基于短语向量和主题加权的关键词抽取方法

孙 新[1,2],盖 晨[1],申长虹[1],张颖捷[1]
(1. 北京理工大学计算机学院北京市海量语言信息处理与云计算应用工程技术研究中心,北京 100081;
2. 北京理工大学东南信息技术研究院,福建莆田 351100)

摘 要: 现有关键词抽取算法缺乏对短语的有效表示,为抽取出更能反映文本主题的关键短语,本文提出一种基于短语向量的关键词抽取方法PhraseVecRank.首先设计基于LSTM(Long Short-Term Memory)和CNN(Convolutional Neural Network)自编码器的短语向量构建模型,解决复杂短语的语义表示问题.然后,利用短语向量对每个候选短语计算主题权重,通过主题加权排序提高关键词抽取的效果.在公共数据集和学术论文数据上的实验表明,本文提出的方法能够有效提取与文本主题信息相关的关键短语,同时利用自编码器构造的短语向量可以更好地表示短语的语义信息.

关键词: 短语向量;自编码器;主题加权;关键词抽取
中图分类号: TP181 **文献标识码:** A **文章编号:** 0372-2112(2021)09-1682-09
电子学报URL: http://www.ejournal.org.cn **DOI:** 10.12263/DZXB.20200014

The Theme-Weighted Keyphrase Extraction Algorithm Based on Phrase Embedding

SUN Xin[1,2],GE Chen[1],SHEN Chang-hong[1],ZHANG Ying-jie[1]
(1. *Beijing Engineering Research Center of High Volume Language Information Processing and Cloud Computing Applications, School of Computer Science and Technology, Beijing Institute of Technology, Beijing* 100081, *China*;
2. *Beijing Institute of Technology Southeast Academy of Information Technology, Putian, Fujian* 351100, *China*)

Abstract: Keyword extraction is a key basic problem in the field of natural language processing. The keyphrase extraction algorithms(PhraseVecRank) is proposed based on phrase embedding. Firstly, a phrase vector construction model based on LSTM(Long Short-Term Memory) and CNN(Convolutional Neural Network) is designed to solve the semantic representation of complex phrases. Then, PhraseVecRank uses phrase embedding to calculate theme weight for each candidate phrase, and uses semantic similarity between candidate phrase embedding and co-occurrence information to calculate edge weight together, which can improve the extraction effect of keyphrases through topic weighted ranking. The experimental results verify that PhraseVecRank can effectively extract keyphrases covering the topic information of text, and the phrase embedding models we proposed can better represent the semantic information of phrases.

Key words: phrase embedding;auto-encoder;theme-weighted;keyphrases extraction

1 引言

关键词抽取可以从文本中抽取重要的、具有主题性的词或短语,是自然语言处理领域的一个核心基本问题,其研究成果可广泛用于文档自动摘要、文本分类等领域.目前关键词抽取算法主要分为有监督的方法和无监督的方法[1].有监督的关键词抽取算法把关键词抽取任务转化为分类任务,首先确定候选项的特征表示方法,通过构造不同的分类器实现关键词提取[2].有监督的方法虽然可以取得不错的效果,但是存在过拟合及数据标注代价大的问题.

无监督的关键词抽取方法采取各种评分指标对候选关键词进行排序,不需要人工标注的数据,只通过文档自身内容就可以抽取关键词,因此使用范围广泛.无监督的关键词抽取方法主要分为基于统计信息的方法、基于主题的方法和基于图的方法.其中,基于图的关键词提取方法考虑了文档中词间语义关联,可以融

收稿日期:2019-12-23;修回日期:2020-11-24;责任编辑:梅志强
基金项目:国家重点研发计划项目(No.2017YFB0803300)

合更多其他特征信息,是目前最有效、并被广泛研究的一类无监督抽取方法.

TextRank[3]是一种常用的基于图的关键词抽取方法. 该算法将文本中的候选词项作为无向图的节点,以候选词项之间的共现关系构造边,根据排序结果获取关键词. 为了引入除共现关系之外的其他信息,一系列基于TextRank的改进算法陆续被提出. SingleRank[4]通过与当前文本具有相同主题的邻域文本集合的方式引入全局信息. TopicalPageRank[5]利用文档的主题分布和主题的单词分布进行排序打分. PositionRank[6]捕捉文本中频繁出现的词,同时考虑它们在文本中的对应位置. 文献[7]采用了一种基于加权超图随机游走的方法. 现有方法通常是在词图模型基础上引入候选词项的位置、共现信息、文档主题等信息,改进边的权重计算方法.

深度学习技术可以从多个显式特征中学习得到融合统一的词向量(word embeddings),利用词向量特征的关键词提取方法的研究相继展开[8]. 预训练模型也给关键短语抽取带来了新的引入外部知识和信息的方法. SIFRank[9]通过引入预训练模型ELMo对文章和短语得到动态的句子向量和短语向量,实现了高质量的关键短语抽取. 然而,已有工作主要集中在对单个词的研究,而没有考虑在多数情况下,尤其是特定专业领域,关键词往往不是单个词而是短语的问题. 因此,仅用词向量不足以满足关键词抽取任务的需要,现有工作缺乏对短语语义的充分利用.

针对已有算法对候选短语的语义表示能力不足的问题,本文首先提出基于长短时记忆神经网络(Long Short-Term Memory,LSTM)和基于卷积神经网络(Convolutional Neural Network,CNN)的自编码器的短语向量构建模型;然后,用短语向量表示候选关键短语,进一步利用短语向量对每个候选短语计算主题权重,通过主题加权排序提高关键词抽取的效果. 最后,在公共数据集以及中文学术论文数据集上进行实验,验证了算法的优化效果和短语向量模型的有效性.

2　短语向量构建模型

短语向量(phrase pmbedding)的构建方法一般分为两类,一类是把短语看作一个不可分割的单元,但是这种方法不能充分利用短语中单词的信息;另一类是通过基于词向量的组合方法研究推断出短语向量[11,12]. 自编码器可以有效压缩信息,是最常用的用于组合词向量的结构.

自编码器(Auto Encoder,AE)是一种结构非常简单的网络模型,包括编码器和解码器两个部分. 它的特殊之处在于:自编码器是以通过隐藏层将输入内容恢复作为输出目标的. 输入序列经过编码器被编码为一个上下文向量,然后通过解码器重建输入序列. 如果能把输入内容恢复,就说明隐藏层很好地压缩了输入内容,同时能保留输入信息.

本文首先利用自编码器对单词向量进行组合构建短语向量,利用长短时记忆神经网络LSTM和卷积神经网络CNN在编码时的特点,提出两种短语向量构建模型:基于长短时记忆神经网络的自编码器(Long Short-Term Memory Auto Encoder, LSTMAE)和基于卷积神经网络的自编码器(Convolutional Auto Encoder, ConvAE),构建语义更加精确的短语向量.

2.1　自编码器

传统自编码器直接使用基础的全连接网络进行编码和解码,层与层之间是全连接的,每层之间的节点是无连接的,即:

$$\boldsymbol{h}=\sigma_e(\boldsymbol{W}_e\boldsymbol{x}\boldsymbol{b}_e) \tag{1}$$

$$\boldsymbol{y}=\sigma_d(\boldsymbol{W}_d\boldsymbol{h}+\boldsymbol{b}_d) \tag{2}$$

其中,$\boldsymbol{x}$为输入序列,$\boldsymbol{y}$为输出序列,$\boldsymbol{h}$为隐藏层向量,$\boldsymbol{W}_e$、$\boldsymbol{b}_e$和$\boldsymbol{W}_d$、$\boldsymbol{b}_d$分别是编码和解码时全连接网络中的权重矩阵和偏置,σ_e、σ_d为全连接网络中使用的非线性变换.

自编码器的训练目标是使输出序列和输入序列尽可能一致:

$$J(\boldsymbol{W},\boldsymbol{b})=\frac{1}{n}\sum_{i=1}^{n}\left(\boldsymbol{x}_i-\hat{\boldsymbol{x}}_i\right)^2 \tag{3}$$

以自编码器对单词向量进行组合来构建短语向量时,可以在编码器部分输入短语中各单词的表示,然后把它们压缩为一个中间隐藏层向量,在解码器部分通过隐藏层向量重新解析出输入的短语,那么这个中间向量就可以被认为是包含了语义信息的短语向量表示.

2.2　基于长短时记忆神经网络的自编码器LSTMAE

以全连接网络为编解码核心的自编码器,其隐藏层之间的节点是无连接的,无法处理类似短语结构中的序列信息. 在循环神经网络RNN(Recurrent Neural Network, RNN)中,隐藏层之间的节点不再是无连接而是有连接的,并且隐藏层的输入不仅包含输入层的输出还包含上一时刻隐藏层的输出,因此RNN适合用来对序列数据进行编码. 然而在RNN的传播过程中,存在着重大的历史信息的遗忘和误差累积的问题,现在通常使用长短时记忆神经网络LSTM来改进. 因此,为更好地编码短语向量,本文使用长短时记忆神经网络来对序列数据进行编码.

LSTMAE由两部分组成:由双向LSTM和全连接网络组成的编码器,以及由单向LSTM和Softmax层组成的解码器,结构如图1所示.

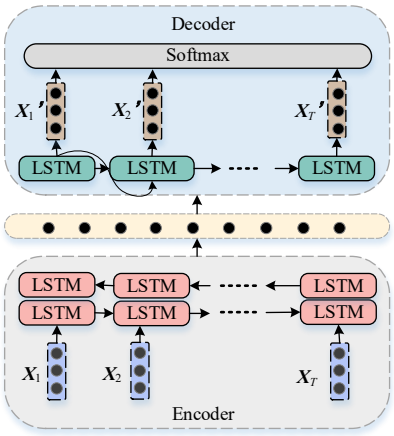

图1 LSTMAE结构图

为了更好地理解短语的语义，在编码器部分使用了双向LSTM，从两个方向获取序列信息．即对于一个输入序列$(\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_T)$，这里$T$表示词项中的单词数量，$\boldsymbol{x}_t$是短语中第$t$个单词的向量表示($t$=1，…，$T$)，在编码器部分，从前后两个方向分别进行计算：

$$(\vec{\boldsymbol{h}}_t,\vec{\boldsymbol{C}}_t)=\mathrm{LSTM}(\vec{\boldsymbol{h}}_{t-1},\vec{\boldsymbol{C}}_{t-1},\boldsymbol{x}_t) \tag{4}$$

$$(\overleftarrow{\boldsymbol{h}}_t,\overleftarrow{\boldsymbol{C}}_t)=\mathrm{LSTM}(\overleftarrow{\boldsymbol{h}}_{t-1},\overleftarrow{\boldsymbol{C}}_{t-1},\boldsymbol{x}_t) \tag{5}$$

其中，$\boldsymbol{x}_t$为t时刻输入的$\boldsymbol{c}_j$中的单词，$\vec{\boldsymbol{h}}_t$、$\vec{\boldsymbol{C}}_t$和$\overleftarrow{\boldsymbol{h}}_t$、$\overleftarrow{\boldsymbol{C}}_t$分别为$t$时刻从左向右和从右向左两个方向上的隐藏层状态和细胞状态，t=1，2，…，T.

在每一个时刻，当前隐藏层状态和细胞状态的计算都要依赖上一个时刻的隐藏层状态、细胞状态和当前输入．最后，取T时刻隐藏层状态$\boldsymbol{h}_T$和细胞状态$\boldsymbol{C}_T$作为最终状态，直接将两个方向上的状态进行连接．

另外，为了给解码部分提供一个固定大小的输入$\boldsymbol{E}_T$，还需要通过一个全连接层对连接后的状态进行处理，计算如下：

$$\boldsymbol{h}_T=\vec{\boldsymbol{h}}_T\oplus\overleftarrow{\boldsymbol{h}}_T \tag{6}$$

$$\boldsymbol{C}_T=\vec{\boldsymbol{C}}_T\oplus\overleftarrow{\boldsymbol{C}}_T \tag{7}$$

$$\boldsymbol{h}'_T=f\left(\boldsymbol{W}_h\boldsymbol{h}_T+\boldsymbol{b}_h\right) \tag{8}$$

$$\boldsymbol{C}'_T=f\left(\boldsymbol{W}_c\boldsymbol{C}_T+\boldsymbol{b}_c\right) \tag{9}$$

其中，$\oplus$为连接符，$\boldsymbol{W}_h$、$\boldsymbol{b}_h$、$\boldsymbol{W}_c$、$\boldsymbol{b}_c$代表全连接网络中的参数矩阵和偏置，f表示全连接网络中的激活函数ReLU，$\boldsymbol{E}_T$是$\boldsymbol{h}'_T$和$\boldsymbol{C}'_T$组成最终提供给解码器的一个元组．

在解码部分，以$\boldsymbol{E}_T$为初始状态使用单向LSTM进行解码，计算如下：

$$\boldsymbol{z}_t=\mathrm{LSTM}\left(\boldsymbol{z}_{t-1},\boldsymbol{E}_T,\hat{\boldsymbol{x}}_{t-1}\right) \tag{10}$$

其中，$\boldsymbol{z}_t$是解码器在t时刻的隐藏层状态，$\boldsymbol{z}_{t-1}$为t−1时刻的隐藏层状态，$\hat{\boldsymbol{x}}_{t-1}$为$t$−1时刻输出的词项中的单词．

根据$\boldsymbol{z}_t$估算当前单词的概率：

$$p\left(\hat{\boldsymbol{x}}_t\right)=\mathrm{softmax}\left(\boldsymbol{W}_s\boldsymbol{z}_t+\boldsymbol{b}_s\right) \tag{11}$$

其中，$\boldsymbol{W}_s$是参数矩阵，$\boldsymbol{z}_t$是解码器在t时刻的隐藏层状态，$\boldsymbol{W}_s\boldsymbol{z}_t+\boldsymbol{b}_s$对每个可能的输出单词进行打分，用softmax归一化得到每个单词$\hat{\boldsymbol{x}}_t$的概率$p\left(\hat{\boldsymbol{x}}_t\right)$.

自编码器的训练目标是使输出正确词项的概率最大，训练时的损失函数L计算方法如下：

$$L=-\sum_{t=1}^{T}\log p\left(\hat{\boldsymbol{x}}_t\right) \tag{12}$$

在自编码器训练结束后，其损失函数值L趋于稳定．此时，$\boldsymbol{E}_T$中的值即为LSTMAE构造的短语向量．

2.3 基于卷积神经网络的自编码器ConvAE

卷积神经网络CNN在对序列进行编码时也有其特有的优点：首先，CNN通过一个固定大小的窗口对序列进行扫描，可以捕捉序列中局部的、位置固定的信息；其次，CNN不需要在时间上迭代计算，相对于LSTM训练速度更快．因此，设计基于CNN的自编码器ConvAE．在编码阶段用CNN代替之前的全连接网络，通过卷积、池化等操作来保留短语上的序列信息．解码阶段使用全连接网络．ConvAE的结构如图2所示．

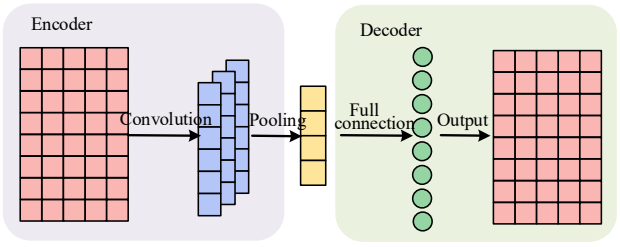

图2 ConvAE结构图

在编码部分，CNN主要包括输入层、卷积层和池化层．在输入层，把词项c_j中各个单词对应的词向量连接起来作为输入，即：

$$\boldsymbol{x}_{1:n}=\boldsymbol{x}_1\oplus\boldsymbol{x}_2\oplus\ldots\oplus\boldsymbol{x}_n \tag{13}$$

其中，$\boldsymbol{x}_i$是输入单元的向量表示，i=1，2，…，n，n为词项中单词的个数．

然后，用一个长度为h的卷积核对输入序列进行卷积，计算卷积核和输入之间的点积，得到一个特征值$\boldsymbol{c}_i$，计算方法如式(14)所示．

$$\boldsymbol{c}_i=f\left(\boldsymbol{w}\cdot\boldsymbol{x}_{i:i+h-1}+\boldsymbol{b}\right) \tag{14}$$

其中，$\boldsymbol{w}\cdot\boldsymbol{x}_{i:i+h-1}$是卷积核$\boldsymbol{w}$在输入序列的某个长度为$h$的子序列$\boldsymbol{w}\cdot\boldsymbol{x}_{i:i+h-1}$上的卷积操作，$\boldsymbol{b}$是偏置项，$f$为ReLU激活函数．

把卷积核$\boldsymbol{w}$应用于全部输入窗口$\{\boldsymbol{x}_{1:h},\ \boldsymbol{x}_{2:h+1},\ \cdots,\ \boldsymbol{x}_{n-h+1:n}\}$，可以产生一个$n$−$h$+1维的激活图$\boldsymbol{c}$：

$$\boldsymbol{c}=\left[\boldsymbol{c}_1,\boldsymbol{c}_2,\ldots,\boldsymbol{c}_{n-h+1}\right] \tag{15}$$

用池化操作选择最能代表特征的部分，这既可以减少数据量，又可以保留特征．为了获取$\boldsymbol{c}$中的整体特

征,这里采用了平均池化方法:

$$\hat{\boldsymbol{c}}=\frac{1}{n-h+1}\sum \boldsymbol{c}_i \tag{16}$$

这样,一个卷积核在输入序列上产生一个特征值 $\hat{\boldsymbol{c}}$,F 个卷积核就能产生 F 个特征值,设定的卷积核的数量就是最终词项的短语向量维度:

$$\boldsymbol{v}=\left[\hat{\boldsymbol{c}}_1,\hat{\boldsymbol{c}}_2,\ldots,\hat{\boldsymbol{c}}_F\right] \tag{17}$$

在解码器部分,采用 n 个全连接网络,n 为词项中的单词数,词项中的每个单词对应一个全连接网络. 在每个全连接网络中,输入层是编码器隐藏层的短语向量,输出层是对应的单词:

$$\hat{\boldsymbol{x}}_i=f\left(\boldsymbol{W}_i\boldsymbol{v}+\boldsymbol{b}_i\right) \tag{18}$$

其中,$\boldsymbol{W}_i$、$\boldsymbol{b}_i$ 为各个全连接网络中的权重矩阵和偏置项,i=1,2,…,n,f 为 ReLU 激活函数.

训练过程中模型的损失函数使用均方误差(Mean Squared Error,MSE),训练目标为使输入词项和输出词项之间的误差尽可能小:

$$\mathrm{MSE}=\frac{1}{n}\sum_{i=1}^{n}\left(\boldsymbol{x}_i-\hat{\boldsymbol{x}}_i\right)^2 \tag{19}$$

通过以上构建的两种短语向量模型 LSTMAE 和 ConvAE,将自编码器中的全连接网络替换为复杂的 LSTM 或 CNN 网络,可以利用短语序列上的信息对词向量压缩,从而得到短语的向量表示.

3　关键词抽取算法 PhraseVecRank

在 TextRank 中,文档中的候选词项构成图,用候选词项在文档中的共现关系构造边,然后通过候选词项之间的相互投票来迭代计算权值,也就是说每个顶点的票数会均匀地投给与它相连的每个顶点. 这样的方法虽然简单,但忽略了文档的主题性,也没有考虑顶点之间的语义关系.

为抽取出更具主题意义的关键短语,本文提出基于短语向量和主题加权的 TextRank 算法(PhraseVecRank). 算法流程示意如图 3 所示,包括候选短语选择、主题权重计算和候选短语排序三个步骤.

3.1　候选短语选择及短语向量表示

考虑到文本的关键信息并不是以单个词的形式存在,而是由多个词组合而成的短语的方式体现. 因此首先需要获取候选短语及短语向量表示.

候选短语主要依靠单词的词性标注和在文档中的相邻位置来确定. 文本中有重要表示意义的词往往是某些固定的词性及其组合,可以把文档中位置相邻的单词视为短语,通过 n-gram 词项和词性标注来选取候选词项,将其加入候选短语集合中. 这里,将第 i 篇文档 $\boldsymbol{d}_i$ 的候选短语集合记作 C_{d_i}.

接着,使用大量候选短语 $\boldsymbol{c}_j=(\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_T)$ 对自编码器 LSTMAE 或 ConvAE 进行训练. 其中,$\boldsymbol{x}_i$ 是候选短语 $\boldsymbol{c}_j$ 中第 i 个单词的词向量表示,T 表示候选短语中的单词数量. 依次输入 $\boldsymbol{x}_i$ 对应的词向量表示,经自编码器 LSTMAE 或 ConvAE 编码获得 $\boldsymbol{c}_j$ 的短语向量表示. 然后,该短语向量经自编码器 LSTMAE 或 ConvAE 解码得到解码序列 $\boldsymbol{x}_i$ 对应的概率值. 自编码器输出的是对应每个单词的概率,训练目标是使输出正确单词的概率最大,在损失函数值趋于稳定时,编码所得的中间向量即为候选短语的短语向量表示.

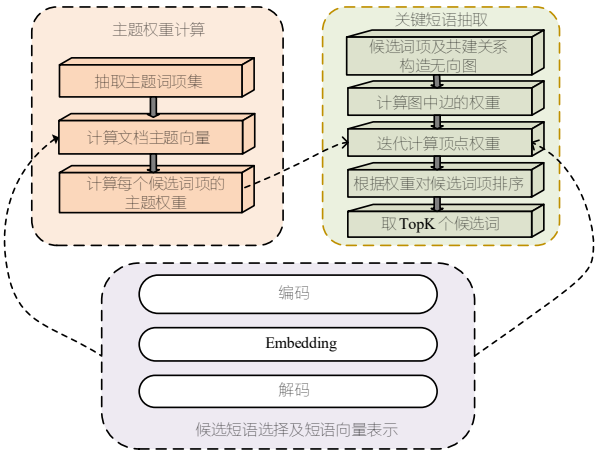

图 3　PhraseVecRank 算法流程示意图

在自编码器训练完成后,当需要获取候选短语的短语向量表示时,只需利用编码部分计算就可获得候选短语的短语向量表示. 短语向量能够从整体角度代表该候选短语的语义信息.

3.2　主题权重计算

每篇文档都具有一定的主题,为了更好地抽取出有主题意义的短语,在基于长短时记忆神经网络的自编码器 LSTMAE 和基于卷积神经网络的自编码器 ConvAE 构建的短语向量基础上,本文进一步提出文档主题向量的概念. 通过主题向量代表文档的主题,然后根据候选短语与文档主题向量之间的相似度计算每个候选短语的主题权重.

定义 1　文档主题短语　依据文档中具有高度概括性的句子或段落,根据词项和词性标注保留 n-gram 作为文档主题短语. 第 i 篇文档 $\boldsymbol{d}_i$ 的主题短语的集合称为文档的主题短语集 T_{d_i},即 $T_{d_i}=\{\boldsymbol{t}_1,\boldsymbol{t}_2,\ldots,\boldsymbol{t}_n\}$,其中,$t$ 表示文档 $\boldsymbol{d}_i$ 的主题短语,n 为文档 $\boldsymbol{d}_i$ 中主题短语集中的短语数目.

定义 2　文档主题向量　文档 $\boldsymbol{d}_i$ 的主题短语集 T_{d_i} 中所有主题短语对应的短语向量的平均值,称为文档主题向量 $\hat{\boldsymbol{T}}_{d_i}$,文档主题向量用于表示整篇文档的主题,即:

$$\hat{\boldsymbol{T}}_{d_i}=\frac{1}{n}\sum_{i=1}^{n}\hat{\boldsymbol{t}}_i \tag{20}$$

其中，$\hat{\boldsymbol{t}}_i$是主题短语$\boldsymbol{t}_i$使用短语向量模型LSTMAE或ConvAE编码后的短语向量表示.

定义3 主题权重 对于候选短语$\boldsymbol{c}_j$，其短语向量表示与文档主题向量之间的相似度称为候选短语$\boldsymbol{c}_j$的主题权重，记作$\boldsymbol{w}_{c_j}^{d_i}$，即：

$$\boldsymbol{w}_{c_j}^{d_i}=\cos\left(\hat{\boldsymbol{T}}_{d_i},\hat{\boldsymbol{c}}_J\right) \tag{21}$$

其中，$\hat{\boldsymbol{c}}_J$是候选词项$\boldsymbol{c}_j$使用短语向量模型LSTMAE或ConvAE编码后的短语向量表示，即LSTMAE: $\boldsymbol{c}_j\rightarrow\hat{\boldsymbol{c}}_J$，或者ConvAE: $\boldsymbol{c}_j\rightarrow\hat{\boldsymbol{c}}_J$，cos表示余弦距离.

候选短语的主题权重能够表示候选短语接近或者代表该文本的程度，权重越大则表示该候选短语距离文本的主题越近，能够更好地表达该主题文本.

3.3 候选短语排序

获得候选短语及其主题权重之后，算法以候选短语作为图中的顶点，以候选短语的共现信息为边来构造图，以候选短语之间的语义相似度和共现信息计算边的权重，迭代计算每个候选短语的得分并排序.

首先，构造无向图. 以文档$\boldsymbol{d}_i$的候选短语集$\boldsymbol{C}_{d_i}$中的所有元素为顶点构造一个无向图，如果两个候选短语$\boldsymbol{c}_j$和$\boldsymbol{c}_k$在一个长度为n的共现窗口中出现，则它们之间存在一条边.

然后，依据LSTMAE和ConvAE构造的短语向量计算候选短语权重，即边的权重.

定义4 候选短语权重 候选短语$\boldsymbol{c}_j$和$\boldsymbol{c}_k$的短语向量表示之间的余弦距离和两者的共现次数的乘积称为候选短语权重，记作$\boldsymbol{w}_{jk}$，即：

$$\boldsymbol{w}_{jk}=\cos\left(\vec{\boldsymbol{c}}_J,\vec{\boldsymbol{c}}_k\right)\times\mathrm{occur}_{\mathrm{count}}\left(\boldsymbol{c}_j,\boldsymbol{c}_k\right) \tag{22}$$

其中，$\vec{\boldsymbol{c}}_J$、$\vec{\boldsymbol{c}}_k$分别是候选短语$\boldsymbol{c}_j$和$\boldsymbol{c}_k$的短语向量表示，cos表示余弦距离，$\mathrm{occur}_{\mathrm{count}}\left(\boldsymbol{c}_j,\boldsymbol{c}_k\right)$表示$\boldsymbol{c}_j$和$\boldsymbol{c}_k$在共现窗口中共同出现的次数.

这里，将两个候选短语的相似度和共现次数相乘的主要目的是用共现次数来加强它们的语义联系，为图中的每条边分配权重.

迭代计算候选短语得分，即迭代计算图中各个顶点的权重. 对于与候选词项$\boldsymbol{c}_j$相连的顶点$\boldsymbol{c}_j^{d_i}$，依据主题权重$\boldsymbol{w}_{c_j}^{d_i}$和短语向量权重$\boldsymbol{w}_{jk}$，计算文档$\boldsymbol{d}_i$的候选短语$\boldsymbol{c}_j$的评分$R\left(\boldsymbol{c}_j^{d_i}\right)$. 计算方法如下：

$$R\left(\boldsymbol{c}_j^{d_i}\right)=(1-\boldsymbol{d})\times\boldsymbol{w}_{c_j}^{d_i}+\boldsymbol{d}\times\sum_{\boldsymbol{c}_k^{d_i}\in\varepsilon\left(\boldsymbol{c}_j^{d_i}\right)}\frac{\boldsymbol{w}_{jk}}{\sum_{\boldsymbol{c}_p^{d_i}\in\varepsilon\left(\boldsymbol{c}_k^{d_i}\right)}\boldsymbol{w}_{kp}}R\left(\boldsymbol{c}_k^{d_i}\right) \tag{23}$$

其中，$\boldsymbol{w}_{c_j}^{d_i}$是文档$\boldsymbol{d}_i$的候选短语$\boldsymbol{c}_j$的主题权重，$\varepsilon\left(\boldsymbol{c}_j^{d_i}\right)$表示与$\boldsymbol{c}_j$相连的顶点集合. d为阻尼系数，通常取值为0.85. 而

$$\boldsymbol{d}\times\sum_{\boldsymbol{c}_k^{d_i}\in\varepsilon\left(\boldsymbol{c}_j^{d_i}\right)}\frac{\boldsymbol{w}_{jk}}{\sum_{\boldsymbol{c}_p^{d_i}\in\varepsilon\left(\boldsymbol{c}_k^{d_i}\right)}\boldsymbol{w}_{kp}}R\left(\boldsymbol{c}_k^{d_i}\right)$$

则表示与$\boldsymbol{c}_j$相连的顶点给$\boldsymbol{c}_j$的投票.

最后，通过迭代计算候选短语得分进行候选短语排序. 在多次迭代后，图中的每个顶点都能得到一个稳定的得分. 依据得分对顶点排序，以得分最高的TopK个候选短语作为最终的关键短语.

详细步骤如算法1所示. 依据短语向量表示，算法通过主题向量引入了文档的主题信息，然后在计算顶点权重时，既考虑了候选词项的主题信息，又考虑了它们之间的语义关系和共现关系，使抽取出的关键词更加精确.

算法1 PhraseVecRank算法

输入 文档$\boldsymbol{d}_i$；候选词项词性；n-gram大小；共现窗口大小window_size；关键词保留个数Top$K$

输出 抽取出的Top$K$个关键词

步骤1：数据预处理，对原始文本分词并标注词性；

步骤2：根据词性保留n-gram，得到候选短语集；

步骤3：计算各候选短语的主题权重：

(1)把主题段落中的主题短语加入主题短语集；

(2)计算文档主题向量；

(3)计算每个候选短语与文档主题向量的相似度即该短语的主题权重；

步骤4：将所有候选短语作为顶点构造图，如果两个候选短语在一个共现窗口出现，则它们之间存在一条边；

步骤5：通过两个顶点的相似度和共现次数，计算图中边的权重；

步骤6：迭代计算顶点权重，直到权重收敛或达到最大迭代次数；

步骤7：将候选短语按照权重由大到小排序，保留前Top$K$个短语作为文档的关键短语.

4 实验结果与分析

为了验证本文算法的有效性，本小节设计了两组实验，尤其通过对中文学术论文中专业词汇的抽取，验证本文提出的短语向量模型对中文短语的语义信息的表达能力.

4.1 数据集及评价指标

英文数据采用Inspec公开数据集. 从Inspec数据库中选取计算机控制、信息技术领域的2000份期刊摘要. 其中，训练集1000篇，验证集500篇，测试集500篇. 平均每篇文档有9.81个关键词，数据集中关键词不在文档中的比例为21.52%.

中文学术论文数据集采用环保、生物、物理、化学、数学、财经、文学等领域的学术论文共计59913篇，数据中包括“论文题目”、“发表年份”、“摘要”、“关键词”和“英文关键词”等. 在关键词抽取实验中，以数据集中的

“论文题目”和“摘要”作为文本内容,“关键词”作为标注数据来验证抽取结果. 平均每篇论文有4.2个关键词,且包含大量专业领域的短语词汇. 数据集中关键词不在论文摘要中的比例为27.44%.

评价指标选用准确率、召回率和$F1$值,具体定义如下,式中N表示集合中的元素数目.

$$P=\frac{N(\text{抽取集合}\cap\text{标注集合})}{N(\text{抽取集合})}\times 100\%$$
$$R=\frac{N(\text{抽取集合}\cap\text{标注集合})}{N(\text{抽取集合})}\times 100\% \tag{24}$$
$$F1=\frac{2\times P\times R}{P+R}\times 100\%$$

4.2 实验参数设置

在训练自编码器时,取数据集中的“关键词”字段作为训练数据,参数如表1所示. 在设定n-gram值时,统计标注关键词长度(单词数),结果如表2所示,全部254376个关键词的平均长度为1.98,约93.9%的关键词的长度在1到3之间. 因此,从文档中选取1-gram、2-gram和3-gram词项加入候选集合中.

表1　训练参数值设置

参数类型	LSTMAE	ConvAE
epoch	500	5000
batch_size	128	1000
LSTM_size	200	-
encoder_embedding_size	100	-
decoder_embedding_size	100	-
filter_len	-	2
filter_num	-	200
learning rate	0.1	1.0

表2　关键词长度分布

关键词长度	数目	占比
1	90161	35.4%
2	115322	45.3%
3	33550	13.2%
4	9175	3.6%
5	3042	1.2%
其它	3126	1.2%

在对学术论文集进行关键词词性统计时,词性标注利用Jieba分词工具完成,统计结果如表3所示. 可以看出,关键词中73.1%的单词词性集中在名词、动词和动名词. 因此,PhraseVecRank方法取文档的名词、动词、动名词及其组合作为候选短语. 在计算主题分布时,从“题目”中抽取候选词项计算文本主题权重. 在进行候选短语排序时共现窗口大小初始化为3,并取Top5、Top10、Top15进行验证.

表3　单词词性分布

词性	数目	占比
n	252862	50.3%
v	84527	16.8%
vn	30302	6.0%
eng	19338	4.0%
x	16666	3.3%
nz	16272	3.2%
a	13667	2.7%
其它	69218	13.7%

4.3 关键词抽取实验结果分析

实验对比的基准算法包括:基于统计信息的TFIDF算法[11]、基于图的TextRank算法[3]及其改进算法SingleRank[4]、TopicalPageRank[5],有监督算法KEA[13]、Maui[14]、有监督的关键词生成算法CopyRNN[15],以及本文提出的算法PhraseVecRank.

对于PhraseVecRank算法,对比四种不同的短语向量的构建方法,包括:词向量平均值(Word-embedding Average, WA)、Li等人提出的基于ITG重排序构建的递归自编码器(ITG-RAE)[12]和本文提出的两种自编码器LSTMAE和ConvAE.

英文Inspec数据集的实验结果如表4所示,可以看出,TFIDF算法通过词频选择文本中的关键词,而TextRank算法在计算时依赖文本中词的共现关系,使得文本中经常使用的但未必能够代表文章主题的词项分值较高,因此,TextRank方法的实验结果并不理想. SingleRank算法改进了TextRank算法,通过外部文本引入了部分主题信息,算法效果相较TextRank方法有一定程度的提升.

表4　Inspec数据集Top10实验结果

关键词抽取算法	准确率P(%)	召回率R(%)	$F1$值(%)
TFIDF	32.7	38.6	33.4
TextRank	22.99	11.44	15.28
SingleRank	34.8	40.4	35.2
Maui	-	-	4.2
KEA	-	-	12.6
CopyRNN	-	-	34.2
PhraseVecRank-WA	34.6	41.6	37.8
PhraseVecRank-ITG-RAE	34.9	42.1	38.2
PhraseVecRank-LSTMAE	**36.0**	**45.7**	**40.3**
PhraseVecRank-ConvAE	35.7	44.5	39.6

利用特征向量表示候选短语的KEA方法和基于统计关键词抽取的Maui是较早期的有监督算法,实验结果不理想. 有监督的CopyRNN算法使用序列到序列(Seq2Seq)的方法为文档生成关键短语. 通常有监督的

方法在特定任务和数据集上能够取得更好的效果. 因此,CopyRNN算法总体$F1$值高于经典的无监督关系抽取方法.

PhraseVecRank算法在准确率、召回率和$F1$值取得了良好的结果. PhraseVecRank-LSTMAE的实验结果最优,说明短语向量LSTMAE能较好地捕捉Inspec数据集上短语的语义信息.

中文学术论文数据集的实验结果如表5所示,在学术论文集上的实验结果整体上比在英文Inspec数据集上的结果偏低,其主要原因在于:首先,中文学术论文数据集中27.44%的关键短语没有在论文摘要中出现,而本实验中对比的方法都属于关键词抽取算法、只能抽出原文中存在的词,这将限制抽取式模型的最终性能. 其次,中文学术论文数据中的关键词是论文作者给出的学术论文内容总结和提炼后的专业名词短语,通过算法自动抽取专业领域的学术短语的难度很大,很难完全匹配标注的关键词. 因此,所有算法在中文学术论文集上的实验结果均低于在英文数据集的结果.

表5 中文学术论文数据集Top实验结果对比

关键词抽取算法	准确率P(%)			召回率R(%)			$F1$值(%)		
	Top5	Top10	Top15	Top5	Top10	Top15	Top5	Top10	Top15
TFIDF	10.89	7.17	5.38	12.99	17.08	19.10	11.85	10.10	8.39
TextRank	9.01	6.08	4.67	10.74	14.48	16.59	9.80	8.56	7.28
SingleRank	11.04	7.79	5.69	13.16	18.54	20.32	12.00	10.97	8.89
TopicalPageRank	11.88	8.25	6.23	14.14	19.64	22.25	12.91	11.61	9.73
PhraseVecRank-WA	12.60	9.69	7.57	15.01	23.11	27.08	13.69	13.65	11.83
PhraseVecRank-ITG-RAE	12.69	9.73	7.59	15.14	23.20	27.16	13.81	13.71	11.86
PhraseVecRank-LSTMAE	13.59	10.32	8.16	16.84	25.41	29.53	15.04	14.68	12.79
PhraseVecRank-ConvAE	**14.02**	**10.39**	**8.30**	**17.26**	**25.57**	**29.91**	**15.47**	**14.78**	**12.99**

对比表5中的结果数据,通过词频选择关键词的TFIDF算法,以及依据文本中词的共现关系TextRank算法实验结果相当,SingleRank算法效果有相应提升. TopicalPageRank算法需要先用LDA在文档中训练主题模型,算法效果进一步得到了改善,但是总体效率不高. 本文提出的PhraseVecRank算法在准确率、召回率和$F1$值均有比较明显的提升,主要原因是PhraseVecRank利用了短语向量,并通过主题权重引入了文档中的主题信息,更能够捕捉文中的关键短语.

4.4 短语向量模型的效果分析

对于PhraseVecRank算法,表5进一步对比了四种不同的短语向量构建方法的实验结果. 使用词向量求平均值的方法PhraseVecRank-WA把各个词的语义进行平均,可能会把短语映射到一个没有意义的语义空间中,因此效果不如使用自编码器对词向量进行压缩的方法. 递归自编码器(ITG-RAE)通过机器翻译任务将词向量进行组合,效果比直接对词向量求平均值要好,但是递归自编码器使用全连接网络直接对词向量进行压缩,忽略了短语结构中的序列信息,因此效果提升有限.

本文提出的两种自编码器结构从不同角度获取短语中的序列信息,取得了最好的效果. 其中ConvAE能够获取短语中固定窗口大小上的组合信息,因此在文本长度偏短而且组合模型相对固定的短语上取得的效果更好;而LSTMAE能够捕捉序列上的上下文信息,更适用于文本较长且更加依赖上下文的短语组合.

参数变化对PhraseVecRank算法的影响情况如表6所示. 对于TopK取值方面,随着K取值的增大,正确的抽取结果会增多,但是,由于保留的抽取结果数量变多,算法的准确率会降低,而召回率则越来越高. 可以看出TopK值取5时,算法的效果可以达到最好. 当TopK取值偏大时保留的冗余数据过多,反而会降低算法的效果.

表6 不同K取值对结果的影响

关键词抽取算法	Top2	Top3	Top5	Top10	Top15
TFIDF	8.12	9.54	11.85	10.10	8.39
TextRank	8.03	9.24	9.80	8.56	7.28
SingleRank	8.22	10.14	12.00	10.97	8.89
TopicalPageRank	8.41	10.53	12.91	11.61	9.73
PhraseVecRank-WA	8.54	11.14	13.69	13.65	11.83
PhraseVecRank-ITG-RAE	8.85	11.45	13.81	13.71	11.86
PhraseVecRank-LSTMAE	10.17	12.74	15.04	14.68	12.79
PhraseVecRank-ConvAE	**10.53**	**12.98**	**15.47**	**14.78**	**12.99**

为了验证不同共现窗口大小对抽取结果的影响,在PhraseVecRank-ConvAE实验中,取TopK值为5,观察窗口大小分别取1~5时算法的准确率、召回率和$F1$值的实验结果如图4所示. 当共现窗口大小为3时,抽取效果最好. 主要原因是,如果共现窗口过小,能够利用的单词之间的共现信息会过少,而如果共现窗口过大,反而会带来更多无用信息,导致算法准确率降低.

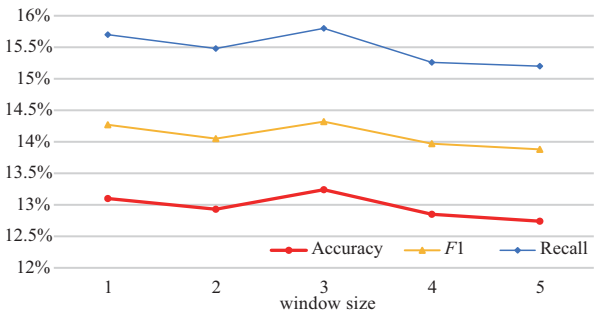

图4　不同共现窗口大小的抽取结果

综上分析,英文公共数据集上的实验结果表明PhraseVecRank方法具有与经典方法相当的关键词抽取性能. 特别地,针对中文学术论文数据集,PhraseVecRank方法能够很好的抽取和表达专业领域的词汇短语.

5　结论

本文研究基于图的无监督关键词抽取算法. 为了更好地获取短语向量表示,设计了基于LSTM和CNN的自编码器LSTMAE、ConvAE来构造短语向量. 在短语向量模型基础上,提出基于短语向量和主题加权的关键词抽取方法PhraseVecRank,利用候选词和短语向量表示为每个候选短语分配主题权重,然后利用候选短语之间的短语向量相似度和共现次数计算网络图中边的权重. 实验结果证明,PhraseVecRank算法能够有效地提高关键词抽取算法的准确率、召回率和$F1$值,同时利用自编码器构造的短语向量可以更好地表示短语的语义信息.

参考文献

[1] Papagiannopoulou E, Tsoumakas G. A review of keyphrase extraction[J]. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2020, 10(2): e1339.

[2] 刘慧婷, 刘志中, 王利利, 等. 一般间隙序列模式挖掘的关键词抽取[J]. 电子学报, 2019, 47(5): 1121 – 1128.
Liu H T, Liu Z Z, Wang L L, et al. Keyphrase extraction using sequential patterns mining algorithm with one-off and general gaps condition[J]. Acta Electronica Sinica, 2019, 47(5): 1121 – 1128.(in Chinese)

[3] Mihalcea R , Tarau P . TextRank: Bringing order into texts [A]. Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing[C]. Barcelona, Spain : ACL, 2004. 404 – 411.

[4] Wan X J, Xiao J G. Single document keyphrase extraction using neighborhood knowledge[A]. Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence [C]. Seattle, Washington: AAAI Press,2008. 855 – 860.

[5] Liu Z, Huang W, Zheng Y, Sun M. Automatic keyphrase extraction via topic decomposition[A]. Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing. Massachusetts[C]. Cambridge, MA: ACL, 2010. 366 – 376.

[6] Florescu C, Caragea C. PositionRank: An unsupervised approach to keyphrase extraction from scholarly documents [A]. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics[C]. Vancouver, Canada : ACL, 2017. 1105 – 1115.

[7] 马慧芳, 刘芳, 夏琴, 等. 基于加权超图随机游走的文献关键词提取算法[J]. 电子学报, 2018, 46(6): 1410 – 1414.
Ma H F, Liu F, Xia Q, et al. Keywords extraction algorithm based on weighted hypergraph random walk[J]. Acta Electronica Sinica, 2018, 46(6): 1410 – 1414.(in Chinese)

[8] Bojanowski P, Grave E, Joulin A, et al. Enriching word vectors with subword information[J]. Transactions of the Association for Computational Linguistics, 2017, 5: 135 – 146.

[9] Sun Y, Qiu H P, Zheng Y, et al. SIFRank: A new baseline for unsupervised keyphrase extraction based on pre-trained language model[J]. IEEE Access, 2020, 8: 10896 – 10906.

[10] Peters M, Neumann M, Iyyer M, et al. Deep contextualized word representations[A]. Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies[C]. New Orleans, Louisiana: ACL, 2018. 2227 – 2237.

[11] Salton G, Buckley C. Term-weighting approaches in automatic text retrieval[J]. Information Processing & Management, 1988, 24(5): 513 – 523.

[12] Li P, Liu Y, Sun M. Recursive autoencoders for ITG-based translation[A]. Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing [C]. Seattle: ACL, 2013. 567 – 577.

[13] Witten I H, Paynter G W, Frank E, et al. KEA: practical automatic keyphrase extraction[A]. Proceedings of the fourth ACM conference on Digital Libraries[C]. Berkeley: ACM, 1999. 254 – 255.

[14] Medelyan O, Frank E, Witten I H. Human-competitive tagging using automatic keyphrase extraction[A]. Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing[C]. Singapore: ACL, 2009. 1318 – 1327.

[15] Meng R, Zhao S Q, Han S G, et al. Deep keyphrase gener-

ation[A]. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics[C]. Vancouver, Canada : ACL, 2017. 582 – 592.

作者简介

孙 新 女，1975年4月生于吉林省长春市.现为北京理工大学计算机学院副教授，硕士生导师. 主要研究方向为人工智能，机器学习.
E-mail: sunxin@bit.edu.cn

盖 晨 男，1996年5月生于河北省石家庄市.现为北京理工大学计算机学院硕士生研究生.主要研究方向为人工智能.
E-mail: 851774342@qq.com

申长虹 女，1995年3月生于河南省.北京理工大学计算机学院硕士生.主要研究方向为人工智能.
E-mail: suishelly@foxmail.com