Computer Science and Application 计算机科学与应用, 2023, 13(12), 2613-2622
Published Online December 2023 in Hans. https://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2023.1312260

Hans汉斯

基于知识图谱嵌入的多模型融合与多任务学习的工业软件组件推荐方法

钟卓伦

广东工业大学自动化学院，广东 广州

收稿日期：2023年11月27日；录用日期：2023年12月23日；发布日期：2023年12月30日

摘　要

随着物联网、云计算、工业互联网等的快速发展，工业软件系统逐渐向扁平化、松耦合、平台化、模块化与组件化的架构模式发展。工业软件组件已逐步演变为跨域、跨空间、跨平台的网络化协同新模式。然而，由于工业软件组件数量的不断的增加，为创建工业软件开发挑选合适的工业软件组件变得更加困难。为了解决这个问题，已经提出了各种方法来推荐工业软件组件以匹配子任务的需求。但目前的方法在特征融合与利用、文本需求理解、子任务与组件的类别等元数据信息利用等方面存在一些挑战。为了克服现有服务推荐模型推荐效率不高以及学习能力不强的问题，本文提出了一种新的基于知识图谱嵌入的多模型融合与多任务学习的工业软件组件推荐模型(KG-MTFM)。模型利用语义特征提取模块来提取子任务需求与工业软件组件描述文档的语义特征，获得语义特征向量。建立组件分类的知识图谱通过表示学习算法获得实体嵌入向量，将其与组件语义特征向量链接形成工业软件组件的特征向量后，引入特征交互模块来对子任务和工业软件组件之间的特征交互进行建模。实验结果表明，我们的方法优于目前流行的主要方法。

关键词

工业软件组件，知识图谱，服务推荐，多模型融合与多任务学习

Multi-Model Fusion and Multi-Task Learning Based on Knowledge Graph Embedding for Industrial Software Component Recommendation

Zhuolun Zhong

School of Automation, Guangdong University of Technology, Guangzhou Guangdong

文章引用: 钟卓伦. 基于知识图谱嵌入的多模型融合与多任务学习的工业软件组件推荐方法[J]. 计算机科学与应用, 2023, 13(12): 2613-2622. DOI: 10.12677/csa.2023.1312260

Received: Nov. 27th, 2023; accepted: Dec. 23rd, 2023; published: Dec. 30th, 2023

Abstract

With the rapid development of the Internet of Things, cloud computing, industrial Internet, etc., industrial software systems are gradually developing into flat, loosely coupled, platform-based, modular and component-based architecture models. Industrial software components have gradually evolved into a new mode of cross-domain, cross-space and cross-platform network collaboration. However, due to the continuous increase in the number of industrial software components, it becomes more difficult to select the right industrial software components for creating industrial software development. To solve this problem, various methods have been proposed to recommend industrial software components to match the needs of subtasks. However, the current methods have some challenges in the aspects of feature fusion and utilization, text requirement understanding, subtask and component classification and other metadata information utilization. In order to overcome the problems of low recommendation efficiency and weak learning ability of existing service recommendation models, this paper proposes a new industrial software component recommendation model (KG-MTFM) based on multi-model fusion and multi-task learning embedded in knowledge graph. The model uses the semantic feature extraction module to extract the semantic features of subtask requirements and industrial software component description documents, and obtain the semantic feature vector. The entity embedding vector is obtained by the representation learning algorithm, which is linked with the component semantic feature vector to form the feature vector of the industrial software component, and the feature interaction module is introduced to model the feature interaction between the subtask and the industrial software component. The experimental results show that our method is superior to the prevailing methods.

Keywords

Industrial Software Components, Knowledge Graph, Service Recommendation, Multi-Model Fusion and Multi-Task Learning

1. 引言

随着近年来物联网、云计算、工业互联网等的快速发展，工业软件系统逐渐向扁平化、松耦合、平台化、模块化与组件化的架构模式发展。工业软件组件已逐步演变为跨域、跨空间、跨平台的网络化协同新模式。工业软件组件的本质是一个小型的应用程序，封装了硬件功能，对外部提供可以被调用的 API，使得用户以及开发者能够通过调用组件以完成特定的功能。通过标准接口，任何兼容的设备都可以在开放的网络环境中轻松调用现有的工业软件组件。

与基于 Mashup 的应用程序开发[1]类似，在工业软件的开发过程中，一个或多个组件可以构成一个集成的子任务工具，让用户可以集成现有的组件资源，以创建复合工业应用程序，并对复杂的业务需求做出反应。与传统的方式相比，利用工业软件组件与子任务工具进行开发使得开发人员不需要在开发新的应用程序时重新从头开始编写代码[2]。开发人员可以从组件库中浏览和寻找潜在有用的工业软件组件，然后集中利用由几个组件构成的子任务工具，快速地将实现所需要功能的工业软件组件应用到他们的工

业软件应用程序中。随着这一技术的发展，现有工业软件组件的数量在过去几年中一直在迅速增长。对于开发人员来说，人工挑选大量的候选工业软件组件也变得越来越困难。与代码推荐[3]和软件库推荐[4]类似，组件推荐技术可以向开发人员推荐适合工业软件具体工业子任务开发的工业软件组件，从而极大的提高组件的利用率以及工业软件的开发效率。

如今，已有很多方法被用在了推荐任务上。这些方法可以概括为基于协同过滤的方法[5] [6]、基于内容的方法[7] [8]以及基于知识图谱的推荐方法[9] [10]。这些方法都起到了不错的效果，但仍然存在以下问题，例如：在特征融合和利用方面做的不足，且当在文本要求中没有提供足够的信息或用户输入描述文本信息不精准时，推荐的结果往往不令人满意。且大多数模型的学习能力还有着很大的提升空间。所以，如何结合各种算法的优势，取长补短，提高推荐的精准性以及如何利用多种元数据来优化推荐模型是一个急切需要解决的难题。

为了克服现有的问题，本文提出了一种基于知识图谱嵌入的多模型融合与多任务学习的工业软件组件推荐模型(KG-MTFM)。模型的核心思想为：首先，利用语义组件来提取子任务需求与工业软件组件描述文档的语义特征，获得语义特征向量。并建立组件分类的知识图谱通过表示学习算法获得实体嵌入向量，将其与组件语义特征向量链接形成工业软件组件的特征向量后，引入特征交互模块来对子任务和工业软件组件之间的特征交互进行建模。

对于工业软件组件推荐任务，我们将其转化为一个多标签分类问题。通过预测子任务需求与所有组件之间的相关性得分来进行排序，从而生成推荐列表。同时，考虑到类别对于工业软件组件和子任务的重要性，我们增加了一个类别判断任务，使我们的框架具有多任务学习能力。多任务学习充分利用多个相关任务训练信号中隐含的特定领域信息的归纳迁移方法，通过确定子任务的类别，将类别判断任务中隐含的训练信号信息作为归纳偏差，提高组件推荐能力。同时，它也可以实现同时推荐工业软件组件和判断子任务的类别。

2. 方法概述

工业软件组件推荐过程分为三个步骤：首先，我们需要构建推荐模型；然后，利用收集的工业软件组件和子任务的相关元数据对推荐模型进行训练;之后，开发人员提交他们用自然语言描述的需求，推荐系统将生成一个候选的工业软件组件集合，同时提出子任务的类别。模型的整体框架图如图 1 所示。在本节中，我们将详细讨论我们的方法与模型构成。

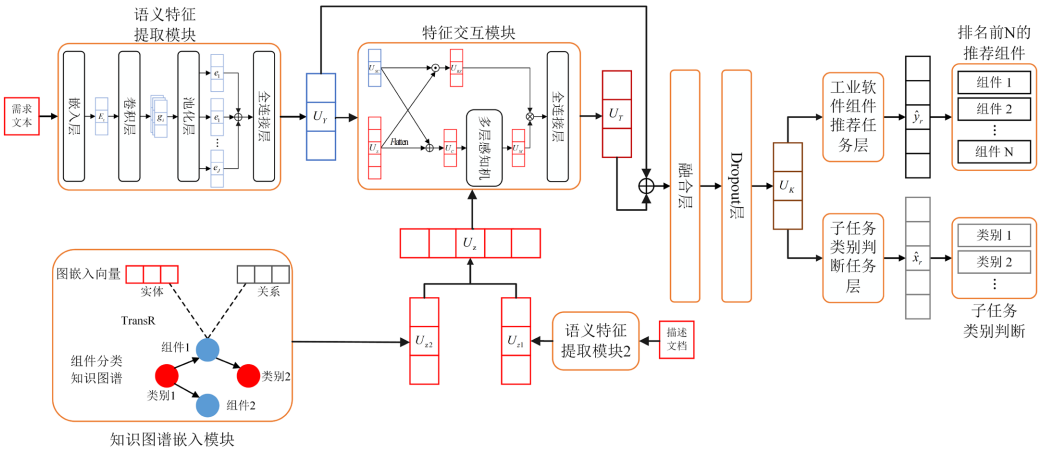


Figure 1. The model diagram of KG-MTFM
图 1. KG-MTFM 模型图

2.1. 问题定义

$r=\{D_r,C_r,T_r,V\}$ 是一个子任务，$r\in R$ 由一个文本描述 D_r 来介绍其功能,一组标签 T_r 来索引它，一组服务组件 C_r 参与组成子任务，以及为子任务做准备的一组元素 V。V 包含数据管道，指定组件图形化排列的布局，在工作中它是可选的。

$z=\langle D_z,T_z,I,O\rangle$ 是一个工业软件组件，$z\in Z$ 由文本描述 D_z 来介绍其功能，一组类别/标签 T_z 来对其进行索引，一组输入端口 I 携带数据进入工业软件组件，一组输出端口 O 携带工业软件组件生成的数据。每个工业软件组件至少有一个输入或输出端口。

w 是一个文本文档，表示创建子任务的用户需求。

将推荐模型定义为一个非线性参数化函数 $\hat{y}=f(\theta,w,Z)$，在确定模型参数 θ 的值后，我们可以使用函数 f 来估计 w 和 Z 之间的相关性得分，然后根据 $\hat{y}\subseteq\{0,1\}^{|z|}$ 对所有工业软件组件进行排名。

2.2. 算法模型

2.2.1. 语义特征提取模块

在子任务、工业软件组件和需求的文本描述中，包含了比其他元数据更丰富的信息。因此，在基于内容的推荐方法中，文本建模或表示技术是必不可少的。在本文中，我们使用 CNN 作为文本文档特征提取的基本组件。在次之前，由于子任务、需求和工业软件组件的文本描述通常包含噪声信息，不能直接用于模型学习。通常需要进行文本过滤、缩写替代、去除单词词缀以及单词矢量化等处理来保留文本中的重要信息。

CNN 的核心思想是从由多个单词组成的滑动窗口中捕获局部特征。对于语义组件，它可以自动组合和过滤特征，以获得不同抽象层次的语义信息。让表示子任务或需求文本 r 对应的词嵌入矩阵，词向量的维数 s 由预训练的语言模型确定。

卷积层中的卷积核 $G_i\in R^{\omega s_j\times d}$ 首先在大小为 ωs_j 的字窗上滑动，执行卷积操作

$$g_i=relu(G_iE_r+d_i) \tag{1}$$

其中 d_i 为偏置项，$relu$ (Rectified Linear Unit)作为激活函数。卷积运算后，每个卷积核得到一个特征图 g_i。然后，为了提取重要的特征，我们对所有的特征映射进行最大池化(Max-Pooling)操作，简化特征的冗余细节，然后将已经池化的特征进行如下连接：

$$e_j=\left[MP(g_i);\cdots;MP(g_I)\right] \tag{2}$$

其中 I 是卷积核的个数。通常，我们需要不同大小的窗口来增强特征提取。在这种情况下，多个窗口的结果也将通过 $e=\left[e_1;e_2;\cdots;e_j;\cdots;e_J\right]$ 连接在一起；其中 e_j 对应大小为 ωs_j 的单词窗口的结果。最后，我们将 e 输入全连接层，实现进一步的非线性变换，得到模块的最终输出：

$$U_Y=sigmoid(H_0e+d_0) \tag{3}$$

其中 H_0 为权重矩阵，d_0 为偏置向量，$sigmoid$ 为激活函数。

2.2.2. 知识图谱嵌入模块

在工业软件的开发中，各种组件之间存在着各种依赖关系，例如组件 1 和组件 2 虽具体功能不同，但都属于机械手运动的类别，如果按照类别做一个聚合的话，若干个组件之间就出现了共性。这种依赖关系可以通过图数据进行存储，例如知识图谱，因此在工业软件组件的推荐系统的建模中，考虑引入知识图谱进行相关性增强，提升推荐系统的模型性能。

本文中，我们首先使用结构化数据构建一个组件分类知识图谱(如图 2 所示)。三元组标识组件归属于某一类，表示为(组件、归属、类别)。因此，图网络包含一条表示组件和归属于的类别之间的“归属于”的边。随后，利用知识图嵌入模型，将表示知识图中完整实体和关系的三元组输入到知识图嵌入模型中，使用 TransR [11]，得到所有实体和关系的向量表示。

TransR 的基本思想为：在平面直角坐标系中，向量表示三元组知识(h,γ,t)。其中 h 表示的是头实体的向量表示；γ 表示的是关系的向量表示；t 指代的是尾实体的向量表示。对于每个三元组(h,γ,t)，将实体空间中的实体通过矩阵 M_γ 投影到 γ 关系空间中，分别为 h_γ 和 t_γ：

$$h_\gamma = hM, t_\gamma = tM_\gamma \tag{4}$$

然后进行得分计算，满足 $h_\gamma + \gamma = t_\gamma$，其损失函数定义为：

$$\mu_\gamma(h,t) + \left\| h_\gamma + \gamma - t \right\|_{L_1/L_2} \tag{5}$$

经过训练后，可以生成一个表示知识图谱中所有实体的嵌入向量 U_{z2} 用于后期的特征交互。

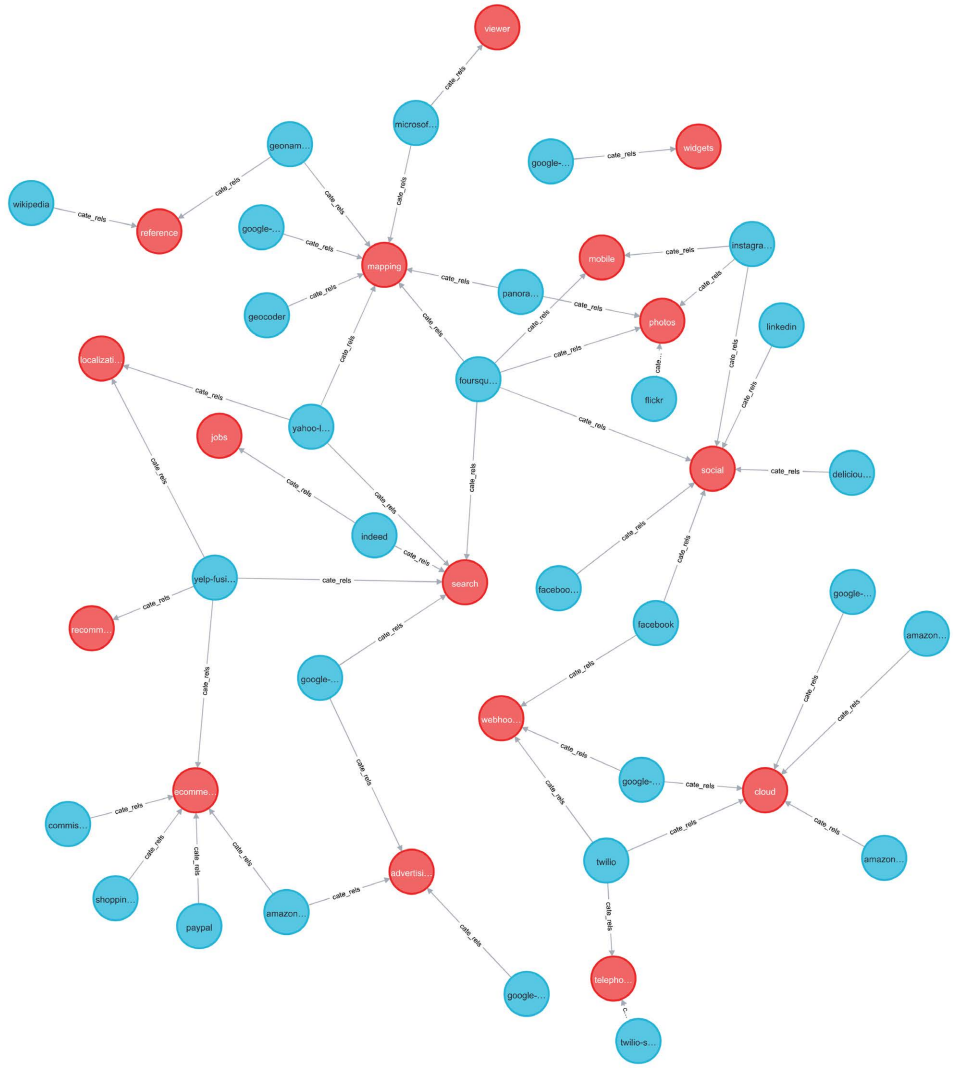


Figure 2. Knowledge graph of component classification
图 2. 组件分类知识图谱

2.2.3. 特征交互模块

在本模块中，我们对子任务描述和组件的相关信息进行特征交互。在 2.2.1 中获得的输出向量U_Y为子任务 r 的嵌入向量，U_z为所有工业软件组件的嵌入向量。U_z由两部分构成：U_{z1}是组件的描述文本经过与 2.2.1 中相同的另一个语义特征提取模块生成的嵌入向量，U_{z2}则是 2.2.2 中知识图谱嵌入模块输出的实体向量。我们将这两个向量连接到每个组件，最终得到$U_z=[U_{z1};U_{z2}]$。

随后，我们生成子任务与工业软件组件的交互向量如下：

$$U_{RZ}=U_Y\cdot U_z^{\mathrm{T}} \tag{6}$$

这种线性操作可以记忆历史数据中那些常见的和高度频繁的交互模式。但是对于子任务和工业软件组件之间那些不常见但有用的交互进行建模的方面却并未关注。且表征低阶交互特征的最佳矢量维数可能与高阶交互特征的矢量维数不同。所以我们再使用 *Concat* 操作来进行特征融合。

$$U_C=\left[U_Y;Flatten\left(U_z\right)\right] \tag{7}$$

其中，$Flatten\left(U_z\right)$是一个将U_z变换成一维向量的平坦运算。之后，将U_C送入多层感知机(MLP)，学习子任务与工业软件组件特性的高阶交互，具体如下：

$$U_M=\delta\left(H_n\left(\cdots\delta\left(H_1U_C+d_1\right)\cdots+d_n\right)\right) \tag{8}$$

其中，H_n和d_n表示 MLP 第 n 层的权重和偏置向量，δ是激活函数。

最后，我们将前两个步骤获得的特征向量U_{RZ}和U_M连接起来得到基于矩阵分解的特征学习模块的最终输出U_T。

2.2.4. 多模型融合与多任务学习

多模型融合的策略是将不同类型的特征或模型进行融合，以提高神经网络的性能。在本文中，我们引入了语义组件和特征交互组件，在生成了不同的方向和角度对应的特征向量后，我们采用多模型融合对特征向量进行聚合。采用早期融合的策略，紧接着是下游任务。

$$\hat{y}_r=sigmoid\left(H_{TL}U_K+d_{TL}\right) \tag{9}$$

其中$U_K=[U_Y;U_T]$，H_{TL}和d_{TL}分别是特定任务层中的权重矩阵和偏置向量。为了抑制模型的过拟合，我们采用了 Dropout 的方法。

多任务学习是一种充分利用多个相关任务训练信号中隐含的特定领域信息的归纳迁移方法。在反向传播过程中，多任务学习允许某一任务专用的共享隐藏层中的特征被其他任务使用，从而实现隐式数据增强和协正则化的效果，这对深度神经网络的模型训练非常有益。为了使我们的模型具有多任务学习的能力，我们引入了两个特定的任务层。

对于工业软件组件推荐这一主要任务，我们将其转化为一个多标签分类问题[12]，再将其分解为多个独立的二元分类问题。集合 Z 中的每个工业软件组件都被视为一个单独的标签。我们训练一个分类器，该分类器可以用 z 的最相关子集自动标记子任务 r。为了通过主要任务训练我们的模型，采用二进制交叉熵损失，其定义如下：

$$L_{RT}\left(\hat{y}_r,y_r\right)=-\frac{1}{|Z|}\sum_{z\in Z}y_r[z]\log\left(\hat{y}_r[z]\right)+\left(1-y_r[z]\log\left(1-\hat{y}_r[z]\right)\right) \tag{10}$$

其中$y_r[z]$对应于目标值，该目标值表示该组件是否符合子任务的要求 r (如果 r 调用 z，则$y_r[z]=1$，否则为 0)，$\hat{y}_r[z]$是对应于$y_r[z]$的预测值。

模型经过训练后，我们可以得到预测相关分数$\hat{y}_r\subseteq\{0,1\}^{|Z|}$，并根据这些分数对所有组件进行排序，

从而生成子任务需求 r 的候选组件的推荐列表。

同时，考虑到类别对子任务和工业软件组件的重要性，我们增加了一个类别判断任务，使我们的框架具有多任务学习能力。与工业软件组件推荐任务类似，我们也将该辅助任务转化为多个独立的二分类问题，每一个类别 $b \in B$ 并采用 BCE 损失函数对模型参数进行优化。

$$L_{ZT}\left(\hat{x}_r, x_r\right) = -\frac{1}{|Z|}\sum_{b\in B} x_r[b]\log\left(\hat{x}_r[b]\right) + \left(1 - x_r[b]\log\left(1-\hat{x}_r[b]\right)\right) \tag{11}$$

通过对两个任务的损失函数进行汇总并施加正则化约束，我们可以得到模型的目标函数为：

$$\min \frac{1}{|R|}\sum_{r\in R}\left(L_{RT} + L_{ZT}\right) + \lambda\left\|\theta'\right\|_2^2 \tag{12}$$

其中 $\theta' = \theta \cup \{H_1, d_1, H_2, d_2, \cdots, H_z, d_z, \cdots, U\}$ 为模型参数，$\lambda\left\|\theta'\right\|_2^2$ 为 $L2$ 正则化项，用于抑制模型过拟合。

3. 实验与分析

在本节中，我们将进行一系列实验来评估我们提出的方法，并通过分析进行评价。

3.1. 数据集

本章中，我们选择了一个符合描述的真实世界数据集进行验证实验，使用 Mashup 类比子任务，Web API 类比工业软件组件。了确保我们实验结果的可靠性，数据集来自对 ProgrammableWeb 网站的爬取，我们共爬取了 22,642 个 Web API。和 8484 个 Mashup。对于每个 Mashup，我们获得四种元数据：名称、描述、类别和 API 调用信息。对于每个 API，我们获得它的三种元数据：名称、描述和类别。由于缺少上述一个或多个相应的主元数据，我们删除了 267 个无效 Mashup 和 20,995 个无效 API。处理后，我们的数据集包括 499 个应用程序类别 8217 个 Mashup、1647 个至少被 Mashup 调用过一次的 API。最后，我们将数据集随机分为训练集和测试集，用于性能测试。

3.2. 评价指标

在我们的场景中，用户可以通过编写和提交查询来进行推荐工业软件组件的请求，然后我们的推荐系统根据相似性向用户返回前 N 个相关的组件。

在本文的实验中我们采用了三个广泛使用的准确性指标：*Precision@N*、*Recall@N* 和 *F1@N*。并将推荐列表的长度设置为 $N = \{5,10,15,20,25,30\}$。具体的评价指标如下：

(1) *Precision@N* 是一种评价推荐系统的精度类型，它表示在所有推荐服务中有多少是被标记原来属于目标子任务的。计算方法如下：

$$Precision@N = \frac{|realZs \cap TOPNZs|}{|TOPNZs|} \tag{13}$$

(2) *Recall@N* 指目标子任务标记的组件中有多少在长度为 N 推荐列表中。计算方法如下：

$$Recall@N = \frac{|realZs \cap TOPNZs|}{|realNZs|} \tag{14}$$

(3) *F1@N* 采用加权调和平均来融合准确率及召回率，避免了两者出现矛盾时无法判断推荐算法优劣的情况，从而可以更好更全面更均衡地去衡量算法性能，计算方法如下：

$$F1@N = \frac{2 \times Precision@N \times Recall@N}{Precision@N + Recall@N} \tag{15}$$

3.3. 对比方法

在实验中，为了更好地验证我们所提出的模型的有效性，本文将我们提出的模型与下列目前常用的几种服务推荐方法进行实验对比。用于对比研究的方法具体如下：

神经协同过滤(NCF)。NCF 是推荐系统中的一种经典神经模型。使用带有 Glove.6B 的语义组件初始化子任务的嵌入。300d 预训练的词向量，并通过组件和子任务之间调用的隐式反馈来训练模型。

贝叶斯个性化排名(BPR)，BPR 被广泛用于预测用户对不同产品的偏好。给定一对 z_i 和 z_j，如果子任务 r 调用 z_i，那么我们得到一个三元组(r,z_i,z_j)，这意味着对于 r, z_i 的排名高于 z_j。初始化子任务的嵌入方式与神经协同过滤的方法中相同。

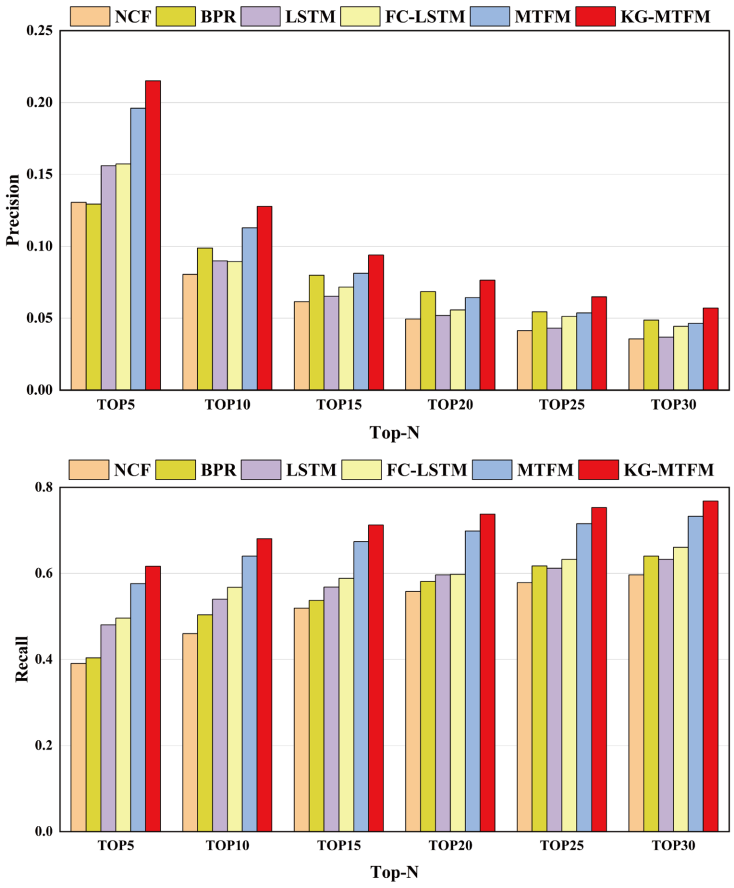
长短期记忆模型(LSTM)。LSTM 作为语义组件被广泛用于构建分类器。在实验中，我们使用带有注意力的双向 LSTM 来提取组件描述的特征向量。该模型使用多标签软边际损失函数进行优化。

基于功能和上下文注意的长短期记忆模型(FC-LSTM)。该模型具有功能注意机制和上下文注意机制，以帮助选择合适的服务。

多任务融合模型(MTFM) [13]。多模型融合的神经网络框架(MTFM)利用语义组件来生成需求的表示，采用基于矩阵分解的特征交互组件，来在子任务和组件的调用矩阵上建模子任务和组件之间的功能交互。

3.4. 结果与分析

图 3 是本文方法和其他对比方法的比较图，展示的是六种方法 *Precision@N* 的值、*Recall@N* 的值与 *F*1*@N* 的值。



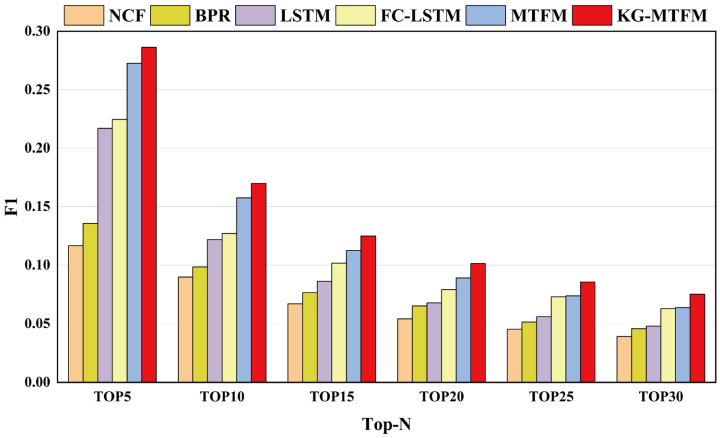


Figure 3. Performance comparison of each method
图 3. 各方法的性能比较

从结果来看，在所选取的对比方法中，神经协同过滤模型表现最差，这是因为基于协同过滤的方法通常比较依赖描述文档的信息来查找具有类似功能的子任务，但是数据集中有关于子任务描述都相对简单，甚至存在信息缺漏的情况，这严重影响了推荐的效果。

相较于基于协同过滤的方法，贝叶斯个性化排名模型的优势在于直接进行排名优化。而长短期记忆模型的优势在于双向长短期记忆具有优越的编码结构和注意机制。基于功能和上下文注意的长短期记忆模型在前一方法的基础上进行了强化，拥有功能和上下文注意两种注意力机制，因此性能强于基础的长短期记忆模型。所以，从图中可以得知，这三种方法的性能优于神经协同过滤模型，但仍然存在着较大的提升空间。

多任务融合模型既有基于协同过滤的方法的优点，也有基于内容的方法的优点，并且利用多任务学习使得模型性能进一步提升。从图中结果可知，在各项指标上都相较于前四种方法有了显著的提升。

而我们所提出的基于知识图谱嵌入的多模型融合与多任务学习的工业软件组件推荐方法，通过在多模型融合与多任务学习方法的基础上，引入了基于知识图谱嵌入的组件分类信息，以及通过语义组件获取的基于组件描述文本的语义特征向量，来构成表示组件的特征向量，进一步强化了模型的性能。由于在数据集中大量的子任务都仅仅调用数量较少的组件，所以在分析中，我们主要考虑推荐列表长度 N 为 5 时候的性能。当 N 为 5 时，我们可以发现不论是 *Precision*、*Recall* 还是 $F1$ 的值，本文提出的方法都要优于其他各个方法，相较于性能已经较好的 MTFM 模型也有着不错的提升。

4. 总结与展望

基于工业软件组件组合应用的工业软件开发模式仅需整合已经开发的现有的工业软件组件，并不需要开发者拥有高超的编程技巧，可以让非专业开发人员但是却对工业流程与业务十分熟悉的技术人员可以使用由一系列工业软件组件组成的子任务工具实现软件的开发，以获得所需的功能。这使得这种新兴开发方式开始变得热门起来。然而，随着时间的发展，工业软件组件的数量和种类愈发的多样化，如何为工业软件的开发者推荐合适的组件成为了一个急需解决的问题。然而，现有的推荐法均存在着一些不足，本文的工作主要是针对这些不足，结合知识图谱表示学习等技术提出了基于知识图谱嵌入的多模型融合与多任务学习的工业软件组件推荐方法。

我们提出的模型通过语义特征提取模块，对描述性信息进行特征提取，并使用知识图谱嵌入模块来获取关于工业软件组件的嵌入向量，然后通过特征交互模块进行子任务和工业软件组件的特征交互。基

于融合特性，我们将子任务分类判断作为辅助任务，通过归纳偏置提高组件推荐这一主要任务的性能。实验结果表明，本文所提出的模型可以有效提高组件推荐的精确性。

目前还有很多需要研究的地方。我们可以继续探索多模型融合多任务模型在工业软件组件推荐任务中的可能性，以及引入内容更丰富的知识图谱来增强模型等，使推荐更加地准确。

参考文献

[1] 高永兵, 吴纪磊, 胡文江, 魏晓东. 基于 Web 服务的 Mashup 应用的研究与实现[J]. 计算机技术与发展, 2010, 20(6): 137-140, 151.

[2] Chowdhury, R.S., Daniel, F. and Casati, F. (2014) Recommendation and Weaving of Reusable Mashup Model Patterns for Assisted Development. *ACM Transactions on Internet Technology* (*TOIT*), **14**, 1-23. https://doi.org/10.1145/2663500

[3] Nguyen, T., Vu, P. and Nguyen, T. (2019) Personalized Code Recommendation. 2019 *IEEE International Conference on Software Maintenance and Evolution* (*ICSME*), Cleveland, 29 September-4 October 2019, 313-317. https://doi.org/10.1109/ICSME.2019.00047

[4] Ouni, A., Kula, R.G., Kessentini, M., *et al*. (2017) Search-Based Software Library Recommendation Using Multi-Objective Optimization. *Information and Software Technology*, **83**, 55-75. https://doi.org/10.1016/j.infsof.2016.11.007

[5] Zheng, Z., Ma, H., Lyu, M.R., *et al*. (2009) Wsrec: A Collaborative Filtering Based Web Service Recommender System. 2009 *IEEE International Conference on Web Services*. Los Angeles, 6-10 July 2009, 437-444. https://doi.org/10.1109/ICWS.2009.30

[6] Wu, H., Yue, K., Li, B., *et al*. (2018) Collaborative QoS Prediction with Context-Sensitive Matrix Factorization. *Future Generation Computer Systems*, **82**, 669-678. https://doi.org/10.1016/j.future.2017.06.020

[7] Xia, B., Fan, Y., Tan, W., Huang, K., Zhang, J. and Wu, C. (2015) Category-Aware API Clustering and Distributed Recommendation for Automatic Mashup Creation. *IEEE Transactions on Services Computing*, **8**, 674-687. https://doi.org/10.1109/TSC.2014.2379251

[8] Cao, B., *et al*. (2014) CSCF: A Mashup Service Recommendation Approach based on Content Similarity and Collaborative Filtering. *International Journal of Grid and Distributed Computing*, **7**, 163-172. https://doi.org/10.14257/ijgdc.2014.7.2.15

[9] Mezni, H. (2022) Temporal Knowledge Graph Embedding for Effective Service Recommendation. *IEEE Transactions on Services Computing*, **15**, 3077-3088. https://doi.org/10.1109/TSC.2021.3075053

[10] Cao, Z., Qiao, X., Jiang, S., *et al*. (2019) An Efficient Knowledge-Graph-Based Web Service Recommendation Algorithm. *Symmetry*, **11**, 392. https://doi.org/10.3390/sym11030392

[11] Ji, G., He, S., Xu, L., *et al*. (2015) Knowledge Graph Embedding via Dynamic Mapping Matrix. *Proceedings of the* 53*rd Annual Meeting of the Association for Computational Linguistics and the* 7*th International Joint Conference on Natural Language Processing* (*Volume* 1: *Long Papers*), Beijing, 26-31 July 2015, 687-696. https://doi.org/10.3115/v1/P15-1067

[12] Zhang, M.L. and Zhou, Z.H. (2013) A Review on Multi-Label Learning Algorithms. *IEEE Transactions on Knowledge and Data Engineering*, **26**, 1819-1837. https://doi.org/10.1109/TKDE.2013.39

[13] Wu, H., Duan, Y., Yue, K., *et al*. (2021) Mashup-Oriented Web API Recommendation via Multi-Model Fusion and Multi-Task Learning. *IEEE Transactions on Services Computing*, **15**, 3330-3343. https://doi.org/10.1109/TSC.2021.3098756