户评论产生,人工处理大量自然语言文本需要耗费极大的人

收稿日期:2021-04-09;修回日期:2021-12-20;责任编辑:梅志强
基金项目:国家重点研发计划(No.2018YFB1402800);国家自然科学基金(No.62172311)

力和时间成本,从而使得从用户评论中快速有效识别和分类需求成为挑战问题.

目前从用户评论中获取需求的相关研究主要关注软件系统功能方面的需求. 然而,最近的需求工程领域工业调研(Naming the Pain in Requirements Engineerin, NaPiRE)结果表明,“不清楚/无法度量”的非功能需求是涉众在开发过程中最棘手的问题之一[1]. 用户评论中的非功能需求与软件质量密切相关,在软件系统的开发和维护过程中起到关键作用. 将非功能需求进行分类,可以帮助开发者更好地理解系统的非功能需求并发现系统中存在的主要质量问题.

Broy[2,3]提出根据结构化的系统模型对需求进行分类. 该方法根据是否描述了系统的行为属性,将需求分为行为型需求和表示型需求. Eckhardt[4]认为基于Broy提出的系统模型能够对需求进行有效的分类. 该系统模型提供了明确清晰的系统概念,可以根据系统属性精准地指定需求类别,使需求的表示更加具体和准确.

基于上述研究背景,本文的主要关注点是:基于系统模型将移动应用用户评论中的非功能需求自动分类为行为型需求和表示型需求. Lu等人[5]基于ISO/IEC 25010 (International Organization for Standardization/International Electrotechnical Commission)软件质量需求标准[6],已对4000条iBooks和WhatsApp用户评论进行了分类,最终得到了1278条非功能需求用户评论. 本文以上述1278条非功能需求用户评论为数据集,从系统模型的角度出发,通过人工标注的方式将非功能需求分类为行为型需求和表示型需求. 之后,分别使用了机器学习算法和深度学习算法对人工标记得到的数据集进行自动分类. 最后,评估和比较了机器学习模型和深度学习模型在自动分类非功能需求时的性能和时间消耗. 本文的主要贡献包括3个方面:(1)提供了基于系统模型将非功能需求标记为“行为型需求”和“表示型需求”的实验数据集[7],为这类研究提供了公共数据;(2)将机器学习算法和深度学习算法应用于基于系统模型的非功能需求自动分类,分别得到了性能最优的机器学习模型组合(SVM与TF-IDF组合)和深度学习模型组合(TextCNN(Text Convolutional Neural Networks)与Word2Vec组合);(3)评估和比较了机器学习和深度学习在将非功能需求自动分类时的性能和时间消耗差异,为非功能需求的自动分类提供了最佳方法和使用建议.

2 相关工作

2.1 需求分类的方式

软件需求分类是需求工程领域的重要任务之一. 最常见的分类方法是将需求分为功能需求和非功能需求. 尽管很多研究采纳了这种分类方式,但是因为“非功能需求”的定义较为模糊,对于什么是“非功能需求”以及如何获取、记录和验证它们,需求工程界尚未达成共识[8]. Glinz[8]指出,将需求分类为功能需求和非功能需求,将会导致定义问题、分类问题和表示问题. 为了解决上述问题,作者提出应该基于“关注点”将需求分为功能需求、性能需求、特定质量需求和约束. 此外,很多研究关注非功能需求的分类. 最常用的非功能需求分类是ISO/IEC 25010[6]标准中的质量模型定义的8大类质量特性类别. 然而该标准中对非功能需求进行分类存在以下问题:(1)非功能需求通常没有被量化,不易测试;(2)在产品规划过程中没有考虑非功能需求;(3)项目中主要关注功能需求,非功能需求通常不被记录;(4)该分类过于抽象,难以为开发者的分类需求提供指导[4]. 尽管存在多种非功能需求的分类方式,但非功能需求没有被集成到软件开发过程中,且缺乏一种普遍接受的方式来提取、记录和分析非功能需求[4].

2.2 需求的自动分类方法

还有一些研究使用自动化方法对需求进行分类. Abad等人[9]使用决策树(Decision Tree,DT)将需求分为功能需求和非功能需求. 还使用LDA(Latent Dirichlet Allocation)、K-means、朴素贝叶斯(Naïve Bayes,NB)等方法将非功能需求进一步分类为可用性、可维护性、性能. 结果表明,NB分类器在对非功能需求进行分类时表现最好. Li等人[10]使用k-Nearest Neighbor、NB、SVM将需求分类为安全性、可靠性、性能、系统接口等子类别. 结果表明,SVM在对需求进行分类时表现最好.

2.3 用户评论分类

目前有很多研究使用自动化方法对用户评论进行分类. Stanik等人[11]分别使用机器学习和深度学习方法将用户评论分类为问题报告、询问评论和无关评论. 结果表明,机器学习模型与深度学习模型在分类用户评论时性能相当. Lu等人[5]将用户评论分为非功能需求(可靠性、可用性、可移植性和性能)、功能需求以及其他. 作者将BoW(Bag-of-Words)、TF-IDF(Term Frequency-Inverse Document Frequency)、CHI2(Chi Squared)和AUR-BoW(Augmented User Reviews - Bag-of-Words)与NB、J48和Bagging结合对用户评论进行分类. 其中,AUR-BoW针对训练集中的用户评论,利用与评论相似的词语对用户评论进行扩展,将扩展后的句子作为BoW的输入. 研究结果表明,AUR-BoW与Bagging结合起来的分类效果最佳. Jha等人[12]从用户评论中提取非功能需求并将其分为可靠性、可用性、性能和可支持性,使用NB和SVM对非功能需求进行分类. 结果表明,SVM的性能优于NB分类器的性能.

在本文中,我们从系统模型的视角对非功能需求

进行分类,并在第3节对使用的技术进行了详细的介绍.

3　相关理论和技术

3.1　基于系统模型的非功能需求分类

Broy[2,3]提出了一种基于结构化的系统模型对需求进行分类的方法. 在系统模型中,系统由接口(Interface)、体系结构(Architecture)和状态(State)3个基本视图构成,并将接口行为作为主要概念. 系统的行为属性包括系统接口上的行为以及接口行为在体系结构和状态转换方面的内部行为(例如点击系统交互界面的响应时间),系统的非行为属性则关注系统在语法构成上和技术层面上的表示、描述、构造、实现和执行方式(例如系统实现要求使用的编程语言). 基于系统模型的需求分类方法根据需求是否描述了系统的行为属性,将其分为"行为型需求"和"表示型需求". 其中"行为型需求"不仅包含了传统意义上的功能需求,还包含了描述系统行为的质量需求;"表示型需求"包含从语法构成和技术层面关注系统表示、描述、构造、实现和执行方式的质量需求. 图1给出了基于系统模型对非功能需求用户评论进行分类的具体方法.

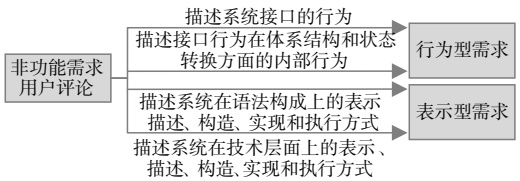

图1　基于系统模型的非功能需求分类

基于系统模型将需求分类,为需求分类提供了新的视角,打破了实践中功能需求与非功能需求的分离[4]. 从实践角度来看,这种分类方法使得非功能需求可以像功能需求一样被提取、分析和记录. Eckhardt[4]基于Broy提出的分类模型对非功能需求进行了分类. 结果表明,大多数所谓的"非功能需求"实际上像功能需求一样都描述了系统的行为,因此被定义为"非功能需求"并不合适. 王田路等人发现[13],在用户评论的非功能需求中超过70%的非功能需求描述了系统的行为,因此它们在本质上不能被统一认为是"非功能"的.

为了解决传统需求分类存在的问题,本文基于Broy提出的系统模型对需求进行分类. 本文首先基于传统的需求分类模型从用户评论中提取非功能需求评论. 之后基于系统模型的需求分类方法,根据用户评论是否描述了系统的行为属性,将其分类为"行为型需求"和"表示型需求",来帮助我们进一步理解非功能需求的本质. 需要说明的是,本文没有关注用户评论中的功能需求,原因是在基于系统模型的需求分类方式中,认为功能需求属于"行为型需求". 表1提供了用户评论句子中"行为型需求"和"表示型需求"的示例(更多分类实例,可参见本文的实验数据集[7]). 在王田路等人前期人工标注和分析工作的基础上[13],本文结合机器学习和深度学习领域的分类算法实现了基于系统模型对用户评论中非功能需求的自动分类. 该自动分类方法可以辅助需求工程师和系统涉众了解和分析用户评论中非功能需求的本质以及非功能需求所描述的系统属性.

表1　用户评论中行为型需求和表示型需求的示例

需求类型	描述	示例
行为型需求	Requirements that describe behavioral properties of the system, including the behavioral of interface, architecture, and state.	App is not responding while sending the pictures.
表示型需求	Requirements that describe the representation, description, construction, implementation, and execution of the system (i.e., the way that a system is syntactically and technically represented).	If there is another version released called ibooks classic with the wooden bookshelf UI, we will be thankful.

3.2　文本特征提取技术

BoW被广泛应用在文本分类任务中. 该模型根据语料库所有文档中的单词构造一个无序字典,将字词在文档中出现的频率作为文档特征. TF-IDF用来评估某一字词对于某个文档集合或语料库中的一个文档的重要程度. 如果某字词在一个文档中频率出现,在其他文档中很少出现,则认为此字词具有很好的类别区分能力,适合用来分类.

3.3　监督式机器学习

监督式机器学习算法接受已知的输入数据集合(训练集)和已知的对数据的响应(输出),然后训练一个模型,为新输入数据的响应生成合理的预测. Jha等人使用机器学习算法对需求进行自动分类时[12],结果表明SVM的性能优于NB. Abad[9]等人在使用LDA、K-means、NB等方法对非功能需求分类时,结果表明NB的表现最好. 为了确定表现最好的分类器,我们选择和比较了文本分类领域具有代表性的监督式机器学习算法NB、LR、DT、RF和SVM. 通过学习已标记的训练集分别训练5种分类器模型,再将训练好的模型应用到测试集中,以实现对非功能需求的自动分类.

3.4　词嵌入技术

Word2Vec[14]是词嵌入(Word Embedding)的方式之一. 它使用无监督方法,通过学习和训练文本,可以把对文本内容的处理简化为向量运算,使用词向量的方式表征词语的语义信息. 使用Word2Vec工具训练得到

的词向量可以有效度量词语与词语之间的相似性. FastText[15]用于从大规模语料库中学习单词的高维向量表示. Word2Vec 没有考虑单词的内部结构,直接学习整个单词的词向量. 而 FastText 首先计算子串的词向量,最后将单词所有子串的词向量组合得到单词的词向量. 因此 FastText 能够识别未出现在语料库中的单词.

3.5 神经网络模型

通过多层神经网络学习数据的特征,深度学习将数据转换为有利于分类任务的更高效的数字表示形式. 卷积神经网络(Conventional Neural Network,CNN)是深度学习领域代表性的算法模型,CNN 由输入层、卷积层、激活函数、池化层和全连接层组成,通过反向传播算法进行参数优化. TextCNN 模型[16]是代表性的 CNN 网络结构,与传统的 CNN 相比,TextCNN 在结构上无明显变化. 其流程是:先将文本分词,通过词嵌入(Word Embedding)得到词向量,将词向量进行卷积、池化操作,最后将输出外接到 Softmax 层做 n 分类. RCNN 模型[17]将单词的上下文信息和单词本身的向量进行整合以得到该单词的词向量. 其流程是:首先通过词嵌入得到词向量. 然后使用一个具有循环结构的隐藏层捕获单词的上下文信息,重新计算单词向量. 再对词向量进行池化操作,以捕获文本的关键单词. 最后将输出外接到 Softmax 层做 n 分类. CharCNN [18]模型将文本作为一种原始字符级信号来处理. 其流程是:首先将一系列编码字符作为输入,使用 one-hot 编码对字符进行向量化操作. 将文本用字符向量表示,再用字符向量得到文档向量. 之后对文档向量进行卷积、池化操作后,将输出外接到全连接层做 n 分类. 本文分别使用 Word2Vec 和 FastText 模型将单词转化为词向量,作为 TextCNN 和 RCNN 模型的输入. CharCNN 本身基于字符计算向量,因此无需以 Word2Vec 和 FastText 词向量作为输入.

4 研究设计

4.1 研究目标与研究问题

本文的研究目标是:基于系统模型,将移动应用用户评论中的非功能需求自动分类为行为型需求和表示型需求. 为了实现该目标,我们使用了机器学习和深度学习算法,并提出了以下 5 个研究问题(Research Question,RQ).

RQ1:机器学习模型中,在将用户评论中的非功能需求自动分类为行为型需求和表示型需求时,哪种特征提取技术(TF-IDF、BoW)表现更好?

在自动分类任务中,文本通常被表示为数字向量. 不同的特征提取技术关注的重点和使用的算法通常不同,因此可能导致不同的分类效果. 本文使用 BoW 和 TF-IDF 来提取文本特征. 此 RQ 的目的是找到适合本文自动分类任务的特征提取技术.

RQ2:在将用户评论中的非功能需求自动分类为行为型需求和表示型需求时,哪种机器学习方法(NB、LR、DT、RF、SVM)性能最好?

不同的分类方法在相同的分类任务中,可能导致不同的分类性能. 不同的机器学习算法适用于不同的分类场景. 本文使用 5 种常用的分类器进行实验. 此 RQ 的目的是找到在本文的自动分类任务中,获得最佳性能的分类器.

RQ3:深度学习模型中,在将用户评论中的非功能需求自动分类为行为型需求和表示型需求时,哪种词嵌入模型(Word2Vec、FastText)表现更好?

深度学习模型的输入为词嵌入或字符嵌入. 不同词嵌入技术在计算词向量时依据的算法是有差异的. Word2Vec 和 FastText 被广泛应用于自然语言处理任务中. 此 RQ 的目的是找到适合本文自动分类任务的词向量技术.

RQ4:在将用户评论中的非功能需求自动分类为行为型需求和表示型需求时,哪种深度学习模型(TextCNN、RCNN、CharCNN)表现最好?

不同的深度学习模型结构有所差异,可能导致不同的分类性能. TextCNN 和 RCNN 模型基于单词表示文本,CharCNN 模型利用字符表示文本. 本文使用 3 种流行的深度学习算法进行实验. 此 RQ 的目的是,找到在本文的自动分类任务中,获得最佳性能的深度学习模型.

RQ5:在将用户评论中的非功能需求自动分类为行为型需求和表示型需求时,从性能和时间消耗 2 个维度比较机器学习模型与深度学习模型,哪种模型表现更好?

深度学习模型已被应用于自然语言处理领域,在文本分类任务中都表现出了非常好的性能. 本文使用 TextCNN 和 RCNN 模型以及 CharCNN 模型自动将非功能需求进行分类. 此 RQ 的目的是,从性能和时间消耗 2 个维度综合评估和比较机器学习模型和深度学习模型在自动分类非功能需求时的表现差异.

研究问题(RQs)间的关联:RQ1 和 RQ2 关注机器学习算法中,不同阶段使用不同技术导致的分类结果的性能差异. RQ1 比较不同的特征提取技术的性能差异. RQ2 则重在关注不同分类器的性能差异. RQ3 和 RQ4 关注深度学习算法中,不同阶段使用不同技术所导致的分类结果的性能差异. RQ3 比较不同词向量模型初始化词嵌入层时的性能差异. RQ4 则关注不同深度学习模型本身的性能差异. RQ5 从性能和时间消耗 2 个维度综合比较了机器学习模型和深度学习模型在

分类非功能需求时的性能差异.

4.2 数据收集

本文所用的数据来源于App Store和Google Play中阅读类和通讯类的2个移动应用程序iBooks和WhatsApp. 选择这2个移动应用程序的原因有以下2点:(1)这2个移动应用拥有大量的活跃用户,且其中的用户评论规模非常庞大,可以为本研究提供丰富的数据;(2)这2个移动应用主要用于在线阅读和信息交流,没有涉及比较专业的领域,涵盖的用户类型较为广泛,因此在对其用户评论标记时不需要研究人员具备某些专业领域的知识,从而使标记得到的分类结果更加可靠.

我们在之前的工作[6]中收集了iBooks和WhatsApp中的用户评论,将其分割为单个句子. 随后从每个移动应用中随机选取2000条评论,基于ISO/IEC 25010软件质量需求标准将其人工标注为功能需求和非功能需求. 最终,识别了1278条"非功能需求"用户评论,这1278条"非功能需求"用户评论将作为本文的数据集.

本文将1278条非功能需求进行人工标记. 具体分为以下2个步骤:(1)预标记. 从1278条评论中随机抽取50条作为预实验数据,由3位作者分别标记这些数据,将50条评论分类为行为型需求和表示型需求,最后3位作者一起讨论和解决预标记过程中产生的分歧. 该过程的目的是使3位作者对基于系统模型进行需求分类的理解达成共识;(2)正式标记. 在对本文分类任务达成一致理解后,由第1、2作者分别独立标记其余的1228条非功能需求用户评论. 使用Cohen's Kappa系数[19]计算2位作者分类结果的一致性,该值约为0.86,表明2人在对用户评论的非功能需求进行分类标记时的结果比较一致. 最后,由第3作者浏览前2位作者的标记结果,3位作者一起讨论和解决标记结果中存在的分歧,就数据集的标记结果达成一致.

最终,我们在1278条非功能需求中识别了899条行为型需求和379条表示型需求. 我们将数据集划分为训练集(1022条,占80%)、验证集(128条,占10%)和测试集(128条,占10%). 本文的数据集已在线提供[7].

4.3 实验步骤

本文使用机器学习和深度学习对非功能需求进行分类. 图2和图3分别展示了使用机器学习和深度学习算法进行非功能需求分类的过程及每个步骤所使用的技术. 在深度学习算法中,由于Word2Vec和FastText在训练词向量时,不需要有标签的数据集,因此我们将收集到的所有用户评论(来自iBooks的6696条原始用户评论和来自WhatsApp的4400条原始用户评论)作为语料库训练Word2Vec和FastText词向量. 在训练字符向量时,我们基于文献[18]提供的69个字符来计算one-hot编码,以此得到字符向量. 本文根据经验选取常用的数值以确定TextCNN、RCNN和CharCNN模型的主要参数. 3个模型的主要参数分别如表2、表3和表4所示.

图2　使用机器学习算法进行非功能需求自动分类的过程

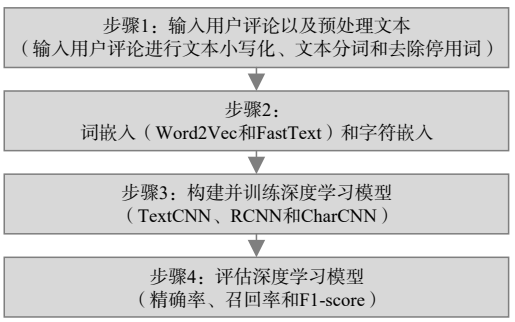

图3　使用深度学习算法进行非功能需求自动分类的过程

表2　TextCNN模型的主要超参数

Hyper-Parameter	Value
Dimension of word vector	200
Length of word sequence	200
Number of convolution kernel	128
Size of convolution kernel	[3~5]
Learning rate	0.001
Batch size	64
Epoch	10

表3　RCNN模型的主要超参数

Hyper-parameter	Value
Dimension of word vector	200
Length of word sequence	200
Size of hidden layer	128
Learning rate	0.001
Batch size	64
Epoch	10

表4 CharCNN模型的主要超参数

Hyper-Parameter	Value
Dimension of character vector	70
Length of character sequence	1014
Size of convolution and pooling layers	([256,7,3], [256,7,3], [256,3,None], [256,3,None], [256,3,None], [256,3,3])
Size of fully-connected	[1024,1024,2]
Learning rate	0.001
Batch size	64
Epoch	10

我们将行为型需求类别作为正类,表示型需求类别作为负类. 因此,精确率(Precision)指的是使用本文的分类模型得到的行为需求类别的用户评论中真正属于行为型需求类别的比例. 召回率(Recall)指的是使用本文的分类模型得到的真正属于行为型需求类别的非功能需求,占数据集中行为型需求类别的非功能需求的比例. F1-score是精确率和召回率的加权调和平均值,当F1-score值较高时,一般说明分类模型的分类效果较好. 式(1)~(3)分别给出了精确率、召回率和F1-score的计算公式.

其中,TP(True Positive)表示被分类模型标记为行为型需求的非功能需求中,实际属于行为型需求的评论数量. FP(False Positive)为被标记为行为型需求的非功能需求中,实际不属于行为型需求的评论数量. FN(False Negative)表示被分类模型标记为表示型需求的非功能需求中,实际不属于表示型需求的评论数量.

$$\text{Precision}=\frac{\text{TP}}{\text{TP}+\text{FP}} \tag{1}$$

$$\text{Recall}=\frac{\text{TP}}{\text{TP}+\text{FN}} \tag{2}$$

$$\text{F1-score}=\frac{2\times\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}} \tag{3}$$

4.4 实验环境

实验环境配置为Intel(R) Core(TM) i5-7200U CPU以及8 GB RAM的一台式计算机,运行Windows 10(64位)操作系统. 本文使用NLTK工具(版本号3.4.5)进行分词,并删除了长度小于等于3的单词. 在机器学习算法中使用scikit-learn工具(版本号0.22.1)提供的特征提取技术和分类器算法(默认参数)进行实验. 在深度学习算法中使用gensim库(版本号3.8.1)提供的词嵌入模型训练词向量,并使用Tensorflow框架(版本号1.14.0)构建和训练深度学习模型. 本文使用的NLTK工具、scikit-learn工具、gemsim库以及Tensorflow框架均基于Python语言.

5 结果与分析

为了回答3.1节中提出的研究问题,我们计算了由机器学习模型(如表5所示)和深度学习模型(如表6所示)处理测试集后得到的精确率、召回率和F1-score.

表5 使用机器学习模型对用户评论中的非功能需求进行分类的结果

	TF-IDF			BoW		
	Precision	Recall	F1-score	Precision	Recall	F1-score
NB	0.914	0.980	0.946	**0.989**	0.908	0.947
LR	0.931	0.969	0.950	0.941	0.979	0.960
DT	0.934	0.867	0.899	0.939	0.949	0.944
RF	0.922	0.969	0.945	0.922	0.969	0.945
SVM	0.941	**0.990**	**0.965**	0.941	0.980	0.960

表6 使用深度学习模型对用户评论中的非功能需求进行分类的结果

	Precision	Recall	F1-score
TextCNN + Word2Vec	0.950	0.969	**0.959**
TextCNN + FastText	**0.989**	0.898	0.941
RCNN + Word2Vec	0.931	0.959	0.945
RCNN + FastText	0.766	**1.000**	0.867
CharCNN	0.876	0.867	0.871

RQ1:我们计算和比较了5种机器学习算法分别与TF-IDF和BoW结合时得到的F1-score值,结果如图4所示. 结果显示,除了SVM分类器外,其他所有分类器在与BoW结合时得到的F1-score值,均比与TF-IDF结合得到的F1-score值高. SVM分类器与TF-IDF组合得到了最高的F1-score,但仅比与BoW组合得到的F1-score高0.05.

RQ1结论:整体上可以认为,在将用户评论中的非功能需求自动分类为表示型需求和行为型需求时,简单的BoW技术比TF-IDF技术表现更好. 这表明,在需求分类任务的特征提取阶段,研究人员应该优先选择表现更好的BoW技术. 但是仍然需要关注使用TF-IDF技术是否能够得到更好的性能.

RQ2:我们计算和比较了5种机器学习模型自动分

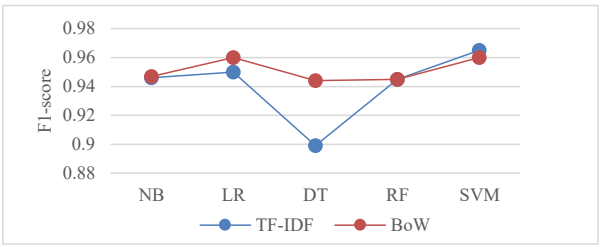

图4 不同机器学习算法与TF-IDF和BoW组合得到的F1-score值的对比结果

类后得到的精确率、召回率和F1-score,结果如表5所示. 结果显示,所有分类器都有较好的分类性能,精确率、召回率和F1-score值几乎都在0.9以上(DT与TF-IDF组合的结果除外). 图5显示,5种分类器与TF-IDF组合时,SVM获得了最佳精确率(0.941)、召回率(0.990)和F1-score(0.965). DT获得了最低召回率(0.867)和F1-score(0.899). 这表明,在需求分类自动分类任务中,使用TF-IDF特征提取技术时,应优先考虑SVM分类器. 图6显示,5种分类器与BoW组合时,SVM和LR获得了最佳F1-score(0.960). SVM获得了最佳召回率(0.980),且与LR(0.979)仅相差0.01. 而NB获得了最高的精确率(0.989). 图6与图5中的结果相似的是,DT获得了最低的F1-score(0.944). 这表明,在需求自动分类任务中,在使用BoW特征提取技术,应优先考虑SVM和LR分类器. 同时也应关注使用NB是否能够得到更好的性能.

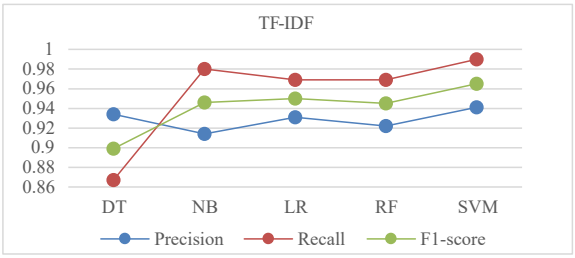

图5　不同机器学习算法与TF-IDF组合时得到的精确率、召回率和F1-score值的对比结果

RQ2结论:整体上看,SVM与TF-IDF组合得到了最佳召回率(0.990)和最佳F1-score(0.965),而NB与BoW组合获得了最高精确率(0.989). 此外,不管是与TF-IDF还是与BoW组合,DT分类器都得到了最低F1-score. 因此,我们认为在将用户评论中的非功能需求分类为表示型需求和行为型需求时,SVM分类器整体性能最好.

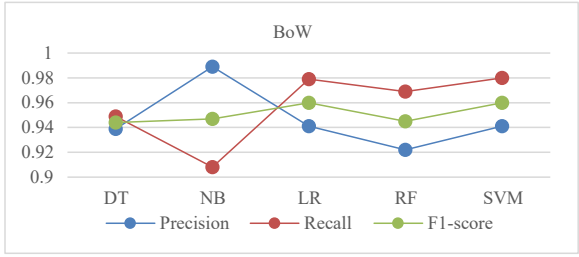

图6　不同机器学习算法与BoW组合时得到的精确率、召回率和F1-score值的对比结果

RQ3:我们计算了TextCNN、RCNN分别与2种词向量组合后得到的F1-score值,结果如图7所示. 实验结果显示,无论是TextCNN还是RCNN,与Word2Vec组合获得F1-score值均高于与FastText组合得到的F1-score.

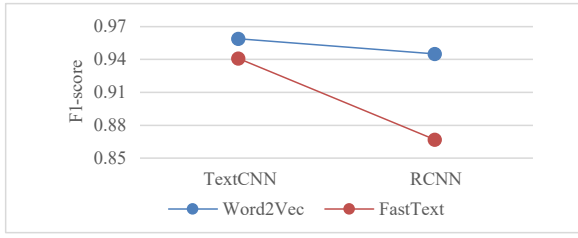

图7　不同深度学习算法与Word2Vec和FastText组合得到的F1-score值的对比结果

RQ3结论:整体上来看,在将非功能需求分类为表示型需求和行为型需求时,Word2Vec模型比FastText模型表现更好.

RQ4:我们计算了TextCNN、RCNN模型分别与Word2Vec、FastText组合,以及CharCNN模型的精确率、召回率和F1-score,结果如表6所示. 图8显示了TextCNN和RCNN分别与Word2Vec组合和CharCNN 3个模型的精确率、召回率和F1-score值的对比结果. 结果显示,TextCNN与Word2Vec组合的精确率(0.950)、召回率(0.969)和F1-score(0.959)均高于其他2个深度学习模型. 图9显示了TextCNN和RCNN分别与FastText组合和CharCNN 3个模型得到的精确率、召回率和F1-score值的对比结果. 结果显示,TextCNN与FastText组合的精确率(0.989)和F1-score(0.941)高于其他2个模型,RCNN与FastText组合获得了最高召回率(1.000).

RQ4结论:整体上看,TextCNN与FastText组合获得了最高精确率(0.989),RCNN与FastText组合获得了最高召回率(1.000),TextCNN与Word2Vec组合获得了最高F1-score(0.959). 尽管RCNN与FastText组合获得了最高的召回率(1.000),却是以所有模型中最低的精确率(0.766)为代价的. 结合图8和图9,我们认为在将非功能需求分类为表示型需求和行为型需求时,TextCNN模型整体性能最好. 但也应该注意到,当关注召回率时,RCNN与FastText组合是非常不错的选择.

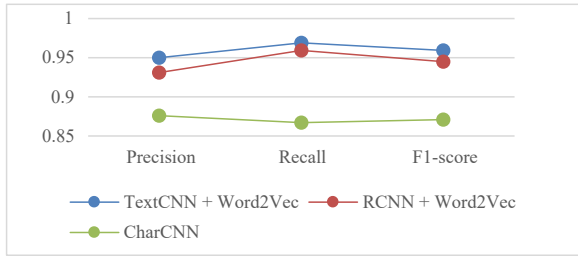

图8　TextCNN与Word2Vec组合、RCNN与Word2Vec组合、CharCNN得到的精确率、召回率和F1-score值的对比结果

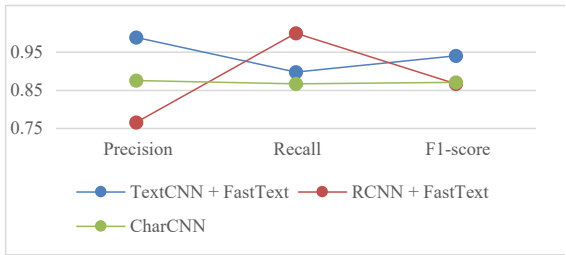

图9 TextCNN与FastText组合、RCNN与FastText组合、CharCNN得到的精确率、召回率和F1-score值的对比结果

RQ5:我们比较了2类模型中获得最高F1-score组合的精确率、召回率和F1-score,结果如图10所示.结果显示,SVM与TF-IDF的召回率(0.990)和F1-score(0.965)值比TextCNN与Word2Vec组合的值高.TextCNN与Word2Vec得到的精确率(0.950)比SVM与TF-IDF得到的精确率高(0.941).很多研究表明,相较于机器学习模型,CNN有更好的分类效果.根据我们的比较结果,机器学习模型得到了较高的召回率和F1-score.深度学习模型的分类效果没有明显的优势,可能的原因有:(1)本文实验的数据集规模较小;(2)深度学习模型存在通过进一步调节参数来达到更好分类效果的可能性.

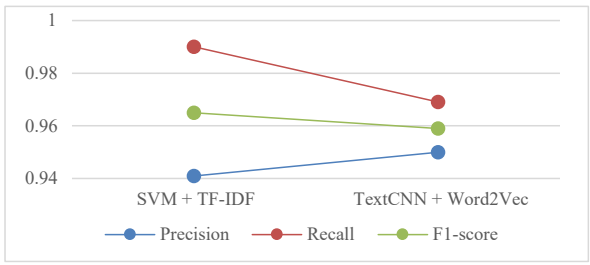

图10 SVM与TF-IDF组合、TextCNN与Word2Vec组合得到的精确率、召回率和F1-score值的对比结果

我们还计算和比较了所有机器学习组合与深度学习组合的时间消耗(建模耗时和模型预测耗时),结果如表7、表8和表9所示.其中,机器学习的建模耗时为完成图2中步骤1~4所需的时间,深度学习建模耗时为完成图3中步骤1~3所需的时间.模型预测耗时分别为完成图2和图3中最后一个步骤所需的时间.在机器学习算法中,建模耗时最长的是RF与BoW组合(994 ms),建模耗时最短的是DT与BoW组合(514 ms).所有机器学习算法的模型预测耗时不超过20 ms.我们注意到,机器学习算法在与TF-IDF组合时的建模耗时普遍比与BoW组合时的建模耗时长(RF除外),而模型预测耗时几乎一致.可能的原因是计算BoW向量比计算TF-IDF向量简单,同时也从时间消耗角度验证了RQ1的回答"简单的BoW技术比TF-IDF技术表现更好".

在深度学习算法中,CharCNN模型的建模耗时最

表7 机器学习算法的建模耗时(Bt)以及模型预测耗时(Pt) ms

	TF-IDF		BoW	
	Bt	Pt	Bt	Pt
NB	548	4	**491**	4
LR	853	4	848	6
DT	585	3	514	3
RF	880	17	994	18
SVM	604	**9**	540	7

表8 深度学习算法(TextCNN、RCNN)的建模耗时(Bt)以及模型预测耗时(Pt) ms

	Word Embedding			
	Word2Vec		FastText	
	Bt	Pt	Bt	Pt
TextCNN	94575	27817	135659	27694
RCNN	224776	82282	277075	72534

表9 深度学习算法(CharCNN)的建模耗时(Bt)以及模型预测耗时(Pt) ms

CharCNN	Character Embedding	
	Bt	Pt
	514391	85078

长(514391 ms)且模型预测耗时也最长(85078 ms).TextCNN和RCNN与FastText组合的建模耗时,比与Word2Vec组合耗时长了约40000到50000 ms.可能的原因是FastText词向量考虑了单词的内部结构,而Word2Vec计算1个完整单词的词向量.但在模型测试时,TextCNN和RCNN与FastText组合的预测耗时,比与Word2Vec组合耗时短100到10000 ms.TextCNN分别与Word2Vec、FastText组合的建模耗时是所有深度学习模型中耗时最短的,模型预测耗时也是最短的.可能的原因是TextCNN直接使用了预训练的词向量进行建模,RCNN在预训练词向量的基础上重新计算了词向量,CharCNN则基于文本字符计算文本向量,因此这2个模型比TextCNN模型消耗更多的建模时间.这从时间消耗角度验证了RQ4的回答"深度学习模型中,TextCNN模型整体性能最好".

在时间消耗方面,深度学习模型耗时较长,主要有2个原因:(1)在特征提取阶段,机器学习模型通过1278条用户评论计算向量值,而深度学习模型则通过11096条用户评论计算词向量和字符向量;(2)深度学习模型需要学习的参数比机器学习模型要多,因此建模耗时和模型预测耗时比机器学习模型耗时要长.

RQ5结论:整体上,机器学习模型完成1次分类的时间仅需500~1000 ms,深度学习模型耗时100000~200000 ms,约为机器学习模型耗时的200倍.无论在召回率、F1-score的比较上,还是在时间消耗的比较上,

机器学习模型都优于深度学习模型,尤其是2者在耗时上的巨大差距. 因此我们认为,在将非功能需求分类为表示型需求和行为型需求时,机器学习模型比深度学习模型表现更好,并且在需求文本分类的分类应用中应首先尝试机器学习模型,在性能相当的情况下,应优先使用耗时更少的机器学习模型,在使用深度学习模型时,需要从性能和时间消耗维度对其进行整体收益评估.

6　结果的效度与局限性

本文依据文献[20]中的效度分析准则,从构造效度、内部效度、外部效度和可靠性讨论对本研究结论有效性的可能威胁,并介绍我们为缓解这些威胁所采取的措施.

6.1　构造效度分析

构造效度关注理论构造是否能被正确地解释和度量. 本文的潜在威胁是人工分类过程中决定每条非功能需求类别时的个人主观性. 为了减少这类威胁,人工分类的过程由3人共同完成(软件工程领域硕士生和教师),在正式分类之前,3人首先随机选取了50条用户评论句子进行预分类,对于存在分歧意见的用户评论句子,3人通过讨论最终达成一致意见. 正式分类由参与预分类的2名硕士生先单独进行分类标记,有不同分类意见时与第3人(教师)讨论,并最终在分类结果上达成一致. 另一项潜在威胁是基于本研究所使用的用户评论是否能够得出合理的结论,为了减少该威胁,我们通过随机抽取的方式从移动应用中选取用户评论样本数据作为本研究的数据集.

6.2　内部效度分析

内部效度关注研究结果是否可以由本研究数据得出,以及是否有其它影响结果的因素未考虑到. 本文的内部效度威胁是机器学习算法和深度学习算法是否存在过拟合或欠拟合. 为了减轻这种威胁的影响,我们使用2种特征提取技术与5种机器学习算法进行组合,并使用2种基于词嵌入的深度学习算法和1种基于字符向量的深度学习算法,最后评估和对比了所有组合的性能和时间消耗. 在当前的数据规模下,结果表明,在对非功能需求分类进行分类时,机器学习模型比深度学习模型在性能和时间消耗方面表现更好. 但不能确定在数据集规模更大时,是否仍能得到与本文一致的结果,需要通过进一步实验来研究.

6.3　外部效度分析

外部效度指研究结果在多大程度上可适用于其它数据集和环境. 本文对2个常用的移动应用的用户评论进行非功能需求分类,通过应用类别和平台环境的多样性来减少研究数据对外部效度产生的威胁. 但是必须承认本研究中采用的移动应用类别数量有限,因此不能确定在使用其它类别应用程序(例如教育类)的用户评论进行非功能需求分类时,结果是否一致. 此外,本文数据来源于国外的移动应用的英文用户评论,因此尚无法保证在对中文的用户评论使用相同的研究方法时能够得到相同的研究结果.

6.4　可靠性分析

可靠性指如果其他研究人员重复本研究是否会得到相同或相似的结果. 本文的人工分类标记过程由3名软件工程领域研究人员共同完成,并明确了分类标记过程,以尽可能地减少个人偏见对分类结果的影响. 此外,本文提供了研究使用的数据集[7],说明了实验步骤和参数设置,给出了实验环境,供其他研究人员使用和验证本研究工作,提升了本研究结果的可重复性. 需要说明的是,本文基于国外的移动应用的用户评论进行研究,研究结果尚未与应用的开发团队进行确认,因此尚无法确定研究结果对这2个应用未来的更新迭代所产生的影响.

7　总结

本文根据Broy[2]等人提出的系统模型,将iBooks和WhatsApp 2个应用程序的1278条非功能需求用户评论人工标注为行为型需求和表示型需求. 并使用了机器学习与深度学习算法进行分类. 在使用机器学习进行分类时,将TF-IDF和BoW技术分别与NB、LR、DT、RF以及SVM进行组合. 在使用深度学习进行分类时,将Word2Vec、FastText分别与TextCNN、RCNN相结合,并构建和训练了基于字符向量的CharCNN深度学习模型. 我们使用上述所有组合将1278条非功能需求分类为行为型需求和表示型需求,最后对所有组合的分类性能和时间消耗进行评估,其中在性能方面我们计算和比较了所有组合的精确率、召回率和F1-score. 结果表明,在将非功能需求分类为行为型需求和表示型需求时(其中第1到第5点分别对应RQ1到RQ5的结果,第6点为综合结果):

(1) 在机器学习模型中,简单的BoW技术比TF-IDF技术在特征提取方面表现更好.

(2) 在机器学习模型中,SVM分类器的整体性能最好.

(3) 在深度学习模型中,Word2Vec模型比FastText模型在计算词向量方面表现更好.

(4) 在深度学习模型中,TextCNN的整体性能最好.

(5) 在性能和时间消耗2个维度上,机器学习模型比深度学习模型表现更好.

(6) TF-IDF与SVM组合得到了最佳F1-score

(0.965),BoW与NB组合、FastText与TextCNN组合获得了最高精确率(0.989),FastText与RCNN组合获得了最高召回率(1.000).

基于本文的研究结果,我们计划从以下5个方面来改进我们的工作:

(1) 本文机器学习模型使用了默认参数,深度学习模型根据经验选取了常用的数值以确定主要参数,未来我们计划通过实验来确定这些模型的最佳参数,并将尝试更多其它主流的分类算法进行非功能需求的自动分类实验,如DCNN(Dynamic Convolutional Neural Network)、HAN(Hierarchical Attention Networks)等.

(2) 本研究数据集来自应用商店中2个类别的应用程序,为了提高研究结果的通用性以及自动分类效果的外部效度,我们拟在数据集中增加移动应用类别的数量.

(3) 本研究数据集规模可能会影响自动分类算法的训练效果,进一步影响分类效果. 我们计划标记更多数量的移动应用用户评论句子,以减少对外部效度的威胁.

(4) 将本文的研究结果与iBooks和WhatsApp应用的开发团队进行确认,进一步提升本文研究结果的可靠性.

(5) 本文基于两个国外的移动应用的英文用户评论进行非功能需求自动分类的研究,我们也计划利用国内开发的移动应用的中文用户评论进行研究,提升研究结果的通用性、自动分类效果的外部效度以及研究结果的可靠性.

参考文献

[1] FERNÁNDEZ D M. Supporting requirements-engineering research that industry needs: The NaPiRE initiative[J]. IEEE Software, 2018, 35(1): 112-116.

[2] BROY M. Rethinking Functional Requirements: A Novel Approach Categorizing System and Software Requirements [M]//Software Technology: 10 Years of Innovation in IEEE Computer. Hoboken, NJ, USA: John Wiley & Sons, Inc, 2018: 155-187.

[3] BROY M. Rethinking nonfunctional software requirements [J]. Computer, 2015, 48(5): 96-99.

[4] ECKHARDT J. Categorizations of Product-related Requirements in Practice[D]. München: Technische Universität München, 2017.

[5] LU M M, LIANG P. Automatic classification of non-functional requirements from augmented app user reviews[C]// EASE'17: Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering. New York: ACM, 2017: 344-353.

[6] Joint Technical Committee ISO/IEC JTC 1. ISO/IEC 25010: 2011[S/OL].[2021-04-07].https://www.iso.org/standard/35733.html.

[7] 李雪莹,王田路,梁鹏,王翀. 基于系统模型的用户评论中非功能需求的自动分类[EB/OL]. [2021-04-07] http://doi.org/10.5281/zenodo.4399602.

[8] GLINZ M. On non-functional requirements[C]//15th IEEE International Requirements Engineering Conference. Piscataway: IEEE, 2007: 21-26.

[9] ABAD Z S H, KARRAS O, GHAZI P, et al. What works better? A study of classifying requirements[C]//2017 IEEE 25th International Requirements Engineering Conference. Piscataway: IEEE, 2017: 496-501.

[10] LI C Y, HUANG L G, GE J D, et al. Automatically classifying user requests in crowdsourcing requirements engineering[J]. Journal of Systems and Software, 2018, 138: 108-123.

[11] STANIK C, HAERING M, MAALEJ W. Classifying multilingual user feedback using traditional machine learning and deep learning[C]//2019 IEEE 27th International Requirements Engineering Conference Workshops. Piscataway: IEEE, 2019: 220-226.

[12] JHA N, MAHMOUD A. Mining non-functional requirements from App store reviews[J].Empirical Software Engineering, 2019, 24(6): 3659-3695.

[13] WANG T L, LIANG P, LU M M. What aspects do non-functional requirements in app user reviews describe? an exploratory and comparative study[C]//2018 25th Asia-Pacific Software Engineering Conference(APSEC). Piscataway: IEEE, 2018: 494-503.

[14] MIKOLOV T, CHEN K, CORRADO G, et al. Efficient estimation of word representations in vector space[EB/OL]. (2013-01-16). https://arxiv.org/abs/1301.3781.

[15] BOJANOWSKI P, GRAVE E, JOULIN A, et al. Enriching word vectors with subword information[J]. Transactions of the Association for Computational Linguistics, 2017, 5: 135-146.

[16] KIM Y. Convolutional neural networks for sentence classification[C]//Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Stroudsburg, PA, USA: Association for Computational Linguistics, 2014: 1746-1751.

[17] LAI S W, XU L H, LIU K, et al. Recurrent convolutional neural networks for text classification[C]//AAAI'15: Proceedings of the Twenty-Ninth AAAI Conference on Arti-

ficial Intelligence. Menlo Park, CA: AAAI, 2015: 2267-2273.

[18] ZHANG X, ZHAO J, LECUN Y. Character-level convolutional networks for text classification[C]// Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1. Cambridge, MA: MIT Press, 2015: 649 – 657.

[19] COHEN J. A coefficient of agreement for nominal scales [J]. Educational and Psychological Measurement, 1960, 20(1): 37-46.

[20] SHULL F, SINGER J, SJØBERG D. I. Guide to Advanced Empirical Software Engineering[M]. Berlin, German: Springer, 2008.

作者简介

李雪莹　女，1993年12月出生于河南省南阳市．2021年毕业于武汉大学计算机学院．主要研究方向包括需求工程、软件体系结构和机器学习．

E-mail: xueyingli@whu.edu.cn

王田路　女，1995年6月出生于河北省邢台市．本硕毕业于武汉大学计算机学院．现在中国银行总行信息科技部工作．主要研究方向为软件体系结构和需求工程．

E-mail: wangtianlu@whu.edu.cn

梁　鹏(通讯作者)　男，1978年10月出生于湖北省荆门市．现任武汉大学计算机学院教授．主要研究方向为软件体系结构和需求工程．

E-mail: liangp@whu.edu.cn

王　翀　女，1981年3月出生于湖北省武汉市．现任武汉大学计算机学院讲师．主要研究方向为面向服务软件工程和需求工程．

E-mail: cwang@whu.edu.cn