软件学报 ISSN 1000-9825, CODEN RUXUEW
*Journal of Software*,2021,32(12):3751−3767 [doi: 10.13328/j.cnki.jos.006102]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

# 基于 GAT2VEC 的 Web 服务分类方法*

肖 勇[1,2], 刘建勋[1,2], 胡 蓉[1,2], 曹步清[1,2], 曹应成[1,2]

[1](服务计算与软件服务新技术湖南省重点实验室(湖南科技大学),湖南 湘潭 411201)

[2](湖南科技大学 计算机科学与工程学院,湖南 湘潭 411201)

通讯作者: 刘建勋, E-mail:ljx529@gmail.com

**摘 要**: 随着SOA技术的发展,Web服务被广泛应用,服务数量增长迅速.正确高效地对Web服务进行分类,对于提高服务发现质量、促进服务组合效率非常重要.然而,现有的Web服务分类技术存在描述文本稀疏、未充分考虑属性信息以及结构关系等问题,难以有效提升Web服务分类的精度.针对此问题,提出一种基于GAT2VEC的Web服务分类方法.首先,针对Web服务之间的结构关系和自身的属性信息分别构建出多个相对应的结构关系图和属性二分图,并采用随机游走算法生成Web服务的结构上下文和属性上下文;然后,利用SkipGram模型对联合上下文进行训练,得到融合多维信息的表征向量;最后,采用SVM模型实现Web服务的分类预测.在ProgrammableWeb真实数据集上进行对比实验,实验结果表明:相比于Doc2vec,LDA,Deepwalk,Node2vec和TriDNR这5种方法,所提出的方法在Macro *F*1值上有了135.3%,60.3%,12.4%,10.5%和4.3%的提升,切实提高了服务分类的精度.

**关键词**: Web服务分类;GAT2VEC模型;随机游走;SVM模型

**中图法分类号**: TP311

中文引用格式: 肖勇,刘建勋,胡蓉,曹步清,曹应成.基于 GAT2VEC 的 Web 服务分类方法.软件学报,2021,32(12):3751−3767. http://www.jos.org.cn/1000-9825/6102.htm

英文引用格式: Xiao Y, Liu JX, Hu R, Cao BQ, Cao YC. GAT2VEC-based Web service classification method. Ruan Jian Xue Bao/Journal of Software, 2021,32(12):3751−3767 (in Chinese). http://www.jos.org.cn/1000-9825/6102.htm

## GAT2VEC-based Web Service Classification Method

XIAO Yong[1,2], LIU Jian-Xun[1,2], HU Rong[1,2], CAO Bu-Qing[1,2], CAO Ying-Cheng[1,2]

[1](Hunan Key Laboratory for Services Computing and Novel Software Technology (Hunan University of Science and Technology), Xiangtan 411201, China)

[2](School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan 411201, China)

**Abstract**: With the development of SOA technology, Web service is widely used and the number of services is growing rapidly. It is very important to classify Web service correctly and efficiently to improve the quality of service discovery and promote the efficiency of service composition. However, the existing Web service classification technologies have some problems, such as sparse description text, insufficient consideration of attribute information, and structural relationship. Therefore, it is difficult to effectively improve the accuracy of Web service classification. In order to solve this problem, this study proposes a GAT2VEC-based Web service classification method. Firstly, according to the structural relationship between Web services and their own attribute information, several corresponding structural diagrams and attribute bipartite diagrams are constructed respectively, and the random walk algorithm is used to generate the structural context and attribute context of Web services. Then, the SkipGram model is used to train the joint context to obtain the word vector which merges the multidimensional information. Finally, the SVM model is used to perform the classification and prediction of Web services.

* 基金项目: 国家自然科学基金(61872139, 61873316, 61702181); 湖南省自然科学基金(2018YFB1402800-04, 2018JJ2139, 2018JJ2136, 2018JJ3190)

Foundation item: National Natural Science Foundation of China (61872139, 61873316, 61702181); Natural Science Foundation of Hunan Province (2018YFB1402800-04, 2018JJ2139, 2018JJ2136, 2018JJ3190)

收稿时间: 2019-11-21; 修改时间: 2020-03-09; 采用时间: 2020-06-12

The experimental results show that compared with the five methods of Doc2vec, LDA, Deepwalk, Node2Vec, and TriDNR, the proposed method has 135.3%, 60.3%, 12.4%, 10.5%, and 4.3% improvement in Macro *F*1 value, which effectively improves the accuracy of service classification.

**Key words**: Web services classification; GAT2VEC model; random walks; SVM model

Web 服务因其跨语言、跨平台、松散耦合、基于开放式标准等特点,成为 SOA(service-oriented architecture)的主流实现技术.随着 SOA 架构和 Web 服务相关标准的日趋成熟,网络上可用的 Web 服务越来越多.例如:截止到 2020 年 3 月 20 日,ProgrammableWeb 网站上已经发布了 7 961 个 Mashup 和 23 368 个 Web API;而当开发人员希望检索与消息传递相关的 Mashup 时,ProgrammableWeb 的搜索引擎将返回 1 695 个搜索结果.因此,在大量服务中快速、准确地发现和选择所需要的服务,成为服务计算领域的关键问题之一.通常情况下,Web 服务缺少规范的形式化的描述模型,如 Web 服务的描述文本内容过少、描述语言不规范等.前者使得服务缺乏足够有效信息,难以被用户发现;后者使得服务描述随意性较大,可能导致相同的服务描述不一,而不同的服务却描述相似,进一步增加了服务查找和发现的难度[1].目前,该问题已引起了众多研究者的注意[2].其中,如何通过自动服务分类减少服务匹配过程中的候选服务数量,以提高服务查找和服务发现的准确性和效率,已成为了近年来的研究重点.

目前,关于 Web 服务分类的研究主要以基于功能语义的服务分类方法为主.例如:Crosso 等人[3]将 WSDL (Web service description language)中的元素进行分割去除停用词后,归至词根,然后利用不同的分类算法进行分类.Katakis 等人[4]考虑了 Web 服务的文本描述和语义标注,解决了 Web 服务在其应用领域的自动分类问题.但是 WSDL 文档通常包含很少的描述内容,导致这些算法通常无法取得较满意的分类效果.随着机器学习的兴起,文档主题生成模型开始引起了众多研究者的关注.Shi 等人[5]提出了一种考虑多重 Web 服务关系的概率主题模型 MR-LDA,其可对 Web 服务之间相互组合的关系以及 Web 服务之间共享标签的关系进行建模.Cao 等人[6]通过注意力机制将 BiLSTM 局部的隐状态向量和全局的 LDA 主题向量结合起来,提出一种基于主题注意力机制 Bi-LSTM 的 Web 服务分类方法.但是主题模型通常是基于大量的已知观测样本来推测隐含的后验主题分布概率,需要大量的辅助信息.为了进一步利用有限的特征信息挖掘出 Web 服务之间的隐含关系,越来越多的深度学习方法被引入到了服务分类领域.Ye 等人[1]将 Web 服务描述文档中的所有离散特征结合起来,利用 Wide & Bi-LSTM 模型对 Web 服务类别进行预测.Chen 等人[7]利用 LSA 模型对移动应用内容文本进行全局主题建模,再通过 BiLSTM 模型对内容文本进行局部隐藏表征,提出一种主题注意力机制增强的移动应用分类方法.但是这些深度学习的方法在耗费了大量计算资源的前提下,对服务分类准确度的提升并不明显.总的来说,上述的方法与技术虽然提高了 Web 服务分类的精度,但普遍存在以下两个问题.

(1) 尽管考虑到了 Web 服务描述文档通常比较短、语料有限等问题,并提出挖掘描述文档中词语的语序和上下文信息或融合标签等辅助信息的方法,更好地实现了短文本建模,但是这些方法利用的离散特征关联性一般,且始终没有较好地解决文档语义稀疏的问题;

(2) 这些方法基本都依赖于文本描述信息和标签等属性信息,而未考虑 Web 服务之间的结构交互关系.在实际情况中,Web 服务之间存在着丰富的对象和链接.例如:在 ProgrammbleWeb 数据集中,存在两个 Mashup(200 Towns 和#haiku),它们都属于 Photos 类,然而二者的标签和主题描述都不相似,因此很难将二者归为一类.但是这两个 Mashup 在结构上都调用了同一个名为 Twitter 的 API.由此可见,结构交互信息在分类任务中起着相当重要的作用.

网络表征学习(network representation learning,简称 NRL)是最近提出的通过学习网络节点连续、低维的潜在特征来解决稀疏性问题的一种重要方法.这些特征涵盖了网络的全局结构,并可以被提取到后续的机器学习任务中.其中,将 Deepwalk[8]算法应用到网络中提取特征并进行表征,成为一种常用的方法.它通常是先通过短随机游走得到节点序列,然后输入到 SkipGram 模型中,得到最终的嵌入向量.直观地说,邻近的节点往往具有相似的上下文(序列),因此具有彼此相似的嵌入.这一基本思想在后来的若干方面得到了扩展[9,10].近年来,Yang 等人[11]证明了 Deepwalk 等价于邻接矩阵 $M$ 的因式分解,并提出了一种通过分解文本关联矩阵结合节点文本特征

的 TADW(text-associated deep walk)模型.Pan 等人[12]提出的 TriDNR(tri-party deep network representation)同时使用结构、文本和嵌入的部分标签信息来进行网络表征.这些 NRL 方法的出现,使得我们充分考虑 Web 服务之间的结构交互关系,并同时融合结构关系和文本属性的想法成为可能.在此基础上,为了解决传统 Web 服务分类模型所存在的问题,本文提出一种基于 GAT2VEC[13]的 Web 服务分类方法(简称 GWSC).通过从服务网络图中获取包含组合调用等信息的结构上下文和服务自身的属性上下文,GWSC 充分保持了 Web 服务之间结构和属性的邻近度,并使用了单一神经层共同学习这些特征,从而通过融合多个信息源提升了 Web 服务分类的精度.值得一提的是:由于 GWSC 将属性文本信息以拓扑结构化的形式来表示,使得在低维空间中学习到的属性文本信息表征也可以重构出原有的服务关系网络,从而可较好地解决文档语义稀疏的问题.

本文第 1 节具体介绍 GWSC 方法.第 2 节进行实验评估及分析.第 3 节介绍相关工作.第 4 节对本文工作进行总结.

## 1 方法介绍

GWSC 方法框架如图 1 所示.首先,对于每个服务网络图中的顶点,我们通过随机游走得到其相对于其他顶点的结构和属性上下文;然后,将这两种上下文语义信息结合起来,学习出一种既保留结构又保留属性近似值的网络嵌入表征;最后,将得到的表征向量输入到 SVM 模型当中进行分类预测.其方法框架如图 1 所示,具体来说,GWSC 由以下网络生成、随机游走、表征学习和 SVM 分类这 4 个阶段组成.

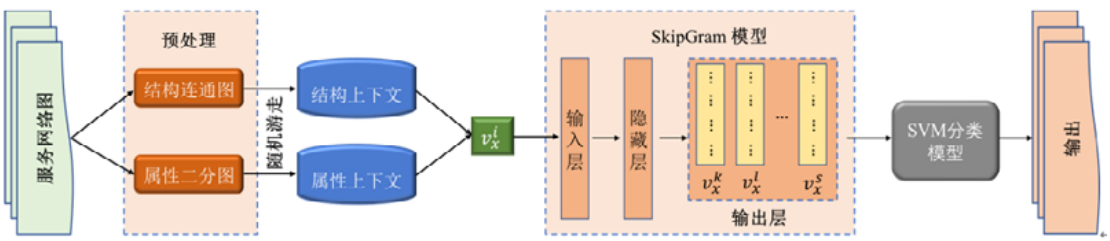



Fig.1 GWSC method framework

图 1 GWSC 方法框架

### 1.1 网络生成

首先,我们考虑一个如图 2 所示的服务网络图 $G=(V,E,A)$,它由一组顶点 $V$、一组边 $E$ 和一个属性函数 $A:V\rightarrow 2^{\mathcal{C}}$ 组成,其中,$\mathcal{C}$是所有可能属性的集合.

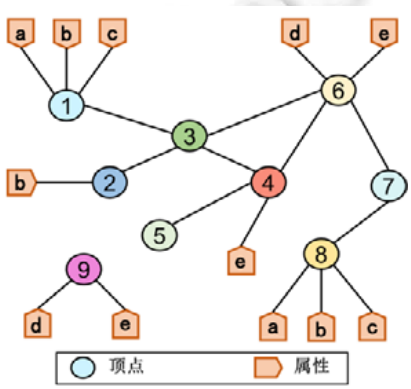



Fig.2 Service network graph

图 2 服务网络图

例如:对于本文的 Web 服务网络,节点可以是 Mashup,如果两个 Mashup 调用了同一个 API,则认为这两个

Mashup 节点之间存在一条边.而属性可以是 Mashup 的名称、关键词、标签等.此外,在本文中,我们将预处理以后的描述文档里所包含的核心语义信息也加入到了属性集合中.我们的目标是学习一个低维网络表示$\sigma:V\to\mathbb{R}^d$,其中,$d<<|V|$是学习表征的维数,从而保存结构和属性上下文信息;同时,$\sigma$也是为后续的分类任务提供的特征输入.在本文中,我们考虑 $G$ 是一个齐次的、非加权的、部分属性的图.

然后,我们从服务网络图 $G$ 中得到两个图.

1. 结构连通图 $G_s=(V_s,E)$,由边集 $E$ 中包含的顶点的子集 $V_s\subseteq V$ 组成.如图 3 所示,我们把 $V_s$ 中的顶点称为结构顶点,而边$(p_s,q_s)\in E$ 编码了节点间的结构关系;
2. 属性二分图 $G_a=(V_a,\mathbb{C},E_a)$,如图 4 所示,它由 3 部分构成.
   (1) 与属性相关联的内容顶点 $V_a\subseteq V$ 的子集;
   (2) 上述服务网络图定义中给出的可能属性顶点集$\mathbb{C}$;
   (3) 将内容顶点连接到由函数 $A$ 关联的属性顶点的边集 $E_a$:

$$V_a=\{v:A(v)\neq\varnothing\},$$
$$E_a=\{(v,a):a\in A(v)\}.$$

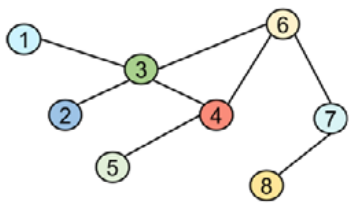



Fig.3 Structural connected graph
图 3 结构连通图

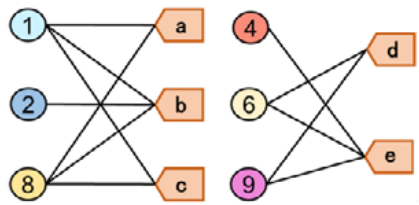



Fig.4 Attribute bipartite graph
图 4 属性二分图

**定义 1**. 两个内容顶点 $u,v\in G_a$ 只能通过属性顶点到达.

在定义 1 中,内容顶点之间的路径包含内容顶点和属性顶点.因此,包含在路径中的内容顶点具有上下文关系,因为它们可以通过某些属性顶点到达,这些内容顶点构成属性上下文.我们方法遵循的是:“如果它们与相似的对象相连,那么这两个实体是相似的”.这种现象可以在许多应用中观察到,例如在二分图“Mashup-标签”中,“Mashup”被定义了“标签”,如果两个 Mashup 被定义了相似的标签,那么二者就是相似的.这种二分网络结构在文献[14]中也被使用过,它建立了文档和文档中的词语之间的网络模型.

### 1.2 随机游走

总的来说,实体之间的相似性可以用几种方式来衡量.例如,它可以根据图中两个节点的距离来测量:在图 4 的属性二分图中,节点 2 和节点 8 都与节点 1 有属性连接,因此我们可以说它们与节点 1 具有相等的相似性;但是有 3 条连接节点 1 和节点 8 的路径,而连接节点 1 和节点 2 的只有一条路径,因此,节点 1 和节点 8 比节点 1 和节点 2 更相似.

随机游走的方法有很多[8−10,12,15],本文采用短随机游走的方法,在 $G_s$ 和 $G_a$ 上进行随机游走,从而获得顶点的结构和属性上下文,因为短随机游走能够有效地捕获具有高相似度节点的上下文.

首先,我们通过在结构连通图 $G_s$ 上的随机游走来捕获结构上下文.对于每个顶点,通过$\gamma_s$ 次长度为$\lambda_s$ 的随机游走来构造一个语料库 $R$.该上下文语料信息将被用于嵌入,目的是保持局部和全局的结构信息.我们把 $r_i$ 表示为随机游走序列 $r\in R$ 中的第 $i$ 个顶点.例如,图 3 中的随机游走路径可以是:$r$=[2,3,4,3,1],步长为 5,从顶点 2 开始,结束于顶点 1.

而在属性二分图 $G_a$ 中,随机游走从一个内容顶点开始,通过属性节点跳转到其他内容顶点.这样,属性顶点充当内容顶点之间的桥梁,从而确定内容顶点之间的上下文关系.即,哪些内容顶点是密切相关的.由于我们感兴趣的是这些顶点在随机游走中展现出的关联度,而不是通过哪些属性将它们连接起来,所以我们忽略了随机

游走中的属性顶点.因此,游走只包含 $V_a$ 的顶点.属性相似性较高的顶点组在随机游走中可能经常出现在一起.类似于 $G_s$,我们执行 $\gamma_a$ 次长度为 $\lambda_a$ 的随机游走并构造一个语料库 $W$,$w_j$ 是指随机游走序列 $w\in W$ 中的第 $j$ 个顶点.例如,图 4 中具有属性的随机游走路径可以是:[2,*b*,1,*c*,8,*b*,2,*b*,8].我们在路径中跳过属性节点,因此,相应的路径是 $w$=[2,1,8,2,8],步长为 5,从顶点 2 开始,以顶点 8 结束.

### 1.3 表征学习

在得到结构和属性上下文以后,我们使用 SkipGram 模型共同学习基于这两个上下文的嵌入.具体来说,我们从每个结构或属性上下文中选择顶点 $v_x\in V_s|V_a$,并将其输入到 SkipGram.输入顶点 $v_x$ 是一个独热编码向量 $\{0,1\}^{|V_s\cup V_a|}$,同时也是目标顶点,输出层产生关联上下文顶点到给定输入顶点的 $2c$ 多项式分布.$c$ 是上下文大小,即在目标顶点之前或之后的预测顶点的数量.同样,输出顶点可以属于结构顶点,也可以属于属性顶点,这取决于它们在随机游走中的共现性.

GWSC 方法的最终目的是,能够最大化目标顶点的结构和属性上下文概率.与以往的研究[8,9]类似,我们假定当给定一个目标顶点时,其上下文顶点的概率彼此独立.因此,我们的目标函数定义如下:

$$L=\sum_{r\in R}\sum_{i=1}^{|r|}\log p(r_{-c}:r_c\mid r_i)+\sum_{w\in W}\sum_{i=1}^{|w|}\log p(w_{-c}:w_c\mid w_i) \tag{1}$$

等式(1)可以写成:

$$L=\sum_{r\in R}\sum_{i=1}^{|r|}\sum_{\substack{-c\leqslant j\leqslant c\\ j\neq i}}\log p(r_j\mid r_i)+\sum_{w\in W}\sum_{i=1}^{|w|}\sum_{\substack{-c\leqslant t\leqslant c\\ t\neq i}}\log p(w_t\mid w_i) \tag{2}$$

其中,$r_{-c}$:$r_c$ 和 $w_{-c}$:$w_c$ 分别对应于 $R$ 和 $W$ 语料库中随机游走长度为 $2c$ 的上下文窗口内的一系列顶点.公式的前半部分使用结构上下文进行学习,后半部分从属性上下文中进行学习.如果$|V_a|$=0,那么模型将变成 Deepwalk,也就是说,只从结构中学习.当 $r_i$ 是结构上下文 $r$ 中的中心顶点时,$p(r_j|r_i)$是第 $j$ 个顶点的概率;而当 $w_i$ 是属性上下文 $w$ 中的中心顶点时,$p(w_t|w_i)$是第 $t$ 个顶点的概率,这些概率可以用 *softmax* 函数来计算.概率 $p(r_j|r_i)$可计算为

$$p(r_j\mid r_i)=\frac{\exp(\varphi(r_j)^T\phi(r_i))}{\sum_{v_s\in V_s}\exp(\varphi(v_s)^T\phi(r_i))} \tag{3}$$

其中,$\varphi(\cdot)$,$\phi(\cdot)$分别表示上下文顶点或目标顶点.同样的,也可以用等式(3)计算 $p(w_t|w_i)$.

由于需要对图的所有顶点进行归一化处理,使得计算量很大.因此,我们用分层的 *softmax* 函数[16]来进行计算.在这之后,使用 Huffman 编码来构造具有顶点作为叶节点的层次 *softmax* 的二叉树[17].因此,为了计算概率,我们只需遵循从根节点到叶节点的路径.所以,叶节点 $r_j$ 出现在结构上下文中的概率是:

$$p(r_j\mid r_i)=\prod_{h=1}^{d}p(s_h\mid\phi(r_i)) \tag{4}$$

其中,$d$=log$|V_s|$是树的深度,$s_h$ 是路径中的节点,$s_o$ 为根节点,且 $s_d$=$r_j$.此外,将 $p(r_j|r_i)$建模为二进制分类器,可以使计算复杂度降至 $O(\log|V_s|)$.这同样适用于计算属性上下文中顶点的概率.而考虑到我们是从两个上下文中计算概率的,所以我们的整体计算复杂度应该是 $O(\log|V_s|+\log|V_a|)$.

### 1.4 SVM分类

在得到了表征向量以后,我们需要将表征向量输入到分类器中进行训练.解决分类问题的方法有很多,主要包括决策树、贝叶斯、*K*-近邻、人工神经网络、支持向量机(SVM)等.对于决策树,数据的准备往往是简单的,且易于通过静态测试来对模型进行评测,但对于那些各类别样本数量不一致的数据,信息增益的结果往往偏向于那些具有更多数值的特征.而贝叶斯所需估计的参数很少,对缺失数据不太敏感,算法也比较简单,但需要知道先验概率,且分类决策存在错误率.*K*-近邻算法简单有效,重新训练的代价也较低,但该算法在分类时有个很明显的缺点是:当样本不平衡时,如一个类的样本容量很大,而其他类样本容量很小时,有可能导致当输入一个新样本时,该样本的 $K$ 个邻居中大容量类的样本占多数.人工神经网络分类的准确度往往较高,并行分布处理能力较强,但通常应用于大规模数据集,且需要大量的参数,如网络拓扑结构、权值和阈值的初始值,学习时间较长,

甚至可能达不到学习的目的.相比之下,SVM 虽然缺乏对数据的敏感度,但可以解决高维和非线性问题,而且在提高了泛化性能力的基础上,可以解决小样本情况下的机器学习问题,避免了神经网络结构选择和局部极小点的情况.

由于本文的数据集较小,不适合用人工神经网络进行分类,且存在类别样本数分布不均的问题,例如在 API 数据集中,仅样本数排名第一的 Tools 类与样本数排名第十的 Science 类就相差了 503 个样本数,所以决策树和 *K*-近邻算法在本文中也不适用.综合考虑之下,我们最终选择 SVM 算法进行分类.具体步骤如下.

- 首先,对于 $k(k\geqslant 2)$类问题,我们根据得到的表征向量和已知的数据构建样本集$(x_1,y_1),\ldots,(x_l,y_l),x_i\in R^n$, $i=1,\ldots,l,y_i\in\{1,\ldots,k\},l$ 表示样本数;
- 然后,把类 1 作为一类,其余 $k-1$ 类视为一类,自然地将 $k$ 分类问题转化为二分类问题,在训练过程中,每个分类函数都需要所有样本参与.其分类函数为

$$f(x)=\arg_{j\in\{1,\ldots,k\}}\sum_{i=1}^{n}(\alpha_i^j y_i^j K(x,x_i^j)+b^j) \tag{5}$$

其中,上标表示第 $j$ 个 SVM 分类器的决策函数,$\alpha_i^j$ 和 $y_i^j$ 分别为第 $j$ 个支持向量的参数和类别标号,$b^j$ 为偏移量.对于待测样本,若:

$$f^l(x)=\max_{j\in\{1,\ldots,k\}}f^j(x) \tag{6}$$

则输入样本属于第 $l$ 类;

- 最后,对所有样本执行上述操作即可完成所有分类任务.

## 2 实验评估及分析

### 2.1 数据集描述及处理

为评估模型,我们从 ProgrammableWeb.com 网站上爬取了 6 415 个 Mashup 和 12 919 个 API 的信息,包括他们的名称、描述文档、主次分类等信息,基本的统计信息见表 1、表 2,完整的数据集可以在 http://kpnm.hnust.cn/xstdset.html 网址上进行下载.

**Table 1** Data statistics of Mashup

**表 1** Mashup 数据统计信息

| Items | 值 |
|---|---|
| Mashup 的个数 | 6 415 |
| Mashup 的类别数 | 324 |
| 平均每个类别包含的 Mashup 个数 | 19.8 |
| 平均每个 Mashup 拥有的标签数 | 3.16 |
| Mashup 调用的 API 数 | 1 471 |

**Table 2** Data statistics of API

**表 2** API 数据统计信息

| Items | 值 |
|---|---|
| API 的个数 | 12 919 |
| API 的类别数 | 383 |
| 平均每个类别包含的 API 个数 | 33.7 |
| 平均每个 API 拥有的标签数 | 3.85 |
| 被 Mashup 调用的 API 数 | 1471 |

在爬取的数据中,类别为“Search”的 Mashup 就有 306 个,而类别“Cities”中仅仅包含了一个 Mashup.同样,类别为“Tools”的 API 有 790 个,而类别“Solar”中仅仅包含了一个 API.因此,我们选取了数量最多的前 10~50 类 Mashup 和 API 用于实验,详细的分布情况请见表 3、表 4.

**Table 3** Top 10 categories with the largest number in Mashup

**表 3** Mashup 数量最多的前 10 类

| 分类 | 数量 | 分类 | 数量 |
|---|---|---|---|
| Mapping | 1 034 | Music | 250 |
| Search | 306 | Video | 176 |
| Social | 299 | Travel | 169 |
| eCommerce | 294 | Messaging | 133 |
| Photos | 256 | Mobile | 126 |

**Table 4** Top 10 categories with the largest number in API

**表 4** API 数量最多的前 10 类

| 分类 | 数量 | 分类 | 数量 |
|---|---|---|---|
| Tools | 790 | Messaging | 388 |
| Financial | 586 | Payments | 374 |
| Enterprise | 487 | Government | 306 |
| eCommerce | 435 | Mapping | 295 |
| Social | 403 | Science | 287 |

由于在构建属性二分图时需要用到大量文本信息,为了提高分类的精确度,我们首先要对 Mashup 和 API 的描述文档进行预处理.过程如下.

(1) 分词(tokenize):将每个单词按照空格分开,且将单词和标点符号也分开,使得文本中的单词、字符变成单独的单元;

(2) 去停用词(stop words):去除英文中一些无意义的词以及标点符号,如“a”“the”“to”“@”等;

(3) 词干化处理(stemming):在英文文本中,同一个单词会因为人称、时态的不同而有不同的表现形式,如“adaptation”“adapted”“adapting”,它们实际上都是同一个单词“adapt”.若将这些单词看作是不同的单词,那么之后的实验结果的准确度将会降低,故需要进行词干化处理.

在完成以上 3 个步骤后,获得了处理好的 Mashup 和 API 描述文档.同时,我们可以注意到:Mashup 数据集中,最多的“Mapping”类有 1 034 个,而排第二的“Search”类只有 306 个.为防止数据集样本分布不均影响实验结果,我们随机选取类别为“Mapping”的子集 434 条作为其实验数据.而 API 数据集分布较为均匀,影响不大,不需要做相关处理,所有的数据集实验统计见表 5、表 6.

**Table 5** Experimental statistics of Mashup dataset

**表 5** Mashup 数据集实验统计

| Items | 数量 | Items | 数量 |
|---|---|---|---|
| $\|V\|$ | 6 415 | $V_a$ | 8 711 |
| $\|E\|$ | 57 497 | $\|\mathcal{C}\|$ | 6 268 |
| $V_s$ | 2 385 | $\|E_a\|$ | 97 117 |

**Table 6** Experimental statistics of API dataset

**表 6** API 数据集实验统计

| Items | 数量 | Items | 数量 |
|---|---|---|---|
| $\|V\|$ | 12 919 | $V_a$ | 11 524 |
| $\|E\|$ | 87 497 | $\|\mathcal{C}\|$ | 10 268 |
| $V_s$ | 4 523 | $\|E_a\|$ | 121 846 |

### 2.2 评估标准

一般来说,Web 服务分类结果有以下 4 种情况.

(1) 属于类 $A$ 的样本被正确分类到类 $A$,将这一类样本标记为 TP;

(2) 不属于类 $A$ 的样本被错误分类到类 $A$,将这一类样本标记为 FP;

(3) 不属于类别 $A$ 的样本被正确分类到了类别 $A$ 的其他类,将这一类样本标记为 TN;

(4) 属于类别 $A$ 的样本被错误分类到类 $A$ 的其他类,将这一类样本标记为 FN.

我们采用 Macro $F1$ 值和 Micro $F1$ 值作为性能评价指标,其中,Web 服务分类的 Macro $F1$ 值计算公式如下:

$$Precision_{ma}=\frac{\sum_{i=1}^{N}\frac{TP_i}{TP_i+FP_i}}{N} \tag{7}$$

$$Recall_{ma}=\frac{\sum_{i=1}^{N}\frac{TP_i}{TP_i+FN_i}}{N} \tag{8}$$

$$Macro\ F1=\frac{2\times Precision_{ma}\times Recall_{ma}}{Precision_{ma}+Recall_{ma}} \tag{9}$$

而 Web 服务分类的 Micro $F1$ 值计算公式如下:

$$Precision_{mi}=\frac{\sum_{i=1}^{N}TP_i}{\sum_{i=1}^{N}TP_i+FP_i} \tag{10}$$

$$Recall_{mi}=\frac{\sum_{i=1}^{N}TP_i}{\sum_{i=1}^{N}TP_i+FN_i} \tag{11}$$

$$Micro\ F1=\frac{2\times Precision_{mi}\times Recall_{mi}}{Precision_{mi}+Recall_{mi}} \tag{12}$$

上述公式中,Precision 表示准确率,Recall 表示召回率,$N$ 表示分类的类别数.

### 2.3 对比方法

我们用 GWSC 比较了最新的嵌入算法:两种基于属性内容的方法(Doc2vec 和 LDA)、两种基于结构关系的方法(Deepwalk 和 Node2vec)以及一种同时使用结构关系和属性内容的方法(TriDNR),且所有方法均采用 SVM 算法进行分类.

- Doc2vec[17]:Doc2vec 是一种无监督的神经网络模型,用于学习句子、段落或文档等可变长度文本的表示.该模型将每个单词、句子和段落都映射到向量空间中,然后将段落向量和词向量级联或者求平均得到特征,从而同时学习单词向量和文档向量.我们将与每个顶点相关的文本输入到模型中,并得到了每个顶点的表示;
- LDA[18]:LDA 是一种非监督机器学习技术,可以用来识别大规模文档集或语料库中潜藏的主题信息,并将文档集或语料库中每篇文档的主题以概率分布的形式给出.它采用了词袋的方法,这种方法将每一篇文档视为一个词频向量,从而将文本信息转化为了易于建模的数字信息;
- Deepwalk[8]:Deepwalk 是一种基于随机均匀游走的方法,它仅利用结构信息来学习网络中顶点的一维特征表示.该方法利用构造节点在网络上的随机游走路径来模仿文本生成的过程,并提供一个节点序列,然后用 Skip-Gram 模型对随机游走序列中每个局部窗口内的节点对进行概率建模,最大化随机游走序列的似然概率,从而学习节点的分布式表示;
- Node2vec[9]:Node2vec 类似于 Deepwalk,主要的创新点在于改进了随机游走的策略,定义了两个参数 $p$ 和 $q$,在广度优先搜索(BFS)和深度优先搜索(DFS)中达到一个平衡,BFS 用于探究图中的结构性质,而 DFS 则用于探究出相邻节点之间的相似性,从而同时考虑到了局部和宏观的信息,并且具有很高的适应性;
- TriDNR[12]:TriDNR 使用 3 个信息源——网络结构、顶点内容和标签信息来学习顶点的表示.它结合了两种模型:Deepwalk 从结构中学习表示,Doc2vec 用于捕获与节点内容和标签信息相关的上下文.最终表示是这两个模型输出的线性组合;
- GAT2VEC[13]:这是本文所用到的向量表征方法.GAT2VEC 框架的基本思路与 Deepwalk 方法类似,二者的主要区别在于:Deepwalk 只考虑到了顶点之间的结构信息;而 GAT2VEC 框架则同时融合了结构

与属性信息,并创造了新的顶点定义方式.这同样也是 GAT2VEC 框架相比于 Node2vec 方法最大的改进之一.另外,虽然 GAT2VEC 框架与 TriDNR 都同时考虑到了顶点的结构与属性信息,但 TriDNR 是直接通过多层感知机训练得到每个顶点相对应的属性信息嵌入向量;而 GAT2VEC 框架则创造性地构建出包含属性信息的二分图,并利用网络表征的方法学习到二分图中每个顶点所对应的属性特征向量.

### 2.4 实验结果

#### 2.4.1 实验设置

在实验中,我们选择 70%的数据作为训练集,30%的数据作为测试集.对于 Mashup 数据集,我们将随机游走次数$\gamma_s$和$\gamma_a$均设置为 30,游走步长$\lambda_s$和$\lambda_a$均设置为 120;表征的维度大小 $d$ 设置为 128,窗口大小 $c$ 设置为 5.对于 API 数据集,我们将随机游走次数$\gamma_s$和$\gamma_a$均设置为 30,游走步长$\lambda_s$和$\lambda_a$均设置为 120;表征的维度大小 $d$ 设置为 128,窗口大小 $c$ 设置为 5.为保证实验的客观性,GWSC 与其他方法之间共有的参数设置为相同的值,而其余的参数被设置为默认的最优值.

#### 2.4.2 分类性能比较

我们分别选取了数量最多的前 10~50 类 Mashup 和 API 进行实验,实验结果见表 7、表 8.此外,我们借助 t-SNE 工具[19]基于 PCA 降维技术对前 20 类分类结果进行了可视化,其效果如图 5、图 6 所示.显然,对于两个数据集,本文所提出的 GWSC 方法在 *Micro F*1 值和 *Macro F*1 值两个指标上均要优于其余 5 种方法.例如:当服务类别数为 10 时,在 Mashup 数据集上,GWSC 相比于 Doc2vec,LDA,Deepwalk,Node2vec 和 TriDNR 在 *Macro F*1 值上分别有 135.3%,60.3%,12.4%,10.5% 和 4.3% 的提升;而在 API 数据集上,GWSC 相比于 Doc2vec,LDA,Deepwalk,Node2vec 和 TriDNR 在 *Macro F*1 值上分别有 137.2%,61.3%,14.3%,8.6%和 1.1%的提升,效果显著.具体来说:

(1) 随着数据集所取分类的类别数增多,分类的效果逐渐下降.其原因可能是:
   1) 分类的类别数越多,包含的信息就越多且越复杂,模型分类的难度也就越大;
   2) 分类是按照每个类别包含的 Mashup 或 API 的数量进行排名的,而排名靠后的类别由于所包含的 Mashup 或 API 数量的减少,其所能利用的信息相对来说就会减少,从而影响分类效果;

(2) 基于结构关系的方法(Deepwalk 和 Node2vec)其效果要远优于基于属性内容的方法(Doc2vec 和 LDA).这个结果表明:对于 Mashup 分类,结构信息比内容信息更重要.因为调用同一个 API 的两个 Mashup 在功能需求上往往一样,那么二者的类别就会更相似.同样,对于 API 分类,结构信息也比内容信息更重要.因为被同一个 Mashup 调用的两个 API 在功能上往往需要满足类似的需求,那么二者的类别也就会更相似;

(3) 对于 Mashup 和 API 两个数据集,同时使用结构关系和属性内容的方法(TriDNR,GWSC),其表现要明显优于只使用单一信息的方法.这也就验证了:对于同一个数据集而言,捕获的信息越多,特征向量的表示往往就越准确,分类的效果也就会随之提升;

(4) GWSC 相比于 TriDNR,在整体上有了一定幅度的提升.这表明:将属性文本信息以结构化的形式来表示,确实有利于 Mashup 和 API 的分类效果.在 GWSC 方法中,Mashup 和 API 的属性信息被构建成了二分图,并以深度游走的方式嵌入到了表征向量中;

(5) 整体来说,Mashup 的分类效果要优于 API 的分类效果.其原因可能是:
   1) API 数据集比 Mashup 数据集更大,种类和标签都更加丰富,包含的信息更复杂,相对应的,数据集中的特征信息会更难以捕获,分类的难度也就会越大;
   2) 在 Mashup 数据集中,每一个 Mashup 都至少调用了一个 API;而在 API 数据集中,有部分 API 并没有被 Mashup 调用,Mashup 数据集的结构更趋完整,且描述文档更加规范,种类相对来说更为集中.

**Table 7** Experimental results of Mashup dataset

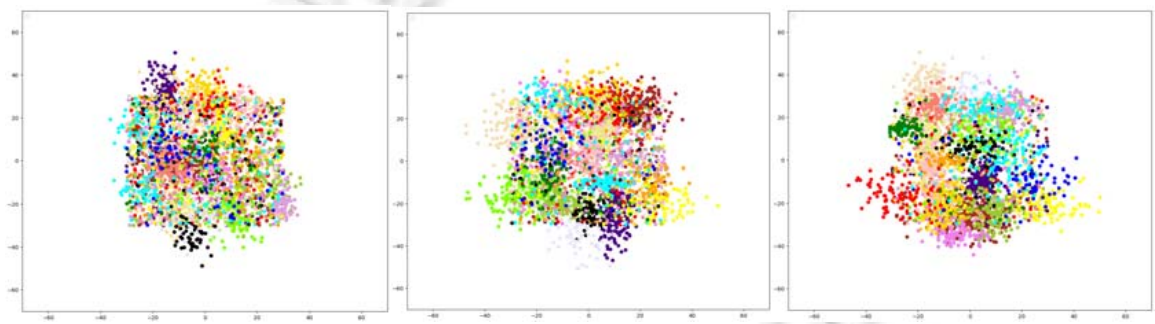
**表 7** Mashup 数据集实验结果

| 对比方法 | 类别数 | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 10 | | 20 | | 30 | | 40 | | 50 | |
| | Mi-*F*1 | Ma-*F*1 | Mi-*F*1 | Ma-*F*1 | Mi-*F*1 | Ma-*F*1 | Mi-*F*1 | Ma-*F*1 | Mi-*F*1 | Ma-*F*1 |
| Doc2vec | 0.421 | 0.312 | 0.235 | 0.196 | 0.192 | 0.125 | 0.135 | 0.098 | 0.106 | 0.074 |
| LDA | 0.538 | 0.458 | 0.393 | 0.367 | 0.325 | 0.213 | 0.261 | 0.186 | 0.184 | 0.135 |
| Deepwalk | 0.685 | 0.653 | 0.612 | 0.603 | 0.531 | 0.504 | 0.412 | 0.369 | 0.358 | 0.342 |
| Node2vec | 0.702 | 0.664 | 0.693 | 0.614 | 0.612 | 0.586 | 0.536 | 0.438 | 0.389 | 0.395 |
| TriDNR | 0.764 | 0.704 | 0.732 | 0.624 | 0.654 | 0.602 | 0.616 | 0.543 | 0.521 | 0.469 |
| **GWSC** | **0.783** | **0.739** | **0.745** | **0.635** | **0.667** | **0.612** | **0.638** | **0.585** | **0.588** | **0.523** |

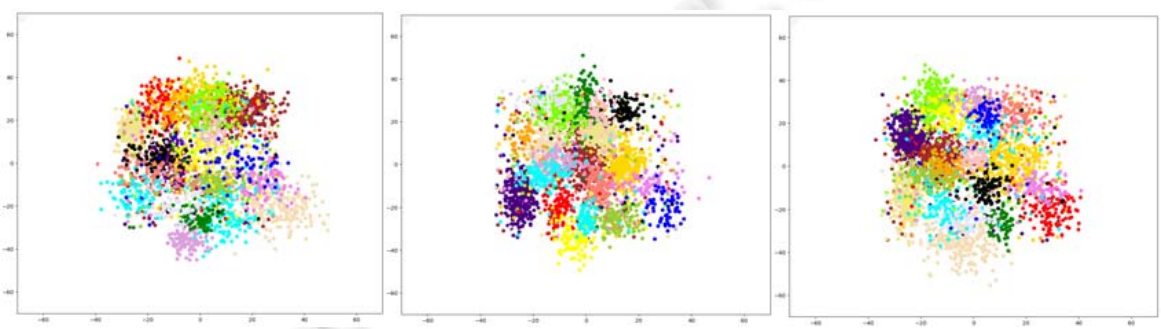
**Table 8** Experimental results of API dataset

**表 8** API 数据集实验结果

| 对比方法 | 类别数 | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 10 | | 20 | | 30 | | 40 | | 50 | |
| | Mi-*F*1 | Ma-*F*1 | Mi-*F*1 | Ma-*F*1 | Mi-*F*1 | Ma-*F*1 | Mi-*F*1 | Ma-*F*1 | Mi-*F*1 | Ma-*F*1 |
| Doc2vec | 0.391 | 0.304 | 0.315 | 0.265 | 0.278 | 0.124 | 0.256 | 0.097 | 0.201 | 0.064 |
| LDA | 0.516 | 0.447 | 0.354 | 0.356 | 0.325 | 0.243 | 0.308 | 0.210 | 0.254 | 0.125 |
| Deepwalk | 0.654 | 0.631 | 0.549 | 0.594 | 0.537 | 0.514 | 0.452 | 0.377 | 0.365 | 0.352 |
| Node2vec | 0.683 | 0.664 | 0.602 | 0.614 | 0.546 | 0.576 | 0.491 | 0.434 | 0.409 | 0.392 |
| TriDNR | 0.759 | 0.713 | 0.713 | 0.622 | 0.657 | 0.586 | 0.597 | 0.513 | 0.586 | 0.479 |
| **GWSC** | **0.771** | **0.721** | **0.742** | **0.630** | **0.663** | **0.594** | **0.618** | **0.555** | **0.588** | **0.521** |



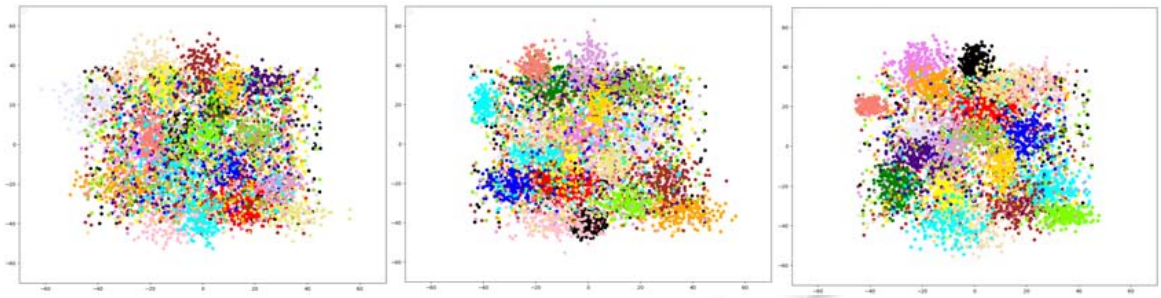
(a) Doc2vec (b) LDA (c) Deepwalk



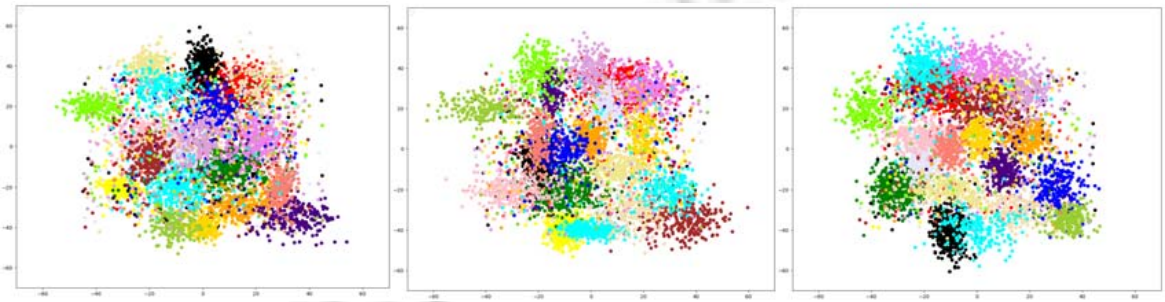
(d) Node2vec (e) TriDNR (f) GWSC

Fig.5 Visualization results with 20 categories classification of Mashup dataset

图 5 Mashup 数据集 20 分类可视化结果



(a) Doc2vec (b) LDA (c) Deepwalk



(d) Node2vec (e) TriDNR (f) GWSC

Fig.6 Visualization results with 20 categories classification of API dataset

图 6 API 数据集 20 分类可视化结果

### 2.5 结构信息与属性信息权重分析

为了验证第 2.4 节第(2)点所提出的结构信息比属性信息更加重要的观点,我们对 GAT2VEC 框架进行了一些修改,从而实现对结构信息与属性信息的权重分析.具体来说,在最终对结构路径和属性路径进行向量表征时,我们不再以相同的权重对二者进行组合,而是分别设置不同的权重(从 0.1 到 0.9)进行实验.例如,权重设置为 0.1,表示在表征向量时,结构信息的权重设置为 0.1,属性信息的权重设置为 0.9.全部实验结果如图 7 和图 8 所示.

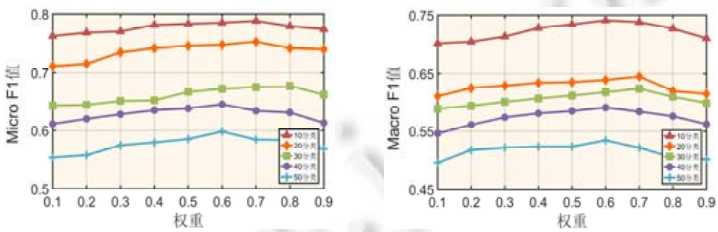



Fig.7 Analysis of the weight with the structure and the attribute information of Mashup dataset

图 7 Mashup 数据集结构和属性信息的权重分析

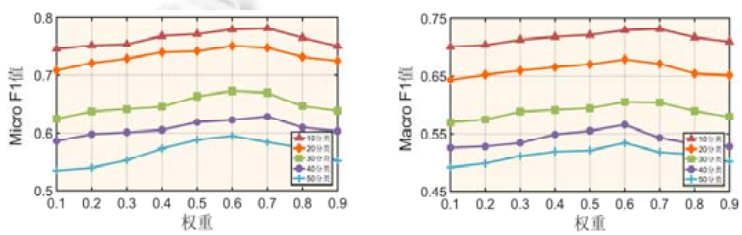



Fig.8 Analysis of the weight with the structure and the attribute information of API dataset

图 8 API 数据集结构和属性信息的权重分析

具体来说:

(1) 整体而言,无论是 Mashup 数据集还是 API 数据集,随着结构属性所占权重的提升,分类效果在不断上升;但是当权重超过 0.6 或 0.7 左右时,分类效果开始下降.这说明对于 Mashup 和 API 的 Web 服务分类来说,结构信息在一定程度上来说确实比属性信息更重要.另外,对于当权重超过一定阈值以后,分类效果会出现下降的情况,其可能的原因是权重设置的越高,属性信息就会相对减少,整体的数据集信息也会随之减少,从而影响分类效果;

(2) 总的来说,数据集所取分类的类别数越多,其分类效果受结构信息与属性信息权重的影响越大,其可能原因是:1) 分类类别越多,数据集就越大且越复杂,更容易受到结构信息与属性信息的影响;2) 分类是按照每个类别包含的 Mashup 或 API 的数量进行排名的,而排名靠后的类别由于所包含的 Mashup 或 API 数量的减少,其拥有的结构调用信息相对来说就会减少,从而影响分类效果.

### 2.6 $\gamma$和$\lambda$参数分析

在向量表征模型中,增加游走次数或游走步长可以收集更多的上下文信息,从而学习更精确的表示.但是过多的游走次数和较大的游走步长都不合适,因为容易产生噪声数据,从而导致较差的网络表示.在本文中,我们针对不同随机游走次数$\gamma$和游走步长$\lambda$下的 Mashup 和 API 分类进行实验比较,以确定最佳分类效果下的参数值.

#### 2.6.1 Mashup 数据集分析

首先,选择不同游走次数$\gamma$(10,20,30,40,50)进行 Mashup 分类实验,游走步长$\lambda$暂设为方法的默认值 80,得到的 *Micro F*1 值和 *Macro F*1 值如图 9 所示.

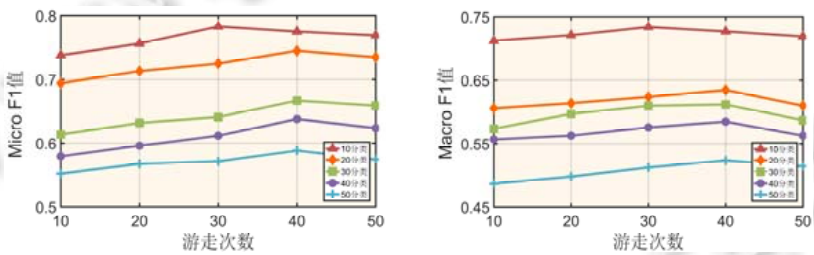



Fig.9 Number of walk parameter analysis of Mashup dataset

图 9 Mashup 数据集游走次数参数分析

从实验结果可以看出:

1) 对于 Mashup 数据集的 10 分类问题,当游走次数$\gamma$设置为 30 时,Mashup 数据集的分类效果最好;

2) 对于 Mashup 数据集的 20~50 分类问题,随着游走次数的增加,*Micro F*1 值和 *Macro F*1 值也逐渐增加;但当$\gamma$超过 40 的时候,分类效果开始下降.

其次,我们选择不同的游走步长$\lambda$(40,80,120,160,200)进行 Mashup 分类实验,$\gamma$设置为上述最佳值,得到的 *Micro F*1 值和 *Macro F*1 值如图 10 所示.

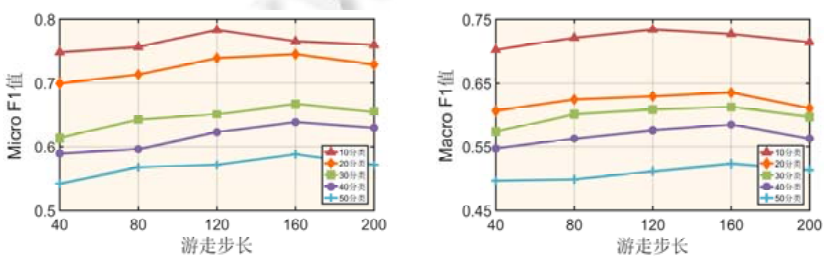



Fig.10 Walk length parameter analysis of Mashup dataset

图 10 Mashup 数据集游走步长参数分析

从实验结果可以看出:

1) 对于 Mashup 数据集的 10 分类问题,游走步长$\lambda$在 120 附近增加或减少时,分类效果有所下降;

2) 对于 Mashup 数据集的 20~50 分类问题,Micro $F1$ 值和 Macro $F1$ 值随着游走步长的增加呈现出上升趋势;但当$\lambda$超过 160 的时候,分类效果开始呈下降趋势.

总体来说,最佳游走步长的数值相对来说较大,这是因为 Mashup 的服务网络图较为稀疏,需要较长距离游走才可以捕获到有价值的网络表征信息.

2.6.2 API 数据集分析

对于 API 数据集,我们选择不同游走次数$\gamma$(10,20,30,40,50)进行 API 分类实验,游走步长$\lambda$同样暂设为方法的默认值 80,得到的结果如图 11 所示.

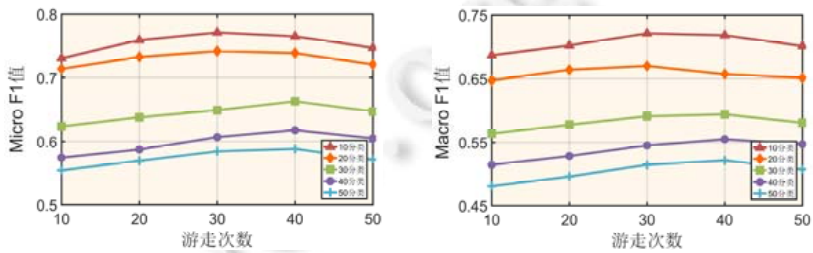



Fig.11 Number of walk parameter analysis of API dataset

图 11 API 数据集游走次数参数分析

从实验结果可以看出:

1) 对于 API 数据集的 10 分类、20 分类问题,随着游走次数的增加,Micro $F1$ 值和 Macro $F1$ 值先上升后下降;当游走次数$\gamma$设置为 30 时,API 的分类效果最好;

2) 对于 API 数据集的 30~50 分类问题,当$\gamma$设置为 40 时,分类效果达到最佳.

接下来,我们选择不同的游走步长$\lambda$(40,80,120,160,200)进行 API 分类实验,$\gamma$设置为上述最佳值,得到的 Micro $F1$ 值和 Macro $F1$ 值如图 12 所示.

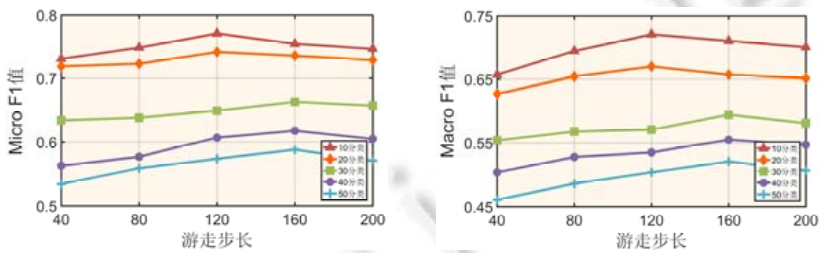



Fig.12 Walk length parameter analysis of API dataset

图 12 API 数据集游走步长参数分析

从实验结果可以看出:(1) 对于 API 数据集的 10 分类、20 分类问题,当游走步长$\lambda$设置为 120 时,API 的分类效果最好;(2) 对于 API 数据集的 30~50 分类问题,Micro $F1$ 值和 Macro $F1$ 值随游走步长的增加先上升后下降,最佳的游走步长为 160.

### 2.7 表征维度分析

适当地增加表征维度的大小可以学习到更多的特征,从而得到更好的分类效果.但随着特征空间维度增加,整个特征空间会变得越来越稀疏;同时,分类器一旦学习了训练数据的噪声和异常,模型就会出现过拟合现象,大大影响分类效果.在本节中,我们设计了多组实验对本文表征学习中涉及的表征维度 $d$ 进行参数调整,以使分类效果达到最好.我们选取了 5 个不同的维度(32,64,128,256,512)进行对比实验,结果如图 13、图 14 所示.

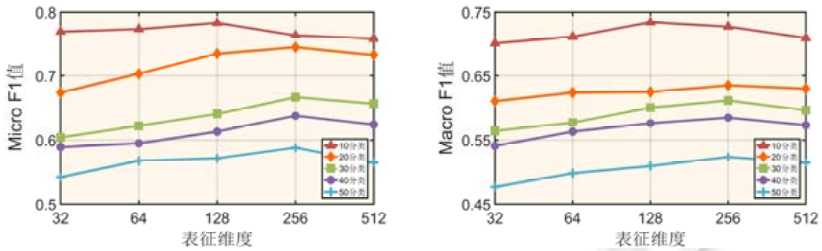



Fig.13 Representation dimension analysis of Mashup dataset

图 13 Mashup 数据集表征维度分析

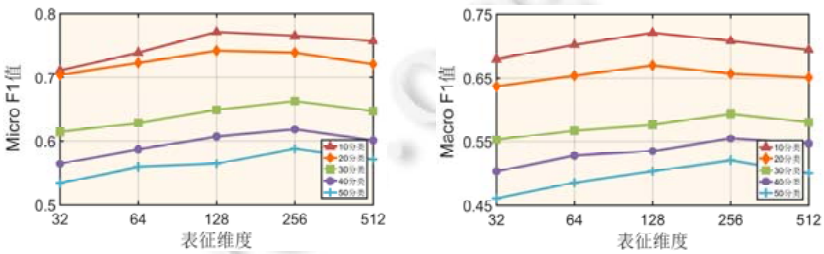



Fig.14 Representation dimension analysis of API dataset

图 14 API 数据集表征维度分析

首先,对于 Mashup 数据集可以看到:

1) 在 10 分类问题中,表征维度 $d$ 在 128 附近增加或减少时,分类效果有所下降;
2) 在 20~50 分类问题中,Micro $F1$ 值和 Macro $F1$ 值随着游走步长的增加呈现出上升趋势;但当表征维度超过 256 的时候,分类效果开始呈下降趋势.

其次,对于 API 数据集可以看到:

1) 在 10 分类、20 分类问题中,分类效果随着表征维度的增加先上升后下降;且当表征维度设置为 128 的时候,分类效果达到最佳;
2) 在 30~50 分类问题中,当表征维度 $d$ 设置为 256 时,Micro $F1$ 值和 Macro $F1$ 值达到最大值.

## 3 相关工作

随着服务计算和云计算的发展,互联网上出现了多种多样的网络服务.其中,Web 服务的发现和挖掘成为一个热门的研究方向.研究表明:正确高效的 Web 服务分类能够有效提高 Web 服务发现的性能[20].目前,对 Web 服务自动分类的研究吸引了不少研究者的注意,主要有基于功能语义的服务分类与基于 QoS(quality of service)的服务分类.

目前,基于功能语义的服务分类方法已有大量的研究[21,22].Kuang 等人[23]提出一种基于语义相似度的 Web 服务分类方法,通过采用自适应反向传播神经网络模型训练服务的特征向量及其类别,从而对 Web 服务进行分类.Miguel 等人[24]提出一种基于启发式的分类系统,通过将新服务与已分类的服务进行比较,预测出新服务可用分类类别的适当性,并生成候选类别的排序列表来进行分类.Crosso 等人[3]利用 Web 服务的类别和在标准描述中常见的信息之间的连接,实现了自动 Web 服务分类.Katakis 等人[4]考虑了服务的文本描述和语义标注,解决了 Web 服务在其应用领域的自动分类问题,并提出了通过扩展特征向量和集成分类器来提高整体分类精度的方法.Kumar 等人[25]通过匹配 Web 服务的服务名称、输入和输出参数、服务描述,确定了它们在本体中的位置和意义,最终利用本体 Web 语言对 Web 服务进行分类.Shi 等人[5]提出一种考虑多重 Web 服务关系的概率主题模型 MR-LDA,其可对 Web 服务之间相互组合的关系以及 Web 服务之间共享标签的关系进行建模.Cao 等人[6]通过注意力机制将 BiLSTM 局部的隐状态向量和全局的 LDA 主题向量结合起来,提出一种基于主题注意力机制

Bi-LSTM 的 Web 服务分类方法.Ye 等人[1]利用 Wide & Bi-LSTM 模型对 Web 服务类别进行深度预测,挖掘 Web 服务描述文档中词语的语序和上下文信息.Chen 等人[7]针对富含全局主题信息与局部语义信息的移动应用内容表征文本,引入注意力机制区分其不同单词的贡献度,提出一种主题注意力机制增强的移动应用分类方法.

此外,基于 QoS 的 Web 服务分类主要考虑 Web 服务的质量,通常使用到的 QoS 包括吞吐量、可用性、执行时间等.Moraru 等人[26]提出一个将语义技术与逻辑推理相结合的混合系统(即 OpenCyc),其通过与数值微积分进行分类、评估,推荐 QoS 感知 Web 服务.Makhlughian 等人[27]提出了一个整体的服务选择和排序,该方法首先根据用户的 QoS 要求和偏好将候选 Web 服务分类到不同的 QoS 级别,然后通过语义匹配对最合适的候选服务进行匹配.TF 等人[28]提出了一种基于 QoS 参数和 *K*NN(*K*-nearest neighbors)算法的 Web 服务有效选择方法,并通过加入新的并行分类模型,提高了系统的性能.

上述方法考虑到了 Web 服务描述文档中的长度、有限语料库和稀疏特征等问题,并在服务语料库的训练过程中引入了辅助信息(如词分类信息、标签信息等)[29],但是它们并没有考虑到 Web 服务之间丰富的链接结构关系.

近年来,NRL 技术的不断成熟为解决此类问题提供了良好的思路,并使得机器学习任务(如分类、预测和推荐)在网络中的应用成为了可能.NRL 的一种常见方法是,使用节点的内容或属性信息来学习图的嵌入向量,例如深度神经网络模型 SkipGram[30]和 Le 提出的段落向量模型[31]等.

当利用来自结构和属性两者的信息时,NRL 所学习的表示往往会更加精确.属性信息的引入,使得模型能够利用上下文信息来学习稀疏连接的、甚至断开的节点的嵌入;而结构信息的引入,使得模型能够在学习的嵌入中保持节点的结构相似性.在这两个信息源的相辅相成之下,节点的低维嵌入学习会更加精确.Yang 等人[11]表明了 Deepwalk 等价于分解邻接矩阵 $M$,并提出一种名为 TADW 的模型,该模型通过对文本相关的矩阵进行分解来融合节点的文本特征.尽管 TADW 已经取得的不错的效果,但存在几个局限性:(1) 它分解近似矩阵,因此表示较差;(2) 它高度依赖于计算昂贵的奇异值分解(SVD);(3) 它不适用于大型的网络表征.

近几年来,人们提出了各种利用标签信息来学习嵌入的半监督学习方法[12,32].标签是与顶点相关联的类,用于对学习嵌入的分类器进行训练,以标记非标记节点.结果表明:通过在学习过程中加入标签,嵌入效果将会更好.使用标签信息的原因是:具有相似标签的节点往往具有很强的互连性和属性的高度相似性,因此也应该具有相似的嵌入.Pan 等人[12]提出的 TriDNR 使用了 3 种信息来源:结构、文本和嵌入的部分标签.该模型使用两层神经网络:第 1 层基于 Deepwalk 学习基于结构的表示,第 2 层使用 Doc2vec[17]学习内容和部分标签的表示.最后的表示是两者的线性组合.

然而,上述方法只适用于同构网络.最新的 NRL 方法学习了异构网络中节点的表示.Dong 等人[33]提出了 Metapath2vec,该模型利用基于元路径的随机游走来生成网络中不同类型节点之间的语义关系.然而,这项工作忽略了相似节点之间的关系,如论文之间的引用等.预测文本嵌入(PTE)[14]学习了一个包含文字、文档和标签的异构文本网络嵌入.随着网络表征学习的广泛应用,通过融合文本属性和结构网络信息来提升分类精度的思想得以实现.

## 4 结 论

本文提出一种基于 GAT2VEC 的 Web 服务分类方法 GWSC,它使用网络结构和顶点属性来学习服务网络图的表征向量,并采用一种新的方法来捕获属性内容.首先,它从网络中提取结构上下文,并从属性二分图中提取属性上下文;然后,使用一个浅层神经网络模型,从这两个上下文中联合学习一个表征向量,通过对与网络相关的多个信息源进行建模,并采用适当的学习方法,使学习得到的信息与原始输入图的信息尽可能保持一致;最后,采用 SVM 分类器进行分类.在真实世界数据集上的广泛实验表明:我们的方法能够准确地表征图中顶点的结构和属性上下文,进而提高了 Web 服务分类精度.对于未来的工作,我们将会考虑利用与不同类型顶点相关的不同信息扩展 GWSC 来学习异构网络中的顶点表示.

**References**:


[1] Ye H, Cao B, Peng Z, *et al*. Web services classification based on Wide & Bi-LSTM model. IEEE Access, 2019,7: 43697−43706.

[2] Bruno M, Canfora G, Penta MD, *et al*. An approach to support Web service classification and annotation. In: Proc. of the IEEE Int'l Conf. on E-Technology, E-Commerce and E-Service. IEEE Computer Society, 2005. 138−143.

[3] Crosso M, Zunino A, Campo M. AWSC: An approach to Web serviceclassification based machine learning techniques. Inteligencia Artificial, 2008,12(37):25−36.

[4] Katakis I, Meditskos G, Tsoumakas G, *et al*. On the combination of textual and semantic descriptions for automated semantic Web service classification. In: Proc. of the IFIP Int'l Conf. on Artificial Intelligence Applications and Innovations. Boston: Springer-Verlag, 2009. 95−104.

[5] Shi M, Liu JX, Zhou D, *et al*. Web service clustering method based on multiple relational topic model. Journal of Computer Science, 2019,42(4):820−836 (in Chinese with English abstract).

[6] Cao Y, Liu J, Cao B, *et al*. Web services classification with topical attention based Bi-LSTM. In: Proc. of the Int'l Conf. on Collaborative Computing: Networking, Applications and Worksharing. Cham: Springer-Verlag, 2019. 394−407.

[7] Chen J, Cao B, Cao Y, *et al*. A mobile application classification method with enhanced topic attention mechanism. In: Proc. of the CCF Conf. on Computer Supported Cooperative Work and Social Computing. Singapore: Springer-Verlag, 2019. 683−695.

[8] Perozzi B, Al-Rfou R, Skiena S. Deepwalk: Online learning of social representations. In: Proc. of the 20th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. ACM, 2014. 701−710.

[9] Grover A, Leskovec J. Node2vec: Scalable feature learning for networks. In: Proc. of the 22nd ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. ACM, 2016. 855−864.

[10] Tang J, Qu M, Wang M, *et al*. Line: Large-scale information network embedding. In: Proc. of the 24th Int'l Conf. on World Wide Web. Int'l World Wide Web Conferences Steering Committee, 2015. 1067−1077.

[11] Yang C, Liu Z, Zhao D, *et al*. Network representation learning with rich text information. In: Proc. of the 24th Int'l Joint Conf. on Artificial Intelligence. 2015. 2111−2117.

[12] Pan S, Wu J, Zhu X, *et al*. Tri-party deep network representation. Network, 2016,11(9):12.

[13] Sheikh N, Kefato Z, Montresor A. Gat2vec: Representation learning for attributed graphs. Computing, 2019,101(3):187−209.

[14] Tang J, Qu M, Mei Q. Pte: Predictive text embedding through large-scale heterogeneous text networks. In: Proc. of the 21th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. ACM, 2015. 1165−1174.

[15] Wang D, Cui P, Zhu W. Structural deep network embedding. In: Proc. of the 22nd ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. ACM, 2016. 1225−1234.

[16] Morin F, Bengio Y. Hierarchical probabilistic neural network language model. Aistats, 2005,5:246−252.

[17] Le Q, Mikolov T. Distributed representations of sentences and documents. In: Proc. of the Int'l Conf. on Machine Learning. 2014. 1188−1196.

[18] Blei DM, Ng AY, Jordan MI. Latent dirichlet allocation. Journal of Machine Learning Research, 2003,3(Jan.):993−1022.

[19] Maaten LV, Hinton G. Visualizing data using *t*-sne. Journal of Machine Learning Research, 2008,9(9):2579−605.

[20] Liu C. Research on a feature vector-based Web service discovery algorithm [Master's Thesis]. Changchun: Jilin University, 2011 (in Chinese with English abstract).

[21] Hou J, Wen Y. Utilizing tags for scientific workflow recommendation. In: Proc. of the Int'l Conf. on Applications and Techniques in Cyber Security and Intelligence. Cham: Springer-Verlag, 2019. 951−958.

[22] Hou J, Wen Y. Prediction of learners' academic performance using factorization machine and decision tree. In: Proc. of the 2019 Int'l Conf. on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData). IEEE, 2019. 1−8.

[23] Kuang L, Wu J, Deng S, *et al*. Service classification using adaptive back-propagation neural network and semantic similarity. In: Proc. of the 2006 10th Int'l Conf. on Computer Supported Cooperative Work in Design. IEEE, 2006. 1−5.

[24] Corella MÁ, Castells P. Semi-automatic semantic-based Web service classification. In: Proc. of the Int'l Conf. on Business Process Management. Berlin, Heidelberg: Springer-Verlag, 2006. 459−470.

[25] Kumar S, Mishra RB. Towards a framework for classification and recommendation of semantic Web service composition approaches. Int'l Journal of Computers and Applications, 2009,31(4):274–281.

[26] Moraru A, Fortuna C, Fortuna B, *et al*. A hybrid approach to QoS-aware Web service classification and recommendation. In: Proc. of the 2009 IEEE 5th Int'l Conf. on Intelligent Computer Communication and Processing. IEEE, 2009. 343–346.

[27] Makhlughian M, Hashemi SM, Rastegari Y, *et al*. Web service selection based on ranking of QoS using associative classification. arXiv preprint arXiv:1204.1425, 2012.

[28] TF MR, SivaPragasam P, BalaKrishnan R, *et al*. QoS based classification using *K*-nearest neighbor algorithm for effective Web service selection. In: Proc. of the 2015 IEEE Int'l Conf. on Electrical, Computer and Communication Technologies (ICECCT). IEEE, 2015. 1–4.

[29] Shi M, Liu J, Cao B, *et al*. A prior knowledge based approach to improving accuracy of Web services clustering. In: Proc. of the 2018 IEEE Int'l Conf. on Services Computing (SCC). IEEE, 2018. 1–8.

[30] Mikolov T, Chen K, Corrado G, *et al*. Efficient estimation of word representations in vector space. CoRR. arXiv:1301.3781, 2013.

[31] Le QV, Mikolov T. Distributed representations of sentences and documents. CoRR.arXiv:1405.4053, 2014.

[32] Huang X, Li J, Hu X. Label informed attributed network embedding. In: Proc. of the 10th ACM Int'l Conf. on Web Search and Data Mining. ACM, 2017. 731–739.

[33] Dong Y, Chawla NV, Swami A. Metapath2vec: Scalable representation learning for heterogeneous networks. In: Proc. of the 23rd ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. ACM, 2017. 135–144.

**附中文参考文献:**

[5] 石敏,刘建勋,周栋,等.基于多重关系主题模型的 Web 服务聚类方法.计算机学报,2019,42(4):820–836.

[20] 刘超.一种基于特征向量的 Web 服务发现算法研究[硕士学位论文].长春:吉林大学,2011.

**肖勇**(1995－),男,硕士生,CCF 学生会员,主要研究领域为服务计算与云计算,大数据处理,GIS 与移动计算.

**刘建勋**(1970－),男,博士,教授,博士生导师,CCF 杰出会员,主要研究领域为服务计算与云计算,大数据处理,GIS 与移动计算.

**胡蓉**(1977－),女,博士,副教授,CCF 专业会员,主要研究领域为服务计算,数据挖掘.

**曹步清**(1979－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为服务计算与云计算,社会网络与软件工程.

**曹应成**(1994－),男,硕士生,主要研究领域为服务计算与云计算,大数据处理,GIS 与移动计算.