基于 Word2Vec 的编程领域词语拼写错误检测算法

刘峻松[1] 唐明靖[2] 薛 岗[1*] 杨成荣[3]

[1](云南大学软件学院 云南 昆明 650000)

[2](云南师范大学生命科学学院 云南 昆明 650000)

[3](六盘水师范学院 贵州 六盘水 553004)

摘 要 Stack Overflow 是一个计算机编程领域的问答社区,其中的文本蕴含大量有价值的信息可供挖掘,但由于其本身存在大量的错误词汇,给文本的分析造成影响。对此,提出一种词语自动检测纠错算法,通过词向量的技术以语义相似度为核心,对错误词汇进行分析,结合改进的编辑距离算法对文本进行自动检测纠错。实验结果表明,该算法能够对诸如此类专业性较强的领域主题文本进行自动检测纠错,并且能够较好地还原标准文段用词。

关键词 词向量 编辑距离 拼写纠错 Word2Vec Stack Overflow

中图分类号 TP391.1 **文献标志码** A DOI:10.3969/j.issn.1000-386x.2022.03.045

SPELLING CHECK ALGORITHM OF WORDS IN THE FIELD OF PROGRAMMING BASED ON WORD2VEC

Liu Junsong[1] Tang Mingjing[2] Xue Gang[1*] Yang Chengrong[3]

[1](*School of Software*, *Yunnan University*, *Kunming 650000*, *Yunnan*, *China*)

[2](*School of Life Sciences*, *Yunnan Normal University*, *Kunming 650000*, *Yunnan*, *China*)

[3](*Liupanshui Normal University*, *Liupanshui 553004*, *Guizhou*, *China*)

Abstract Stack Overflow is a question-and-answer community in the field of computer programming, where text contains a lot of valuable information to mine, but there are a lot of wrong words in it, which affects the analysis of text. This paper proposes an automatic error check and correction algorithm of words. It analyzed the wrong words by using the technique of word vector and focusing on semantic similarity, and combined the modified editing distance algorithm to automatically check and correct the text. Experiments show that the proposed algorithm can automatically check and correct the subject text of such professional areas, and can restore the standard text semantics well.

Keywords Word vector Edit distance Spelling correction Word2Vec Stack Overflow

0 引 言

Stack Overflow 是一个热门的计算机编程领域的问答社区,它为世界范围内的计算机编程爱好者提供了一个解决问题的平台。因此论坛中的问答文本具有很高的价值,每年都有很多人以 Stack Overflow 中的问题答案文本为研究对象,在海量的文本数据中挖掘不同的信息,为不同领域的研究提供数据基础。

由于 Stack Overflow 是一个开放式的问答社区平台,其中所有的文本数据均为来自世界各地用户的输入,因此其文本数据中存在大量的拼写错误。在对文本进行分析时,拼写错误对基于统计学理论的很多分析方法来说是相对致命的。以分析热门问题和热搜问题为例,在通过关键词进行分析和检索的过程中,如果某段文本的语义中心词存在拼写错误,根据计算机的

收稿日期:2019-12-23。**刘峻松**,硕士生,主研领域:文本分析,微服务架构应用。**唐明靖**,副教授。**薛岗**,副教授。**杨成荣**,硕士生。

模式匹配原则,该文段将会被错误地认知或归类,当错误词汇出现的频率较高时,对于统计结果乃至最终的分析结果都会产生较大的影响。绝大多数人类输入的文字都会出现文本拼写错误,而诸如 Stack Overflow 这种开放平台下的自然语言文本来说,其中拼写错误文本的数量更是不可忽视。

本文提出了一种基于词向量的文本拼写错误自动检测算法,通过结合文段语义及部分计算机输入习惯所造成的常见错误情况,对 Stack Overflow 中计算机编程领域的文本数据进行自动的单词拼写检测和纠正。实验结果表明,与现有的以编辑距离为基础的候选词检测和纠错方式相比,使用本文算法对文本进行自动校正后,所获得的结果文本与标准文本对比,语义相似度更高,针对部分计算机编程领域的专业词汇及缩写等情况的检测和纠正效果更好,且在面对海量文本数据时能够做到快速自动检测和纠正,从而验证了基于 Word2Vec 的计算机编程领域词语拼写错误检测算法在针对计算机编程领域自然语言文本的单词拼写自动纠错问题中具有较好的效果。

1 研究背景

单词拼写错误的检测和纠正在自然语言处理领域是一个很早就已经出现的问题,Kukich[1] 使用 UNIX 实现了英文文本的拼写检查方法,同时提出了单词拼写错误应包括非词错误(Non-word error)和真词错误(Real-word error),这些理论为后续的单词拼写检测和纠错提供了基础。Levenshtein[2] 提出了编辑距离的概念,如今编辑距离被广泛应用于单词拼写检测和纠错中,Soleh 等[3] 提出了使用词法分析和查找字典的方式检测错误词汇,通过错误词汇编辑距离构建候选词集合,最后使用隐马尔可夫模型对词汇文本进行分析进而对候选词集合的所有词汇进行排序,选取序列中排列首位的词汇作为错误词汇的改正词汇进行替换。谢文慧等[4] 提出在编辑距离的计算中引入键盘物理布局这一因素,将键盘键位间的最短距离直接引入到编辑距离算法中,但该文使用绝对的物理距离作为参数,实际上用户的键盘输入误差仅存在于周围的键位当中,更远的键位距离值会对最终的判别产生负向的影响。且上述所有方法均是以字典和编辑距离为核心判断标准,因此对于部分专业领域较强的特殊词汇及字典中没有记录的网络新兴词汇的检测能力不强,甚至可能会出现误判的情况,而且对于网络开放社区的文本来说存在大量诸如用户名、邮箱地址等特定且无实际意义的词汇,该类词汇可能由某个具有实意的单词演变而来且二者编辑距离极有可能很小,对该类词汇的误判会对文段的语义产生较大影响。

Bergsma 等[5] 将 N-gram 模型引入到拼写纠错问题当中,基于统计语言模型,分别利用了有监督和无监督的方法,结合上下文语义对单词进行拼写纠错。Kim 等[6] 结合了单词的相似性和 N-gram 模型,使用 N-gram 模型计算的语义相似性对单词的拼写相似度进行修正,提高了拼写纠错的准确性,但是 N-gram 模型具有参数空间大且数据稀疏严重的弊端,因此在处理大量文本时效率较低。

目前从文本拼写纠错领域的研究情况看,大部分方法是基于文本拼写特征或基于统计的词汇替换方法进行词语拼写矫正,上述方法存在准确度低、速度较慢等问题,而本文算法以 Word2Vec 运算的词向量构建文本的向量空间,通过余弦相似度构建与检测词汇语义相似词汇的集合,结合余弦相似度、词频、基于键盘键位改进的文本编辑距离的复合评分标准来对错误词汇进行检测和纠正。相较于上述已有的方法,本文提出的方法复合了多种对词汇正误判断及候选集合选取有影响的因素。通过实验表明,本文方法能够在保证语义的前提下自动对大量文本进行检测和纠错,并且对部分专业性较强的生僻词汇、新词汇、缩写词汇有较好的检测和纠正效果。

2 相关技术

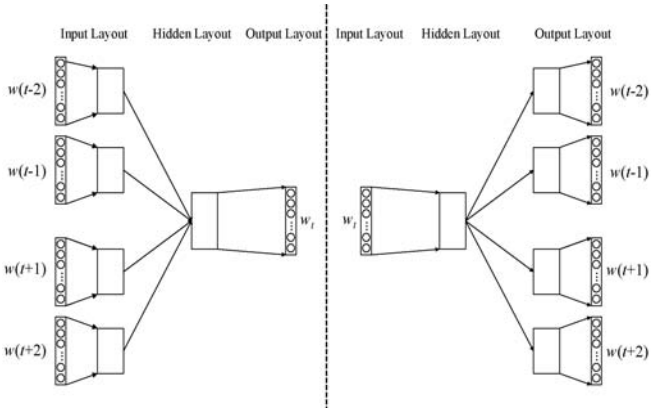
2.1 词向量技术

为了表达词与词之间的关系,Hinton[7] 提出了词语的分布式表达形式,每个词对应的分布式表达是一个低维度的实值向量,其中每一个维度均可以表示一个词的潜在特征。通过对大量文本语料的分析和训练,将已知文本中的每一个词汇映射为低维向量空间中的一个向量,这个向量空间称为词向量空间,其中的每一个向量称之为词向量。在这个空间中引入“距离”的概念,这个“距离”一般使用向量间的余弦值,多维向量的余弦值由欧几里得向量点积公式推导得出,以此值作为两个词语的余弦相似度[8]。假设空间内现有两个 n 维向量 $\boldsymbol{a}=(A_1,A_2,\cdots,A_n)$、$\boldsymbol{b}=(B_1,B_2,\cdots,B_n)$,向量夹角为 θ,余弦相似度计算式表示为:

$$\cos\theta=\frac{\boldsymbol{a}\cdot\boldsymbol{b}}{\boldsymbol{a}\times\boldsymbol{b}}=\frac{\sum_1^n(A_i\times B_i)}{\sqrt{\sum_1^n A_i^2}\times\sqrt{\sum_1^n B_i^2}} \tag{1}$$

由于词向量本身包含了词语潜在的上下文特征,因此通过对向量间余弦值的计算可以判断其对应词汇之间在语义或者上下文使用上的相似度。

Word2Vec 是在 2013 年由 Google 的 Mikolov 等[9-10]提出并实现的一种工具,用于快速地对文本进行训练并获得低维词向量,其核心是一个浅层的神经网络。Word2Vec 中包含了两种训练模型[10],分别为 CBOW 和 Skip-gram,两种模型如图 1 所示。



(a) CBOW 模型　　(b) Skip-gram 模型

图 1　Word2Vec 中的两种训练模型

可以看出,两种模型均是包含输入层、输出层及映射层的浅层神经网络模型,核心理论是贝叶斯条件概率,研究 w 和 $Context(w)$ 之间的条件概率关系,即 $P(w \mid Context(w))$ 或 $P(Context(w) \mid w)$,此处 $Context(w)$ 定义为词语 w 的上下文,数学表达如下:

$$Context(w_i) = w_{i-t}, \cdots, w_{i-1}, w_i, w_{i+1}, \cdots, w_{i+t} \quad (2)$$

式中:w_i 表示当前词汇;t 表示纳入上下文计算的词汇数量,即从当前词汇开始计算前后需要纳入计算的连续词汇的数量。CBOW 模型是通过输入上下文对其中词汇进行预测,而 Skip-gram 与之相反,通过词汇对上下文进行预测。Word2Vec 为了提高训练的效率,还提供了两种优化算法,分别是 Hierachy Softmax 和 Negative Sampling,通过使用 Word2Vec 训练可以输出一组质量相对较高的低维词向量,并且语义相近的词汇将被映射到空间距离相近的位置上。

2.2　编辑距离

编辑距离(Levenshtein Distance)是 Levenshtein[2]提出的方法,用于表示一个字符串转变为另一个字符串所需的最小操作步数。一步操作包括删除一个字符、增加一个字符和修改一个字符三种情况,假设现有字符串 A 和字符串 B,使用 A_i 表示 A 字符串前 i 个字符构成的子串,同理使用 B_j 表示 B 字符串前 j 个字符构成的子串,用 $LD(i,j)$ 表示字符串 A 和 B 之间的编辑距离,则根据编辑距离算法可得计算式:

$$LD(i,j) = \begin{cases} 0 & i=0, j=0 \\ i & i>0, j=0 \\ j & i=0, j>0 \\ \min\{LD(i-1,j)+1, LD(i,j-1)+1, \\ \quad LD(i-1,j-1)+F\} & i>0, j>0 \end{cases} \quad (3)$$

式中:min{}表示选取最小值函数;$LD(i-1,j)+1$ 表示在 B_j 字符串的末尾加入一个字符并增加一次编辑次数;$LD(i,j-1)+1$ 表示在 B_j 字符串末尾删除一个字符并增加一次编辑次数;$LD(i-1,j-1)+F$ 表示将 B_j 字符串的最后一个字符修改为 A_i 的最后一个字符;$F = \begin{cases} 0, A[i] = B[j] \\ 1, A[i] \neq B[j] \end{cases}$ 表示一次判断,若最后一位字符相同则无需增加编辑次数,反之增加一次编辑次数。一般而言,两个字符串的编辑距离越小,则拼写相似度越高。

3　基于词向量的单词拼写识别方法

本文以文本词向量为词义相似度的评判基础,通过改进的编辑距离模型对词义相似度的模型进行修正,综合考虑文本的语义和编辑距离的影响提出一种文本相似度计算方法,以此为基础提出了一种文本单词拼写检测纠错的算法。本节通过对编辑距离模型、综合文本相似度模型及单词拼写错误检测方法三个方面进行概述。

3.1　基于物理键盘输入方式的编辑距离模型

Levenshtein[2]提出的编辑距离可以一定程度的描述两个单词之间的拼写相似程度,但是 Stack Overflow 是一个开放的网络社区,其中绝大多数词汇都是通过计算机键盘进行输入的,因此有一部分词汇错误是键盘键位相近导致的误操作所造成的。本文将在原始编辑距离公式上进行改进,将因键盘键位相近导致误操作的情况纳入编辑距离计算中。

本文使用无向图的方式表示键盘键位,根据国际标准 QWERTY 键盘的物理键位位置,构建如图 2 所示的无向图。文献[4]使用无向图中的最短路径作为距离引入到编辑距离当中,但实际上针对国际标准键盘布局,有一种较为常用的输入指法,在该指法下,用户在输入的过程中,不同的输入错误情况出现的概率会根据指法中键位的分布而存在偏差,键盘指法的分布如图 2 所示。

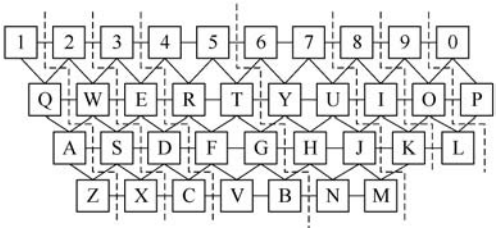


图 2 键盘布局和键盘指法分布图

文献[11]中针对键盘指法提出了三种输入错误的类型:(1) 错误字母与正确字母位于同一个手指负责的区域(此类错误情况定义为 W_1);(2) 错误字母与正确字母位于同一只手的相邻手指负责的区域(此类错误情况定义为 W_2);(3) 错误字母与正确字母位于不同手的相邻手指负责的区域(此类错误情况定义为 W_3)。

以单词“word2vec”为例,与字母“w”相邻部分的键位如图 3 所示,用户在执行键入“W”的操作时,若错误输入为“2”“S”则属于 W_1 情况,若错误输入为“3”“Q”“E”“A”则属于 W_2 情况。

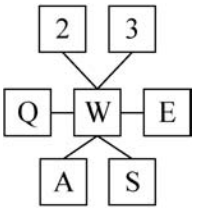


图 3 字母“W”相邻布局图

文献[11]通过大量的统计实验表明,上述三种错误情况出现的概率满足如下关系:

$$\begin{cases} P(W_1) = 2 \times P(W_2) \\ P(W_1) = 3 \times P(W_3) \end{cases} \tag{4}$$

式中:W_1、W_2、W_3分别代表上文提及的发生三种输入错误类型的事件;$P(W)$表示不同输入错误类型所代表的事件的发生概率。因此,将上述无向图改为加权无向图,将边赋予不同的权值。同样以“word2vec”为例,如果使用图的最短距离直接作为键盘键位对编辑距离的影响因子,则“mord2vec”和“tord2vec”的影响程度是不一样的,但是实际上,一旦超过“相邻”键位这个范畴,这种词语中字符的区别则更倾向于不同单词或其他错误情况,因此本文在上述基础上引入一个阈值,当其最短距离超过阈值时,则认为该字符差异不是由键盘物理键位的误操作引起的。

根据上述思路,首先根据三种错误情况出现的概率对键盘键位图中各边的权值进行设定,根据上述规则,设 $W_1 = 1$、$W_2 = 2$、$W_3 = 3$。尽管某些情况下,同一手指负责的区域出错的可能性较大。由于两个字母按键相隔距离较远时,其混淆输入的可能性将大幅度下降,因此在加权图的距离计算时将距离乘跳数作为其距离的最终值,同时引入阈值 $T = 4$,将误操作范围界定于图 3 所示的范围内。则任意两个键盘可输入字符串 A 和 B 之间的距离 D_k的计算公式如下:

$$D_k(A,B) = \begin{cases} +\infty & D_h(A,B) \geqslant T \\ D_h(A,B) & 0 < D_h(A,B) < T \\ 0 & D_h(A,B) = 0 \end{cases} \tag{5}$$

式中:$D_h(A,B) = D(A,B) \times h(A,B)$;$D(A,B)$表示加权图中 A、B 两个节点的最短距离;$h(A,B)$表示加权图中 A、B 两个节点最短路线上所需的跳数。在原始编辑距离公式中,字符的一处变化都会导致编辑距离加 1,可以将编辑距离等于 1 解释为在无任何因素影响下编辑差异步长为 1 的情况,因此本文将上述 D_k使用 Sigmoid 函数将值映射到[0,1]之间。Sigmoid 函数表达式如下:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{6}$$

则推导可得任意两个字符串 A 和 B,改进后编辑距离的影响因子 $I(A,B)$的计算式如下:

$$I(A,B) = \frac{1}{1 + e^{-(D_k(A,B))}} \tag{7}$$

综上,对原始编辑距离公式修正为:

$$LD_k(A[i],B[j]) = \begin{cases} 0 & i=0, j=0 \\ i & i>0, j=0 \\ j & i=0, j>0 \\ \min\begin{cases} LD_k(A[i-1],B[j]) + I(A[i],B[j]), \\ LD_k(A[i],B[j-1]) + I(A[i],B[j]), \\ LD_k(A[i-1],B[j-1]) + F_k \end{cases} & i>0, j>0 \end{cases} \tag{8}$$

式中:$F_k = \begin{cases} 0 & A[i] = B[j] \\ A[i] = B[j] & A[i] \neq B[j] \end{cases}$;$A[i]$表示 A 字符串的前 i 个字符构成的子串;$B[j]$表示 B 字符串的前 j 个字符构成的子串。

3.2 词汇综合相似度模型

基于词向量关注每个词汇上下文情况,而不关注词汇拼写本身的特性,且绝大部分拼写错误词汇,输入者所想表达的语义与其对应的正确词汇是一致的,因此错误词汇的上下文特征与正确词汇的上下文特征相似度较高,也就是在向量空间中二者词向量间的夹角余弦值较小,因此将词向量间的余弦相似度值与上述改进的编辑距离同时纳入到综合相似度评分的计算中。

对任意两个词 A 和 B 的综合相似度评分 $S(A,B)$进行计算,$S(A,B)$与 A、B 对应词向量的余弦相似度成

正比,与 LD_k 成反比,由此可得 $S(A,B)$ 计算公式为:

$$S(A,B)=\begin{cases}\dfrac{\cos(\boldsymbol{a}\cdot\boldsymbol{b})}{LD_k(A,B)} & LD_k(A,B)<\max(len(A),len(B))\\ 0 & LD_k(A,B)=\max(len(A),len(B))\end{cases} \tag{9}$$

式中:$\boldsymbol{a}$、$\boldsymbol{b}$ 表示词语 A、B 所对应的词向量;$\cos(\boldsymbol{a}\cdot\boldsymbol{b})$ 表示 A、B 词语对应词向量的余弦相似度;$LD_k(A,B)$ 表示改进的词语 A、B 的编辑距离;max( )表示选取最大值函数;len( )表示字符串长度。若两个词语的编辑距离等于最长词语的字符数,则意味着在本文模型中,这两个词汇没有任何相似之处,因此将其相似度综合评分直接定为 0。

3.3　基于词向量的自动拼写错误识别

本文提出的算法会对文本中每一个词语进行分析。对于每一个被检测词语,首先通过 Word2Vec 计算的模型获得与当前词语向量余弦语义相似度最高的十个词语组成候选词集合,分别对当前词语和候选词集合中的所有词语计算综合相似度评分,获取评分最高的词语,对比两个词语的词频。若当前被检测词语的词频低于候选集中评分最高的词语,则使用该词语替换当前词语,达到词语纠错的目的。因此要对文本语料进行处理和训练,获得词向量模型。首先对文本进行预处理,原始 Stack Overflow 的文本数据如下:

```
<p><a href=\"http://pyxml.sourceforge.net\"
rel=\"nofollow noreferrer\"
title=\"PyXML\">PyXML</a> works well.
</p>\n\n<p>You didn't say what platform you're using,
however if you're on Ubuntu you can get it with
<code>sudo apt-get install python-xml</code>. I'm sure
other Linux distros have it as well. </p>\n\n<p>If you're
on a Mac, xpath is already installed but not immediately
accessible. You can set
<code>PY_USE_XMLPLUS</code> in your
environment or do it the Python way before you import
xml.xpath:</p>\n\n<pre><code>if
sys.platform.startswith('darwin'):\n
os.environ['PY_USE_XMLPLUS'] =
'1'\n</code></pre>\n\n<p>In the worst case you may
have to build it yourself. This package is no longer
maintained but still builds fine and works with modern
2.x Pythons. Basic docs are <a
href=\"http://pyxml.sourceforge.net/topics/howto/section
-XPath.html\" rel=\"nofollow
noreferrer\">here</a>.</p>\n
```

Stack Overflow 的原始文本是按照 HTML 的格式组织的,其中包含大量的 HTML 标签和无意义的格式信息,因此对上述原始数据的处理步骤如下:

(1) 解析 HTML 结构文本获得自然语言文本。在解析 HTML 文本的过程中,包含两类标签,一类是诸如 <p>、
 仅对文本格式或样式设置的标签,此类标签应直接删除而保留标签内的文本;另一类是诸如 <a>、<code> 等包含链接、代码等与自然语言文段语义不相关的内容,此类标签应与标签内文本一同删除。

(2) 清理转义字符。对文本中部分转义字符进行识别和删除,该部分字符内嵌于 HTML 标签内大部分用于对格式进行调整或表达某类符号,无实际语义。

(3) 删除常规标点符号。由于 Stack Overflow 文本数据主题是计算机编程领域,因此文本中包含大量特殊字符,该类字符少量与语义相关,例如“C”“C#”,在计算机编程领域中这两个词汇代表着两种不同的技术,若删除特殊符号“#”对文本语义产生影响较大,因此本步仅对常规标点符号进行删除。

(4) 分词处理。由于 Stack Overflow 本身是一个国外论坛,因此其中所有的文本均为英语文本,英语文本直接使用空格作为分界进行分词即可。

(5) 对文本进行词类还原。同一个词汇会存在不同形式,但其语义相同,因此需要对词汇进行词类还原,包括动词时态还原以及名词复数还原,这里使用 NLTK[12] 中的词类还原工具 WordNetLemmatizer 对文本中的词性进行还原。

经过上述五步的数据处理,可以获得最终用以训练的文本数据。针对样例数据,上述五步处理过程的结果如表 1 所示。

表 1　样例数据处理过程

操作	结果输出
解析 HTML	pyxml works well. you didn"t say what platform you"re using, however if you"re on ubuntu you can get it with. i"m sure other linux distros have it as well. if you"re on a mac, xpath is already installed but not immediately accessible. you can set in your environment or do it the python way before you import xml.xpath: in the worst case you may have to build it yourself. this package is no longer maintained but still builds fine and works with modern 2.x pythons. basic docs are here.
删除标点符号和转义字符	pyxml works well you didn t say what platform you re using however if you re on ubuntu you can get it with i m sure other linux distros have it as well if you re on a mac xpath is already installed but not immediately accessible you can set in your environment or do it the python way before you import xml xpath in the worst case you may have to build it yourself this package is no longer maintained but still builds fine and works with modern 2 x pythons basic docs are here

续表 1

操作	结果输出
词性还原	pyxml work well you didn t say what platform you re use however if you re on ubuntu you can get it with i m sure other linux distros have it a well if you re on a mac xpath be already instal but not immediately accessible you can set in your environment or do it the python way before you import xml xpath in the worst case you may have to build it yourself this package be no longer maintain but still build fine and work with modern 2 x python basic doc be here

接下来使用 Word2Vec 工具对处理好的文本进行训练，Word2Vec 自身集成了两种词向量训练模型，以及两种训练优化算法，因此共有四种词向量训练框架，如表 2 所示。

表 2 Word2Vec 四种词向量训练框架

模型	Hierachy Softmax	Negative Sampling
CBOW	CBOW + HS	CBOW + NS
Skip-gram	Skip-gram + HS	Skip-gram + NS

本文中数据量较大，且存在较多的专业生僻词，因此选取 Skip-gram 算法进行计算，而 Hierachy Softmax 使用了哈夫曼编码，因此效率较高，所以本文使用 Skip-gram + HS 的框架训练词向量。

词向量训练完成后，即可对文本的每一个词汇进行扫描识别，但是在计算机编程领域存在较多专业性生僻词汇和缩写词汇的特殊词汇情况，一个特殊词汇可能是标准英语词汇字典中不存在的词汇，也有可能是某个较长词汇或者词组的缩写。一般情况下，在特定的专业领域内，该类情况所衍生出的一系列特殊词汇的拼写和形式都较为统一，因此使用大量的同一领域内的语料进行训练后，针对专业性生僻词汇和缩写应存在以下两种情况：

（1）专业性生僻词汇。此类词汇的特征是在标准英语词汇字典中不存在却真实的表现了该领域的某个含义，此类词汇反映在向量空间中应是作为一个语义独立的单独词汇，因此从向量特征或是语义的角度来说不存在可以替换的词汇也不会被作为错误词汇处理。

（2）缩写词汇。此类词汇的特征是拼写长度相对较短，其语义可以代表另一个单词，这种单词应分为两个方面处理：当缩写词汇拼写长度过短时一个字母发生变化，尽管编辑距离很短，但是对于这类词来说极大概率会变成毫不相关的另一个词汇，而且在长度极短的词汇中出现错误的概率也相对较低，因此应当根据文本情况设定文本检测的最小单词长度，在检测的过程中跳过过于短小的词汇；而对于长度较长的缩写词汇，反映在向量空间模型中应该与其代表的原有词汇相似度最高，但一般缩写词汇与原有词汇的编辑距离都相对较长，因此也不会进行更改处理。即便缩写词汇与原有词汇的拼写差距较小，那么被修改为原有词汇对原文的文意也没有影响。

综上所述应按照相关规则检测所有符合要求的单词，对每个单词，识别工作流程如图 4 所示。

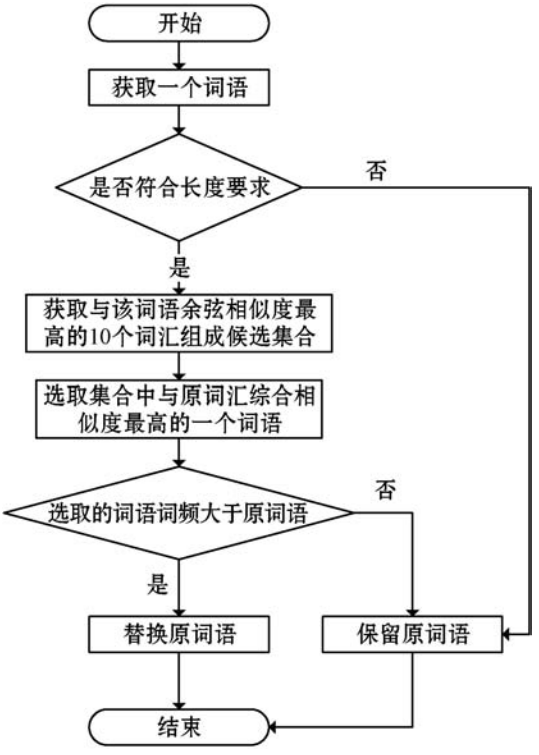


图 4 词语拼写纠错流程

通过词向量模型构建与之余弦相似度最小的 10 个词汇候选集，每个词与被检测词计算综合相似度评分，选取评分最高的一个词汇，与原词汇对比在整体文本中出现的词频，词频更高的单词视为该语境中更接近正确的词语，若候选集中的词语词频更高，则替换原词汇，反之保留原词汇。

4 实 验

为了验证本文所提算法的实际效果，本文从两个方面来进行实验验证：计算机编程领域内专业词语错误检测纠正效果和纠错后文本语义还原效果。本文验证实验环境为 CPU：Intel Core i5-4590，内存：8 GB RAM，操作系统：Microsoft Windows 10 1903；实验程序使用 Python 语言编写，Python 版本 3.7；调用 NLTK[12] 及 Gensim[13] 相关的 Python 库；文本数据来源于开源计算机问答社区 Stack Overflow[14] 中的文本数据。对

比算法为基于经典编辑距离和词频统计的错误词汇检测纠正算法。

4.1　错词检测纠正效果验证

错词检测纠正率验证的实验主要针对本文提出算法模型对计算机领域特定词汇的错误检测纠正效果进行实验验证。在所有 Stack Overflow 文本数据中,随机选取 500 句具有专业领域词汇的句子进行人工的错词纠正,获得 500 句词汇拼写标准的句子集合作为实验数据的对照数据,对上述 500 句文本中 675 个涉及计算机编程领域的词汇进行人工给错,此处错误的类型包括各类的输入错误,每类错误均在 Stack Overflow 全部文本数据中进行检索,确保该类型错误为真实存在的错误情况。

同时对上述 500 句人工给错的数据分别执行本文提出的基于 Word2vec 和基于经典编辑距离的检测纠错算法,根据数据实际情况,仅对拼写长度大于 2 的词汇进行检测,两种算法的训练数据均为全部 Stack Overflow 文本数据。实验分别将两种算法计算的结果与对照数据进行对照,对错误词汇的识别数、改正数及正确词汇的误判数三个方面进行统计和对比,结果如表 3 所示。

表 3　错词纠正效果实验结果

算法	错误词汇总数	错误词汇识别数	错误词汇改正数	正确词汇误判数
经典编辑距离算法	675	487	389	227
基于词向量纠错算法	675	507	477	152

通过实验结果所示,经典编辑距离算法和基于词向量的纠错算法的纠正效果如表 4 所示。

表 4　纠正效果对比(%)

算法	识别率	纠正率	误判率
经典编辑距离算法	72.15	57.63	33.63
基于词向量的纠错算法	75.11	70.67	22.52

通过实验验证可得,相比于经典的编辑距离算法,本文提出的基于 Word2Vec 的拼写错误检测算法的识别率提高了 4.1%,纠正率提高了 22.63%,误判率减少了 49.33%,整体效果提升显著。

4.2　语义还原效果验证

语义还原效果的实验验证使用的数据与错词检测纠正效果实验相同,引入 BLEU 值[15] 作为语义还原度的评价标准,BLEU 原用于评价机器翻译的翻译准确度,本文使用该值评价修改后的文本与原文本的语义相似程度,通过上述实验中两种算法纠正完成后的 500 句文本语句与标准对照文本进行 BLEU 的计算,结果如表 5 所示。

表 5　两种算法结果平均 BLEU 值对比

算法	平均 BLEU 值
经典编辑距离算法	0.699 215
基于词向量纠错算法	0.752 213

通过 BLEU 值对比,本文提出的基于 Word2Vec 的拼写检测纠错算法,在语义还原度上比经典的编辑距离算法提高了 7.58%,具有较好的语义还原性。

综合上述实验结果表明,本文提出的基于 Word2Vec 的计算机编程领域词语拼写错误检测算法,能够在保证较高语义还原度的基础上,较好地对计算机编程领域文本进行自动拼写错误检测和校正。

5　结　语

目前,在自然语言分析领域,语料中的错误词汇依然影响着数据统计和挖掘的精确性,本文提出了一种基于词向量计算词语相似度,再通过改进的编辑距离算法对相似度进行修正,针对 Stack Overflow 开放性问答社区中计算机编程领域的大量文本实现快速自动纠错。经过实验验证,相比较于现有的文本纠错方法,本文算法能够在保证原文语义的前提下自动将大部分文本中的错误进行纠正,取得了较好的效果。

参　考　文　献

[1] Kukich K. Techniques for automatically correcting words in text[J]. ACM Computing Surveys,1992,24(4):377 - 439.

[2] Levenshtein V I. Binary Code Capable of Correcting deletions, insertions and reversals[J]. Doklady Akademii Nauk SSSR,1966,163(4):845 - 848.

[3] Soleh M Y, Purwarianti A. A non word error spell checker for Indonesian using morphologically analyzer and HMM [C]//Proceedings of the 2011 International Conference on Electrical Engineering and Informatics. IEEE,2011:1 - 6.

[4] 谢文慧,易荣庆,彭涛. 基于键盘距离和依存分析的拼写纠错方法[J]. 吉林大学学报(理学版),2018,56(5):145 - 152.

[5] Bergsma S, Pitler E, Lin D K. Creating robust supervised classifiers via web-scale n-gram data [C]//Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics. ACL,2010:865 - 874.

[6] Kim J, Lee E, Hong T, et al. Correcting misspelled words in Twitter text [C]//2016 7th International Conference on

Big Data Technologies and Applications(BDTA). Springer, 2016: 83 - 90.
[7] Hinton G E. Learning distributed representations of concepts [C]//8th Conference of the Cognitive Science Society,1989.
[8] Nguyen H V, Bai L. Cosine similarity metric learning for face verification[C]//Proceedings of the 10th Asian Conference on Computer vision. Springer,2010:709 - 720.
[9] Mikolov T, Chen K, Corrado G, et al. Efficient estimation of word representations in vector space[EB]. arXiv:1301, 3781,2013.
[10] Mikolov T, Sutskever I, Chen K, et al. Distributed representations of words and phrases and their compositionality [C]//Proceedings of the 26th International Conference on Neural Information Processing Systems. Curran Associates Inc., 2013:3111 - 3119.
[11] Yin K T, Shan W, Liu Z R, et al. A data quality improvement method based on non-word errors correction[C]//2014 IEEE 5th International Conference on Software Engineering and Service Science. IEEE,2014:945 - 949.
[12] NLTK[EB/OL]. [2019 - 12 - 23]. http://www.nltk.org/.
[13] Genism[EB/OL]. [2019 - 12 - 23]. https://radimrehurek.com/gensim/.
[14] Stack overflow[EB/OL]. [2019 - 12 - 23]. https://stackoverflow.com/.
[15] Papineni K, Roukos S, Ward T, et al. BLEU: A method for automatic evaluation of machine translation[C]//Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics. ACL,2002:311 - 318.

(上接第 276 页)

式特征赋予合理的先验分布,建立隐式特征生成过程的联合概率表达,借助贝叶斯后验概率预测隐式特征后的显式反馈;将推荐结果转化为非合作博弈中用户效益最大化的纳什均衡求解,最终活动推荐给用户的活动集合。与其他三种推荐算法相比,本文算法有较高的查准率、查全率和平均绝对误差。但先验模型参数的获取仅靠经验,如何进一步降低参数的敏感性将是后续研究的重点。

参 考 文 献

[1] Ortega F, Hernando A, Bobadilla J, et al. Recommending items to group of users using matrix factorization based collaborative filtering[J]. Information Sciences,2016,345:313 - 324.
[2] Wang W, Zhang G Q, Lu J. Member contribution-based group recommender system[J]. Decision Support Systems, 2016,87:80 - 93.
[3] Hong M, Jung J J, Camacho D. GRSAT: A novel method on group recommendation by social affinity and trustworthiness[J]. Cybernetics and Systems,2017,48(3):140 - 161.
[4] Zhang L M, Zhou R, Jiang H X, et al. Item group recommendation: A method based on game theory[C]//Proceedings of the 26th International Conference on World Wide Web Companion. International World Wide Web Conferences Steering Committee,2017:1405 - 1411.
[5] Lai C H, Hong P R. Group recommendation based on the analysis of group influence and review content[C]//Asian Conference on Intelligent Information and Database Systems. Springer,2017:100 - 109.
[6] Lu Z Y, Li H, Mamoulis N, et al. HBGG: A hierarchical Bayesian geographical model for group recommendation [C]//Proceedings of the 2017 SIAM International Conference on Data Mining. Society for Industrial and Applied Mathematics,2017:372 - 380.
[7] Guo Q J, Ji W T, Zhou R Y. Algorithm study under big data environment of personalized recommendation based on user interest model[C]//2017 IEEE/ACI 16th International Conference on Computer and Information Science (ICIS). IEEE,2017:167 - 172.
[8] He R N, McAuley J. Fusing similarity models with Markov chains for sparse sequential recommendation[C]//2016 IEEE 16th International Conference on Data Mining (ICDM). IEEE,2017:191 - 200.
[9] 武俊芳,吴婷. 用户和相似性填充相融合的协同推荐模型[J]. 计算机工程与设计,2018,39(2):458 - 462.
[10] 徐毅,叶卫根,戴鑫,等. 融合用户信任度与相似度的推荐算法研究[J]. 小型微型计算机系统,2018,39(1):78 - 83.
[11] 张小鹏,吕学强,李卓,等. LDA 与词汇链相结合的主题短语抽取方法[J]. 小型微型计算机系统,2018,39(11):107 - 113.
[12] Qi Z F, Liu Q Q, Wang J, et al. Battle damage assessment based on an improved Kullback-Leibler divergence sparse autoencoder[J]. Frontiers of Information Technology & Electronic Engineering,2017,18(12):1991 - 2000.
[13] Sherchan W, Nepal S, Paris C. A survey of trust in social networks[J]. ACM Computing Surveys,2013,45(4):1 - 33.
[14] Chen J, Wang C K, Wang J M, et al. Recommendation for repeat consumption from user implicit feedback[J]. IEEE Transactions on Knowledge and Data Engineering,2016,28 (11):3083 - 3097.
[15] Guo L, Ma J, Chen Z M, et al. Learning to recommend with social contextual information from implicit feedback[J]. Soft Computing,2015,19(5):1351 - 1362.
[16] Anderson A, Kumar R, Tomkins A, et al. The dynamics of repeat consumption[C]//Proceedings of the 23rd International Conference on World Wide Web. ACM,2014:419 - 430.