第5章

基于span级对比表示学习的软件知识实体和关系联合抽取方法

5.1 引言

第4章在软件知识实体识别的基础上，根据预定义的实体关系类型对软件知识实体对的语义关系进行分类，实现了软件知识实体和关系的抽取任务。此类基于流水线式的方法将实体抽取和关系抽取建模为两个彼此独立的任务，模型实现较为简单和灵活，但忽略了两个任务的彼此交互和关联，带来了一些特有的问题[36]。首先，由于实体抽取和关系抽取的串行任务关系，上一阶段任务的质量好坏直接影响下一阶段任务的性能。其次，实体抽取和关系抽取采用分开独立建模，模型训练过程中的参数不共享、信息不交互，造成语义信息和依赖信息丢失，会产生没有匹配关系的冗余实体，使模型的错误率增大，影响任务的整体性能。

针对上述问题，实体和关系联合学习方法通过端到端的方式将实体抽取任务和关系分类任务建模为一个任务，利用联合损失函数来增强任务关联和信息共享，模型的整体性能得到了提升，发展成信息抽取任务的主流方法[37,38]。

根据建模对象的不同，基于联合学习方法的实体关系抽取主要分为基于参数共享的方法和基于序列标注的方法。其中，基于参数共享的方法[52,53]能缓解错误传播和任务间关系依赖被忽视的问题，但由于该类方法本质上还是继承了两个子任务之间的先后关系，会产生没有匹配关系的冗余实体，影响联合抽取的质量。针对冗余实体问题，基于序列标注的方法[55,56]将联合学习模型转换为序列标注模

型，从而实现实体和关系同时抽取[37]。但是，该类方法通常属于Token级的任务，对实体和关系的标签归属进行了限定，造成重叠实体的关系分类依然存在问题。为解决上述问题，基于span的方法[148,149]将句子序列的一个或多个单词建模为span单元，生成所有可能的span，并从span级对句子序列进行建模表示。由于不受句子序列的顺序限制，基于span的方法能对重叠实体进行选择和独立特征表示，较好地缓解了Token级句子序列建模产生的错误传播问题和实体重叠问题。

软件知识社区文本是非结构化的用户生成内容，存在领域实体语义关系复杂的情况，采用传统的序列建模方式进行软件知识实体和关系抽取会带来实体重叠和错误传播问题，影响软件知识图谱构建的质量。当句子中包含多个相互重叠的实体时，就会出现实体重叠问题。例如，在句子序列“*GetHashCode is Method of Base Object Class of .NET Framework*”中，“*.NET*” *and* “*.NET Framework*”就是重叠的实体。

因此，针对传统流水线方法存在的弊端和软件知识社区文本存在的实体重叠问题，我们提出一种基于span级对比表示学习的软件知识实体和关系联合抽取方法。一方面，从建模对象的角度以span为单元对句子序列进行建模，能避免Token级句子序列建模的弊端，有效缓解实体和关系联合抽取任务的实体重叠问题。另一方面，从特征学习的角度在实体抽取和关系抽取两个任务中引入对比表示学习的思想，可以获得实体span和实体对更具区别性的特征表示，提升分类预测的准确性。具体地，我们的主要工作如下：

(1) 针对流水线方法存在的任务依赖问题和软件知识社区文本存在的实体重叠问题，提出一种基于span级对比表示学习的软件知识实体和关系联合抽取模型，从软件知识社区的用户生成内容中联合抽取软件知识实体及其语义关系。具体而言，在span表示层将句子序列建模为span单元，并生成丰富的实体span正、负样本，以避免无法选择重叠实体的问题。

(2) 在实体分类层和关系分类层，提出有监督的实体对比表示学习和关系对比表示学习，通过数据增强和对比损失构建，获取实体、实体对更适于分类的特征表示，从而提升软件知识实体和关系联合抽取的性能。

(3) 设计了一个联合训练算法获取模型训练最优解，并基于软件知识社区文本构建了涵盖8个实体类型和5个关系类型的软件知识实体和关系联合抽取标注数据集。

5.2 相关工作

5.2.1 基于 span 的信息抽取

传统基于序列标注的实体和关系抽取方法大多属于 Token 级的任务，受限于固定单一的表示和模型的序列特征，模型处理过程中会产生级联错误，并无法解决实体和关系重叠问题。为解决上述问题，相继提出基于 span 的方法，并得到广泛应用。基于 span 的信息抽取方法将句子序列中的一个或多个单词作为 span 单元，生成所有可能的 span，并在 span 级对句子序列进行建模。该类方法克服了句子序列固有的序列特征，可以选择重叠的实体并表示特征，缓解了 Token 级句子序列建模带来的错误传播和实体重叠问题。

Dixit 等[148]利用 BiLSTM 提出一个基于 span 的模型用于实体和关系联合抽取，并在 ACE2005 数据集上取得较好的性能。Luan 等[149]提出一种基于共享 span 表示的多任务分类框架，通过实体识别、关系分类和指代消解等任务，在科学文献摘要数据集 SciERC 上构建科学文献知识图谱。为了增强不同任务间的交互，Luan 等[150]提出一种基于动态 span 图的信息抽取框架 DYGIE，该框架通过图传播的方式捕获 span 间的交互信息，在不需要额外句法分析和处理的情况下，提升了实体识别和关系抽取的性能。Wadden 等[151]在 DYGIE 框架的基础上，将编码器替换成 BERT，并通过对文本 span 进行枚举、精炼和打分捕获局部和全局上下文，在实体识别、关系抽取和事件抽取上获得较好的性能。Eberts 等[152]以预训练语言模型 BERT 为核心，通过强负样本采样、span 过滤和局部上下文表示等机制，解决了部分实体重叠问题，提升了实体和关系联合抽取模型的预测精度。Ding 等[153]将实体和关系联合抽取方法应用到特定军事领域，提出一个融合 span 方法和图结构的混合模型，并结合特定领域的词汇和句法知识提升模型的抽取性能。

5.2.2 对比表示学习在自然语言处理的应用

对比表示学习作为一种判别性的自监督学习，利用数据样本的相似性或差异性来学习具有监督信息的编码器；编码器对同一类型的数据尽可能采用相似的特征表示，对不同类型的数据尽可能采用不同的特征表示，以捕获更适合下游任务的特征表示[91]，在自然语言领域、图像领域和图数据领域得到了广泛应用。

在文本表示任务方面，Giorgi 等[154]受深度度量学习(Deep Metric Learning，DML)启发，提出了一种基于对比学习的无监督句子特征表示学习方法，并通过实验证明了对比学习在句子表征任务上的可行性。为了得到更好的句子特征表示，

Gao 等[155]提出一种基于对比表示学习的训练方法，利用 Dropout 等数据增强方式产生正、负样本数据，通过对比损失函数构建，使得语义相近的样本相距尽可能近，而语义不相似的样本相距尽可能远，从而增强句子向量表示；并在下游的 7 个文本语义匹配(Semantic Textual Similarity，STS)任务中取得较好的性能。针对预训练模型 BERT “坍缩”问题，美团知识图谱团队[156]提出一种基于对比表示学习的句子表示迁移框架。该框架通过构建对比学习任务，在目标领域的无标签数据集上对 BERT 预训练模型进行 Fine-tune，使得生成的句子表示更适合下游任务的数据分布。同时，分别在无监督、有监督的情况下对文本语义匹配任务 STS 进行了验证，实验结果表明该方法的性能均超过基准模型。

在实体和关系抽取任务方面，Peng 等[143]为了更好地捕获句子上下文信息和实体类型信息，提出了一种基于实体遮蔽的对比表示学习框架进行关系预训练。该框架利用远程监督的方法通过外部知识图谱生成正、负样本，将具有相同关系的实体对的句子视为正样本，将具有不同关系的实体对的句子视为负样本，并采用随机遮蔽实体的方法进行预训练，获得了较好的关系表示。针对预训练模型不能捕获文本中的事实知识，Qin 等[157]提出一种预训练阶段的对比表示学习框架，通过实体判别和关系判别两个对比学习任务增强模型对实体对及其语义关系的理解，在关系提取、阅读理解和实体分类任务上取得了较好的性能。Su 等[158]通过同义词替换、随机交换和随机删除等数据增强方式，利用对比表示学习的方法改善了预训练模型 BERT 的文本表示，进而提升了生物医学关系抽取的效果。

5.3　模型与方法

5.3.1　任务建模与方法分析

我们提出的基于 span 级软件知识实体和关系联合抽取方法的任务目标是从非结构化的软件知识社区文本中，自动识别所有可能的软件知识实体 span，并预测其对应的实体类型；再根据预定义的关系类型对实体 span 对的语义关系进行分类，从而得到软件知识关系三元组。因此，基于 span 级软件知识实体和关系联合抽取任务可以形式化定义为一个 7 元组 $SKG=(X,S,Y_e,Y_r,\delta,\varepsilon,NA)$，其中：

$X=(x_1,x_2,\cdots,x_n)$为软件知识社区文本的句子序列；

$S=(s_1,s_2,\cdots,s_n)$为枚举 X 产生的一个候选 span 集；

$Y_e(s_i)\in\delta\cup\{NA\}$为预测候选 span 实例 s_i 实体类型的函数，并产生候选软件知识实体集 $E=(e_1,e_2,\cdots,e_{|E|})$，$s_i=(x_i,x_{i+1},\cdots,x_{i+k})$；

$Y_r(e_i,e_j)\in\varepsilon\cup\{NA\}$为预测软件知识实体对$(e_i,e_j)$语义关系的函数，并产

生软件知识实体关系集合 $R=(r_1,r_2,\cdots,r_{|R|}),e_i,e_j\in E$；

δ 为预定义的实体类型集合；

ε 为预定义的关系类型集合；

NA 为非实体或者没有语义关系。

例如，给定软件知识社区文本的句子序列“*GetHashCode is method of base Object class of .Net Framework.*”，软件知识实体和关系联合抽取的目标是准确识别实体对(e_i,e_j)“*GetHashCode*”和“*.Net Framework*”，并预测该实体对之间的语义关系 r_{ij} 为“*inclusion*”，从而获得软件知识关系三元组$\langle e_i,r_{ij},e_j\rangle$。

结合上述软件知识实体和关系联合抽取的任务目标和存在的问题，我们提出一个基于 span 级对比表示学习的软件知识实体和关系联合抽取模型 SCL-SKG，其由输入嵌入层、span 表示层、实体分类层和关系分类层组成，整体架构图如图 5-1 所示。

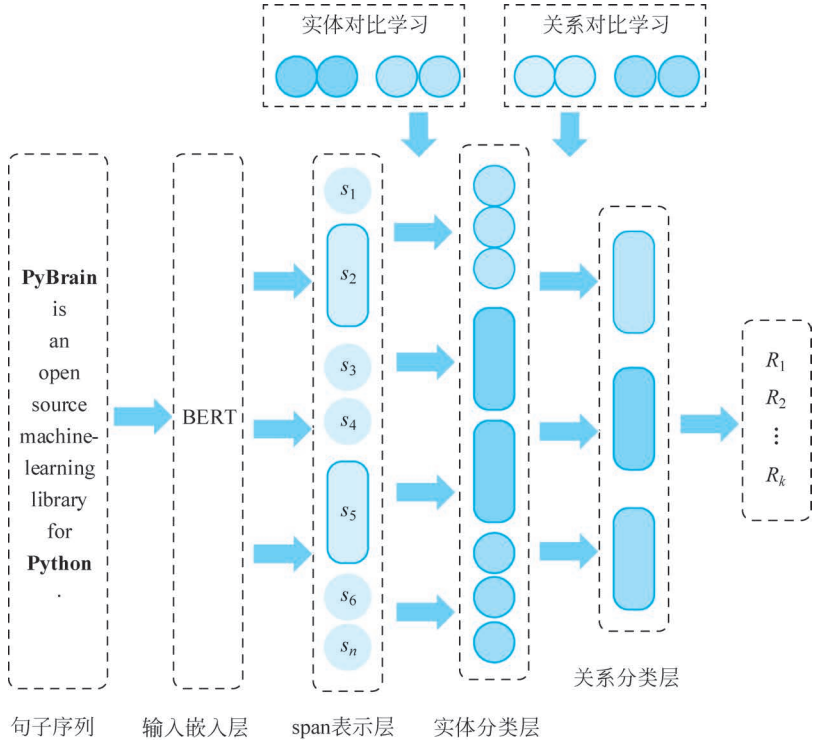

图 5-1 CS-SKG 模型整体架构图

SCL-SKG 模型中各层的主要功能如下：

在输入嵌入层，利用软件工程领域的精调预训练语言模型 SWBERT 对输入句子

序列进行特征编码，生成句子序列的动态词向量表示，并作为 span 表示层的输入。

在 span 表示层，基于词性过滤机制去除含义较少的停用词，保留动词和名词等实体词汇，生成句子序列所有可能的 span 表示，为下一步的对比表示学习产生了丰富的实体 span 正、负样本。

在实体分类层，提出一个 span 级的实体对比表示学习方法，通过正、负样本增强和实体对比损失函数构建，约束模型学习实体 span 更好的特征表示，并根据预定义的实体类型对实体 span 进行分类。

在关系分类层，提出一个关系对比表示学习方法，通过正、负样本增强和关系对比损失函数构建，获得候选实体对更好的特征表示，并根据预定义的关系类型对实体对的语义关系进行预测。

5.3.2　输入嵌入层

为了获取不同语境下句子序列的动态词向量表示，从而提升软件知识实体和关系联合预测的准确性，我们利用前面章节已构建的软件工程领域预训练语言模型 SWBERT 对输入句子序列进行特征编码，捕获句子序列的动态词向量表示。具体过程如下：

(1) 对于软件知识社区文本的输入句子序列 $X=(x_1,x_2,\cdots,x_n)$，经过基于块的分词、句子序列的首尾添加标志符[CLS]和[SEP]，得到句子序列对应的 Token 序列。

(2) 针对 Token 序列的每个 Token 产生 Token 向量、分割向量和位置向量，三个向量经过求和，得到 BERT 模型的输入向量表示，$\boldsymbol{E}=(\boldsymbol{E}_1,\boldsymbol{E}_2,\cdots,\boldsymbol{E}_n)$。

(3) 经过特征编码，得到句子序列 X 的动态词向量表示：

$$\boldsymbol{H}=[h_1,h_2,\cdots,h_n]=\mathrm{SWBERT}(x_1,x_2,\cdots,x_n) \tag{5.1}$$

经过上述预训练语言模型 SWBERT 编码，得到一个长度为 $n+1$ 的词向量序列 $\boldsymbol{W}=(w_{\mathrm{cls}},w_1,w_2,\cdots,w_n)$，其中，$w_{\mathrm{cls}}$ 表示整个句子序列的分类信息。由此，获得软件知识实体和关系联合抽取模型 SCL-SKG 的输入嵌入层的词向量表示，并作为 span 表示层的输入。

5.3.3　span 表示层

针对软件知识社区文本存在的实体重叠问题，我们参考相关工作[148,152]，在软件知识实体和关系联合抽取模型 SCL-SKG 中构建一个面向 span 级的句子序列表示层，以 span 为单元对句子序列进行建模表示。通常，基于 span 的方法采用迭代句子序列中所有单词的方式产生 span 表示，这样会带来模型算法计算开销太大的问题。

为了提升模型计算效率，使得生成的 span 更具领域专业性，将对句子序列的单词进行过滤操作，去除含义较少的停用词，保留动词和名词等实体词汇；并依此生成不同长度的 span 表示，得到实体 span 表示的集合 $S=(s_1,s_2,\cdots,s_n)$，其中，实体 span 实例表示为 $s_i=(x_i,x_{i+1},\cdots,x_{i+k})$，$k$ 为 span 的长度，表示该实体 span 所包含单词的个数。同时，根据对软件知识社区文本语料的观察，实体 span 的长度过长，表示实体的可能性降低，会增加模型计算的复杂度，因此 k 值不宜设置过大。

例如，对于软件知识社区文本的句子序列“*GetHashCode is method of base Object class of. Net Framework.*”，经过过滤操作后的句子序列为“*GetHashCodemethod base Object class. Net Framework.*”，进而生成 span 表示集“*GetHashCode*”“*method*”“*GetHashCodemethod*”“*methodbase*”“*Object*”“*base Object*”“*class*”“*. Net*”“*. Net Framework*”等。

由此，SCL-SKG 模型通过上述实体 span 表示层可以生成句子序列对应的实体 span 表示，并产生了丰富的实体 span 的正、负样本，为下一步的实体 span 对比表示学习提供了数据增强方式。

5.3.4 实体分类层

我们利用“聚合正样本，分离负样本”的特征提取思想，在实体分类层提出一种实体对比表示学习方法，通过正、负样本数据增强和损失函数构建，约束模型学习实体 span 更具区别性的特征表示，进而提升实体 span 分类预测的准确性。因此，在模型的实体分类层包括实体对比表示学习和实体分类两个步骤。

1. 实体对比表示学习

为了利用软件知识实体的标签信息，我们在对比式自监督学习的基础上进行了扩展，提出一种有监督的 span 级实体对比表示学习方法，获取更适于下游任务的实体 span 特征表示。有监督的实体对比表示学习区别于自监督对比表示学习，其利用软件知识实体的标签信息（如实体的类型信息），通过特定的数据增强方法生成有关原始数据样本的多个正样本、负样本视图，并利用对比损失函数构建，约束模型学习更适应于软件知识实体分类任务的实体 span 特征表示，进而提升实体分类的性能。

通常，对比表示学习的通用框架主要包括数据增强、编码器（Encoder）、Projection 网络和对比损失函数四个组件[96]，下面就有监督实体对比表示学习方法涉及的各个组件进行详细介绍。

（1）数据增强。数据增强组件是对比表示学习的关键组成部分，它通过数据增强技术随机生成关于原始数据样本的视图，该视图既要与原始数据样本有所区别，又不能区别太大，否则会破坏原始数据样本的结构信息或语义信息，给模型带

来太大干扰。自然语言领域相较于图像领域，数据增强方法较为困难，通常有随机删除单词、随机插入单词、随机互换单词和近反义词替换等方法[159]；但是，这些方法对句子的结构和语义信息会造成干扰，直接应用到实体抽取、关系抽取等下游任务，会造成模型性能下降。

在有监督实体对比表示学习的数据增强组件中，我们结合 span 表示层产生的实体 span 表示集，利用软件知识实体的类型标签进行数据增强，生成实体 span 实例的多个正、负样本。具体地，有监督实体对比表示学习将同一类型的实体 span 视为正样本，构建实体 span 正样本集 $P(i)$；将同一 batch 内的其他类型实体 span 和非实体 span 作为负样本，构建实体 span 负样本集 $N(i)$。

(2) 编码器(Encoder)。在编码器组件中，SCL-SKG 模型利用预训练语言模型 SWBERT 将输入的软件知识社区文本句子序列 $X=(x_1,x_2,\cdots,x_n)$转换为动态词向量表示，进而提取文本特征，表示为

$$\boldsymbol{h}_i = f(x_i) = \mathrm{SWBERT}(x_i) \tag{5.2}$$

(3) Projection 网络。借鉴 Chen[96] 在图像领域的工作，SCL-SKG 模型在 Projection 网络组件中利用多层感知器将向量表示投射到另一个表示空间，以获得训练阶段更好的特征表示，表示为

$$\boldsymbol{z}_i = g(\boldsymbol{h}_i) = \boldsymbol{W}^2\sigma(\boldsymbol{W}^1\boldsymbol{h}_i) \tag{5.3}$$

式中：$\boldsymbol{W}^1$ 和 $\boldsymbol{W}^2$ 为隐层的权重；σ 为 ReLU 激活函数。

此 Projection 网络组件和向量 $\boldsymbol{z}_i$ 只在对比表示学习的训练过程中使用，训练结束后，将编码器及向量 $\boldsymbol{h}_i$ 用于下游实体分类任务。

(4) 对比损失函数。在对比损失函数组件中，由于自监督对比表示学习的损失函数无法处理实体的类型标签信息，会带来多正样本问题，SCL-SKG 模型参考 Khosla[160] 的工作，对自监督对比损失函数进行了扩展，得到有监督实体对比表示学习的损失函数，表示为

$$L^{ec} = \sum_{i\in B(i)} \frac{-1}{|P(i)|} \sum_{p\in P(i)} \log \frac{\exp(\boldsymbol{z}_i \cdot \boldsymbol{z}_p/\tau)}{\sum\limits_{n\in N(i)} \exp(\boldsymbol{z}_i \cdot \boldsymbol{z}_n/\tau)} \tag{5.4}$$

式中：$\boldsymbol{z}_i$ 为当前实体 span 实例；$\boldsymbol{z}_p$ 为 $\boldsymbol{z}_i$ 的正样本实例；$\boldsymbol{z}_n$ 为 $\boldsymbol{z}_i$ 的负样本实例；$B(i)$为 batch 内的数据样本集；$|P(i)|$为正样本集合的基数；$N(i)$为负样本集合。

通过以上对比损失函数，使得当前实体 span 的实例 $\boldsymbol{z}_i$ 与其正样本集的特征一致性最大化；同时，使得当前实体 span 的实例 $\boldsymbol{z}_i$ 与其负样本集的特征区别性最大化，进而学习到实体 span 更好的特征表示。

至此，基于软件知识社区文本的 span 级实体对比表示学习可以描述为算法 5-1。

算法 5-1　span 级实体对比表示学习算法

Input:实体 span 的实例 s_i,预训练语言模型 SWBERT, Projection 网络 g
Output:实体 span 的实例 s_i 对应的特征向量 $\boldsymbol{h}_i$

```
Begin
    获取实体 span 实例 s_i 的向量表示 h_i                    //式(5.2)
    将向量 h_i 投射到另一个向量表示空间,获得 z_i             //式(5.3)
    按实体 span 数据增强方法,生成 z_i 的正、负样本表示 z_p, z_n
    计算实体对比损失 L^ec                                   //式(5.4)
    更新 SWBERT 和 Projection 网络 g 的参数,使得 L^ec 最小化
    返回实体 span 实例 s_i 的向量表示 h_i
End
```

由算法 5-1 可知,面向软件知识社区的 span 级实体对比表示学习的目标是学习实体 span 更好的特征表示,并将句子序列的实体 span 特征向量 $\boldsymbol{H}=(h_1, h_2, \cdots, h_n)$作为下游实体分类任务的输入,提升实体 span 类型预测的性能。

2. 实体分类

实体分类层的目标是对候选实体 span 的类型进行预测,同时将非实体 span 进行过滤,可以形式化描述为给定候选实体 span 的实例 $s_i=(x_i, x_{i+1}, \cdots, x_{i+k})$和预定义的实体类型集 δ,实体分类的目标是将候选实体 span 的实例 s_i 映射到集合 $\delta \cup \mathrm{NA}$,其中,NA 表示非实体 span,将被模型过滤。

因此,具体的实体分类过程包括以下两个步骤:

(1) 向量拼接。通过上述 span 层的实体对比表示学习,获得候选实体 span 实例 $\boldsymbol{s}_i$ 的最终向量表示为

$$\boldsymbol{s}_i=[\boldsymbol{h}_i; \boldsymbol{s}_{\text{width}}; \boldsymbol{w}_{\text{cls}}] \tag{5.5}$$

式中:$\boldsymbol{h}_i$ 为对应实体 span 实例的向量表示;$\boldsymbol{s}_{\text{width}}$ 为对应实体 span 实例长度的向量表示;$\boldsymbol{w}_{\text{cls}}$ 为特殊的分类信息表示。

(2) 实体类型预测。通过向量拼接操作后,将候选实体 span 实例的向量表示输入 softmax 层进行实体类型预测:

$$\boldsymbol{e}_i=\text{softmax}(\boldsymbol{W}_i \cdot \boldsymbol{s}_i+b_i) \tag{5.6}$$

式中:$\boldsymbol{W}_i$ 为权重矩阵;b_i 为偏置项。

5.3.5　关系分类层

与上述实体分类层的设计思想一致,在获得软件知识实体集后,利用对比表示学习思想,提出一种关系对比表示学习方法,通过正、负样本数据增强和损失函数构建,约束模型学习实体对更适于关系分类的特征表示,进而提高关系分类预测的准确性。因此,在模型的关系分类层包括关系对比表示学习和关系分类两个步骤。

1. 关系对比表示学习

软件知识社区文本的句子序列经过实体抽取后，获得了软件知识实体集 $E=(e_1,e_2,\cdots,e_{|E|})$。为了提升软件知识实体关系分类的性能，我们提出一种有监督的 span 级关系对比表示学习方法，通过特定的数据增强方法和关系对比损失构建，获取候选实体对更适于下游分类任务的特征表示。

与有监督实体对比表示学习方法相比较，有监督关系对比表示学习的编码器(Encoder)和 Projection 网络保持不变，在数据增强和对比损失函数两个组件上有所不同。

在数据增强组件，有监督关系对比表示学习利用软件知识实体的关系标签信息进行正、负样本划分，将具有相同关系类型的实体对视为正样本，并构建正样本集 $P(i)$，将同一 batch 内具有其他关系类型或没有关系的实体对作为负样本，并构建负样本集 $N(i)$。

因此，在对比损失函数组件中，有监督关系对比表示学习的对比损失函数定义如下：

$$L^{rc}=\sum_{i\in B(i)}\frac{-1}{|P(i)|}\sum_{p\in P(i)}\log\frac{\exp(\boldsymbol{z}_i\cdot\boldsymbol{z}_p/\tau)}{\sum_{n\in N(i)}\exp(\boldsymbol{z}_i\cdot\boldsymbol{z}_n/\tau)} \tag{5.7}$$

式中：$\boldsymbol{z}_i$ 为当前实体对实例；$\boldsymbol{z}_p$ 为 $\boldsymbol{z}_i$ 的正样本实例；$\boldsymbol{z}_n$ 为 $\boldsymbol{z}_i$ 的负样本实例；$B(i)$为 batch 内的候选实体对集；$|P(i)|$为正样本集合的基数；$N(i)$为负样本集合；τ 为温度参数；符号“$\cdot$”为向量间相似度计算的内积操作。

通过关系对比损失函数的构建，使得当前实体对实例 $\boldsymbol{z}_i$ 与其正样本集的特征一致性最大化；同时，使得当前实体对实例 $\boldsymbol{z}_i$ 与其负样本集的特征区别性最大化，进而学习到实体对更好的特征表示。

同理，基于软件知识社区文本的 span 级关系对比表示学习可以描述为算法 5-2。

算法 5-2　关系对比表示学习算法

Input: 候选实体对实例$(s_i,\ s_j)$，预训练语言模型 SWBERT，Projection 网络 g
Output: 候选实体对实例$(s_i,\ s_j)$的特征向量 $\boldsymbol{h}_i$

```
Begin
    获取实体对实例(s_i, s_j)的向量表示 h_i
    将向量 h_i 投射到另一个向量表示空间，获得 z_i
    按实体关系的数据增强方法，生成 z_i 的正、负样本表示 z_p、z_n
    计算关系对比损失 L^rc                    //式(5.7)
    更新 SWBERT 和 Projection 网络 g 的参数使得 L^rc 最小化
    返回实体对的特征向量表示 h_i
End
```

由算法 5-2 可知，基于软件知识社区文本的 span 级关系对比表示学习的目标是学习候选实体对更好的特征表示，并作为下游关系分类任务的输入，提升关系类型预测的性能。

2. 关系分类

关系分类层的目标是对候选实体对的关系类型进行预测，可以形式化描述为给定候选实体 span 集 $S=(s_1,s_2,\cdots,s_n)$、实体对实例 $s_i=(x_i,x_{i+1},\cdots,x_{i+k})$ 和 $s_j=(x_j,x_{j+1},\cdots,x_{j+k})$、预定义的关系类型集 ε，关系分类的目标是将候选实体对 (s_i,s_j) 的关系类型映射到集合 $\varepsilon \cup \mathrm{NA}$，其中，NA 表示没有语义关系。

因此，具体的关系分类过程包括以下两个步骤：

(1) 向量拼接。通过上述关系对比表示学习，可获得候选实体对的向量表示 $\boldsymbol{h}_i$ 和实体对之间的上下文向量表示 $\boldsymbol{c}_{ij}$。通过向量拼接，可获得用于关系分类的候选实体对的增强向量表示：

$$\boldsymbol{s}_{ij}=[\boldsymbol{h}_i;\boldsymbol{c}_{i,j}] \tag{5.8}$$

(2) 关系类型预测。向量拼接操作后，将候选实体对的关系向量表示 s_{ij} 输入一个全连接层进行关系分类：

$$\boldsymbol{r}_{ij}=\sigma(\boldsymbol{W}_{ij}\cdot\boldsymbol{s}_{ij}+b_{ij}) \tag{5.9}$$

式中：σ 为 sigmoid 激活函数；$\boldsymbol{W}_{ij}$ 为权重矩阵；b_{ij} 为偏置项。

综上所述，基于 span 级软件知识实体和关系联合抽取方法，首先以领域预训练语言模型 SWBERT 为输入特征编码器，获取句子序列的动态词向量表示；然后以 span 为单元建模句子序列，生成丰富的实体 span 表示，以避免无法选择重叠实体问题；最后在实体分类和关系分类任务中引入有监督对比表示学习方法，通过数据增强和对比损失构建，获取更适于分类任务的实体 span 和实体对特征表示，提升实体分类和关系分类的预测性能。因此，基于 span 级软件知识实体和关系联合抽取的过程可以描述为算法 5-3。

算法 5-3　基于 span 级软件知识实体和关系联合抽取算法

Input: 软件知识社区文本的句子序列 $X=(x_1,x_2,\cdots,x_n)$
Output: 软件知识实体关系三元组

```
Begin
  for each epoch do
    for each batch do
      通过 SWBERT 生成句子序列 X 的词向量表示
      生成句子序列的 span 表示 S=(s1,s2,…,sn)
        for si from S do
          调用算法 5-1 获取实体 span 实例 si 的向量表示 hi
          通过向量拼接获取实体 span 实例 si 的向量表示      //式(5.5)
```

```
        对 span 实例 s_i 的类型进行预测                    //式(5.6)
      endfor
      获得软件知识实体集 E,并选取实体对实例(s_i, s_j)
      for s_i, s_j from S do
        调用算法 5-2 获取实体对实例(s_i, s_j)的向量表示 h_i
        通过向量拼接获取实体对实例(s_i, s_j)的增强向量表示   //式(5.8)
        对实体对实例(s_i, s_j)的关系类型进行预测           //式(5.9)
      endfor
    返回软件知识实体关系三元组
  endfor
endfor
End
```

5.4　实验与分析

为验证我们所提出的软件知识实体和关系联合抽取方法的实际效果，以软件知识社区 StackOverflow 为例开展实验与分析，并通过模型对比实验和模型消融实验对模型性能进行分析和评价。实验的软/硬件环境与前面章节保持一致。

5.4.1　数据集构建

由于软件知识实体和关系联合抽取任务缺乏公开、适用的标注数据集，我们利用前面章节的软件知识实体抽取、软件知识实体关系抽取的数据集来构建软件知识实体和关系联合抽取数据集。在软件知识实体和关系的类型方面，构建了涵盖编程语言、系统平台、软件 API、软件工具、软件开发库、软件框架、软件标准、软件开发过程 8 个预定义实体类型，以及使用关系、包含关系、兄弟关系、同义关系和其他语义关系 5 个预定义关系类型，详细信息如表 5-1 所示。

表 5-1　软件知识实体和关系类型

类型	名　称	含　义	实　例	简写
实体类型	Programming Language	编程语言	Java,Python,C	prla
	Software Platform	软件平台	Weblogic,Dolphin	plat
	Software API	软件 API	QueryManager,isNumeric	api
	Software Tool	软件工具	Pycharm,Firebug	tool
	Software Library	软件开发库	jQuery,NumPy	lib
	Software Framework	软件框架	Hibernate,Django	fram
	SoftwareStandard	软件标准	HTTP,utf-8	stan
	Development Process	软件开发过程	umlet,LoadRunner	depr

续表

类型	名　称	含　义	实　例	简写
关系类型	Use	使用关系	Windows 10,Cortana	use
	Inclusion	包含关系	Python,PyBrain	inc
	Brother	兄弟关系	C,Java	bro
	Consensus	同义关系	JavaScript,JS	con
	Semantic	其他语义关系	Virtual Network,VPN	sem

在数据集标注方面，我们在软件知识实体关系抽取数据集的基础上对实体和关系进行联合标注，利用 JSON(JavaScript Object Notation)文件格式对句子序列、实体类型、实体 span 的起始位置和结束位置、关系类型和头尾实体等信息进行标注，形成软件知识实体和关系联合抽取标注数据集。同时，将该数据集按 7∶1∶2 的比例划分为训练集、验证集和测试集，用于 CS-SKG 模型的训练和测试。数据集由 19013 个句子、43769 个实体和 25183 关系组成，其中包含 452 个与重叠实体的关系实例。数据集的详细信息见表 5-2。

表 5-2　数据集的详细信息

组　成	训　练　集	验　证　集	测　试　集	合　计
句子	15016	196	3801	19013
实体	34592	429	8748	43769
关系	19875	234	5074	25183

5.4.2　超参数设置

在 CS-SKG 模型的训练过程中，对于基于 span 的模块部分，输入嵌入层的 SWBERT 预训练语言模型的词向量维度设置为 768 维，Batch size 设置为 3，span 的最大值均设置为 10，实体 span 负样本和关系负样本最大值均设置为 100，Adam 作为优化器，初始学习率设为 5e－5；对于对比表示学习的模块部分，采用对比损失作为模型的损失函数，温度参数较小更有利于训练，但由于数值不稳定性，极低的温度参数较难以训练，因此，温度参数设置为 0.1。模型的超参数设置如表 5-3 所示。

表 5-3　模型的超参数设置

模型名称	参数名称	参　数　值
span-based model	Word embedding dimension	768
	Batch size	3

续表

模型名称	参数名称	参数值
span-based model	Learningrate	5e−5
	Max_span_size	10
	Num_epoch	100
	Num_negative_entity	100
	Num_negative_relation	100
	Optimizer	Adam
	Dropout	0.5
Contrastive Learning model	τ	0.1

5.4.3　对比实验结果与分析

为了验证我们提出的软件知识实体和关系联合抽取模型 CS-SKG 的性能，选取了三个基于共享参数方法、基于联合解码方法的实体关系联合抽取模型作为基线方法进行对比实验，实验数据基于我们构建的软件知识实体和关系联合抽取标注数据集。

Muti-head 模型[56]是一种基于共享参数方法的实体和关系联合抽取模型。该模型利用 BILOU 标注方法和 CRF 解码实现实体抽取，通过多头选择算法实现关系抽取，利用 sigmoid 层实现关系抽取。

SPERT 模型[152]是一种基于共享参数方法的实体和关系联合抽取模型。该模型抛弃了传统基于 BIO/BILOU 标注的方法，利用预训练语言模型 BERT 获取句子序列的词向量表示，并通过枚举句子序列中所有可能的实体 span，实现实体和关系联合抽取。

NovelTagging 模型[55]是一种基于联合解码方法的实体和关系联合抽取模型。该模型基于一种新序列标注框架和 LSTM 网络实现了一个端到端的实体和关系联合抽取。

我们所提出的软件知识实体和关系联合抽取模型 CS-SKG 是从 span 级对句子序列进行实体和关系抽取，其抽取结果的评价准则：若软件知识实体 span 的边界及其类型均预测正确，则表示实体识别结果正确；若软件知识实体 span 的边界、实体类型及实体间关系均预测正确，则表示关系抽取结果正确。模型对比实验结果如表 5-4、图 5-2 和图 5-3 所示。

表 5-4 对比实验结果

模型	实体			关系		
	P/%	*R*/%	*F*1/%	*P*/%	*R*/%	*F*1/%
Multi-head	68.20	65.23	66.68	61.03	56.82	58.85
NovelTagging	75.67	65.39	70.16	67.15	63.31	65.17
SPERT	83.71	67.17	74.53	72.27	69.64	70.93
SCL-SKG *	82.93	78.69	**80.75**	80.56	76.39	**78.42**
SCL-SKG	82.19	78.91	**80.52**	80.37	76.03	**78.14**

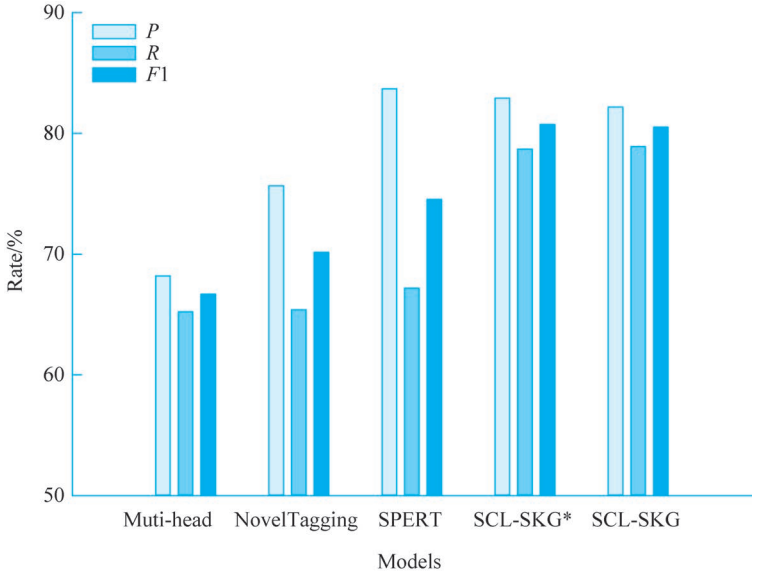

图 5-2 实体抽取结果对比

由模型对比实验结果可见，软件知识实体和关系联合抽取模型 CS-SKG 的 $F1$ 值均高于其他三个基线模型，取得了较好的性能。

从软件知识实体抽取任务来看，CS-SKG 模型与 Muti-head 模型相比，精确率 P 值和 $F1$ 值分别提升了 13.99%和 13.84%；CS-SKG 模型与 NovelTagging 模型相比，精确率 P 值和 $F1$ 值分别提升了 6.52%和 10.36%；CS-SKG 模型与 SPERT 模型相比，精确率 P 值下降 1.52%，$F1$ 值提升了 5.99%。

从软件知识实体关系抽取任务来看，CS-SKG 模型与 Muti-head 模型相比，精确率 P 值和 $F1$ 值分别提升了 19.34%和 19.29%；CS-SKG 模型与 NovelTagging

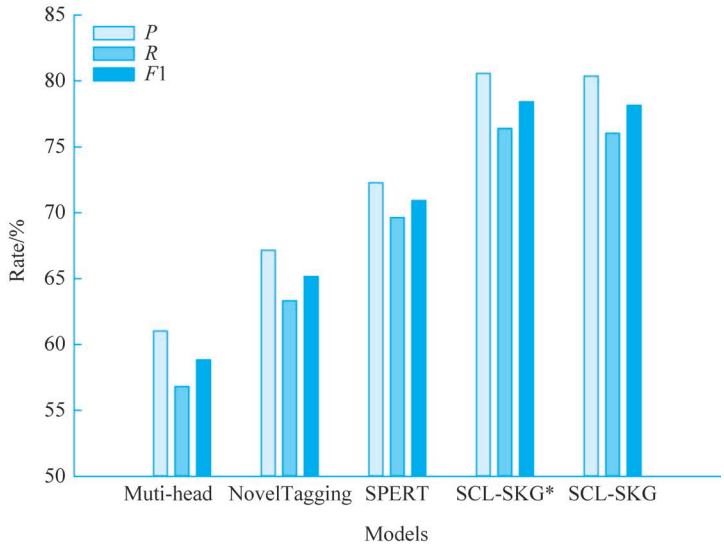

图 5-3　关系抽取结果对比

模型相比，精确率 P 值和 $F1$ 值分别提升了 13.22%和 12.97%；CS-SKG 模型与 SPERT 模型相比，精确率 P 值和 $F1$ 值分别提升了 8.1%和 7.21%。

从句子序列建模对象的层面上看，由于基于 span 的方法不受序列建模的顺序限制，能对重叠的实体进行选择，从而提高了模型的预测精度，相较于 Muti-head 模型和 NovelTagging 模型，SPERT 模型和 CS-SKG 模型分别获得最高的实体抽取精确率和关系抽取精确率。同时，相较于 SPERT 模型，CS-SKG 模型在基于 span 方法的基础上，在实体抽取和关系抽取阶段引入了对比表示学习方法，获得了更适应于分类的实体 span 和实体对的增强特征表示，实体抽取和关系抽取的 $F1$ 值分别提升了 5.99%和 7.21%。

同时，我们探索了 SCL-SKG 模型在剔除重叠实体的数据集上的性能表现。如表 5-4、图 5-2 和图 5-3 所示，CS-SKG * 和 CS-SKG 分别表示剔除重叠实体和保留重叠实体的模型性能。在剔除重叠实体后，SCL-SKG 模型的实体抽取和关系抽取的 $F1$ 值分别仅提高 0.23%和 0.28%。这表明 SCL-SKG 模型在这两种不同的数据集上的性能保持相对稳定。

另外，针对软件知识实体和关系联合抽取模型 CS-SKG 对每个预定义软件知识实体类型和关系类型的预测结果进行了分析，实验结果如表 5-5、表 5-6、图 5-4 和图 5-5 所示。

表 5-5 各实体类型的抽取结果

实 体 类 型	*P*/%	*R*/%	*F*1/%
tool	85.41	81.13	83.21
plat	91.53	88.90	**90.19**
stan	78.59	73.41	75.91
lib	90.16	87.39	88.75
api	76.35	74.46	75.39
prla	89.19	85.51	87.31
fram	76.86	74.31	75.56
depr	69.42	66.17	67.75

表 5-6 各关系类型的抽取结果

关 系 类 型	*P*/%	*R*/%	*F*1/%
inclusion	87.32	85.08	**86.18**
brother	79.83	76.69	78.23
semantic	66.18	59.21	62.50
use	85.25	80.03	82.56
consensus	83.29	79.15	81.17

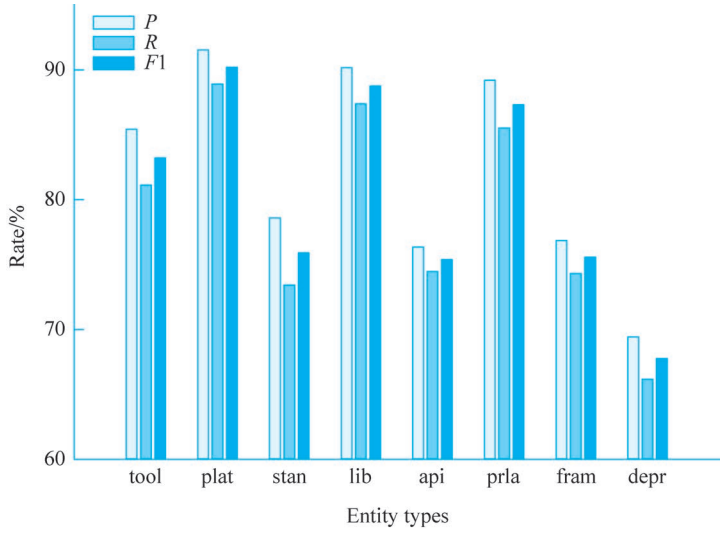

图 5-4 各实体类型的抽取结果

从实验结果来看，软件知识实体和关系联合抽取模型 CS-SKG 的实体抽取任

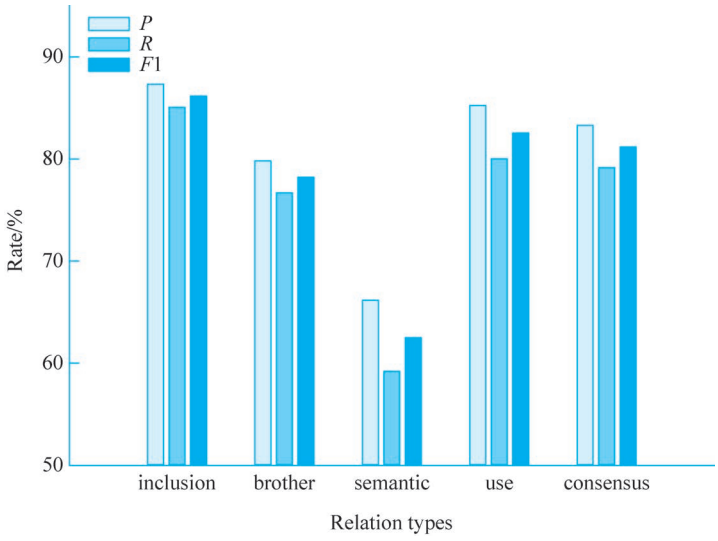

图 5-5　各关系类型的抽取结果

务要比关系抽取任务质量更高。在实体识别任务中，模型在软件工具(Software Tool)、软件开发库(Software Library)和编程语言(Programming Language)等实体类型上都有较好的性能，并在软件平台(Software Platform)实体类型上取得了最高的 $F1$ 值，但在软件开发过程(Development Process)实体类型上出现了最低的 $F1$ 值。在关系分类任务中，包含关系(Inclusion)关系类型取得最高 $F1$ 值，在其他语义关系(Semantic)关系类型上出现了最低的 $F1$ 值。

5.4.4　消融实验结果与分析

我们对软件知识实体和关系联合抽取模型 CS-SKG 进行模型消融实验，其主要目标是验证所提出的实体对比表示学习、关系对比表示学习以及领域预训练语言模型 SWBERT 对软件知识实体和关系联合抽取的性能影响。为保证实验结果的合理性，在模型消融实验的过程中各模型参数保持相同的设置。

1. 对比表示学习对模型性能的贡献评价

软件知识实体和关系联合抽取模型 CS-SKG 以 span 方法为基础，通过引入有监督的对比表示学习框架，实现实体 span 和实体对的特征增强表示，进而提升软件知识实体和关系抽取的性能。为此，我们选取 CS-SKG 模型作为基准模型，验证实体对比表示学习和关系对比表示学习对软件知识实体和关系联合抽取任务的影响，实验结果如表 5-7、图 5-6 和图 5-7 所示。

表 5-7 对比表示学习对模型性能的影响

模 型	实体对比表示学习	关系对比表示学习	实 体			关 系		
			P/%	*R*/%	*F*1/%	*P*/%	*R*/%	*F*1/%
CS-SKG-NN	×	×	68.43	59.61	63.72	65.32	58.34	61.63
CS-SKG-EC	√	×	81.97	76.69	79.24	69.74	62.57	65.96
CS-SKG-RC	×	√	69.11	60.93	64.76	71.51	65.72	68.49
CS-SKG-ALL	√	√	**82.19**	**78.91**	**80.52**	**80.37**	**76.03**	**78.14**

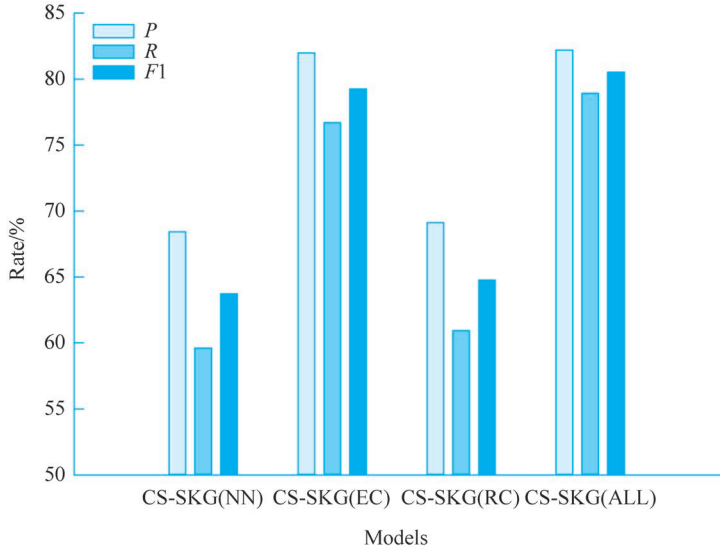

图 5-6 对比表示学习对实体抽取性能的影响

在表 5-7 中，模型 CS-SKG-NN 表示未引入实体对比表示学习和关系对比表示学习，模型 CS-SKG-EC 表示只引入实体对比表示学习，模型 CS-SKG-RC 表示只使用关系对比表示学习，模型 CS-SKG-ALL 表示同时引入了实体对比表示学习和关系对比表示学习。

从对比实验结果来看，单独引入实体对比表示学习后，实体抽取和关系抽取的 $F1$ 值分别提高 15.52%和 4.33%；单独引入关系对比表示学习后，实体抽取和关系抽取的 $F1$ 值分别提高 0.04%和 6.86%；同时引入实体对比表示学习和关系对比表示学习后，实体抽取和关系抽取的 $F1$ 值达到最高。

实验结果表明，基于聚合正样本、分离负样本思想的实体对比表示学习和关系

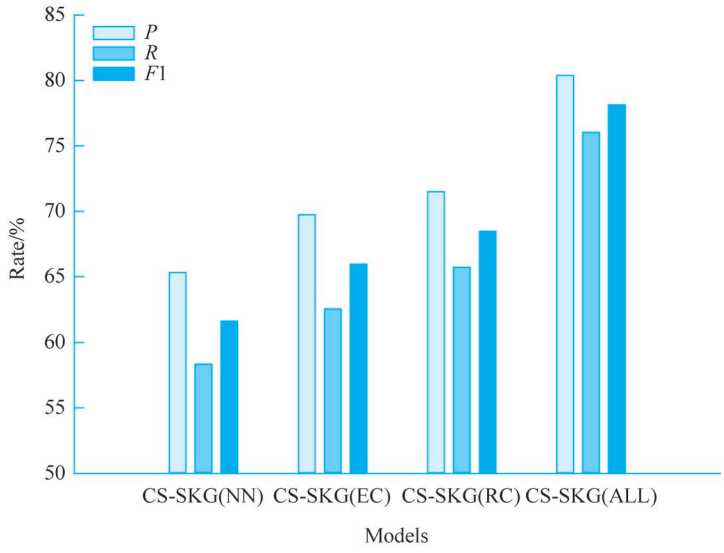

图 5-7　对比表示学习对关系抽取性能的影响

对比表示学习能获取实体 span 和实体对更适于下游分类任务的特征表示，有助于提升软件知识实体和关系联合抽取任务的性能。

2. SWBERT 模型对模型性能的贡献评价

在软件知识实体和关系联合抽取模型 CS-SKG 的输入嵌入层，利用已精调的领域预训练语言模型 SWBERT 对输入句子序列进行编码，生成相应的词向量表示。为了评价领域预训练语言模型 SWBERT 对软件知识实体和关系联合抽取任务的性能贡献，我们选取 CS-SKG 模型作为基准模型，从预训练语言模型的角度对模型进行消融实验。实验结果如表 5-8、图 5-8 和图 5-9 所示。

在表 5-8 中，模型 CS-SKG-NN 表示未引入预训练语言模型，模型 CS-SKG-BERT 表示引入通用领域 BERT 模型，模型 CS-SKG-SWBERT 表示引入已精调的软件领域预训练语言模型 SWBERT。

表 5-8　预训练语言模型对模型性能的影响

模　型	预训练模型	实　体			关　系		
		P/%	*R*/%	*F*1/%	*P*/%	*R*/%	*F*1/%
CS-SKG-NN	×	77.26	69.71	73.29	71.84	68.27	70.01
CS-SKG-BERT	√	80.95	73.19	76.87	76.36	70.17	73.13
CS-SKG-SWBERT	√	**82.19**	**78.91**	**80.52**	**80.37**	**76.03**	**78.14**

由实验结果可知，模型引入通用领域预训练语言模型 BERT 后，实体抽取和关系抽取的 $F1$ 值分别提高 3.58%和 3.12%；模型引入软件领域预训练语言模型 SWBERT 后，实体抽取和关系抽取的 $F1$ 值分别提高 7.23%和 8.13%；并且，模型引入软件领域预训练语言模型 SWBERT 与引入通用领域预训练语言模型 BERT 相比，实体抽取和关系抽取的 $F1$ 值分别提高 3.65%和 5.01%。

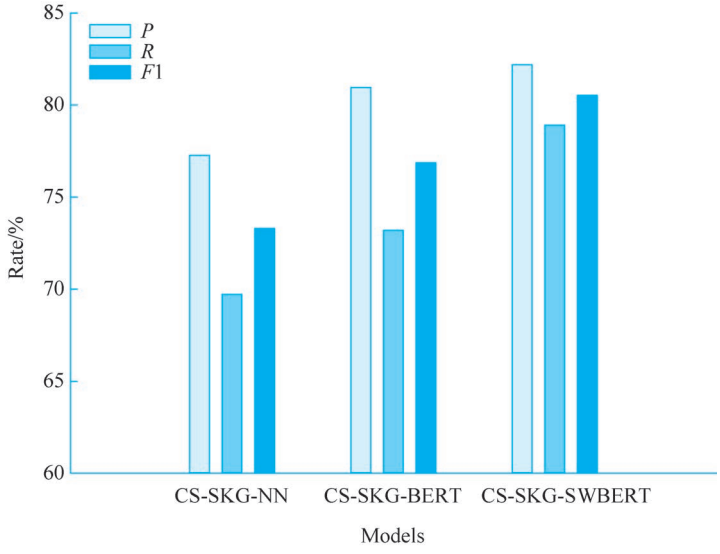

图 5-8 预训练语言模型对实体抽取性能的影响

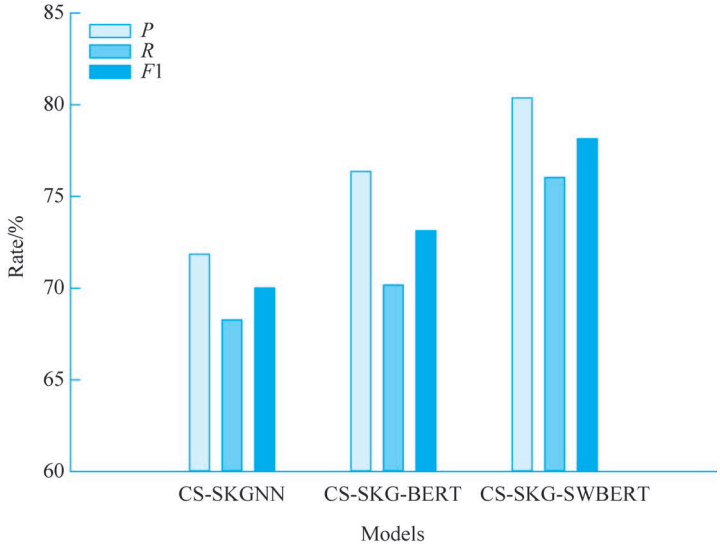

图 5-9 预训练语言模型对关系抽取性能的影响

实验结果表明，特定软件领域的预训练语言模型能更好地捕获软件知识社区文本中单词的领域特征，生成高质量的词向量表示，有助于提升下游的实体分类和关系分类任务性能。

5.4.5　公共数据集的实验结果与分析

为了进一步探索我们提出的 SCL-SKG 模型性能，在三个公共数据集上进行了模型实验与结果分析。CoNLL04 数据集[161]是一个带有标注信息的新闻文章数据集，包括 4 种实体类型和 5 种关系类型。SciERC 数据集[149]来自 4 个人工智能社区的 500 份人工智能会议论文集的摘要。ADE 数据集[162]源于药物使用的医疗报告，包括 2 种实体类型和 1 种关系类型。

根据前面工作的评价方法，我们对 CoNLL04 数据集和 ADE 数据集采用宏观平均值的方法进行评价，对 SciERC 数据集采用微平均值的方法进行评价，实验结果如表 5-9 所示。

表 5-9　公共数据集的实验结果

数据集	模　型	实　体			关　系		
		P/%	*R*/%	*F*1/%	*P*/%	*R*/%	*F*1/%
CoNLL04	Multi-head	83.75	84.06	83.90	63.75	60.43	62.04
	Multi-head+AT[163]	—	—	83.61	—	—	61.95
	Table-filling[51]	81.20	80.20	80.70	76.00	50.90	61.00
	SPERT	85.78	86.84	86.25	74.75	71.52	72.87
	SCL-SKG	86.45	87.35	**86.93**	76.13	71.31	**73.12**
SciERC	SciIE[149]	67.20	61.50	64.20	47.60	33.50	39.30
	DyGIE[150]	—	—	65.20	—	—	41.60
	DyGIE++[151]	—	—	67.50	—	—	48.40
	SPERT	68.53	66.73	67.62	49.79	43.53	46.44
	SCL-SKG	68.24	66.28	67.21	50.37	44.69	**47.56**
ADE	BiLSTM+SDP[164]	82.70	86.70	84.60	67.50	75.80	71.40
	Multi-head	84.72	88.16	86.40	72.10	77.24	74.58
	Multi-head+AT[163]	—	—	86.73	—	—	75.52
	SPERT	88.99	89.59	89.28	77.77	79.96	78.84
	SCL-SKG(without overlap)	88.91	89.16	89.03	75.67	80.81	78.16
	SCL-SKG(with overlap)	87.53	90.25	88.89	75.35	80.33	77.91

实验结果表明，SCL-SKG 模型在 CoNLL04 数据集上获得了性能提升，实体抽取和关系抽取的 *F*1 值分别提高了 0.7%和 0.3%。对于 SciERC 数据集，关系抽

取的性能也有所提高，关系抽取的 $F1$ 值提高了 1.1%。对于 ADE 数据集，包含 120 个含有实体重叠的关系实例，与 SPERT 模型相比，SCL-SKG 模型在实体抽取和关系抽取方面没有实现性能提升。当包含重叠实体时，SCL-SKG 模型的实体抽取和关系抽取的 $F1$ 值分别仅下降 0.14%和 0.25%。

5.4.6 联合训练方法分析

我们基于对比表示学习和 span 方法提出软件知识实体和关系联合抽取模型 CS-SKG，对软件知识实体抽取和关系抽取进行联合建模，包括实体分类和关系分类两个子任务，需要根据联合损失函数对实体分类和关系分类两个任务进行联合训练。

因此，软件知识实体和关系联合抽取模型 CS-SKG 的损失函数由实体对比表示学习损失 $\boldsymbol{L}^{ec}$、实体分类损失 $\boldsymbol{L}^{e}$、关系对比损失 $\boldsymbol{L}^{rc}$ 和关系分类损失 $\boldsymbol{L}^{r}$ 四部分组成，其中，实体分类损失 $\boldsymbol{L}^{e}$ 采用 Categorical Cross Entropy 作为损失函数，关系分类损失 $\boldsymbol{L}^{r}$ 采用 BinaryCross Entropy 作为损失函数。

为了获得 CS-SKG 模型最佳的联合训练结果，我们尝试了损失函数相加、损失函数相乘和损失函数线性结合三种不同的联合训练方法，具体公式如下：

$$\boldsymbol{L}=\boldsymbol{L}^{e}+\boldsymbol{L}^{ec}+\boldsymbol{L}^{r}+\boldsymbol{L}^{rc} \tag{5.10}$$

$$\boldsymbol{L}=\boldsymbol{L}^{e}*\boldsymbol{L}^{ec}+\boldsymbol{L}^{r}\times\boldsymbol{L}^{rc} \tag{5.11}$$

$$\boldsymbol{L}=\boldsymbol{L}^{ec}+\lambda(\cdot)\boldsymbol{L}^{e}+\boldsymbol{L}^{rc}+\lambda(\cdot)\boldsymbol{L}^{r} \tag{5.12}$$

损失函数相加表示实体对比损失 $\boldsymbol{L}^{ec}$、实体分类损失 $\boldsymbol{L}^{e}$、关系对比损失 $\boldsymbol{L}^{rc}$ 和关系分类损失 $\boldsymbol{L}^{r}$ 四个损失相加，见式(5.10)；损失函数相乘表示实体对比损失和实体分类损失相乘，关系对比损失和关系分类损失相乘，见式(5.11)；损失函数线性结合表示在实体分类损失和关系分类损失上加线性函数，并结合实体分类和关系分类两部分损失，见式(5.12)。

由模型联合训练方法的实验结果图 5-10 和图 5-11 可知，损失函数相乘的方法会导致模型无法收敛，取得较差的结果；损失函数相加和损失函数线性结合的方法能完成模型训练与测试，线性结合方法取得了更好的效果。因此，我们采用损失函数线性结合的方法对软件知识实体和关系联合抽取模型进行训练。

5.4.7 实例研究与分析

通过以上实验结果分析，我们所提出的基于 span 级对比表示学习的软件知识实体和关系联合抽取模型 CS-SKG，能较好地解决流水线式方法的任务依赖问题，缓解了实体重叠问题，在基于社区文本的软件知识领域取得了较好的性能。我们

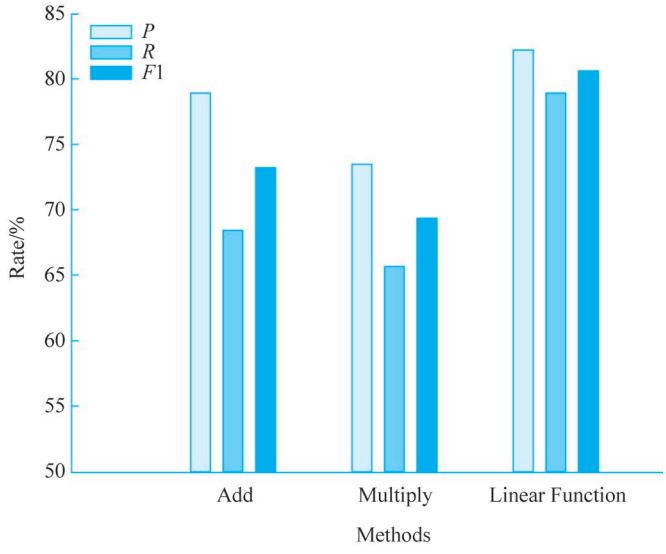

图 5-10　实体抽取的联合训练方法

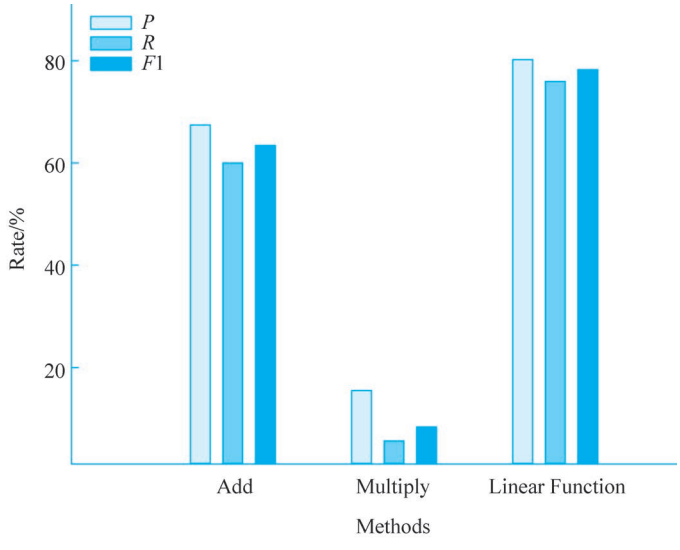

图 5-11　关系抽取的联合训练方法

构建了包含 43769 个实体实例和 25183 个关系实例的软件知识图谱，其中，50 个节点和 1000 个节点的概览图分别如图 5-12 和图 5-13 所示。

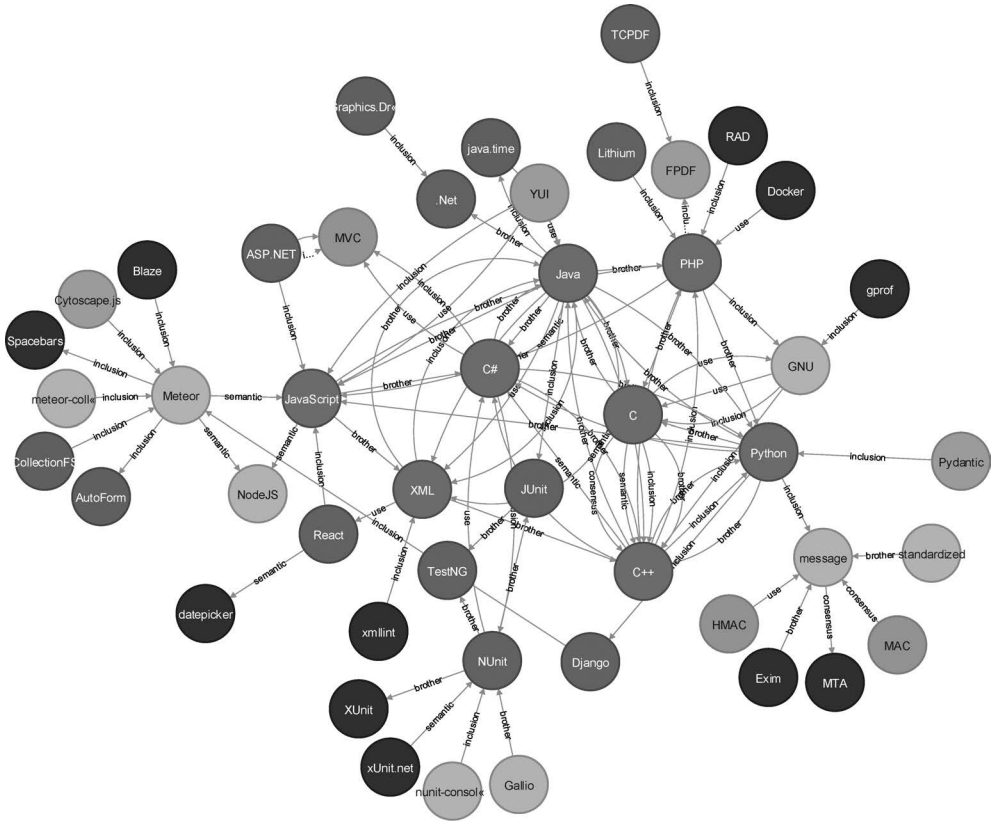

图 5-12　50 个节点的概览图

从图 5-12 和图 5-13 可以看出，软件知识图的节点代表软件知识实体的实例，不同类型的软件知识实体采用不同的颜色来区分。例如，节点“Java”“PHP”“c＃”的实体类型为“Programming Language”，节点“JUnit”“TestNG”“React”的实体类型为“Software Framework”。软件知识实例〈Pydantic，Inclusion，Python〉描述了一个软件知识事实，即头实体“Pydantic”通过“inclusion”边链接到尾实体“Python”。

软件知识实体和关系联合抽取模型 CS-SKG 尽管取得了较好的性能，但也存在一些具体的问题。下面就 CS-SKG 模型的软件知识实体和关系联合抽取实际案例进行分析，分析结果如表 5-10 所示，其中，符号“[]”表示抽取的软件知识实体。

图 5-13　1000 个节点的概览图

表 5-10　软件知识实体和关系联合抽取实例分析

实　例	抽 取 结 果
实例 1	SpriteKit is [[**Apples**] **framework**] for creating 2D games for [[**IOS**] **7**], [[**macOS**] **10.9**], [[**tvOS**] **9**] and [[**watchOS**] **3**]
实例 2	StarlingFramework is an [**ActionScript 3**] library for hardware accelerated 2D graphics
实例 3	[**BlackBerry**] offers a variety of development tools, including the [**BlackBerry Dynamics SDK**], [**Cylance REST APIs**], [**BlackBerry Workspaces APIs**] and SDKs, BlackBerry QNX development and BlackBerry UEM REST APIs
实例 4	[**CUDA**] ([**Compute Unified Device Architecture**]) is a parallel computing platform and programming model for NVIDIA GPUs (Graphics Processing Units)

在实例 1 中，模型 CS-SKG 不仅能抽取软件知识实体“Apples”和“IOS”，还能准确抽取软件知识实体“Apples framework”和“IOS 7”。经统计，在 453 个实体重叠的关系实例中，有 267 个被准确识别出来，这说明模型 CS-SKG 能有效解决实体重叠问题。

在实例 2 中，由于实体 span 的边界错误，模型 CS-SKG 没有准确识别出软件知识实体“ActionScript 3 library”，造成关系抽取出错。

在实例 3 中，模型 CS-SKG 抽取出软件知识实体“BlackBerry”“BlackBerry Dynamics SDK”“Cylance REST APIs”，并准确识别软件知识实体“BlackBerry”和“BlackBerry Dynamics SDK”之间的关系为“Inclusion”。同时，在训练数据集中虽然没有标注软件知识实体“BlackBerry Dynamics SDK”和“Cylance REST APIs”之间的关系，但模型却预测出两个实体间的关系为“Brother”。

在实例 4 中，模型 CS-SKG 将软件知识实体“CUDA”和“Compute Unified Device Architecture”分别抽取为“Software Platform”类型和“Software Tool”类型，造成实体抽取错误，导致没有正确识别两者之间的“Consensus”关系。

5.5 本章小结

针对传统 Pipeline 方法存在的任务依赖问题和软件知识社区文本存在的实体重叠问题，我们提出了一种基于 span 级对比表示学习的软件知识实体和关系联合抽取方法，并以软件知识社区 StackOverflow 为例进行了实验与分析。实验结果表明，span 级对比表示学习方法以 span 为单位建模句子序列，能缓解软件知识实体重叠问题；同时，有监督的实体对比表示学习和关系对比表示学习能获取实体 span 和实体对的增强特征表示，有助于提升软件知识实体和关系分类的性能。