智能家居场景联动中基于知识图谱的隐式冲突检测方法研究

肖 丁 王乾宇 蔡 铭 李 秀

(浙江大学计算机科学与技术学院 杭州 310027)

摘 要 智能家居场景联动是智能家居系统提供的一种自动化服务,它通过各类智能设备的互联、互通和互动,从而为住户提供更为舒适、安全和节能的家居环境.然而,由于各个智能场景的侧重点不同、管理策略各异,当多个智能场景联动和叠加时,会引起作动器设备之间的竞争、干扰和矛盾等控制冲突现象,降低智能场景用户体验,缩短智能设备使用寿命,甚至危及住户的人身财产安全.因此,智能家居场景联动中的控制冲突问题亟需大量实验研究并找到解决方案.本文中,我们将控制冲突分为两大类,一类是同一作动器上执行明显相反动作的显式冲突,一类是不同作动器之间产生相互干扰动作的隐式冲突.现有的控制冲突检测技术对于显式冲突检测的研究较为成熟,而在隐式冲突检测方面性能相对较弱,其原因可归结为自动化程度低与漏判率高两大缺点.本文提出了一种基于知识图谱语义分析的作动器隐式冲突检测方法,可以充分整合并有效挖掘各类作动器功能描述的常识性知识,进而实现对作动器隐式冲突的自动化检测.该方法有两个主要环节,分别是作动器设备注册环节与规则制定环节.我们在作动器设备注册环节提取知识图谱中的常识性知识,对作动器功能之间的效果冲突进行识别判断,并以隐式冲突矩阵的形式将判断结果在边缘设备上进行存储,实现存在于不同类型作动器之间的干扰检测;在规则制定环节,借助隐式冲突矩阵的冲突信息进行字典式查询与匹配,以判断规则之间的隐式冲突.此外,基于知识图谱中设备功能短语自身特点,设计了基于语义贡献度的权重分配策略,完成词向量到短语向量的转化.为解决聚类后相同簇中近反义功能短语混合杂糅的问题,本文提出了相对极性指标和近反义关系一步扩展方法,并借由图模型染色算法将反义关系的短语拆分到不同子类,优化了聚类效果.通过利用 IFTTT 数据集对 10 个典型户型的测试,我们的方法对隐式冲突的查全率达到 0.8301,查准率达到 0.9681,F-score 值为 0.8938.实验结果表明,我们的方法相比于当前检测技术,查全率至少高出 61.93%,F-score 值至少高出 54.56%.并且与已有的方法相比,我们的方法不需要人工标注,可以自动化地进行隐式冲突检测,显著提高了检测效率,同时极大降低了冲突的漏判率.

关键词 智能家居场景联动;控制冲突;隐式冲突;知识图谱;权重分配;极性分析;智能家居

中图法分类号 TP391 **DOI 号** 10.11897/SP.J.1016.2019.01190

Research on Implicit Interference Detection Based on Knowledge Graph in Smart Home Automation

XIAO Ding WANG Qian-Yu CAI Ming LI Xiu

(*College of Computer Science and Technology, Zhejiang University, Hangzhou* 310027)

Abstract Smart home automation is a service which provides residents with more comfortable, approved and energy-saving home environments via the connection, interoperation and interaction of smart devices. However, due to different concerns and management strategies of each smart scene, interferences such as competition, interference and contradiction among devices frequently occur when multiple automation scenes encounter. They influence user experience of smart home automation, shorten service life of smart devices and even endanger the personal and property

收稿日期:2018-05-11;**在线出版日期**:2019-03-16.**本课题得到国家自然科学基金**(51775496,51875503)**资助**.**肖 丁**,博士研究生,主要研究方向为物联网、人工智能等.E-mail:derll.xiao@gmail.com.**王乾宇**,硕士研究生,主要研究方向为物联网、嵌入式系统等.**蔡 铭**(通信作者),博士,副教授,中国计算机学会(CCF)会员,主要研究方向为物联网、人工智能、嵌入式系统等.E-mail:cm@zju.edu.cn.**李 秀**,硕士,主要研究方向为物联网、嵌入式系统等.

safety of residents. Therefore, interference problems in smart home automation require enough researches and solutions urgently. In this paper, we divide the interference into two categories: one is the explicit interference that performs as obviously opposite actions on the same actuator, the other is the implicit interference that manifests as interference between different actuators. The state-of-the-art interference detection technology is relatively mature on explicit interferences. While in respect to implicit interferences, the performance of these detection technology is relatively weak, and the reasons of that can be concluded as two major disadvantages, low degree of automation and high false negative rate. This paper proposes an implicit interference detection method. Through analyzing information from knowledge graph, our method is able to integrate and utilize commonsense knowledge of actuators in different categories substantially, and then detect the implicit interference among actuators in automation. Our method contains two main phases, which are actuator registration phase and rule subscription phase. During the actuator registration phase, our method utilizes commonsense knowledge included in the knowledge graph to evaluate the effects caused by the actuator, judge the interference of effects automatically and at last store the results as the implicit interference matrix at the edge devices; While in the rule subscription phase, our method matches and queries the interference of effects between actuators recorded in the implicit interference matrix to help judge the interference from rules. Besides, based on the characteristics of functionality phrases stored in the knowledge graph, we analyse the grammar of these phrases and design a strategy to determine the weight of words according to the contribution degrees of each word to the whole phrases, and then updates and clusters on the basis of the actuator function lists. For the semantic non-relevance problem led by word properties in natural language texts, we propose the relative polarity index and relative polarity calculation method, which transforms the division problem into a graph coloring problem, and thus it can classify the same and opposite operations from relevant functions. Using the rule data collecting from IFTTT website, we design an experiment set based on the 10 typical floor plans. The experiment result shows the recall rate of our method reaches 0.8301, the precision rate reaches 0.9681 and the *F*-score is 0.8938. The experiment results show that our method outperforms state-of-the-art technologies by more than 61.93% in recall rate and 54.56% in *F*-score. Besides, compared with existing methods, our method does not require manual labeling and has the ability to automatically detect the implicit interferences. Therefore, our method significantly improves the detection efficiency and reduces the missed rate.

Keywords　smart home automation; interference; implicit interference; knowledge graph; weight distribution; polarity analysis; smart home

1　引　言

智能家居场景联动是智能家居系统提供的一种自动化服务.它通过智能设备感知环境以及场景变化,自动执行相应的动作,从而有效提高家庭环境的舒适度、安全性和节能化程度,全面提升用户体验[1].例如,当智能摄像头检测到住户走进家门时,自动开启“回家模式”:打开楼道灯光,客厅背景音乐缓缓响起,中央空调调整至舒适的温度;当智能手环监测到人体进入睡眠状态,自动执行“睡眠模式”:关闭全家灯光,小夜灯自动开启,调高卧室空调温度,避免人受凉,室内切换至安防模式.

区别于传统家电的控制操作全部依赖于用户行为,智能家居场景联动集成了智能传感、无线网络、机器对机器通信、自动化控制等一系列技术,使得各类智能设备能在预先设定的条件情形下进行互联、互通和互动,从而为用户提供更加便捷、舒适和安全的智能化服务.

参与智能家居场景联动的智能设备分为两大

类:传感器(Sensors)和作动器(Actuators)[2].传感器是监测环境因素变化的设备,能够感知并收集温度、光线、湿度、烟雾、音频、视频以及其他环境数据,是数据收集者;而作动器是实施动作和提供服务的功能设备,如空调控制器、窗帘控制器、照明控制器等,是指令执行者.

智能家居场景联动提供的服务以规则集合的形式来互联互通各类的智能设备,并驱动各类智能场景[3].目前主流的智能家居系统,如三星的 SmartThings、苹果的 HomeKit、小米智能家庭平台(米家)、云端规则定制平台 IFTTT[4]及开源智能家居系统 HomeAssistant 等,均采用 Trigger-Action 模式来定制各类规则.传感器负责收集数据以验证环境是否达成规则的触发条件(Trigger),作动器则负责按照规则制定的方式执行对应的动作(Action).例如,用户制定规则"当室内二氧化碳的浓度超过一定范围的时候,则打开窗户通风".其中触发条件就是"二氧化碳浓度超过一定范围",这一信息是由二氧化碳传感器负责感知采集的数据.如果当前的触发条件满足,作动器会接受来自智能家居系统的命令,执行开窗动作.

然而,由于 Trigger 多种多样,同时作动器行为严格遵循规则中的 Action 命令,因此随着制定的规则数量增多,规则之间产生控制冲突现象的可能性也随之提高[5].这些控制冲突会降低智能场景的用户体验,缩短智能设备使用寿命,甚至危及住户的人身财产安全[6].因此在智能家居系统中对于控制冲突的检测非常必要且尤为关键.

对于控制冲突,考虑以下规则组:

规则 1. 当太阳落山时,打开窗户;

规则 2. 当太阳落山时,关闭窗户;

规则 3. 当太阳落山时,打开暖气;

规则 4. 当太阳落山时,打开空调制冷模式.

其中,规则 1 和规则 2 的 Trigger 是相同的,但是使得同一个作动器"窗户"执行了相反的 Action,因此产生了控制冲突;规则 3 与规则 4 在相同 Trigger 条件下,命令不同的作动器"暖气"和"空调"产生动作,然而两者动作之间产生的效果是矛盾的(暖气制热,空调制冷),因此同样判定为冲突;同时,规则 1 与规则 3、规则 1 与规则 4 两两之间也是相互冲突的,因为打开暖气或者空调时,如果窗户被打开,由于空气流通,其会严重影响暖气或者空调的运作.在上述例子之中,第一组冲突的检测更为直观,因为同一个作动器实施了相反的 Action.相比之下,第二组和第三组冲突的检测较为困难,因为规则涉及不同作动器,要进一步判断不同作动器之间的行为是否有冲突,这需要引入必要的常识性知识进一步判断.

因此,针对两种不同情形,我们将控制冲突分为两类:

定义 1. 在短时间内能够同时满足的条件下,同一作动器同时产生明显相反动作的控制冲突称为显式冲突.

定义 2. 在短时间内能够同时满足的条件下,不同作动器产生的动作导致效果相反的控制冲突称为隐式冲突.

现有工作对于冲突检测的理论成因和逻辑关系的研究相对成熟[7-8],然而针对隐式冲突检测的实际应用的研究仍有不足.现有的隐式冲突检测方法主要是通过引入环境变量作为判断中介,同时人工标注每一条规则对环境变量的影响来进行判断.例如,规则 3 与规则 4 中,空调和暖气都影响了环境变量"温度".因此,标注规则 3 和规则 4 对于温度的影响呈现相反效果,进而检测出隐式冲突.但是这类检测方法面临两大困难与挑战:

(1)自动化程度低.现有的隐式冲突依赖于人工对智能设备的功能标注信息,工作量大,自动化程度低.由于智能家居平台的多样性,大量的人工标注和额外规定不仅给开发人员带来负担,同时也不利于检测方法在不同平台上的扩展应用.此外,当智能家居场景联动中加入全新种类设备或全新服务类型时,可能会与原场景中其他原有设备或服务产生冲突或干扰,因此原有方法需要很高的维护成本持续对新设备或新服务进行功能标注.

(2)漏判率高.由于智能设备具有多功能性、标注者背景知识、能力水平也参差不齐,使得人工对规则的标注难以保证全面性和一致性,导致冲突的漏判.例如,对窗户仅标注了"采光"、"吹风"功能,而遗漏了"通风"、"升温"、"降温"等功能,则会遗漏"空调制冷"与"打开窗户"之间存在的隐式冲突.因为窗户打开后会将室内产生的冷空气流出室外,影响空调制冷效果,加大能耗.

由于知识图谱中的知识语料有着来源广泛、涵义丰富、结构清晰和关系严谨等特点,知识图谱在消除歧义、问答系统等方面得到了广泛的应用.因此,本文提出了一种基于知识图谱语义分析的作动器隐式冲突检测方法,可以充分整合并有效挖掘各类作动器功能描述的常识性知识,进而实现对作动器隐式冲突的自动化检测.与传统方法相比,该方法对于

自动化程度和漏判率都得到明显的改善，且跨平台适应性良好. 本文的贡献如下：

(1) 将控制冲突划分为显式冲突和隐式冲突两类. 针对隐式冲突检测，本文首次提出了一种自动化的隐式冲突检测方法——KGIID (Knowledge Graph based Implicit Interference Detection)，基于知识图谱抽取作动器常识性知识，对不同类型作动器之间隐式冲突实现自动化的判断.

(2) 为解决聚类后的相同簇中近反义功能短语混合杂糅问题，首次提出反义关系子类拆分的图模型染色方法，将反义关系的功能短语拆分到不同子类，实现聚类效果优化，降低了误判概率.

(3) 开展了与现有的隐式冲突判别方法的对比实验，进行了与人工标注方法的检测效果对比. 实验结果表明本文提出的基于知识图谱的隐式冲突规则检测方法具有相较于其他方法更好的检测性能.

本文第 2 节介绍相关工作；第 3 节介绍 KGIID 方法的流程设计；第 4 节介绍作动器功能列表的获取与处理；第 5 节介绍作动器功能聚类；第 6 节介绍隐式冲突矩阵的生成；第 7 节介绍实验设计与评估；第 8 节全文总结.

2　相关工作

Hu 等人[8]提出了一个名为 SPIDER 的检测方法来检测用户策略(即用户制定的规则)间的冲突. SPIDER 自行定义了一个针对智能家居的基于本体的语义语境模型，并在该模型中将参与智能家居系统的成员分为元件(传感器和作动器)、服务、策略和特征. 在此基础上，该模型将用户制定的规则以模型中的各成员进行基本逻辑运算的形式进行结构化解析，从而能够表达出规则间的相互关系和冲突成因. 但该模型仅仅在理论层面上对冲突检测进行了研究，并未在实验方面考虑隐式冲突如何检测识别等问题.

Munir 等人[9]设计了 DepSys 系统，DepSys 通过解决一系列依赖关系(包括需求，名称和控制依赖关系)，提供了全面的策略来检测和解决智能家居的冲突问题. DepSys 将冲突检测模块划分为静态检测模块和动态检测模块. 静态检测模块在设备安装过程中进行冲突可能性分析，而动态检测模块则在运行时进行检测. 针对隐式冲突检测，DepSys 采用人工标注规则动作对于环境变量(如温度、湿度、光照等)的影响趋势(上升或下降)来判断不同作动器之间是否产生了冲突效果，并将标注好的结果形成结构化的文档提供给系统. 然而，DepSys 对于人工标注的文件格式具有严格的规定，不利于冲突检测模块在不同平台上的移植.

Sun 等人[10]提出了一个基于形式化规则的控制冲突检测模型 UTEA. UTEA 将规则之间的关系定义为 11 种类型，由关系间不同的组合构成 5 种类型的冲突. 针对隐式冲突检测，该模型引入了环境实体(Environment Entities)模块，通过标注不同规则对于环境实体影响的目标值(或范围)，来判断各个规则之间是否有冲突. 但是，目前智能家居系统中并未对环境实体模块进行独立配置，因此该检测方法只适用于其模型自身.

3　KGIID 方法流程

KGIID 方法的整体流程主要分为两个环节：一个是作动器设备注册环节，另一个是规则定制环节，整体流程如图 1 所示.

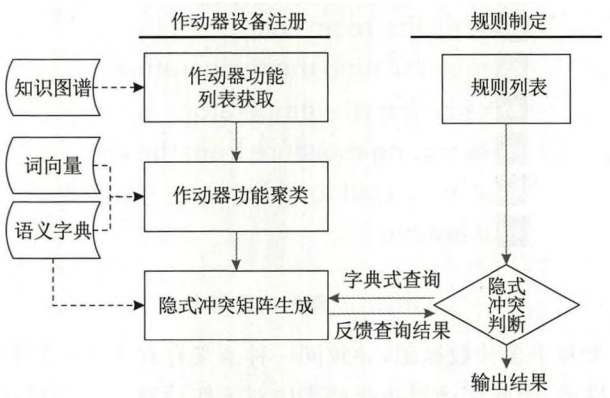

图 1　KGIID 方法整体流程

3.1　作动器设备注册环节

主要包含作动器功能列表获取、作动器功能聚类和隐式冲突矩阵生成三个步骤，主要目的是将智能家居系统发现的所有作动器设备的功能进行冲突预判断，并以隐式冲突矩阵的形式进行存储，方便规则制定环节中进行隐式冲突判断. 具体处理流程如下：

(1) 作动器功能知识条目获取与处理. 在知识图谱中检索新注册的作动器类型，获得该作动器的功能知识条目. 图 2 所示的是以“空调”(air conditioning)为例，在知识图谱中所获取的功能知识条目列表，简称功能列表. 然后根据作动器功能列表，结合词向量工具计算每个功能知识短语的短语向量，为聚类作准备.

(2) 作动器功能聚类. 由于在知识图谱中的知识

图 2 空调的功能列表

来源于多种数据源，导致同一种语义存在多种不同的描述，因此作动器功能列表内对于作动器功能的描述有着大量的重复和冗余，例如图 2 中，空调功能列表中直接表达“cool”的相关功能条目达到了 11 条，间接表达制冷效果的有 3 条，它们都可视为空调制冷功能的重复表达. 为了消除重复冗余功能信息，我们引入词向量，将作动器功能进行聚类，得到初步聚类结果. 由于初步聚类结果中将相关性高的功能知识条目聚为了一类，而相关性包含相同和相反. 因此我们引入语义字典，将初步聚类结果的同一簇中相反语义的功能知识条目进行子类拆分，确保每一个子类内部并不包含语义相反的功能短语，从而获得作动器功能簇列表.

(3) 隐式冲突矩阵生成. 结合语义字典，我们将不同作动器的功能簇列表进行两两比较，可以得到功能簇之间的相关关系，以判断作动器之间各个功能簇的作用为相同或相反关系. 出于数据传输负载和数据隐私的考虑，智能家居内的敏感数据处理都应在家庭范围内完成[11]. 因此我们将相同或相反关系的判断结果，以隐式冲突矩阵的形式存储在配置智能家居系统的边缘设备(如家庭网关、边缘路由器等)上，方便规则制定环节进行隐式冲突判断.

3.2 规则制定环节

当用户制定新规则后，规则引擎将新加入规则和已有的规则生成规则列表，放入隐式冲突判断模块进行冲突识别. 隐式冲突模块在判断规则触发条件能够同时满足的情况下，将规则之中的作动器设备名称和动作，与隐式冲突矩阵中的作动器名称和功能簇进行匹配. 隐式冲突矩阵将查询结果反馈给隐式冲突判断模块. 如果两条规则之间的动作所对应的功能簇是相反关系，则判断这两条规则出现了隐式冲突.

规则制定环节中，隐式冲突判断环节依赖于作动器设备注册环节中的输出——隐式冲突矩阵. 因此 KGIID 方法的技术关键在于如何准确、高效地生成隐式冲突矩阵. 本文将在后续 4、5、6 三节中分别详细介绍作动器设备注册环节中三个步骤的技术原理.

4 作动器功能列表获取与处理

如图 3 所示，作动器功能列表的获取与处理主要分为三个步骤：一是提取知识图谱中的作动器功能列表；二是对功能列表中功能短语的词性标注和词形还原；三是对功能短语进行向量化表示，以方便功能聚类.

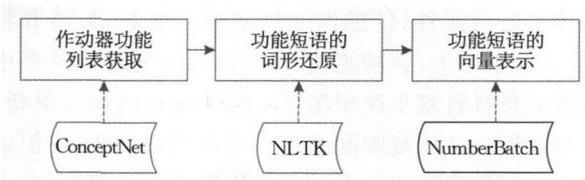

图 3 作动器功能列表获取与处理流程

4.1 作动器功能列表获取

目前，常用的知识图谱有 Freebase[12]、Wikidata[13]、DBpedia[14]、ConceptNet[15] 等. 由于 ConceptNet 5.0 中含有 35 种边关系(即语义关系)，并支持在线 API 接口，相比之下更为全面，因此我们选用 ConceptNet 5.0 知识图谱作为获取作动器功能列表的数据源. 在 ConceptNet 5.0 的作动器所有边关系中，“used for”边关系与作动器功能信息最为相关(如图 2 所示空调的“used for”边关系)，因此作动器功能列表主要提取“used for”边关系. 对于少量缺乏

“used for”边关系的作动器条目（如“humidifier”、“air purifier”），我们提取其“is capable of”和“derived from”边关系作为补充.

4.2　功能短语的词性标注与词形还原

作动器功能列表中的各功能知识条目，都是以功能短语的形式存储. 由于英语自身的语法特点，需要进行词形还原（Lemmatization），即将功能短语中的各单词的屈折形态或者派生形态简化或归并为原形的基础形式[16]，以便于提高短语距离计算的准确性. 例如，“removing moisture”中“removing”应还原为原形“remove”，“breathe better”中“better”需还原为原形“good”. 而词性标注是进行词形还原之前所做的预处理步骤，是为每个单词标注一个词性，以确定该词是名词、动词、形容词或者其他词性的过程.

在实现方案上，我们采用NLTK（Natural Language Toolkit，自然语言工具箱）结合词性标注的方式进行词形还原. NLTK 是一组用 Python 语言开发的用于自然语言处理的模块工具包. 其本身提供与 WordNet 的良好接口，在词形还原时能够对目标单词在 WordNet 中进行访问查询，并进行词缀删除和转换，从而获得目标单词的有效原形. 同时，词性标注的准确率直接影响着词形还原的准确率. 在词性标注器选择方面，我们采用 NLTK 包自带的 pos_tag 方法（part-of-speech tagging）进行标注.

在词性标注的过程中，我们需要结合该单词在 ConceptNet 中的例句进行标注，才能使得词形还原更为准确. 例如对单词“cooling”在脱离上下文环境的情况下进行标注，容易标注其为 NNS（名词复数形式）；但将其放入原句“It is used for cooling an office”中后进行标注，才能准确判别其词性为 VBG（动词现在进行时）.

4.3　功能知识条目的短语向量表示

在功能聚类的指标选取方面，考虑到自然语言处理中，词向量技术（Word Vector，又称词嵌入，Word Embedding）是基于文本训练而得到，保证了词汇之间的相对相似度和语义相似度，因此选取词向量作为聚类指标进行功能聚类较为合适. ConceptNet NumberBatch（简称为 NumberBatch）是词向量技术中的一种[17-18]，它作为 ConceptNet 开源数据项目的一部分，构建之初便结合了来自 ConceptNet、Word2vec[19]、GloVe[20]等数据并加以改进[20]. 因此综合考虑，本实验选用 NumberBatch 进行词义分析和极性判别最为合适.

经过大量查阅 ConceptNet 中的各类作动器词条，我们观察到表达作动器功能知识条目的短句均为动词短语结构（VP 结构短语），例如“cooling a room”、“keep the room cold”等，词向量技术无法直接使用. 我们需要运用语义线性放缩方法进行短文本内容计算，因为通过语义放缩更有利于分类、比较和查找[21]. 因此，为了度量两个短语之间的相关程度，我们提出了一种话题语义的线性放缩方法，对功能短语建立可计算的模型，将语句转化为短语向量的形式，为聚类作准备.

为求得功能短语的向量表示，我们有以下两个定义：

定义 3.　$S=\{w_1,\cdots,w_n\}$：某一个功能短语 S，由其包含的所有单词 W_i 的词向量 w_i 组成.

定义 4.　$K=\{k_1,\cdots,k_n\}$：功能短语 S 的向量倍数集合 K，由各个词向量 w_i 对应的向量倍数 k_i 组成，并满足式（1）

$$\sum_{i=1}^{n}k_i=1 \tag{1}$$

因此，我们可以得到式（2），其中 $T(S)$ 表示功能短语 S 的向量表示.

$$T(S)=\frac{\sum_{i}^{n}S.w_i\cdot K.k_i}{n} \tag{2}$$

式（2）中，词向量 w_i 可由 NumberBatch 直接获取，向量倍数 k_i 的赋值与分配策略是影响短语向量值的关键. 我们将向量倍数 k_i 视作单词 W_i 语义对整个短语的语义贡献度. 经研究发现，实义义动词组成的动词短语中，动词对于整个短语的语义贡献度最大，如“cooling a residence”中“cooling”的语义更重要；而由使役动词或系动词组成的动词短语中，动词无实意，此时其他词汇对整个短语的语义贡献度最大，如“keep the room cold”中“room”和“cold”两个单词的语义贡献度更大.

因此我们提出一种基于语义贡献度的权重分配策略，该策略的核心思路是根据每个词向量 w_i 对于短语语义的贡献程度来赋予向量倍数 k_i 的权重值. 具体策略如下：

（1）在实义动词的 VP 结构短语中，实义动词本身对于短语语义的贡献程度较大，例如“cool a resident”. 因此对于实义动词的向量倍数 k_i 我们赋予统一的高权重 p，同时将代词、冠词、介词等无实义的词汇的向量倍数设为 0 后，剩余词语平均分配余下的倍数权重.

（2）使役动词、系动词、助动词或情态动词组成

的 VP 结构. 这类动词本身含义较弱，对整个短语语义贡献较小，而宾语部分语义贡献更大，例如“let the light in”. 因此对于这一类动词向量倍数设为 0 权重，同时将代词、冠词、介词等无实义的词汇的向量倍数设为 0 后，剩下的单词平均分配倍数权重.

经实践发现，在语料不完全充分的情形下，基于语义贡献度的权重分配策略相比深度学习的注意力训练机制在准确率上更具有优势，可以减小短语向量化时次要语义成分的干扰，确保短语向量的极性倾向不会被影响. 经过反复实验，我们可以调整权重 p 值的大小到合适数值，使得功能短语的向量表示呈现理想结果.

5 作动器功能聚类

基于作动器功能短语向量之间的向量距离，我们可以对每个作动器的功能列表进行聚类. 如图 4 所示，作动器功能聚类的主要流程分为三个步骤：首先用聚类方法进行初步聚类，得到初步聚类结果；然后根据单词之间的相对极性指标获得初步聚类结果中短语之间的相对极性指标；最后以短语之间相对极性指标为依据，用图染色算法将初步聚类结果进行子类拆分，完成最终的聚类. 其中，第二步和第三步是对初步聚类结果的进一步优化.

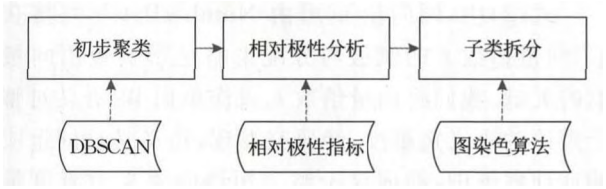

图 4 作动器功能聚类流程

在初步聚类之前，我们需要对作动器功能短语作预处理. 为了减少信息干扰，在预处理时需要忽略停用词(stop word). 停用词是指在英语用语习惯中出现较为频繁和普遍，而实际含义作用较弱，只是维持语法结构的作用的一类词，包含冠词(如“the”，“a”)、介词(如“of”，“for”)、代词(如“that”，“this”)和使役动词(如“let”，“make”)等. 因此对于停用词，我们在预处理过程中将其忽略.

预处理过程中，还需要额外注意负极性词(negative polarity items)，它们的出现会将语义变为否定含义，极性倾向也会发生逆转，例如“not”、“stop”等. 因此在相对极性判别之前，我们需要额外注意句式中是否出现负极性词汇，如果出现，则需要将整个句子的极性取反. 例如，“air conditioning”设备的功能短语中包含“removing moisture from the air”，因为“remove”表达的是“去除”、“消除”之意，后面无论接上任何名词，都代表该名词被去除，不再含有该名词本身的效果，因而判定“removing”为负极性词汇.

5.1 功能短语初步聚类与效果分析

由于在知识图谱中的知识来源于多种数据源，导致同一种语义存在多种不同的描述，作动器功能列表内对于作动器功能的描述有着大量的重复和冗余. 因此我们将同种作动器下的功能列表进行聚类，按照语义相似度关系将功能知识短语拆分为若干簇，使得同一簇中功能短语的语义相似度高，不同簇之间功能短语的语义相似度低，尽可能使得每一个簇中的所有功能短语表达该作动器的同一项功能作用.

我们选择余弦距离作为度量语义相似度的指标. 余弦距离是一个大小在 0 至 1 之间的值，用来衡量两个向量之间的相似程度，其大小等于 1 减去这两个向量之间余弦相似度. 因此，余弦距离越接近 1，表明这两个向量越相似；反之，若余弦距离越接近 0，表示这两个向量越倾向于无关.

在聚类的方法上，我们选择了 DBSCAN(Density-Based Spatial Clustering of Applications with Noise，具有噪声的基于密度的聚类方法)[22]，DBSCAN 适用于支持加速区域查询的数据库，对数据库中点的排序不敏感，可用于高密度的连通区域的划分. 同时拥有着不需要指定类别数量、对异常值具有较强鲁棒性和易于寻找任意形状集群的特点，非常适合功能知识条目的聚类任务.

由 DBSCAN 得到初步聚类结果，以空调“air conditioning”为例，在表 1 中展示.

表 1 “air conditioning”的功能知识初步聚类结果

簇编号	功能知识条目
1	cooling a room, cool down, cool the environment, cool your home, cool a house, cooling, cooling an office, cooling a hotel, keep the room cold, cooling an automobile, manipulating the temperature, reducing the temperature, removing the moisture from the air, storing cold food items, warming, cooling a residence, cool a room or building
2	altering the environment of a room
3	be comfortable in hot weather
4	breathe better

观察初步聚类结果，我们发现有将不同功能聚在一类中的错误. 例如空调的作动器功能列表中，功能相反的“warming”和“cooling a room”都在第 1 簇，同时，除湿功能“removing the moisture from

the air”也与降温功能同在一个簇中. 这些错误并非由聚类算法造成的. 事实上，短语间的余弦距离并不能完全区分出相反或冲突功能，例如，“warming”和“cooling a room”余弦距离接近于 1. 进而根据 NumberBatch 词向量分别计算语义相似度，发现作为反义词的“warm”和“cool”的语义相似度高达 0.491，非常接近于同义词之间的语义相似度(0.5 及以上)，而无关词汇之间语义相似度都低于 0.3. 除湿功能与降温功能聚为一类，是因为“temperature”和“moisture”两者的余弦距离高达 0.582. 这样的现象是词向量本身的特点所导致的.

因为词向量是基于自然语言数据的分布属性来量化和分类不同的单词，单词的分布取决于单词之间是否经常出现在相似的语言环境中. 因此，两个词向量的余弦距离较大，表达的确切涵义是两个单词使用场景比较相似，两者之间可能是同义词(如“cold”和“cool”)，也可能为反义词(如“warm”和“cold”)，或者是同领域的关系(如“temperature”和“moisture”)等. 因此，词汇之间的余弦距离大并不代表他们是近义或反义词，只是使用场景很接近，我们称为相关. 功能聚类的初步结果是将相似度高的词汇聚在了一起.

5.2　相对极性分析

初步聚类的结果对作动器功能区分的粒度过粗，将很多相反或相近类型的功能混淆为一类功能. 我们需要将相似词汇中的相反与相近关系区分出来，并拆分为不同类，以最大化地保证同一类中的作动器功能相近. 因此，本节引入了“相对极性”指标. 相对极性指标表达的是两个短语或词汇极性之间的相互关系，取值为三种离散数值：-1 代表词汇或短语之间具有相反含义，1 代表词汇或短语之间具有相同含义，0 代表词汇或短语之间的含义既不相同也不相反.

在隐式冲突判断的研究场景中，我们需要重点解决相对极性为 -1 的情形，比如“warming”和“cooling a residence”，以及不同领域词汇“temperature”和“moisture”. 对同一簇下的功能知识条目依照相对极性指标进行拆分，将相对极性指标为 -1 的功能短语拆分到不同子类中，得到最终的功能簇列表.

5.2.1　词之间的相对极性

定义 5.　$p(A,B)$ 为单词 A 和单词 B 的相对极性指标，并满足 $p(A,B)\in\{-1,0,1\}$.

$p(A,B)$ 的具体取值需要参考近义词典和反义词典. 在实验过程中，我们整合了 WordNet 和 ConceptNet 中的近义和反义关系，并生成了相应的近义、反义词典. 具体做法为：在 WordNet 中，对目标单词的所有语义的同义词和反义词进行整合；在 ConceptNet 中，我们将 Synonym 关系中的边权重大于 0.5 的英文单词作为目标单词的同义词，将 Antonym 关系中的边权重大于 0.5 的单词作为目标单词的反义词.

由于 WordNet 和 ConceptNet 中，目标单词的近义、反义关系中的单词仅为了释义说明，并未包含所有相关的近反义词词汇. 因此我们需要对单词的近反义关系进行拓展. 由定义 5 我们不难发现：

$$p(A,C)\approx p(A,B)\cdot p(B,C) \tag{3}$$

我们称式(3)为相对极性的扩展性. 单词 A 和单词 C 的相对极性指标值，近似于单词 A、B 之间和单词 B、C 之间的相对极性指标乘积. 理论上，按式(3)能对近反义关系进行任意多步扩展. 然而事实上，由于单词之间的近义反义关系是指目标单词之间某个语义下的具有相近或相反的含义，而一个单词会往往对应多个语义，因此相对极性的多步扩展下会将偏差累积从而导致更多的错误判断. 实践证明，一步扩展的判断效果最好.

图 5 以无向图的形式展示了单词间的相对极性指标的一步扩展方式：初始的相对极性指标由近反义词典来获取，图 5 中的实线代表从近反义词典中获取的相对极性指标，虚线代表的是一步扩展后获取的相对极性指标. 一步扩展后的边权重可按式(3)的方式计算. 例如单词“warm”和“hot”之间的相对极性指标值，等于“warm”与“cool”、“hot”与“cool”的相对极性指标值的乘积.

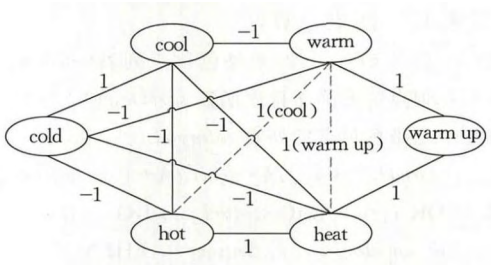

图 5　相对极性指标的一步扩展

5.2.2　短语间的相对极性指标

短语间的相对极性指标计算依赖于两个短语中每个词汇之间的相对极性指标，同时要考虑到两个短语中的负极性词汇的影响. 设短语 $S=\{w_1,w_2,\cdots,w_n\}$，w_i 代表组成该短语的有效词(去除无意义词汇后的单词). 单词 w_i 和 w_i' 之间的相对极性指标为 $p(w_i,w_i')\in\{-1,0,1\}$. 由于在子类拆分任务中，更

关注的是相对极性指标为-1的情况，因此令短语S和S'的各单词之间出现的相对极性指标为-1的次数为$c(S,S')$，计算方法如式(4)所示，其中m和n分别为短语S和S'的有效词总数.

$$c(S,S')=\sum_{i}^{m}\sum_{j}^{n}\frac{|p(w_i,w'_j)|-p(w_i,w'_j)}{2} \tag{4}$$

考虑到短语本身负极性词汇的影响，令短语S和S'的中负极性词汇数量分别为η和η'. 那么两个短语之间的相对极性指标$P(S,S')$的计算方法如式(5)所示.

$$P(S,S')=(-1)^{\eta+\eta'+c(S,S')} \tag{5}$$

由式(5)可知$P(S,S')\in\{-1,1\}$，因此在去除个别异常值(去除无意义词后短语为空、与相反极性词的相对极性指标均为1等异常情况)后，短语列表的子类拆分可转化为一个二分类问题.

5.3 短语列表的子类拆分

短语列表的子类拆分主要目标为将同一簇中相对极性指标为-1的功能知识短语拆分到不同子类别中、相对极性指标为1的功能知识短语在同一子类别中，确保把不同极性的功能知识短语尽可能分在不同类别中.

为了实现子类拆分，我们提出了一种图染色算法予以实现. 在无向图中，令每一个节点代表一个功能短语，相连的两个节点代表两条短语的相对极性指标为-1，因此相连的节点用不同的颜色进行染色. 由于相对极性指标本身特点，一个词无法同时分别成为一对反义词的反义词，因此有边的顶点用两种颜色就能区分，无边的顶点(异常值)用第三种颜色标记. 图染色算法的步骤如算法1所示.

算法1. 图染色算法.

输入：色号$k\in0,1,2$，未染色顶点列表*uncolorList*，按顶点的度降序排序函数*sortDesc*()

输出：已染色顶点的列表*coloredList*

```
coloredList←[ ], k←0, listA←[ ], listB←[ ]
FOR i=0→len(uncolorList) DO
  IF uncolorList[i].degree=0 THEN
    uncolorList[i].color←k
    listA.append(uncolorList[i])
  ELSE
  uncolorList[i].flag←0
  listB.append(uncolorList[i])
k←1, listB←sortDesc(listB)
FOR i=0→len(listB) DO
  IF listB[i].flag=0 THEN
    listB[i].color←k
    listB[i].flag←1
    FOR j=0→len(listB[i].unAdjacent) DO
      IF listB[i].unAdjacent[j].color = undefined THEN
        listB[i].unAdjacent[j].color←k
    k←2
  ELSE
    FOR j=0→len(listB[i].adjacent) DO
      IF listB[i].color=listB[i].adjacent[j].color THEN
        IF listB[i].adjacent[j].flag=0 THEN
          listB[i].adjacent[j].color.clear()
        ELSE listB[i].color←k
    listB[i].flag←1
coloredList←[listA,listB]
RETURN coloredList
```

图6展示了一个图染色的例子. 色号0，1和2分别对应斜线色，竖线色和格子色. 图中的顶点集为$\{A、B、C、D、E、F、G\}$，每个顶点代表一个功能知识短语，顶点的度为该点连接边的数量. 按照算法1，子类拆分步骤如下：

(1) 首先遍历整个顶点集，将度为0的顶点F和顶点G染为0号色并踢出待染色队列，将剩余待染色队列按照顶点的度降序排列，更新后的待染色队列为$\{B、D、A、E、C\}$.

(2) 取队首顶点B，检查此点发现未染色，则将此点染为1号色并剔除染色队列. 然后遍历待染色队列，将与此点不连通且未染色的点$\{A、C\}$同样染为1号色；

(3) 取染色队列顶点D，检查此点发现并未染色，则将此点染色为2号色并剔除待染色队列，再将待染色队列中与此点不连通且未染色的点$\{E\}$，染

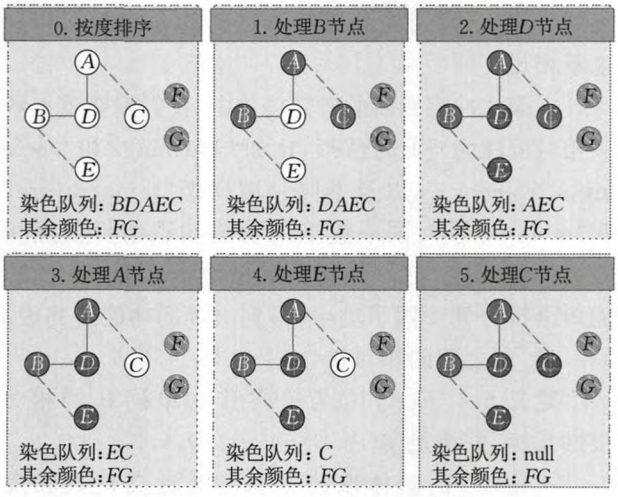

图6 图染色算法

色为 2 号色；

(4) 取队首顶点 A，发现 A 已经染色，则检查与 A 相连的点是否同色，发现 C 与 A 同色，但 C 还在染色队列，因此将 C 点颜色去除清空.

(5) 以此类推，处理 E 节点和 C 节点，最终将 C 节点染为 2 号色. 当染色队列清空后，图染色算法结束.

以“air conditioning”为例，通过图染色算法进行极性判别和子类拆分后，聚类结果如表 2 所示. 与表 1 相比，表 2 中新增了 3 类，分别是“warming”单独一类，“reducing the temperature”和“manipulating the temperature”一类，以及“removing the moisture from the air”一类. 这样将空调的“制热”、“调节温度”和“除湿”三项功能分离出来，符合我们子类拆分的目标.

表 2　“air conditioning”的子类拆分后聚类结果

簇编号	功能知识条目
1	warming
2	breathe better
3	altering the environment of a room
4	reducing the temperature, manipulating the temperature
5	be comfortable in hot weather
6	removing the moisture from the air
7	cooling a room, cool down, cool the environment, cool your home, cool a house, cooling, cooling an automobile, cooling a hotel, cool a room or building, cooling an office, cooling a residence, keep the room cold, storing cold food items

6　隐式冲突矩阵生成

经过功能聚类和子类拆分，实验得到了每一种作动器的功能簇列表. 为了检测作动器之间是否存在隐式冲突，我们设计并实现了隐式冲突矩阵. 如图 7 所示.

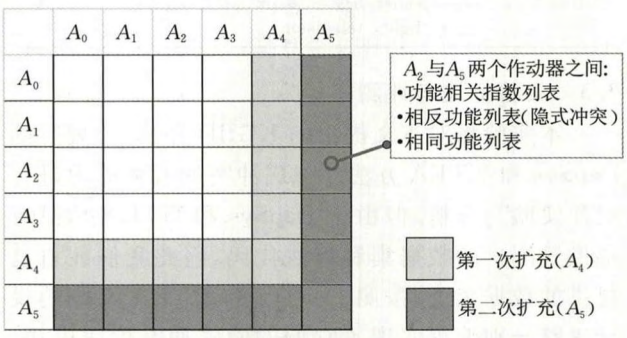

图 7　隐式冲突矩阵

隐式冲突矩阵存储了设备之间存在功能冲突的判断结果，可以提高规则制定环节中的隐式冲突判断效率，其主要特点如下：

(1) 字典式查询映射. 当进行两条规则之间隐式冲突判断时，可以在隐式冲突矩阵中进行字典式查询映射，查询效率为 $O(1)$.

(2) 增量式扩充. 当有新的作动器加入智能家居系统时，我们可以在现有隐式冲突矩阵的基础上进行增量式扩充. 由于新增的作动器设备需要与现有的作动器设备进行两两比较，因此算法效率为 $O(M)$. 其中 M 为当前已有的作动器类别数量.

本文中，我们将隐式冲突矩阵中的各个节点定义为两个作动器之间的相关性信息. 相关性信息包含三个列表，分别是功能相关指数列表、相反功能列表和相同功能列表. 三种列表信息的计算方式如算法 2 所示.

算法 2.　相关性信息计算.

输入：相关指数阈值 θ，ab 之间相关指数 $CoIndex(a,b)$、相对极性 $Rep(a,b)$，功能列表 $functionList1$，$functionList2$

输出：功能相关性信息 CI

```
FUNCTION CorrelationalInfo(funcList1, funcList2)
  FOR i=0→len(funcList1) DO
    FOR j=0→len(funcList2) DO
      CIL[i][j]←CoIndex(funcList1[i],
                        funcList2[j])
      IF CIL[i][j]>θ THEN
        list←[i,j]
        IF Rep(funcList1[i],funcList2[j]) is -l
        THEN
          OFL.append(list)
        ELSE
          SFL.append(list)
  CI←[CIL,OFL,SFL]
  RETURN CI
```

由上述算法可以得到两个作动器之间的冲突列表. 判断两条规则是否存在隐式冲突，我们可以用冲突映射的方式去查询. 完整流程为，在触发条件判断一致之后，提取两条规则 rule1 和 rule2 的作动器类别名称 type1 和 type2，并匹配到知识图谱的对应功能 func1 和 func2，再映射到隐式冲突矩阵中的对应节点 CI(type1, type2)，查询两个对应作动器的相反功能列表 OFL，如果列表中存在这组功能(func1, func2)，就判断这两条规则 rule1 和 rule2 之间存在隐式冲突，否则就判断不存在隐式冲突.

7　实验与评估

7.1　数据集的配置

实验所需要用到的数据集包含四部分，分别是

规则集、知识图谱、词向量和语义字典. 本实验中各数据集的配置如表 3 所示.

表 3 数据集的配置

数据集	选用版本
知识图谱	ConceptNet 5.0
词向量	NumberBatch 17.06
语义字典	WordNet 2.1
规则集	IFTTT

其中,知识图谱、词向量与语义字典都能在线获取,而规则集需要进行人工收集.

IFTTT 是一个知名的云端规则定制平台,具有丰富的智能家居任务规则. IFTTT 中的规则是广义上的 Trigger-Action 形式. 通过对其规则结构的分析,我们提取其中 triggerChannelTitle(触发设备名称)、triggerId(触发条件 ID)、actionChannelTitle(作动器名称)和 actionDesc(动作描述)作为规则集的输入. 实验中,我们总共获取了 11 859 条规则,包含 22 种作动器共 89 种设备型号.

7.2 规则集的预处理

为了模拟真实的用户住户环境,我们需要对智能家居的设备与规则在住所房间内进行配置. House Plans① 网站上提供了大量真实的户型平面设计图,本实验选取了其中 10 种常见的家庭住宅户型,并按照这些户型平面图规划房间功能区域,为设备注册与规则配置做准备.

每个作动器不可能配置全部规则,因此我们根据该房间作动器数量的多少,合理分配规则数量,且单个作动器的规则不能超过 20 条,完成作动器规则设定后,我们整理得到如表 4 所示数据. 表 4 中主要的属性分为户型编号、户型名称、房间数、规则总数和规则组合数,详细介绍如下:

(1)户型编号和户型名称. 为了实验标注方便,我们将 10 个户型按照首字母的顺序进行排序并编号,然后分别进行实验.

(2)规则总数. 规则总数是一个户型中用户配置的规则总量. 设其中一个户型中有 n 个房间,编号依次是 $1,2,3,\cdots,n$. 设第 i 个房间所对应的规则数量为 R_i,该户型内规则总数 SR 如式(6)所示:

$$SR=\sum_{i=1}^{n}R_i \tag{6}$$

(3)规则组合数. 在判断隐式冲突的过程中,需要将规则之间进行两两比较,因此我们将规则之间两两配对组合的总数称为规则组合数. 由于在单个户型内,不同的房间之间是不连通的,因此作用在不同房间内作动器的规则我们认为是互不干扰的. 一个户型内的规则组合数为各个户型中所有房间内的规则组合数之和 CN,如式(7)所示:

$$CN=\sum_{i=1}^{n}C_{R_i}^{2} \tag{7}$$

表 4 各户型中规则配置情况

户型编号	户型名称	房间数	规则总数	规则组合数
1	Apartment	5	563	38842
2	Bungalow House	10	1040	62946
3	Contemporary House	12	1185	68461
4	Cottage House	10	953	52317
5	Country House	13	1264	75658
6	Craftsman	15	1458	85779
7	Farmhouse	17	1523	81229
8	Modern House	12	1437	79423
9	Ranch House	13	1229	70158
10	Traditional House	12	1207	76807
—	总计	119	11859	691620

在各个户型中,不同的房间区域内需要配置相应的作动器. 由于实验应符合真实住户的家居环境,一个房间不可能将所有作动器都配置在内,且每个房间出于其本身的职能关系会拥有不同的作动器配置. 因此本实验选取了智能家居领域中常见的 22 种作动器产品,按照作动器功能与房间的适配性将这 22 种作动器配置在每个户型的各个房间中,表 5 展示的是 Apartment 户型的作动器配置表.

表 5 Apartment 中房间内作动器配备方案示例

房间名称	作动器设备	规则数
Living Room	air conditioning, air purifier, speaker, camera, dehumidifier, humidifier, window, door, light, television, water heater	167
Kitchen	refrigerator, alarm, coffeemaker, cooker, oven, window, door, light	115
Porch	window, door, light	60
Bathroom	door, dryer, washer	43
Bedroom	air conditioning, air purifier, humidifier, dehumidifier, heater, alarm, window, printer, door, light, television	178

7.3 实验标注与评测指标

本实验将对本文提出的 KGIID 方法,与现有的 DepSys 和 UTEA 方法对隐式冲突的检测能力进行对比实验与分析. 但由于 DepSys 和 UTEA 均没有提供其测试的数据集和算法代码,因此我们在自己提供的数据集上,按照 DepSys 和 UTEA 系统的设计思路分别编程实现了它们的隐式冲突检测模块,从而使得 DepSys、UTEA 和 KGIID 能在相同的实验环境下进行隐式冲突检测性能的评估.

① House Plans. https://www.houseplans.com 2018-05-20

实验设置中，对于触发条件的逻辑判断统一采用 TriggerChannelId 与 TriggerChannelTitle 相结合的方式，三种方法的差异主要体现在对于动作 action 的判断与检测上. 按照 DepSys 与 UTEA 的实验要求，由 3 名软件工程师对作动器 ActionList 文件中各个 action 对环境变量的影响进行人工标注，另安排 1 人负责校对标注结果. 标注方式如下：

(1) DepSys. 标注对于环境变量的影响趋势.

(2) UTEA. 标注对于环境影响的目标范围.

(3) KGIID. 不需要人工标注.

三者之间人工标注情况的对比如表 6 所示. 相比 DepSys，UTEA 的标注过程更加精细，有时还要根据作动器的不同模式来确定变化目标区间范围.

表 6　DepSys、UTEA 与 KGIID 的人工标注情况

人工标注	DepSys[9]	UTEA[10]	KGIID
环境变量名称	√	√	—
变化趋势	√	—	—
变化目标范围	—	√	—
标注复杂程度	中等	复杂	无

对于冲突检测，我们选择查全率(recall rate)与查准率(precision rate)以及综合指标 F-score 作为评价三种方法的指标. 查准率可以衡量检测方法输出结果的正确率，即是否存在误判；查全率可以反应方法对于识别所有冲突关系的全面程度，即是否存在漏判；F-score 结合了查全率与查准率两项指标，综合评定方法的性能.

7.4　实验结果与分析

根据表 4 中 10 个户型的规则配置状况，结合人工核验的方式，我们可以得到 10 个户型中规则显式冲突和隐式冲突的数量，如图 8 所示. 由图 8 可知，隐式冲突占所有冲突数量的比例相对较高，总体达到了 25.5%.

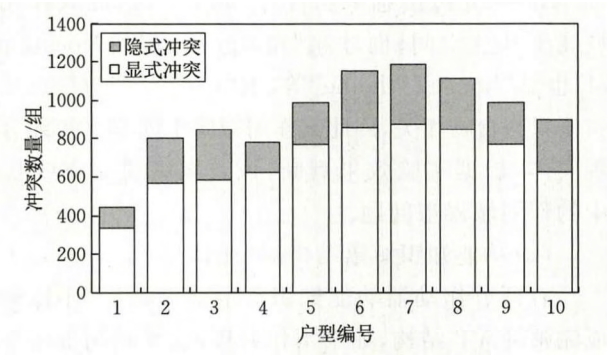

图 8　10 个户型中隐式冲突和显式冲突数量

针对这 10 个户型的作动器和规则配置，我们分别用 DepSys、UTEA 和 KGIID 方法进行独立实验，其结果如图 9 所示.

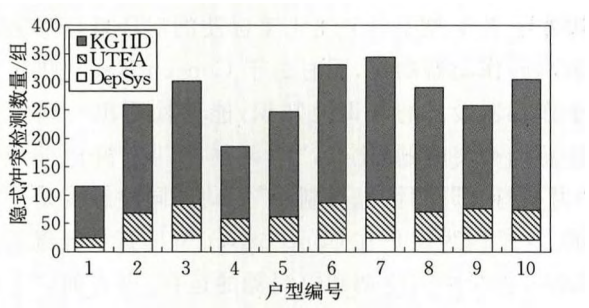

图 9　三种方法的隐式冲突检测数量对比

综合所有户型中的数据信息，我们可以得到 DepSys、UTEA 和 KGIID 三种方法各项检测指标结果，见表 7. 其中，测准数是指对应方法判断正确的隐式冲突数量，误判数是对应方法判断错误的隐式冲突数量，漏判数是对应方法判断时遗漏的隐式冲突数量.

表 7　DepSys、UTEA 与 KGIID 的冲突检测指标

对比指标	测准数	误判数	漏判数	查全率	查准率	F-score
DepSys[9]	200	0	2143	0.0854	1	0.1574
UTEA[10]	494	0	1849	0.2108	1	0.3482
KGIID	1945	64	398	0.8301	0.9681	0.8938

查全率、查准率与 F-score 的指标对比情况如图 10 所示.

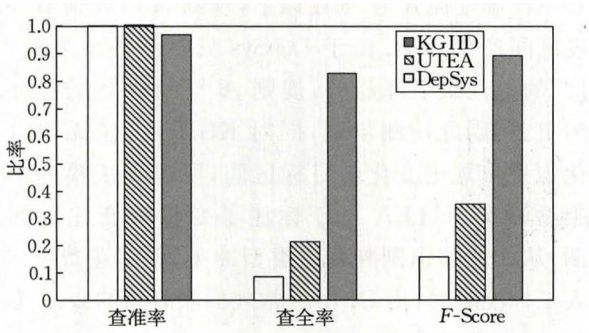

图 10　三种方法的查准率、查全率和 F-score 指标对比

通过图 9、表 7 与图 10 可得知，KGIID 的冲突检测数量和比例明显高于 UTEA 与 DepSys. 其中查全率方面，KGIID 的查全率为 0.8301，比 DepSys 高出了 74.47%，比 UTEA 高出了 61.93%；而查准率方面，KGIID 的查准率为 0.9681，比 UTEA 与 DepSys 都低了 3.19%. 综合查全率和查准率两项指标，KGIID 的 F-score 值为 0.8938，分别比 DepSys 和 UTEA 高出了 0.7364 和 0.5456.

参考实验结果，我们进行了漏判与误判的对比分析.

7.4.1　漏判分析

KGIID 比两种方法具有更少的冲突漏判数，原

因在于其不仅能检测出人工标注的对环境变量造成影响的作动器动作，同时由于 ConceptNet 提供了关于作动器设备的常识性知识，能够检测出一些不易注意的冲突问题. 比如，“空调制冷”与“窗户打开”，“窗户打开”和“恒温器制冷”等动作同时触发是冲突的. 这是因为窗户在 ConceptNet 里具有空气流通的知识，会影响到空调和恒温器的运作. 再比如，“空调制冷”与“加湿器打开”同时触发也是冲突的，原因在于空调具有“除湿”的作用，与加湿器作用相反，会使得加湿器的加湿效果受到限制. 这类常识性知识在人工标注过程中往往容易被遗漏，导致漏判的发生.

然而即便如此，KGIID 仍然有 17％的漏判率. 经过对实验数据的查看检验，我们发现这 17％的漏判的规则，KGIID 与 DepSys 两者都同时漏判了，只有 UTEA 检测出来. 原因在于类似恒温器这一类作动器，会对温度变化的目标范围具有严格要求，如果目标范围不一致也会导致冲突的问题. 例如，规则 A：“如果室内温度小于 20 摄氏度，则打开恒温器并保持室内温度在 24 度至 26 度区间”；规则 B：“如果室内温度小于 20 摄氏度，则打开暖气至最大功率.”按照规则 B 对应作动器的设置，最大功率的暖气会将室内温度提升至 28 度以上，规则 A 与规则 B 在温度区间产生矛盾. 由于 DepSys 只标注环境变量“温度”的上升或下降趋势，规则 A 与规则 B 的温度均为上升，因此检测不到，同时 KGIID 方法无法自动化识别出规则变化的目标区间，所以无法检测出该类型冲突. UTEA 由于标注了变量的变化区间范围，从而能够识别判断此类型冲突，但需花费更多的人工标注时间，是 DepSys 常规标注时间的 2～3 倍.

7.4.2 误判分析

KGIID 中总共存在 64 组误报，原因在于部分冲突的判断确认需要依据作动器当前运行状态进行，而无法仅仅依靠规则内容进行静态判断. 例如，在相同触发条件下，“空调打开”与“暖气打开”这两条规则之间无法确定是否存在隐式冲突. 其原因在于：空调打开时，我们并不清楚此时运行模式处于制冷模式还是制热模式，而只有当前空调处于制冷模式才会与暖气构成隐式冲突.

在 KGIID 方法中，当无法确定作动器动作所映射的功能时，将对该作动器所有功能都进行相互判断. 由于空调中含有与暖气产生功能冲突的功能簇，因而判断为存在隐式冲突，导致了误判. 相比之下，由于“空调打开”动作对于环境变量的影响不确定（可能是制冷或者制热），因而 DepSys 与 UTEA 均不对该规则进行标注，避免了误判.

综合来看，KGIID 相比 DepSys 和 UTEA，具有更高的查全率和 F-score 值，可以适应更多的生产生活环境，但是在查准率方面有一定缺陷，在一些对于查准率要求比较高的特殊场合下，KGIID 有待进一步提高和完善.

7.5 实验细节与讨论

（1）作用域误判问题

作用域是指因为作动器的功能影响区域. 作动器的作用域大体分为三类：作用于开放空间、作用于自身密闭空间和作用于客观实体，后两者统称为作用于非开放空间. 即使在短语向量相对极性指标为负的情况下，如果其中任何一个作动器的作用域不属于开放空间，则仍然不会发生冲突. 作用域误报问题就是指隐式冲突判断方法在识别作用域问题不准确从而导致的误报现象. 例如，“冰箱”与“加热器”在 KGIID 方法中容易被判断为隐式冲突. 原因在于冰箱中有“cooling food”的功能子类，而加热器有“heat the house”的功能子类，在短语向量的判定中被视作极性相反. 加热器作用于一个开放空间，而冰箱仅仅作用在它自身内部空间或者某食物，属于非开放空间，因此它们的作用范围并不相同，这样的冲突判断属于误判.

KGIID 采用了基于宾语分类的检测方式来解决这个问题. 通过观察 ConceptNet 的作动器知识我们发现，作动器的“used for”和“location of”（作动器位于某个位置）边信息中，如果宾语出现相同词汇，则该作动器作用域为开放空间，否则将作用域视为非开放空间. 例如，加热器“heater”，其“used for”和“location of”边信息中均含有“house”，因此其作用域属于开放空间；而冰箱“refrigerator”的“location of”也是“house”、“home”等，但“used for”边信息中宾语只与食物相关，因此其作用域属于内部空间或客观实体. 根据实验效果判断，该方案解决了 KGIID 中的作用域误报问题.

（2）功能知识短语结构缺失问题

有部分作动器功能知识短语成分缺失，不能构成完整的 VP 结构，如果不作处理，会影响词性标注和词义识别的准确性，从而有造成误判的风险. 例如，“lock”有一项功能知识短语只有一个单词“closing”. 依据 WordNet 单独对“closing”进行词性标注和词义识别，那么有可能会造成词性标注和词义识别的

错误（“closing”本身有名词和形容词属性，含义各不相同）.

为了解决此问题，我们将缺失成分的作动器功能知识短语采用“… is used for …”的形式进行补全. 上述例子中，我们将原功能知识短语补全为“lock is used for closing”，这样结合标注，WordNet就能识别出“closing”为动词的现在进行时态，从而避免错误.

8 总　结

智能家居场景联动为住户提供了丰富多样的智能服务，检测并规避控制冲突是智能家居系统安全性的重要保障.

本文提出了一种基于知识图谱的隐式冲突检测方法 KGIID，引入知识图谱将常识性的知识与作动器的功能相结合，通过设备注册与规则制定两个环节，借助词向量、语义字典等工具，实现了规则之间隐式冲突的自动判别. 通过利用 IFTTT 数据集对 10 个典型户型的测试，KGIID 的查全率达到 0.8301，查准率达到 0.9681，F-score 值为 0.8938.

与现有方法相比，KGIID 提高了隐式冲突判别的自动化程度并减少了隐式冲突判别方法的漏判率，在综合性能上更具优势. 但在一些对误判容忍程度较低的特殊场合下，KGIID 在查准率方面仍有改进的空间. 同时，我们后续也将进一步开展基于中文知识图谱的隐式冲突判别技术的研究.

参　考　文　献

[1] Stankovic J A. Research directions for the Internet of Things. IEEE Internet of Things Journal, 2014, 1(1): 3-9

[2] Stojkoska B L R, Trivodaliev K V. A review of Internet of Things for smart home: Challenges and solutions. Journal of Cleaner Production, 2017, 140(1): 1454-1464

[3] Meng Z, Lu J. A rule-based service customization strategy for smart home context-aware automation. IEEE Transactions on Mobile Computing, 2016, 15(3): 558-571

[4] Ovadia S. Automate the Internet with “if this then that” (FTTT). Behavioral & Social Sciences Librarian, 2014, 33(4): 208-211

[5] Shehata M, Eberlein A, Fapojuwo A. Using semi-formal methods for detecting interactions among smart homes policies. Science of Computer Programming, 2007, 67(2): 125-161

[6] Robinson W N, Pawlowski S D, Volkov V. Requirements interaction management. ACM Computing Surveys, 2003, 35(2): 132-190

[7] Shehata M, Eberlein A, Fapojuwo A O. A taxonomy for identifying requirement interactions in software systems. Computer Networks, 2007, 51(2): 398-425

[8] Hu H, Yang D, Fu L, et al. Semantic Web-based policy interaction detection method with rules in smart home for detecting interactions among user policies. IET Communications, 2011, 5(17): 2451-2460

[9] Munir S, Stankovic J A. DepSys: Dependency aware integration of cyber-physical systems for smart homes//Proceedings of the ACM/IEEE 5th International Conference on Cyber-Physical Systems (with CPS Week 2014). Berlin, Germany, 2014: 127-138

[10] Sun Y, Wang X, Luo H, et al. Conflict detection scheme based on formal rule model for smart building systems. IEEE Transactions on Human-Machine Systems, 2015, 45(2): 215-227

[11] Shi Wei-Song, Sun Hui, Cao Jie, et al. Edge computing: An emerging computing model for the internet of everything era. Journal of Computer Research and Development, 2017, 54(5): 907-924(in Chinese)
（施巍松，孙辉，曹杰等. 边缘计算：万物互联时代新型计算模型. 计算机研究与发展，2017，54(5)：907-924）

[12] Bollacker K, Evans C, Paritosh P, et al. Freebase: A collaboratively created graph database for structuring human knowledge//Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data. New York, USA, 2008: 1247-1250

[13] Vrandečić D, Krötzsch M. Wikidata: A free collaborative knowledgebase. Communications of the ACM, 2014, 57(10): 78-85

[14] Mendes P N, Jakob M, Bizer C. DBpedia: A multilingual cross-domain knowledge base//Proceedings of the Language Resources and Evaluation Conference. Istanbul, Turkey, 2012: 1813-1817

[15] Liu H, Singh P. ConceptNet: A practical commonsense reasoning tool-kit. BT Technology Journal, 2004, 22(4): 211-226

[16] Korenius T, Laurikkala J, Järvelin K, et al. Stemming and lemmatization in the clustering of Finnish text documents//Proceedings of the 13th ACM International Conference on Information and Knowledge Management. Washington, USA, 2004: 625-633

[17] Robert S, Joshua C. An ensemble method to produce high-quality word embeddings. arXiv preprint arXiv:1604.01692v1, 2016

[18] Robert S, Joshua C, et al. ConceptNet 5.5: An open multilingual graph of general knowledge//Proceedings of the Association for the Advance of Artificial Intelligence (AAAI) 2017. San Francisco, USA, 2017: 4444-4451

[19] Mikolov T, Chen K, Corrado G, et al. Efficient estimation of word representations in vector space//Proceedings of the International Conference on Learning Representations (ICLR). Scottsdale, USA, 2013: 1-12

[20] Pennington J, Socher R, Manning C. GloVe: Global vectors for word representation//Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Doha, Qatar, 2014: 1532-1543

[21] Chen Fu, Lin Chuang, Xue Chao, et al. Vector semantic computing method study for short sentence. Journal on Communications, 2016, 37(2): 11-19(in Chinese)
(陈福, 林闯, 薛超等. 短句语义向量计算方法. 通信学报, 2016, 37(2): 11-19)

[22] Ester M, Kriegel H P, Sander J, et al. A density-based algorithm for discovering clusters in large spatial databases with noise//Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining. Portland, USA, 1996: 226-331

XIAO Ding, Ph. D. candidate. His research interests include Internet of Things and artificial intelligence, etc.

WANG Qian-Yu, M. S. candidate. His research interests include Internet of Things and embedded systems, etc.

CAI Ming, Ph. D., associate professor. His research interests include Internet of Things, artificial intelligence, embedded systems, etc.

LI Xiu, M. S. Her research interests include Internet of Things and embedded systems.

Background

This paper is related to interference detection in smart home systems. With the popularization of smart home and the continuous development of industry, interferences in smart home systems have received more attention from people. There are many solutions for interferences, such as the eventual consistency principle, priority control, trigger judgment, and manual labeling. These solutions systematically solve the interference problems when different operation requests apply to the same actuators. Unfortunately, with regard to the interference between different actuators, current solutions have two major drawbacks: low degree of automation and high missed rate of detection.

Therefore, this paper proposes a interference detection method called KGIID, which is especially aimed at detecting the interference between different actuators. KGIID has two main phases, which are the actuator registration phase and rule subscription phase. During the actuator registration phase, KGIID introduces the knowledge graph which collects commonsense knowledge to evaluate the effects caused by the actuator, judge the interference of effects automatically and at last store the results as the implicit interference matrix at the edge; While in the rule subscription phase, KGIID matches and queries the interference of effects between actuators recorded in the implicit interference matrix to help judge the interferences from rules. By introducing the knowledge graph, we incorporate commonsense knowledge into judgement conditions. Thus we can save lots of manual work costs and reduce the missed rate of detection. At last we conduct the experiment compared with two typical interference detection systems, DepSys and UTEA. The experiment result shows the recall rate of KGIID reaches 0.8301, the precision rate reaches 0.9681 and the *F*-score is 0.8938. It proves the effectiveness of KGIID in strengthening automation issues and reducing the missed rate.

This work is mainly supported by the National Natural Science Foundation of China Nos. 51775496, 51875503.