IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

收稿日期2019年12月20日，收稿日期2020年1月4日，发表日期2020年1月9日，现版日期2020年1月17日。*数字对象标识符10.1109/ACCESS.2020.2965087*

# SIFRank:一种基于预训练语言模型的无监督关键词提取新基线

孙毅[1]，邱杭平[1]，郑宇[2]，王忠伟[1]，和张超然[1]

[1]陆军工程大学指挥与控制工程学院，南京210001

[2]工业和信息化部第五研究所赛普瑞(南京)实验室，南京211800

通讯作者:邱杭平(qiuhp_zy@163.com)

本文受装备发展基金项目资助，项目编号6141B08010101。

在社交媒体时代，面对海量的知识和信息，需要在信息检索和自然语言处理中应用准确有效的关键词提取方法。传统的关键词提取模型很难包含大量的外部知识信息，但随着预训练语言模型的兴起，解决这一问题有了新的途径。基于上述背景，我们提出了一种新的基于预训练语言模型SIFRank的无监督关键短语提取基线。SIFRank将句子嵌入模型SIF和自回归预训练语言模型ELMo相结合，在短文档关键词提取中具有最佳性能。我们通过文档分割和上下文词嵌入对齐来加快SIFRank的速度，同时保持其准确性。对于长文档，我们通过位置偏置权重将SIFRank升级为SIFRank+，大大提高了其在长文档上的性能。与其他基线模型相比，我们的模型在三个广泛使用的数据集上达到了最先进的水平。

**关键词提取**，预训练语言模型，句子嵌入，位置偏置权重，SIFRank。

## I. 介绍

关键词提取是从文档中选择一组可以概括文档中讨论的主要主题的单词或短语的任务[1]。关键词提取可以大大加快信息检索的速度，帮助人们快速、准确地从冗长的文本中获取第一手信息。

### A. 动机

关键词提取可以分为两种主要的方法:有监督和无监督。监督方法在特定的领域任务上表现更好，但需要大量的劳动来注释语料库，并且训练后的模型可能会过拟合而不能很好地处理其他数据集。传统的主要无监督方法主要分为基于统计的模型和基于统计的模型图。统计模型通常使用词频、n-gram特征、位置、文档语法等不同的信息特征，但这类信息很难反映文档中单词之间的复杂关系。基于图的模型将人类语言视为一个复杂的网络[2]，使用图来对文档中单词或短语之间的关系进行建模。最典型的模型是TextRank，后期的模型使用不同的算法或外部信息对TextRank进行优化[3]。

尽管基于图的模型是有效的，但通过引入外部知识或附加特征可以更好地提高关键词提取的效果。使用预训练的语言模型是可以提供大量外部知识的方法之一。根据Papagiannopoulou和Tsoumaka[4]的综述论文，这被概括为基于嵌入的模型。随着预训练语言模型的兴起，ELMo[5]、Bert[6]、XLNet[7]等深度神经网络模型很好地解决了自然语言处理中的许多监督任务。

协调本稿审稿并批准发表的副主编是韩帅。

词的特征不再是静态的词嵌入，如Word2Vec，而是动态的，实时的和上下文的词嵌入，如ELMo。预训练的语言模型是在大规模的未标记语料库上进行预训练的，文本的表示可以根据不同的语境进行动态调整。基于上述优点，在关键词提取中使用预训练的语言模型可以将统计模型和基于图的模型的优点结合起来。

句子嵌入是句子或文档的表示。获取句子嵌入的方法有很多，由于注意模型的存在，这些方法可以很好地应用于不同的下游监督任务。注意模型可以用监督训练嵌入的权值。然而，在无监督的关键词提取任务中，词嵌入、句子嵌入和文档主题之间的关系需要用合适的模型来解释。

有时候，仅仅使用原始的句子嵌入进行关键词提取是不够的。以词袋句子嵌入模型为例，这种模型不包含文档的单词或短语的位置信息。然而，众所周知，位置信息在关键词提取中起着重要的作用，特别是对于长文档。

### B. 贡献

本文的主要贡献总结如下:

i)我们引入句子嵌入模型SIF[8]来解释句子嵌入与文档主题之间的关系。然后将自回归预训练语言模型ELMo与SIF相结合，计算短语嵌入和文档嵌入。余弦相似度用于计算候选日期短语与主题之间的距离。我们的模型叫做SIFRank。

SIFRank动态实时地计算文本的表示，并根据领域数据信息进行优化。SIFRank在两个短文档数据集(Inspec和DUC2001)上实现了最先进的效果。此外，我们的模型比以前的SOTA模型embed更健壮。

ii)我们提出了一种称为文档分割的方法来加快长文档中词嵌入的计算过程。然而，由于文档被分割成更小的部分，关键词提取的效果会下降。我们取同一词在不同位置和上下文的上下文嵌入的平均值作为嵌入锚，然后用嵌入锚替换上下文词嵌入来计算结果，模型的性能明显反弹。

iii)为了提高模型对长文档数据集的关键词提取能力，我们提出了位置偏置权值。我们使用短语第一次出现偏移位置的倒数作为位置偏置权重。然后使用softmax使位置偏置权重均匀且平滑。通过在SIFRank中增加余弦相似度的位置偏置权重，该模型在长文档数据集DUC2001上的性能得到了显著提高，在短文档数据集上的性能没有受到太大影响。这个模型被称为SIFRank+，它在数据集DUC2001上获得了最先进的结果。

## II. 相关工作

在本节中，相关工作主要分为以下3部分:无监督关键字提取、预训练语言模型和基于嵌入的关键字提取。

### A. 无监督关键字提取

无监督关键词提取方法主要利用文档的不同特征，如词频特征、位置特征、语言特征、主题特征、长度特征、词间关系、外部基于知识的信息等。

基于图的关键词提取是一种最有效、应用最广泛的无监督关键词提取方法。受PageRank[9]的启发，Mihalcea和Tarau提出了TextRank[3]，该模型将文档抽象成一个图，其中单词或短语是图中的节点，单词之间的关系是边。在此之后，人们提出了各种方法来扩展文档图的信息。Wan和Xiao提出了ExpandRank[10]，该方法利用少量的最近邻文档提供更多的知识来改进单个文档的关键词提取。Bougouin等人提出TopicRank[11]，应用该模型通过候选关键词聚类为每个主题分配显著性分数。使用TextRank排名模型对主题进行评分，并通过从每个排名最高的top-ics中选择最具代表性的候选人来提取关键短语。Boudin提出了Multipartite[12]，它在一个多部图结构中编码主题信息，该模型利用关键词的相互增强关系来提高候选排名。Florescu和Caragea提出了PositionRank[13]，该模型将单词出现的位置信息纳入有偏差的TextRank中，显著提高了TextRank对长文档的处理效果。位置偏置权重为单词在文档中逆位置的the$_{\text{P}}$sum，$p(w_i)=\ _k1/p_k$ $(w_i,\ d)$。

### B. 预训练的语言模型

预训练语言模型一般是通过神经网络结构在大规模未标记语料库上进行训练，然后通过提取网络特征或共享网络参数来应用于下游任务的模型(Peters等人[14]认为主要有两种范式:特征提取和微调)。预训练语言模型的发展大致经历了三个阶段:静态文本嵌入模型、上下文文本嵌入模型和微调模型。

静态文本嵌入模型的权重是固定的，文本的表示是固定的。经典的词嵌入模型如Word2Vec[15]和GloVe[16]。Joulin等人提出了FastText[17]，该模型在Word2Vec基础上增加了一个基于字符的n-gram模型，使得计算词汇外词的嵌入(OOV)成为可能。句子嵌入是更高粒度的文本表示。Le和Mikolov在Word2Vec的基础上提出了Doc2Vec[18]。Kiros等人提出了skip-thoughts[19]，它训练一个编码器-解码器模型来重建编码段落的周围句子。Arora等人提出SIF[8]，其中句子被表示为单词嵌入的加权平均值。Pagliardini等人提出Sent2Vec[20]，它使用单词的n-gram特征来生成句子嵌入。

上下文文本嵌入模型可以基于上下文动态地计算文本嵌入。McCann等人[21]提出的CoVe将静态词嵌入GloVe输入到监督神经机器翻译任务中，得到基于上下文的嵌入。Peters等人[5]提出的ELMo模型是一种深度情境化表示方法。词嵌入是深度双向语言模型(biLM)内部状态的学习函数，该模型在大型语料库上进行预训练。

微调型预训练语言模型的预训练参数是解冻的，可以在新任务上进行微调。这种类型的模型不再提取文本的表示。Devlin等人提出了一种自编码预训练语言模型BERT[6]，一种深度双向变形模型，该模型引入了两个任务:掩模语言模型(mask language model, MLM)和下一句预测。在微调步骤中，不同的任务仅在输入和输出层有所不同。Yang等人提出了一种集成Transformer-XL[7]的广义自回归预训练语言模型XLNet，该模型可以通过最大化分解顺序的所有排列的期望似然来学习双向上下文。

根据预训练的方法，将预训练好的语言模型分为自回归(AR)和自编码(AE)两类。AR语言模型，如ELMo和XLNet，试图用自回归模型估计文本语料库的概率分布。与前者不同，BERT等AE语言模型及其变体如RoBERTa[22]并不执行显式密度估计，而是旨在从损坏的输入中重建原始数据。类bert模型的一个大问题是，它们在预训练期间使用了像[MASK]这样的人工符号，但它们不存在于下游任务的文本中。pre -fine - tune的差异对我们的关键词提取模型有一定的影响，这将在后面的实验分析中讨论。

### C. 基于嵌入的关键词提取

预训练语言模型为关键词提取提供了新的研究方向。预训练的模型包含了大量的信息，可以很好地表示单词或短语之间的关系。因此，近年来，基于嵌入的关键词提取取得了很好的性能。

Wang等人[23]提出使用深度信念网络(Deep Belief network)对关键词嵌入的层次关系进行建模。这种方法可以清晰地将目标文档与其他文档区分开来。Papagiannopoulou和Tsoumaka提出了RVA[24]，这是一种局部词向量引导关键词提取模型，该模型以GloVe作为*参考向量,使用在单个文件上训练的所有候选短语嵌入的平均值*，然后计算候选关键词嵌入与参考向量之间的相似度，并将其作为评分进行排序。Bennani-Smires等人提出了embed算法[25]，该算法利用候选关键词的嵌入与文档的句子嵌入之间的余弦相似度。在embed算法中，使用Doc2Vec[18]和Sent2Vec[20]两个句子嵌入模型来获取文档的表示。此外，它们还增加了使用最大边际相关性(MMR)来增加关键词的覆盖率和多样性。

## III. 模型概述

在本节中，我们首先描述了SIFRank的整体结构，然后通过句子嵌入模型SIF合理地解释了词嵌入、句子嵌入和文档主题之间的关系。最后，我们简要介绍了ELMo及其主要特点和用法。

### A. 总体结构

SIFRank模型的框架见图1。在这个模型中，我们遵循了关键词提取的一般过程。主要步骤如下:

步骤1:对文档进行标记，并将词性标记为带有词性标记的标记序列。

步骤2:使用NP-chunker(正则表达式编写的模式)根据词性标签从序列中提取名词短语(NPs)。从文档中提取的NPs是候选关键短语。

步骤3:将token序列放入预训练的语言模型中，提取每个token的表示。在这种情况下，表示可能是具有不同特征的多层词嵌入。

步骤4:通过句子嵌入模型，将NPs与文档的嵌入转化为NP嵌入与文档嵌入。此时，它们具有相同的层数和维数。

步骤5:计算NP嵌入与文档嵌入之间的余弦距离。我们将这个距离视为候选关键词与文档主题之间的相似度。从最相似的候选关键字中选择Top-N作为最终关键字。这是对候选关键字进行排名的最重要因素。

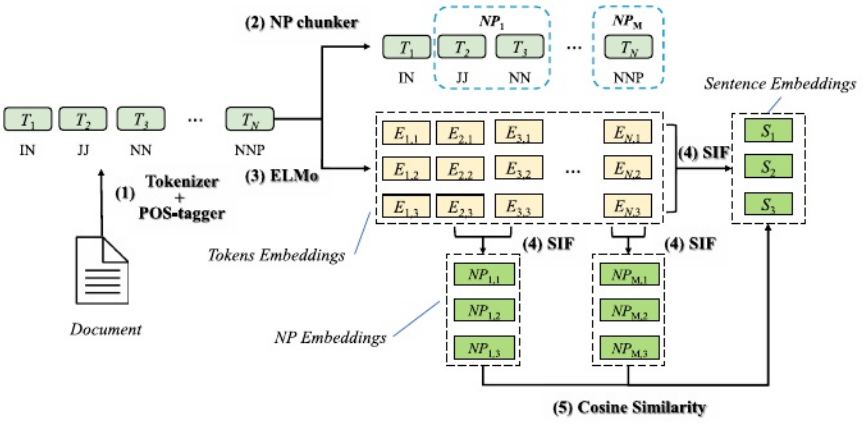



图1。SIFRank模型的框架。

### B. 句子嵌入模型SIF

在本文中，我们选择句子嵌入模型SIF[8]来获得NPs和文档的嵌入。这不仅是因为它能很好地与大多数预训练的语言模型配合，还因为该模型获得的句子嵌入能很好地反映文档的主题。

根据SIF，对于文档$d \in$d，句子$s$的生成是一个动态随机游走过程。在步骤$k$生成$k^{th}$单词$w_k$，假设文档的主题在此过程中没有太大变化。也就是说，所有单词的生成都是由单个主题$cd \in$R决定的。因此，对于给定的句子$s$，句子嵌入是决定整个文档的主题嵌入的最大似然估计。因此，计算候选关键词嵌入与文档嵌入之间的距离，就是计算候选关键词与文档主题之间的相似度。

Arora等人[8]在他们的论文中提出了两个“平滑”的假设。一种是假设一些单词由于上下文而没有出现。另一种假设是高频词(如“the”、“and”)的出现与句子的主题无关。基于这些假设，以$c_d$为主题的句子s的生成概率为:

$$Pr[s|c_d] = \prod_{w \in s} Pr(w|c_d) = \prod_{w \in s} \left[\alpha f_w + (1-\alpha)\frac{\exp(< v_w, \tilde{c}_d >)}{Z_{\tilde{c}_d}}\right] \quad (1)$$

其中$Zc \sim d$ = Pw∈V exp(< c ~ d, $vw$ >),且$c \sim d = \beta c\ 0 + (1-\beta)c\ d$,$c_0 \perp$cd。$f_w$是一个词出现在大型语料库上的统计概率。

最后，句子向量(主题的最大似然估计)可以表示为:

$$v_s = \frac{1}{|s|}\sum_{w \in s}\frac{a}{a+f_w}v_w = \frac{1}{|s|}\sum_{w \in s}\text{Weight}(w)v_w \quad (2)$$

根据经验，超参数$a$很可能适用于[$10^{-3}$，$10^{-4}$]。

### C. 预训练的语言模型emlo

ELMo[5]生成的词嵌入有3层，本文用L0、L1和L2表示，每层有1024个维度。L0是Char Encode Layer(字符编码层)。token的静态嵌入是由这个卷积神经网络层生成的。L1和L2是由biLM(论文中是biLSTM)生成的上下文词嵌入。根据Peters等人在论文中的描述，L1更好地捕获语法信息，L2更好地捕获上下文相关的语义信息。这三层的嵌入可以被加权，然后输入到不同的下游任务中。

## IV. sifrank和sifrank +

在本节中，我们将更详细地介绍SIFRank和SIFRank+模型，包括模型域自适应、文档分割和嵌入对齐、候选关键短语的主题相似度和长文档的位置偏置权重。

### A. SIFRANK

对于给定文档$d$, $d$的嵌入为$v_d$。候选关键词NP的嵌入为$v_{NP}$。SIFRank定义为$v_d$与$vNP$之间的相似度或相关性评分:

$$\text{SIFRank}(v_{NP_i}, v_d) = \text{Sim}(v_{NP_i}, v_d) \quad (3)$$

相似度一般可以用余弦距离来计算:

$$\text{Sim}(v_{NP_i}, v_d) = \cos(v_{NP_i}, v_d) = \frac{\vec{v}_{NP_i} \cdot \vec{v}_d}{\|\vec{v}_{NPi}\|\|\vec{v}_d\|} \quad (4)$$

当使用欧几里得距离来计算相似度时，应该对嵌入的权重进行归一化。

SIFRank的值在0到1之间，它越接近1，候选关键字与文档主题的相关性越高。反之，该值越接近0，则表示该短语与主题越不相关。

### B. 模型域自适应

对于不同域中的文本，单词的概率分布可能是不同的。在某些特定的领域中，通常罕见的词可能是常见的。

为了更好地使模型适应不同领域的任务，我们在句子嵌入计算的过程中改变了词的权函数。这个权值是公共语料库和领域语料库中的加权和:

$$\begin{aligned}\text{Weight}(w) &= \lambda \text{Weight}_{com}(w) + (1-\lambda)\text{Weight}_{dom}(w) \\ &= \lambda \frac{a}{a+f_w} + (1-\lambda)\frac{a'}{a'+f'_w} \end{aligned} \tag{5}$$

其中λ∈[0,1]，$\text{Weight}_{com}(w)$在大规模语料库维基百科上计数，$\text{Weight}_{com}(w)$在任务的领域语料库上计数。

如果统计中没有发现该词出现的频率，则采用该词权重最大化的方法:

$$\text{Weight}(w_i) = \min\{1, \max_{w_j \in s, i \neq j} \text{Weight}(w_j)\} \tag{6}$$

一旦找不到某个单词的频率，则将权重设置为句子中其他单词的最小权重为1，最大权重为1。

### C. 文档分割和嵌入对齐

同一个词在不同的上下文中或不同的位置有不同的词嵌入。因此，将单词嵌入定义为*vswjp*，其中*wi*代表单词，*sj*是

我

句子中单词所在的位置，*p*是单词在句子中的位置。

#### 1) 文档分割

当整个文档放入ELMo时，计算嵌入需要很长时间。当文档被分割成几个部分作为一个批处理。它们可以独立计算，也可以并行计算。

设**MSL**为最小序列长度。将文档分割成不短于MSL的实例。这意味着每个实例由几个完整的句子组成，并且每个实例的长度刚好大于或等于MSL。**文档分割(DS)**的细节如**算法1所示**。

**Algorithm 1** Document Segmentation

**Input:** Document $d$, sentence $s \in d$, MSL
**Output:** The batch after document segmentation
1. batch ← empty list
2. instance ← empty list
3. **for all** $s$ **in** $d$ **do**
4. **if** len(instance) > MSL **then**
5. **add** instance **to** batch
6. instance ← empty list
7. **else**
8. **add** $s$ **to** instance
9. **end if**
10. **end for**
11. **add** instance **to** batch

#### 2) 嵌入的对齐

但是随着文档的分割，模型失去了文档的完整上下文。模型的性能会受到影响。因此，我们使用一种称为**嵌入对齐(embeddings** alignment, EA)的方法来维持模型性能。

根据Schuster等人[26]的研究，对于一个非同音同义词，嵌入锚点大致位于所有上下文嵌入的点云中心。将单词$w_i$的嵌入锚定义为$\bar{v}_{w_i}$。因此，

嵌入锚定义如(7)所示，

$$\bar{v}_{w_i} = \frac{1}{n}\sum_{s_j,p} v_{w_i}^{s_j,p} \tag{7}$$

也就是说，一个词的嵌入锚是不同句子和位置的所有上下文嵌入的平均值。在从被分割的文档中计算出所有的上下文嵌入之后，使用嵌入锚来替换所有的嵌入，以对齐文档中的嵌入。

### D. 长文档的位置偏置权重

对于大多数长文档，作者倾向于写文档的主题，这意味着最重要的关键短语往往出现在文档的开头。

由于SIFRank是一种词袋模型，因此在对长文档(特别是多段落的文档)进行关键词提取时，有必要将位置信息考虑到候选关键词的重要性中。

在Florescu和Caragea[13]的研究中，位置偏置权重是一个词在文档中的逆位置之和。出现在$2^{th}$、$5^{th}$和$10^{th}$的单词，其权重$p(w_i)$= 1/2 + 1/5 + 1/10 = 0.8。

由于词频信息是通过SIF中的嵌入叠加来计算的。为了防止重复计数，我们只考虑候选关键词第一次出现的位置。位置偏置的权重是短语第一次出现偏移量的倒数:

$$p(NP_i) = \frac{1}{p_1 + \mu} \tag{8}$$

其中$p_1$是相对位置$NP_i$第一次出现的次数(所有候选关键字的顺序)，$p_1 \in N^*$。μ是一个超参数，用于优化开头候选关键字的位置偏置权重，尤其是第一个短语$\mu \in R^*$。

为了进一步缩小相邻候选关键字位置偏置权值的差距，使用softmax函数对其进行归一化:

$$\tilde{p}(NP_i) = \text{soft}\max(p(NP_i)) = \frac{\exp(p(NP_i))}{\sum_{k=1}^{N}\exp(p(NP_k))} \tag{9}$$

表1。三个数据集的分析(词干提取后计算关键短语的缺失量)。Missing in doc是指标注的关键短语没有出现在文档中。在候选中缺失意味着标注的关键字不在候选关键字中。

| Dataset | Documents | | | Keyphrases | | | |
|---|---|---|---|---|---|---|---|
| | Type | Number | Tokens average | Total | Average | Missing in doc | Missing in candidates |
| **Inspec** | Abstracts | 500 | 134.4 | 4912 | 9.8 | 24.90% | 36.69% |
| **SemEval2017** | Paragraph | 493 | 194.7 | 8529 | 17.3 | 0.15% | 42.96% |
| **DUC2001** | News | 308 | 828.4 | 2481 | 8.1 | 1.85% | 9.88% |

综上所述，对于较长的文档，SIFRank将会变成(10)所示的形式，我们称之为**SIFRank+**。

$$\text{SIFRank+}(NP_i, d) = \tilde{p}(NP_i) \cdot \text{Sim}(v_{NP_i}, v_d) \qquad (10)$$

## V. 评价

在本节中，我们在三个公共关键字提取数据集上对SIFRank和SIFRank+进行了综合评估。

### A. 数据集

本文使用Inspec、DUC2001和SemEval2017三个公共数据集来评估我们的模型。三个数据集的统计数据如表1所示。Inspec数据集的平均文档长度最短，DUC2001是最长的。值得注意的是，并不是所有的黄金关键字都出现在原文中，也不是所有的关键字都可以被识别为候选关键字。因此，从理论上讲，实现100%的关键字提取是不可能的。

**Inspec**数据集[27]由2000个从科学期刊摘要中选择的短文件组成。其中1000个文档用于训练，500个用于验证，500个用于测试。我们在本文中选择测试部分来验证我们的模型。

**SemEval2017**数据集[28]是SemEval2017竞赛中的任务10。它包含493个段落，选自ScienceDirect期刊，涵盖计算机科学、材料科学和物理学。每篇文档都由一名本科生和一名专家用关键短语注释。

**DUC2001**数据集[10]由来自TREC-9的308篇报纸文章组成。这些文章来自几家报纸，分为30个主题。

可以发现，在表2中，DUC2001中每个主题的文档都有相似的关键词。约有18.11%(1795年为325个)的关键词出现不止一次，这些关键词的频次之和占所有关键词总频次的40.75%(2481年为1011个)。因此，DUC2001中的关键短语在整个数据集的语料库中出现的频率更高，这意味着从该数据集中统计的频率信息可能不起作用。

表2。关键词重复分析。总数是指非冗余单词的数量。M-T-O是指关键词出现超过一次的次数。比例表示M-T-O关键词个数的比例。频率比例是指M-T-O关键词出现频率的比例。

| Dataset | Nonredundant Keyphrases | | | |
|---|---|---|---|---|
| | Total | M-T-O | Proportion | Frequency proportion |
| **Inspec** | 4550 | 235 | 5.16% | 12.15% |
| **SemEval2017** | 7609 | 518 | 6.81% | 16.86% |
| **DUC2001** | 1795 | 325 | **18.11%** | **40.75%** |

### B. 与其他基线进行比较

我们将我们的模型与3种无监督关键字提取方法进行了比较:统计模型、基于图的模型和基于嵌入的模型。统计模型为TFIDF和YAKE[1][29]。基于图的模型[2]有TextRank[3]、SingleRank[10]、TopicRank[11]、Position-Rank[13]和Multipartite[12]。基于嵌入的模型有RVA[24]和embed[25]。

对于TFIDF, n-gram窗口长度设置为3。对于YAKE，窗口大小为1，重复数据删除阈值为0.9,n-gram长度为3。TextRank 和 SingleRank 的窗口大小分别为 2 和 10。TopicRank的最小聚类相似度阈值设置为0.74。PositionRank的窗口大小为10。Multipartite中控制权重调整强度的超参数设为1.1。RVA的词嵌入分别由手套在每个单独的文档上生成，参考向量在全文(不只是标题和摘要)上计算，维数为100。embeddrank3使用了DBOW在维基百科语料库上训练的Doc2Vec模型

在本文提出的SIFRank和SIFRank+中，我们使用了allenlp .5预训练的原始ELMo模型当文档长度小于128时，我们使用ELMo层L0，当文档长度大于128时，我们使用ELMo层L1。

在相同的环境下，所有的模型都使用相同的工具进行标记化、词性标注和名词短语分块。我们使用斯坦福CoreNLP[6]进行标记化和词性标记。

1https://github.com/LIAAD/yake
2https://github.com/boudinfl/pke
3https://github.com/swisscom/ai-research-keyphrase-extraction
4https://github.com/jhlau/doc2vec
5https://allennlp.org/elmo
6https://stanfordnlp.github.io/CoreNLP/

表3。我们的SIFRank和SIFRank+模型与其他基线的比较。N是模型从单个文档中提取的数量。

| N | Method | Inspec | | | SemEval2017 | | | DUC2001 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | P | R | F1 | P | R | F1 | P | R | F1 |
| | Statistical model | | | | | | | | | |
| | TFIDF | 16.72 | 8.51 | 11.28 | **28.32** | **8.18** | **12.70** | 12.05 | 7.46 | 9.21 |
| | YAKE | **23.32** | **11.87** | **15.73** | 26.40 | 7.63 | 11.84 | **13.88** | **8.59** | **10.61** |
| | Graph-based models | | | | | | | | | |
| | TextRank | 36.16 | 18.40 | 24.39 | 36.63 | 10.59 | 16.43 | 18.24 | 11.29 | 13.94 |
| | SingleRank | 36.60 | 18.63 | 24.69 | **40.65** | **11.75** | **18.23** | 28.21 | 17.45 | 21.56 |
| 5 | TopicRank | 33.76 | 17.16 | 22.76 | 38.13 | 11.02 | 17.10 | 26.64 | 16.49 | 20.37 |
| | PositionRank | **37.36** | **18.99** | **25.19** | **40.65** | **11.75** | **18.23** | **32.64** | **20.19** | **24.95** |
| | Multipartite | 34.19 | 17.39 | 23.05 | 38.78 | 11.21 | 17.39 | 28.60 | 17.69 | 21.86 |
| | Embedding-based models | | | | | | | | | |
| | RVA | 32.48 | 16.53 | 21.91 | 43.69 | 12.63 | 19.59 | 26.58 | 16.44 | 20.32 |
| | EmbedRank d2v(Prev. SOTA) | 40.32 | 20.52 | 27.20 | 45.07 | 13.03 | 20.21 | 28.44 | 17.59 | 21.74 |
| | SIFRank(Ours) | **43.16** | **21.97** | **29.11** | **50.39** | **14.56** | **22.59** | 31.75 | 19.63 | 24.27 |
| | SIFRank+(Ours) | 42.24 | 21.50 | 28.49 | 48.03 | 13.88 | 21.53 | **40.39** | **24.99** | **30.88** |
| | Statistical models | | | | | | | | | |
| | TFIDF | 13.76 | 14.01 | 13.88 | 22.19 | 12.83 | 16.26 | 9.61 | 11.89 | 10.63 |
| | YAKE | **18.90** | **19.24** | **19.07** | **24.77** | **14.32** | **18.14** | **12.83** | **15.88** | **14.20** |
| | Graph-based models | | | | | | | | | |
| | TextRank | **33.44** | **33.71** | **33.58** | 35.25 | 20.38 | 25.83 | 17.63 | 21.77 | 19.48 |
| | SingleRank | 33.06 | 33.33 | 33.19 | **37.85** | **21.88** | **27.73** | 23.44 | 28.94 | 25.90 |
| 10 | TopicRank | 27.34 | 27.26 | 27.30 | 30.87 | 17.84 | 22.62 | 20.27 | 24.99 | 22.39 |
| | PositionRank | 31.52 | 31.54 | 31.53 | 35.91 | 20.75 | 26.30 | **26.28** | **32.45** | **29.04** |
| | Multipartite | 28.11 | 28.28 | 28.19 | 32.39 | 18.72 | 23.73 | 21.93 | 27.05 | 24.22 |
| | Embedding-based models | | | | | | | | | |
| | RVA | 30.05 | 30.21 | 30.13 | 37.63 | 21.75 | 27.57 | 22.82 | 28.17 | 25.22 |
| | EmbedRank d2v(Prev. SOTA) | 34.89 | 35.08 | 34.98 | 40.39 | 23.34 | 29.59 | 23.43 | 28.92 | 25.89 |
| | SIFRank (Ours) | **38.69** | **38.90** | **38.80** | **44.85** | **25.92** | **32.85** | 24.82 | 30.64 | 27.43 |
| | SIFRank+(Ours) | 36.67 | 36.87 | 36.77 | 42.96 | 24.83 | 31.47 | **30.20** | **37.28** | **33.37** |
| | Statistical models | | | | | | | | | |
| | TFIDF | 11.44 | 17.47 | 13.83 | 18.01 | 15.62 | 16.73 | 8.51 | 15.80 | 11.06 |
| | YAKE | **16.69** | **25.49** | **20.17** | **22.12** | **19.18** | **20.55** | **12.16** | **22.57** | **15.80** |
| | Graph-based models | | | | | | | | | |
| | TextRank | **30.09** | **44.14** | **35.78** | 32.86 | 28.46 | 30.50 | 16.29 | 30.03 | 21.12 |
| | SingleRank | 29.79 | 43.69 | 35.42 | **34.19** | **29.60** | **31.73** | 20.49 | 37.77 | 26.57 |
| 15 | TopicRank | 24.28 | 34.26 | 28.42 | 26.85 | 23.17 | 24.87 | 17.05 | 31.40 | 22.10 |
| | PositionRank | 28.17 | 40.72 | 33.30 | 32.95 | 28.48 | 30.55 | **22.61** | **41.68** | **29.32** |
| | Multipartite | 25.36 | 36.91 | 30.07 | 28.96 | 25.07 | 26.87 | 19.02 | 35.03 | 24.65 |
| | Embedding-based models | | | | | | | | | |
| | RVA | 28.02 | 40.80 | 33.22 | 33.67 | 29.15 | 31.25 | 19.30 | 35.59 | 25.04 |
| | EmbedRank d2v(Prev. SOTA) | 30.74 | 44.76 | 36.45 | 36.57 | 31.66 | 33.94 | 20.16 | 37.15 | 26.14 |
| | SIFRank(Ours) | **33.38** | **48.62** | **39.59** | **41.05** | **35.54** | **38.10** | 21.49 | 39.60 | 27.86 |
| | SIFRank+(Ours) | 32.74 | 47.68 | 38.82 | 39.10 | 33.85 | 36.29 | **24.86** | **45.83** | **32.24** |

正则表达式{<NN。*|jj>*<NN。*>}用于提取名词短语作为候选关键短语。

由于在不同的文档中标注的关键短语的数量不同，因此提取的关键短语的数量N被设置为5、10和15。本文采用宏精密度(P)、召回率(R)和F1值(F1)对各个模型进行评价。当将提取的结果与标注的结果进行比较时，所有的单词都进行了小写和词干处理。

如表3的结果所示，基于嵌入的模型相对于中的基于图的模型具有明显的优势

表4。ELMo不同层的精度。提取N个关键短语的个数设置为5。

| N | Method | Inspec | | | SemEval2017 | | | DUC2001 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | P | R | F1 | P | R | F1 | P | R | F1 |
| 5 | SIFRank | 41.12 | 20.93 | 27.74 | **50.39** | **14.56** | **22.59** | **31.75** | **19.64** | **24.27** |
| | ELMo-lstm-L0 | **43.16** | **21.97** | **29.11** | 46.45 | 13.42 | 20.83 | 20.33 | 12.58 | 15.53 |
| | ELMo-lstm-L1 | 42.24 | 21.50 | 28.49 | 44.26 | 12.79 | 19.85 | 16.94 | 10.48 | 12.95 |
| | ELMo-lstm-L2 | 42.76 | 21.76 | 28.85 | 47.91 | 13.85 | 21.48 | 27.43 | 16.97 | 20.97 |
| | ELMo-lstm-L0L1L2 | 42.84 | 21.80 | 28.90 | 45.72 | 13.21 | 20.50 | 11.60 | 7.17 | 8.86 |
| | ELMo-Transformer | | | | | | | | | |

表5所示。SIFRank中手套和BERT的精度。提取N个关键短语的个数设置为5。

| N | Method | Inspec | | | SemEval2017 | | | DUC2001 | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | P | R | F1 | P | R | F1 | P | R | F1 |
| 5 | SIFRank | 31.84 | 16.21 | 21.48 | 33.87 | 9.79 | 15.19 | 19.67 | 12.17 | 15.04 |
| | GloVe-100d | 35.20 | 17.92 | 23.75 | 38.30 | 11.07 | 17.17 | **23.71** | **14.67** | **18.12** |
| | GloVe-300d | 36.12 | 18.38 | 24.36 | 38.17 | 11.03 | 17.12 | 18.31 | 11.33 | 13.99 |
| | BERT-768d | 38.12 | 19.40 | 25.72 | 37.97 | 10.97 | 17.03 | 14.40 | 8.91 | 11.01 |
| | BERT-1024d | 37.28 | 18.97 | 25.15 | 37.12 | 10.73 | 16.65 | 11.21 | 6.93 | 8.57 |
| | RoBERTa-768d | 36.28 | 18.46 | 24.47 | 37.97 | 10.97 | 17.04 | 12.25 | 7.58 | 9.36 |
| | RoBERTa-1024d | 41.68 | 21.21 | 28.11 | 45.84 | 13.25 | 20.56 | 18.11 | 11.20 | 13.84 |
| | XLNet-768d | **41.92** | **21.34** | **28.28** | **47.34** | **13.68** | **21.22** | 20.19 | 12.49 | 15.43 |
| | XLNet-1024d | | | | | | | | | |

短文档数据集Inspec和SemEval2017。但是对于长文档数据集DUC2001, PositionRank比简单的基于嵌入的模型工作得更好。我们的模型SIFRank在短文档Inspec和SemEval2017上具有最先进的结果。我们的具有位置偏置权重的模型SIFRank+在长文档数据集DUC2001上获得了最先进的结果。

### C. elmo不同层的性能

如表4所示，我们比较了不同层ELMo的性能。ELMo-lstm-Lx表示仅使用Lx层。*ELMo-lstm-L0L1L2*表示使用L0, L1, L2两层的平均值。*ELMo- transformer*表示将ELMo从2层biLSTM改为6层bittransformer，并得到6层的平均值[30]。

结果表明，上下文层L1在短数据集Inspec上表现最好。静态层L0在较长的数据集SemEval2017和DUC2001上表现最好。这意味着上下文嵌入(L1和L2)可以更好地从短文本中提取单词的特征。但是当在长文本上运行时，使用上下文嵌入(L1或L2)的模型的性能会大大降低。

它还可以表明，在无监督任务中，Transformer结构在提取单词表示时比LSTM更弱。Transformer可能更适合于监督任务微调类型的任务。

### D. 与其他预训练的语言模型比较

我们比较了用GloVe和BERT的词嵌入替换ELMo的效果，如表5所示。

GloVe词嵌入在Wikipedia 2014和Gigaword 5语料库上进行训练，它们分别是50、100、200和300维。[7]

BERT词嵌入使用BERT -as-service[8]，并分别从$BERT_{BASE}$(12层Transformer, 768维)和$BERT_{LARGE}$(24层Transformer, 1024维)的倒数第二层(-2层)中提取词嵌入

RoBERTa的使用方式与BERT相同，也有两个不同尺寸的版本，$RoBERTa_{BASE}$(12层Transformer, 768维)和$RoBERTa_{LARGE}$(24层Transformer, 1024维)

我们使用XLNet的第一层，它表现出最好的性能。XLNet也有两个版本XLNet- base和XLNet- large，不同的是它们都使用Transformer-XL[31]而不是Transformer.11

实验结果表明，对于大多数预训练模型的词嵌入，维数越高，关键短语提取的效果越好。

从具有Transformer结构的AE语言模型BERT和RoBERTa中提取的词嵌入在较短的Inspec数据集上效果良好，而在较长的文档数据集DUC2001上效果较差。我们也尝试使用数据集语料库对BERT和RoBERTa进行增量预训练，但它并没有给模型带来稳定的改进。

7https://nlp.stanford.edu/projects/glove/
8https://github.com/hanxiao/bert-as-service
9https://github.com/google-research/bert
10https://github.com/pytorch/fairseq/tree/master/examples/roberta
11https://github.com/zihangdai/xlnet

表6所示。鲁棒性与嵌入的比较。提取N个关键短语的个数设置为5。

| N | Method | | Inspec | SemEval2017 | DUC2001 |
|---|---|---|---|---|---|
| 5 | SIFRank -ELMo | SIF+Remove ooc | 43.16 | 50.39 | 31.86 |
| | | SIF | 41.64 | 49.78 | 31.86 |
| | | Δ | **-1.52** | **-0.61** | **0.0** |
| | EmbedRank | d2v+Remove ooc | 40.32 | 45.07 | 28.44 |
| | | d2v | 38.64 | 42.68 | 25.99 |
| | | Δ | **-1.68** | **-2.39** | **-2.54** |

AR语言模型XLNet在GLUE和SQuAD等监督数据集上的性能与BERT和RoBERTa相差不大(RoBERTa甚至比XLNet表现得更好)，但XLNet在关键字提取任务上的性能大大提高。我们认为这是因为AE模型在预训练时使用了[MASK]这样的人工符号，会造成预训练-微调的差异。同时，我们论文中提出的关键词提取算法是一种无监督模型，它对这种差异非常敏感。

即使XLNet使用Transformer-XL，它可以比rnn和transformer更好地学习长期依赖关系，但与ELMo甚至GloVe相比，它在长文档数据集DUC2001上的表现仍然很差。因此，我们推断，隐藏层越深，我们任务中提取单词表示的能力就越差(可能单词表示越复杂，但这并不适合我们的任务)。

### E. 消融实验

嵌入在计算句子嵌入时去除不相关的词，只保留名词和形容词，提高了模型的效果。但是当它去除了这部分的时候，性能会大大下降。由于三个评价指标(P、R、F)在提取的关键短语数量(N)确定时是正相关的，为了节省纸张空间，我们选择了其中一个(P)来呈现，下表相同。

在表6中，我们比较了SIFRank和嵌入在删除或不删除不相关单词时精度的变化。*Remove ooc*意味着删除不考虑的单词。

我们的SIF模型即使不去除不考虑词也表现良好，可以更好地避免不相关词的影响，表现出很好的鲁棒性。

### F. 参数敏感实验

我们在这三个数据集上验证了超参数λ对领域自适应方法关键词提取精度的影响。提取的关键词个数N为5个。

结果如FIGURE和表7所示。检验、SemEval2017和DUC2001的λ最佳值分别为0.5、0.6和1.0。观察到适当的λ可以提高上述检验算法的性能

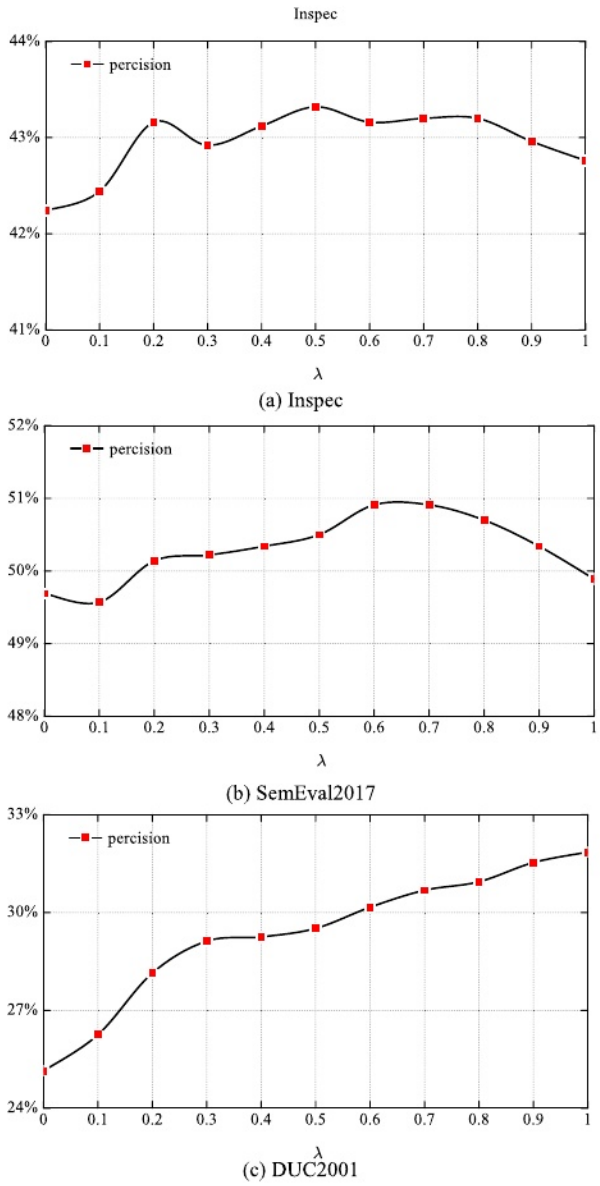



FIGURE2。精度与参数超参数λ的关系。

表7所示。超参数λ对关键词提取精度的影响。将提取的关键词个数N设置为5。

| N | Method | | Inspec | SemEval2017 | DUC2001 |
|---|---|---|---|---|---|
| 5 | SIFRank | λ=1.0 | 42.76 | 49.90 | 31.86 |
| | | λ=0.5 | 43.32 | 50.50 | 29.51 |
| | | Δ | **0.56** | **0.60** | **-2.35** |

SemEval2017在某种程度上。然而，由于DUC2001数据集的特殊性，对该数据集的领域词频统计并不能帮助提高性能。

### G. 文档分割和嵌入对齐的性能

如FIGURE所示，在没有文档分割的情况下，ELMo计算词嵌入所需的时间随着文档长度的增长而迅速增加。与

表8所示。文档分割(DS)和嵌入对齐(EA)对SIFRank图像的影响。

| N | Method | Inspec | SemEval2017 | DUC2001 |
|---|---|---|---|---|
| | SIFRank | 43.16 | 46.45 | 20.33 |
| 5 | SIFRank+DS | 43.08 ↓ | 45.88 ↓ | 21.23 ↑ |
| | SIFRank+DS+EA | **43.48** ↑ | **46.97** ↑ | **25.41** ↑ |

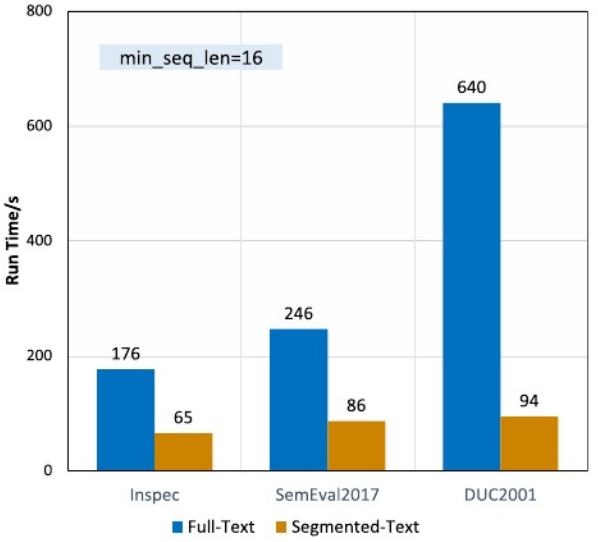



FIGURE3。完整文档和文档分割下SIFRank的操作次数。

文档分割，将MSL设置为16可以大大加快模型的速度。

但随后，通过上下文词嵌入(ELMo L1和L2层)降低了提取关键短语的精度。因此，通过嵌入对齐的方法，可以使模型的性能得到恢复。如表8所示，分别是全文档下的SIFRank计算结果、文档分割后的结果(DS)和文档分割和嵌入对齐后的结果(EA)。

### H. 案例研究

为了观察SIFRank和embed的区别，我们在Inspec中随机选择一个文档。两种模型计算出的关键词的相关分数以热图的形式呈现。

如图4所示，文本中带有粗体斜体和下划线的短语是带注释的关键短语。不同的颜色代表了两个模型对候选关键短语的相关性评分。右边的热图条显示了候选关键词的相关性分布。

可以看出，本文SIFRank算法计算的候选关键字相关性具有很好的区分度，可以很好地区分不相关的候选关键字。相反，embed计算出的不同候选关键词的相关分数之间的差异并不明显。

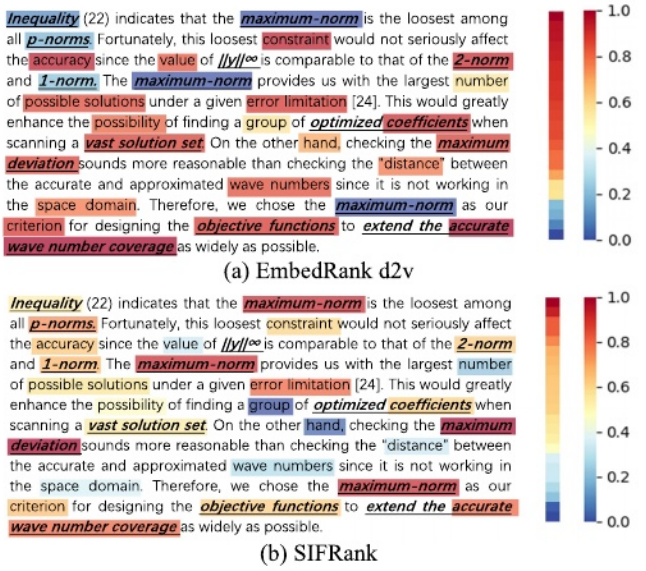



FIGURE4。embed和SIFRank对候选关键字的相关性评分热图。

## VI. 结论

在本文中，我们提出了一种新的基于预训练语言模型SIFRank的无监督关键短语提取基线。我们将句子嵌入模型SIF和自回归预训练语言模型ELMo引入到SIFRank中，在短文档数据集上获得了最先进的结果。我们还加快了SIFRank，同时通过文档分割和嵌入对齐保持其准确性。将SIFRank算法升级为具有位置偏置权重的SIFRank+算法，其在长文档数据集上的性能得到了很大的提高。

未来，仍有一些问题需要进一步研究。**首先**，尽管不同的预训练模型或层具有不同的特征，但很难构建具有多个不同子模型的高效集成无监督关键词提取模型。在我们的实验中，任何集成模型的性能都不如它最好的子模型。**其次**，位置偏置权值的使用极大地改善了长文档数据集上的模型。还有没有其他的信息可以用来改进模型，如何进行整合?权值的归一化方法是否必须与句子嵌入模型具有相同的分布?

## 参考文献。

[1] K. S. Hasan and V. Ng, ‘‘Automatic keyphrase extraction: A survey of the state of the art,’’ in *Proc. 52nd ACL*, 2014, pp. 1262–1273.

[2] R. F. I. Cancho and R. V. Sole, ‘‘The small world of human language,’’ *Proc. Roy. Soc. B, Biol. Sci.*, vol. 268, no. 1482, pp. 2261–2265, 2001.

[3] R. Mihalcea and P. Tarau, ‘‘TextRank: Bringing order into text,’’ in *Proc. EMNLP*, 2004, pp. 404–411.

[4] E. Papagiannopoulou and G. Tsoumakas, ‘‘A review of keyphrase extraction,’’ *Wiley Interdiscipl. Rev., Data Mining Knowl. Discovery*, to be published, doi: 10.1002/widm.1339.

[5] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, ‘‘Deep contextualized word representations,’’ in *Proc. NAACL-HLT*, 2018, pp. 2227–2237.

[6] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, ''BERT: Pre-training of deep bidirectional transformers for language understanding,'' in *Proc. NAACL-HLT*, 2019, pp. 4171–4186.

[7] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, ''XLNet: Generalized autoregressive pretraining for language understanding,'' 2019, *arXiv:1906.08237*. [Online]. Available: https://arxiv.org/abs/1906.08237

[8] S. Arora, Y. Liang, and T. Ma, ''A simple but tough-to-beat baseline for sentence embeddings,'' in *Proc. ICLR*, 2017.

[9] L. Page, S. Brin, R. Motwani, and T. Winograd, ''The PageRank cita-tion ranking: Bringing order to the Web,'' Stanford InfoLab, Palo alto, CA, USA, Tech. Rep. SIDL-WP-1999-0120, 1999. [Online]. Available: http://ilpubs.stanford.edu:8090/422/

[10] X. Wan and J. Xiao, ''Single document keyphrase extraction using neighborhood knowledge,'' in *Proc. AAAI*, 2008, pp. 855–860.

[11] A. Bougouin, F. Boudin, and B. Daille, ''TopicRank: Graph-based topic ranking for keyphrase extraction,'' in *Proc. IJCNLP*, 2013, pp. 543–551.

[12] F. Boudin, ''Unsupervised keyphrase extraction with multipartite graphs,'' in *Proc. NAACL-HLT*, vol. 2, 2018, pp. 667–672.

[13] C. Florescu and C. Caragea, ''A position-biased PageRank algorithm for keyphrase extraction,'' in *Proc. AAAI*, 2017, pp. 4923–4924.

[14] M. Peters, S. Ruder, and N. A. Smith, ''To tune or not to tune? Adapt-ing pretrained representations to diverse tasks,'' 2019, *arXiv:1903.05987*. [Online]. Available: https://arxiv.org/abs/1903.05987

[15] T. Mikolov, K. Chen, G. Corrado, and J. Dean, ''Efficient estimation of word representations in vector space,'' 2013, *arXiv:1301.3781*. [Online]. Available: https://arxiv.org/abs/1301.3781

[16] J. Pennington, R. Socher, and C. Manning, ''Glove: Global vectors for word representation,'' in *Proc. EMNLP*, 2014, pp. 1532–1543.

[17] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, ''Bag of tricks for efficient text classification,'' in *Proc. EACL*, vol. 2, 2017, pp. 427–431.

[18] Q. Le and T. Mikolov, ''Distributed representations of sentences and documents,'' in *Proc. ICML*, 2014, pp. 1188–1196.

[19] R. Kiros, Y. Zhu, R. Salakhutdinov, R. S. Zemel, A. Torralba, R. Urtasun, and S. Fidler, ''Skip-thought vectors,'' in *Proc. NIPS*, vol. 2, 2015, pp. 3294–3302.

[20] M. Pagliardini, P. Gupta, and M. Jaggi, ''Unsupervised learning of sentence embeddings using compositional N-gram features,'' in *Proc. NAACL-HLT*, 2018, pp. 528–540.

[21] B. McCann, J. Bradbury, C. Xiong, and R. Socher, ''Learned in translation: Contextualized word vectors,'' in *Proc. NIPS*, 2017, pp. 6297–6308.

[22] Y. Liu, M. Ott, and N. Goyal, ''RoBERTa: A robustly optimized bert pretraining approach,'' 2019, *arXiv:1907.11692*. [Online]. Available: https://arxiv.org/abs/1907.11692

[23] Y. Wang, Y. Jin, X. Zhu, and C. Goutte, ''Extracting discriminative keyphrases with learned semantic hierarchies,'' in *Proc. COLING*, 2016, pp. 932–942.

[24] E. Papagiannopoulou and G. Tsoumakas, ''Local word vectors guiding keyphrase extraction,'' *Inf. Process. Manage.*, vol. 54, no. 6, pp. 888–902, 2018.

[25] K. Bennani-Smires, C. Musat, A. Hossmann, M. Baeriswyl, and M. Jaggi, ''Simple unsupervised keyphrase extraction using sentence embeddings,'' in *Proc. CoNLL*, 2018, pp. 221–229.

[26] T. Schuster, O. Ram, R. Barzilay, and A. Globerson, ''Cross-lingual alignment of contextual word embeddings, with applications to zero-shot dependency parsing,'' in *Proc. NAACL-HLT*, vol. 1, 2019, pp. 1599–1613.

[27] A. Hulth, ''Improved automatic keyword extraction given more linguistic knowledge,'' in *Proc. EMNLP*, Jul. 2003, pp. 216–223.

[28] I. Augenstein, M. Das, S. Riedel, L. Vikraman, and A. McCallum, ''SemEval 2017 task 10: ScienceIE-extracting keyphrases and relations from scientific publications,'' in *Proc. SemEval*, 2017, pp. 546–555.

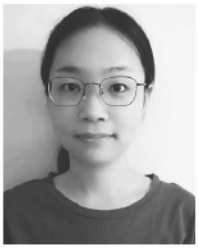
[29] R. Campos, V. Mangaravite, A. Pasquali, A. M. Jorge, C. Nunes, and A. Jatowt, ''YAKE! Collection-independent automatic keyword extractor,'' in *Proc. ECIR*, 2018, pp. 806–810.

[30] M. Peters, M. Neumann, L. Zettlemoyer, and W. T. Yih, ''Dissecting contextual word embeddings: Architecture and representation,'' in *Proc. EMNLP*, 2018, pp. 1499–1509.

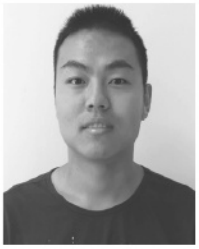
[31] Z. Dai, Z. Yang, and Y. Yang, ''Transformer-XL: Attentive language models beyond a fixed-length contex,'' 2019, *arXiv:1901.02860*. [Online]. Available: https://arxiv.org/abs/1901.02860

孙毅，2018年获PLA陆军工程大学信息工程专业硕士学位，现攻读博士学位。主要研究方向为自然语言处理和网络用户分析。他在2016年ACM国际大学生编程竞赛中获得银奖。

邱杭平，1990年获通信工程学院计算机应用专业硕士学位，2009年获PLA科技大学信息工程专业博士学位。现任PLA陆军工程大学教授。主要研究方向为数据分析、信息管理、系统集成。



于正，2016年获河北大学理学士，2019年获PLA陆军工程大学计算机应用硕士学位。现为赛普瑞(南京)实验室软件工程师。她的研究兴趣包括软件测试和机器学习。



王忠伟，2017年毕业于PLA陆军工程大学，获理学士学位，现攻读硕士学位。主要研究方向为信息推荐、网络用户分析。

张超然，2018年毕业于大连民族大学通信工程专业，获学士学位。目前在解放军陆军工程大学计算机应用专业攻读硕士学位。主要研究方向为自然语言处理、深度学习、光通信。