文章编号:1004-9037(2014)01-0019-11

单词嵌入——自然语言的连续空间表示

陈恩红[1] 邱思语[2] 许 畅[2] 田 飞[1] 刘铁岩[3]

(1. 中国科学技术大学计算机科学与技术系,合肥,230027;2. 南开大学计算机科学与信息安全系,天津,300071;
3. 微软亚洲研究院,北京,100080)

摘要:单词嵌入是指运用机器学习的方法,将位于高维离散空间(维数为词典单词数目)中的每个单词映射到低维连续空间的实数向量的技术。在很多文本处理的任务中,单词嵌入提供了更好的语义级别的单词特征表示,从而为文本处理任务带来了诸多便利。同时,大数据时代海量的未标注文本数据,以及以深度学习为代表的机器学习技术的发展使高效的单词嵌入技术成为可能。本文将给出单词嵌入的定义以及实际意义,同时将综述目前单词嵌入技术的几种典型方法,包括基于神经网络的方法、基于受限玻尔兹曼机的方法以及基于单词与上下文共生矩阵分解的方法。本文将详细介绍不同模型的数学定义、物理意义以及训练方法,并给出他们之间的比较。

关键词:机器学习;自然语言;单词嵌入;文本处理

中图分类号:TP391 **文献标志码**:A

Word Embedding: Continuous Space Representation for Natural Language

Chen Enhong[1], *Qiu Siyu*[2], *Xu Chang*[2], *Tian Fei*[1], *Liu Tieyan*[3]

(1. Department of Computer Science, University of Science and Technology of China, Hefei, 230027, China;
2. College of Computer and Control Engineering, Nankai University, Tianjin, 300071, China;
3. Microsoft Research Asia, Beijing, 100080, China)

Abstract: Word embedding refers to a machine learning technology which maps seach of word lying in high-dimensional discrete space (with dimension to be the number of all words) to a real number vector in low-dimensional continuous space. Word embedding provides better semantic word representations, and thus greatly benefits text processing tasks. Meanwhile, huge amount of unlabeled text data, together with the development of advanced machine learning techniques such as deep learning, make it possible to effectively obtain high quality word embeddings. Besides, the definition and practical value of word embedding are given, and some classical methods are also reviewed to obtain word embedding, including neural network based methods, restricted Boltzmann machine based methods, and methods based on factorization of context co-occurrence matrix. For each model, its mathematical definition, physical meaning are introduced in detail, as well as training procedure. In addition, all these methods are compared in the aforementioned three aspects.

Key words: machine learning; natural language; word embedding; text processing

引 言

随着大数据时代的到来,海量的未标注的语料数据为文本处理的相关研究(诸如自然语言处理、信息检索、文档建模等)带来了新的机遇与挑战——一方面,大量的未标注文本中将为文本处理任务提供非常多的有用信息;另一方面,如何充分挖掘这些潜藏的有用信息,并将其应用于不同的文本处理任务等问题也随之而来。基于半监督学习的方法[1-3]可以部分利用这些无标记文本,并在特定的自然语言处理任务上有着良好的表现。但这

收稿日期:2013-10-17;**修订日期**:2013-12-22

些方法依赖于某一特殊的模型，其训练出的信息很难适用于其他的有监督任务。

为了解决这一问题，越来越多文本处理领域的研究人员开始专注于单词嵌入的方法。具体来说，单词的嵌入表示定义为每个单词关联的某种数学对象，通常来讲该数学对象是一个实数值向量。单词嵌入表示的目标在于学习到每个单词的向量表示，并将这种向量表示用于不同的文本处理任务。学习到的单词向量既可以作为完全的单词特征输入到某些特定任务的有监督学习算法中，也可以作为依赖于不同任务所特定提取特征的有益扩充。

为了将单词与实数向量关联起来，一个最简单的方法在于使用只有一位为 1，其他位全为 0 的向量(One hot representation)。具体来说，假设词典为 V，对于 V 中的第个单词，其关联的向量为 $\boldsymbol{w}_i=(0,0,\cdots,1,0,\cdots)\in\{0,1\}^{|V|}$，$\boldsymbol{w}_i$ 的第 i 位为 1，其他位都为 0。这种方法非常简单，但却有两个主要的缺点：(1)单词向量维度是词典大小，而词典中的单词数目往往很大，从而导致向量维度太大，引起计算上的不便；(2)该种表示唯一记录的是单词在词典中的索引，并没有刻画单词之间的相似度，从而没有为后续的文本处理任务带来更多的有用信息。

为了解决上述 One hot 表示方法的缺点，研究人员开始将注意力转到从大量的无标记文本语料中学习到更有效的单词嵌入表示。这里“有效”有两层含义，一是相对于词典大小，单词嵌入向量的维度非常低，可以认为每一维均对应某种语义的表示而没有冗余信息；二是对于类似的单词(比如 'cat' 和 'dog')，它们的向量表示也相近。海量的文本语料无疑为实现这种有效性提供了很大的帮助：文本预料中包含的单词共现以及单词先后顺序等机构化信息提供了刻画单词相似度的来源，从而为学习到语义层面有效的单词嵌入表示带来了极大的方便。

1 主要方法简述

为了充分挖掘无标记语料中的信息以获取有效的单词嵌入表示，研究人员开发了多种新的机器学习方法，其中主要包括基于神经网络的方法、基于受限玻尔兹曼机的方法以及基于单词与上下文相关性的方法。

在基于神经网络的方法中[4-7]，单词的嵌入表示常常作为神经网络的权重矩阵，神经网络通过优化某个目标函数更新其权重矩阵，从而学习到较优的单词嵌入表示。通常来讲，神经网络优化的目标函数是极大化文本语料的生成概率[4]，或者是尽可能符合某种具体任务的标记信息，例如词性标注(Pos tagging)任务中的标注信息。

与基于神经网络的方法类似，基于受限玻尔兹曼机的方法[8]的目标同样是极大化文本语料的生成概率。二者区别在于具体模型的构建——在该方法中，受限玻尔兹曼机被用来建模文档的概率，单词嵌入向量作为受限玻尔兹曼机的参数。因为受限玻尔兹曼机的目标函数的梯度无法精确求得，其训练过程是近似的梯度下降，这也与训练传统的反向传播神经网络有很大的不同。

基于单词与上下文相关性的方法首先构建单词与上下文的共生矩阵，这里上下文可以是所有文档、每个单词的左窗口、右窗口等；然后对共生矩阵做矩阵分解，从而得到每个单词的低维表示。与以上两种方法不同，基于单词与上下文相关性的方法一般不是概率模型，其训练方法一般是矩阵分解。

2 基于神经网络的方法

2.1 神经网络概率语言模型

使用神经网络以及单词嵌入技术构建语言模型的思想首先由 Yoshua Bengio 等人在文献[4,5]中提出。为了叙述方便，用缩写神经网络概率语言模型(Neural network probabilistic language model，NNLM)来代表该模型，同时引入一些记号：以 V 来代表词典中的所有单词的集合，即词典；训练样本是一串单词序列 $w_1,w_2,\cdots,w_T$，对于任意 $t\in\{1,2,\cdots,T\}$，均有 $w_t\in V$；每个单词嵌入向量的维度为 m，将所有单词嵌入向量的矩阵记为 $\boldsymbol{C}\in\mathbf{R}^{|V|\times m}$，即 V 中的第 i 个单词被映射成 $\boldsymbol{C}$ 的第 i 行 $C_i\in\mathbf{R}^m$。

有了上述记号，给出神经网络模型的训练目标：学习到给定前 $n-1$ 个单词的条件下，出现当前单词的概率，即 $P(w_t\mid w_{t-n+1},\cdots,w_{t-1})$。记 $f(w_t,w_{t-1},\cdots,w_{t-n+1})=P(w_t\mid w_{t-n+1},\cdots,w_{t-1})$，显然，为了满足概率性质，需要函数 f 满足 $\sum_{i=1}^{|V|}f(i,w_{t-1},\cdots,w_{t-n+1})=1$。同时，为了将单词嵌入向量加入到目标函数中，通常采取两层映射的形式，即

$$f(i,w_{t-1},\cdots,w_{t-n+1})=g(i,\boldsymbol{C}_{w_{t-1}},\cdots,\boldsymbol{C}_{w_{t-n+1}})$$

式中：第一层映射为矩阵 $\boldsymbol{C}$，将离散的单词映射成了连续的向量；第二层映射为函数 g，g 使用多层神经网络来建模。具体来说，有

$$P(w_t \mid w_{t-1},\cdots,w_{t-n+1}) = f(w_t,w_{t-1},\cdots,w_{t-n+1}) = \frac{e^{y_{w_t}}}{\sum_i e^{y_i}} \quad (1)$$

其中 y_i 是未经正则化的第 i 个单词概率的对数，按照式(2)计算

$$y_i = b_i + a'\tanh(d + \boldsymbol{W}\boldsymbol{x} + \boldsymbol{U}\boldsymbol{C}'_i) \quad (2)$$
$$\boldsymbol{x} = (\boldsymbol{C}_{w_{t-1}},\cdots,\boldsymbol{C}_{w_{t-n+1}})'$$

需要学习的参数是 $\theta=\{a,b,d,\boldsymbol{U},\boldsymbol{W},\boldsymbol{C}\}$，$a\in\mathbf{R}^h$，$b\in\mathbf{R}^{|V|}$，$d\in\mathbf{R}^h$，$U\in\mathbf{R}^{h\times m}$，$W\in\mathbf{R}^{h\times(n-1)m}$，$\boldsymbol{C}\in R^{|V|\times m}$，$h$ 为隐层节点个数。该式对应下述的 3 层神经网络：

(1) 输入层：通过查找表 $\boldsymbol{C}$ 将 $(w_{t-1},\cdots,w_{t-n+1})$ 映射到向量 $\boldsymbol{x}$，将词典中的第 i 个单词映射成向量 $\boldsymbol{C}_i$；

(2)隐层：通过 $\tanh(d+\boldsymbol{W}\boldsymbol{x}+\boldsymbol{U}\boldsymbol{C}_i)$ 函数将向量 $(\boldsymbol{x},\boldsymbol{C}'_i)$ 映射到隐层 h 维向量；

(3)输出层：将隐层向量通过线性变换 (a,b) 输出到最后一层，并使用 softmax 函数将其转化成概率形式。

训练神经网络使用随机梯度下降法，处理第 t 个单词时的参数更新为 $\theta \leftarrow \theta + \in \frac{\partial \log P(w_t \mid w_{t-1},\cdots,w_{t-n+1})}{\partial\theta}$，其中 $\in$ 是学习率。

2.2　树状加速方法

上述神经网络概率语言模型提供了一种学习单词嵌入的方法，同时利用学到的单词嵌入矩阵 $\boldsymbol{C}$，该模型可以建立更好的语言模型。但是该方法存在计算复杂度过高的问题：注意到在式 (1) 和 (2) 中，在计算给定前 $n-1$ 个单词 $(w_{t-1},\cdots,w_{t-n+1})$ 单词 w_t 的条件概率时，每个词典中的单词 i 对应 y_i 的值均需要计算；即在计算隐层变量 $\tanh(d+\boldsymbol{W}\boldsymbol{x}+\boldsymbol{U}\boldsymbol{C}')$ 时，共需要 $\Theta(hm(n-1)+|V|hm)$ 次操作。对于词典非常大的情况，例如 $|V|\approx 20\,000$，计算该条件概率无疑非常耗时。为了克服这一问题，Bengio 等人在文献[6]中提出了一种利用树状层次结构加速这一计算的技术，这一技术现在已经被广泛应用在了各种神经网络概率语言模型的工作中[9-12]。

该技术的基本思想很直观：注意到在上一小节陈述的基本模型中，事实上，由式(1)，每个单词(的条件概率)均由所有的单词来表示，即每个单词均由 $|V|$ 位的信息来表示，这导致了计算一个单词的条件概率的复杂度是 $|V|$ 阶的。考虑将表示一个单词的位数降低。可以采取一种层次结构：一个单词首先属于某个大类，自顶向下分别再属于某些子类，直至到达这个单词。以单词‘tiger’为例，它的层次类别可以如下：‘所有单词’→‘noun(名词)’→‘living things(生物)’→‘animal(动物)’→‘mammals(哺乳动物)’→‘cats(猫科动物)’→‘tiger(老虎)’，可见通过 7 个节点即可以表示单词‘tiger’，远远低于词典的大小 $|V|$。

具体来讲，词典 V 中的每个单词 v 均关联一个二值的向量 $(b_1(v),\cdots,b_{l_v}(v))$，$l_v$ 是 v 关联的向量的长度。该向量可以解释为 l_v 个二值的决策，例如 $b_1(v)=1$ 代表单词 v 属于最顶层的类 1，$b_1(v)=0$ 代表单词 v 属于顶层的类 2。为了得到每个单词的该二值向量表示，可以构建一棵叶子节点为所有单词的二叉树，对于每个单词，从二叉树的根节点到该单词对应的叶子节点的路径即是该单词对应的二值向量表示(例如，取转向左儿子为 1，转向右儿子为 0)。构建二叉树的方式，可以采用从知识库(例如 Wordnet)中学习到单词分类的方式[6]，也可以采用按照语料中的单词分布构建的方式以加快运算速度[10]，Mnih 和 Hinton 在文献[9]中对各种不同方式构建的二叉树的性能给出了实验性的总结。

使用该二值向量，式(1)可以替换为 $P(v|w_{t-1},\cdots,w_{t-n+1}) = \prod_{j=1}^{l_v} P(b_j(v) \mid b_1(v),\cdots,b_{j-1}(v),w_{t-1},\cdots,w_{t-n+1})$。即每个单词的条件概率分解成从二叉树根节点到该单词对应的叶子节点的 l_v 条边的概率的乘积。不失一般性，考虑建模 $P(b|\text{node},w_{t-1},\cdots,w_{t-n+1})$，其中 node 代表当前考虑的内部节点，它代表了从根节点到当前内部节点的一串 0-1 序列。类似式(2)，有

$$P(b=1 \mid \text{node},w_t,\cdots,w_{t-n+1}) = \text{sigmoid}(\beta_{\text{node}} + \alpha'\tanh(d+\boldsymbol{W}\boldsymbol{x}+\boldsymbol{U}\boldsymbol{N}'_{\text{node}})) \quad (3)$$

其中 $\text{sigmoid}(x)=1/(1+\exp(-x))$；$\beta_{\text{node}}$ 类比于(2)中单词的偏置项 b_i；α 类比于式(2)中的 a；$d,\boldsymbol{W},\boldsymbol{U}$ 与(2)中对应的符号含义相同；类似矩阵 $\boldsymbol{F}$ 代表所有单词的嵌入向量，矩阵 $\boldsymbol{N}$ 给出了所有二叉树内部节点的嵌入向量。

使用这种树状加速方式，构建的二叉树很容易做到最长路径长度是 $O(\ln|V|)$ 阶，则计算一次条件概率的复杂度由 $O(|V|)$ 降低到了 $O(\ln|V|)$，极大提升了运算效率。

2.3　Word2Vec 嵌入模型

在 2012 年和 2013 年，Thomas Mikolov 等人

在 Google Research 的工作单词向量表示(Word to vector,Word2Vec)[7,10]引起了学术界和工业界的广泛关注。Word2Vec 模型计算简单,并且在一些有趣的任务上取得了很好的效果,比如寻找单词之间的线性关系(Tokyo-Japan+France=?)等。

Word2Vec 同样属于神经网络语言模型,也采取了树状加速方式,但与经典的神经网络概率语言模型[4]相比,Word2Vec 有以下显著不同:

(1)不是定义给定前 $n-1$ 个单词的情况下出现第 n 个单词的概率,而是给定一个窗口大小(比如 n),计算给定该窗口其他 $n-1$ 个单词的条件下出现窗口中心的单词的概率;即此时考虑的上下文不再仅仅是左窗口(上文),而是包括左窗口(上文)和右窗口(下文)。

(2)关于上下文向量 $\boldsymbol{x}$ 的构建:在 Bengio 等人的工作中,上下文向量 $\boldsymbol{x}$ 是单词 w_t 之前所有 $n-1$ 个单词嵌入向量的拼接,从而有 $x\in\mathbf{R}^{(n-1)\times m}$,$m$ 是单词嵌入向量的维度;而 Word2Vec 与此不同:Word2Vec 采取两种模型,一是 CBow 模型,在该模型下,x 是窗口中所有 $n-1$ 个上下文单词嵌入向量的平均值;二是 Skip-Gram 模型,在该模型下,x 是每个上下文单词的嵌入向量,对于每个预测的窗口中心词,一共训练 $n-1$ 次。故而在 Word2Vec 中,$x\in\mathbf{R}^m$。

(3)不再设置隐层,而是直接将输入的单词嵌入向量与内部节点嵌入向量作用输出条件概率。

(4)在构建二叉树的过程中,使用了哈夫曼编码,而不是类比之前工作中使用 WordNet 构建;这样做的好处是使得运算速度更快,这是因为哈夫曼编码保证了词频高的单词对应的路径短(l_v 小),从而在预测该高频单词时需要更新参数的二叉树内部节点个数少。

综上所述,在 Word2Vec 模型中,类比于式(3)的条件概率为

$$P(v=1\mid \text{node},w_{\text{I}})=\text{sigmoid}(N'_{\text{node}}x) \tag{4}$$

其中 w_{I} 是上下文单词,$\boldsymbol{x}$ 是 w_{I} 嵌入向量,其他符号类似。

2.4 CW08 嵌入模型

Collobert 和 Weston 等人在文献[12,13]中提出了一种可以解决不同自然语言处理问题的通用架构,为了叙述方便将这个架构简写为 CW08。他们的贡献不在于解决了某个单一的自然语言处理问题,而是提出了一个对于多种自然语言处理问题通用的深层神经网络模型结构,使用这种网络结构可以生成一份通用的单词嵌入向量,来完成自然语言处理里面的各种任务,比如词性标注、命名实体识别、语句切分、语义角色标注等。

上述深层神经网络模型从逻辑上可以分为 3 个部分:单词嵌入层(查找表)、特征提取层和传统神经网络分类层。图 1 列出了这种网络结构。任务 1 和任务 2 共享同一个单词嵌入向量和卷几层。这种共享结构对于大于两个任务的情况也适用。

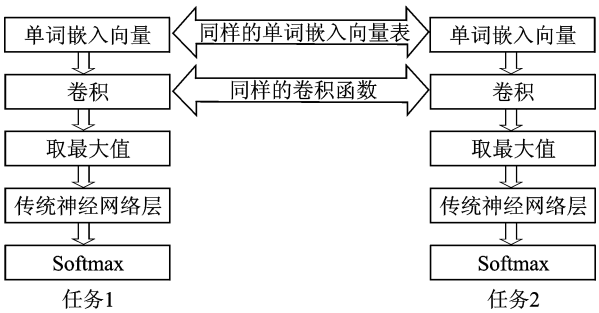

图 1 一个多任务学习深层神经网络的例子

Fig. 1 An example of multi-task deep neural network

为了方便叙述,此处再次引入一些符号:L 层前向传导神经网络记作 $f^1(\cdot)$,每一层神经网络记作,整个网络即为

$$f(\cdot)=f^L(f^{L-1}(\cdots f^1(\cdot)\cdots)) \tag{5}$$

对于一个矩阵 $\boldsymbol{A}\in\boldsymbol{R}^{d_1\times d_2}$,用 $A_{i,j}$ 来表示其第 i 行 j 列的元素,用 $\boldsymbol{A}_j$ 来代表 A 的第 j 列。用 $\langle\boldsymbol{A}\rangle_i^{d_{\text{win}}}$ 来表示以 A 的第 i 列为中心的 d_{win} 列所拼成的列向量。即

$$\begin{aligned}\langle\boldsymbol{A}\rangle_i^{d_{\text{win}}}=&(A_{1,i-d_{\text{win}}/2}\cdots A_{d_1,i-d_{\text{win}}/2},\cdots,\\&A_{1,i+d_{\text{win}}/2}\cdots A_{d_1,i+d_{\text{win}/2}})'\end{aligned} \tag{6}$$

这 3 层的结构和作用如下:

(1)单词嵌入层(查找表):给定一个单词序列 $(w_1,\cdots,w_T)$,单词嵌入层的输出可以表示为

$$f^1([w]_1^{\text{T}})=(C'_{w_1},\cdots,C'_{w_T}) \tag{7}$$

此处的参数 $\boldsymbol{C}$ 即是需要学习的单词嵌入矩阵,$\boldsymbol{C}$ 的第 i 行代表词典中第 i 个单词的嵌入向量。

(2)特征提取层:这里跟卷积神经网络非常类似:输入是由 T 个连续单词嵌入向量构成的矩阵 $\boldsymbol{f}^1$(上一层的输出),对每个宽度为 d_{win} 的连续单词窗口的嵌入向量表示,均使用同一个线性变换(W,b)将其映射到新的空间。此时该层输出的第个列向量(对应第 t 个单词窗口)可以表示为

$$f_t^2=\boldsymbol{W}\cdot\langle f^1\rangle_t^{d_{\text{win}}}+\boldsymbol{b} \tag{8}$$

这里权重矩阵 $\boldsymbol{W}$ 和 $\boldsymbol{b}$ 偏置是要训练的参数。

对于所有的这些列向量,取出每一维的最大

值，作为第三层的输出，即有

$$f_t^3 = \max_i f_{i,t}^2 \quad \forall 1 \leqslant i \leqslant n_2 \tag{9}$$

其中 n_2 表示第二层节点的数量，即矩阵 $\boldsymbol{W}$ 的行数是 n_2。

(3)传统神经网络分类层：有了上面一层被降维且被提取了特征的向量，就可以用传统神经网络的方法解决上述自然语言处理问题，最常用的办法就是构造传统的神经网络分类器。

特殊地，对于语言模型问题，在该工作中没有建模条件概率 $P(w_n|w_1,\cdots,w_{n-1})$，而是直接用神经网络最后的输出值作为联合概率分布 $P(w_1,w_2,\cdots,w_n)$ 的估计，通过对 n 个词语序列打分来判断这几个词以某种次序出现在一起的合理性。对于随机替换中间词的得分应该比合理序列得分低。用数学方法表示为最小化

$$\sum_{x\in \boldsymbol{X}}\sum_{w\in \boldsymbol{V}} \max\{0,1-f(x)+f(x^{(w)})\} \tag{10}$$

$\boldsymbol{X}$ 表示所有输入文本窗口的集合，$\boldsymbol{V}$ 表示字典集合，$\boldsymbol{x}^{(w)}$ 表示将窗口 x 的中心词随机替换成单词 w 所形成的新窗口。

接下来通过深度多任务学习就可以来训练整个网络。训练的目标是最小化所有任务的平均损失函数值。具体算法如下：

(1)选择下一个任务。

(2)为这个任务选一个随机的训练样本。

(3)用梯度下降算法来更新整个网络的参数。

(4)回到第一步。

至此得到了一份通过多个自然语言处理任务训练的单词嵌入表示。经试验证实，这种方法得到的单词嵌入表示在多种自然语言处理任务上都有比较好的表现。

2.5　递归神经网络语言模型

递归神经网络(Recurrent neural network，RNN)是一种特殊的前向传播神经网络结构。标准的 RNN 由 3 层构成：输入层、隐层和输出层，它的特别之处在于隐层通过一个重现矩阵和自己相连。重现矩阵可以传播延迟信号，从而使得 RNN 获得短期记忆的属性。具体来讲，RNN 将上一个阶段隐层的信息保留下来，记作 h_{t-1}，在当前阶段，隐层的输出信息 h_t 不仅仅与当前阶段输入 w_t 有关，而且与上一阶段隐层记忆的信息 h_{t-1} 有关。基于这两部分，隐层值 h_t 得以更新，RNN 模型也这种“递归”性质而得名。

具体到语言模型上，如果给定一串单词序列 $(w_1,w_2,\cdots,w_T)$，$w_t\in \boldsymbol{V}$。RNN 将利用式(11)计算相应的隐层序列 $(h_1,h_2,\cdots,h_T)$ 和输出序列 $(y_1,y_2,\cdots,y_T)$

$$\begin{aligned} h_t &= \tanh(\boldsymbol{W}w_t + \boldsymbol{U}h_{t-1} + b_h) \\ y_t &= \boldsymbol{A}h_t + b_y \end{aligned} \tag{11}$$

图 2 递归神经网络语言模型总结了这种神经网络语言模型。这里 w_t 代表第 t 个单词的 one-hot 表示。需要学习的模型参数集合是 $\theta=\{b_h,b_y,\boldsymbol{W},\boldsymbol{U},\boldsymbol{A}\}$。对于隐层激活函数的选取除了上述的双曲正切(tanh)函数，还可使用 sigmoid 函数。输出层可进一步使用 softmax 函数将线性输出转换为概率形式。最终学习到的矩阵 W 即是单词的嵌入矩阵。

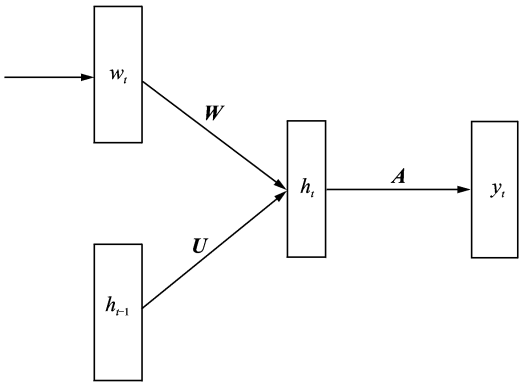

图 2　递归神经网络语言模型

Fig. 2　Recurrent neural network language model

如此可见，在处理第 t 个单词 w_t 时，RNN 将前 $t-1$ 个单词均考虑进来，而不是像经典的前向传播神经网络模型一样只考虑某个固定大小的窗口内的上下文单词(例如 $(w_{t-n+1},\cdots,w_{t-1})$)。正因为这种不需要显式指定上下文长度的优点，RNN 可以刻画更大范围的词间关系依赖。

虽然 RNN 的优点很明显，但是，当使用梯度下降法训练 RNN 却容易出现梯度爆炸和梯度消损的问题。梯度爆炸是指在训练的过程中错误信号在向后反馈时呈指数型增加的现象，梯度消损指的是相反方向，即错误信号以指数速率衰减为 0。这两种现象是由长期信息爆炸式增加造成的。为了解决这两个问题，RNN 模型在原有基础上又有许多扩展。Tomas Mikolov 和 Geoffrey Zweig 提出了一个 RNN 的扩展模型[14](参见图 3 加入特征层的 RNN 模型)：在原有的 3 层基础上加入特征层，与隐层和输出层相连。即，在当前词汇前，抽取

一段固定长度的词汇，利用潜在狄利克雷分配[15]计算词汇的话题分布。一个近似的特征层输入可以写为

$$f(s)=\frac{1}{Z}\prod_{k=0}^{K}\boldsymbol{t}_{w_{s-k}} \tag{12}$$

其中：$\boldsymbol{t}_{w_i}$是单词w_i的 LDA 话题分布向量，Z使得结果归一化。在这样的模型下，单词的嵌入表示相应的变为

$$h_t=\tanh(\boldsymbol{W}w_t+\boldsymbol{U}h_{t-1}+\boldsymbol{F}f(t)+\boldsymbol{b}_h) \tag{13}$$

有了$f(t)$提供补充信息，长期的信息不会如之前一样以指数速率归一为 0，从而避免了上文中梯度消损的问题。

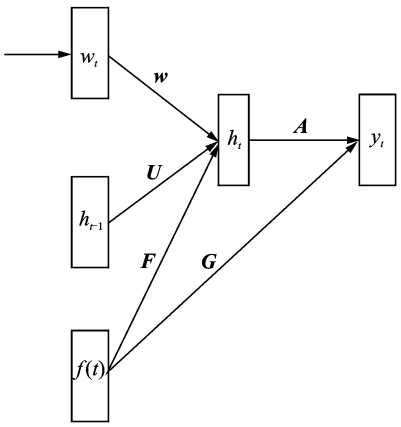

图 3 加入特征层的 RNN 模型

Fig. 3 RNN model with feature representation layer

3 基于受限波尔兹曼机的方法

3.1 受限玻尔兹曼机

受限玻尔兹曼机[16,17]（Restricted Boltzmann machine，RBM）是机器学习界普遍采用的一种生成模型，它是一种无向概率图模型，也称为马尔科夫随机场。主要用来建模观测数据的生成概率，需要说明的是 RBM 处理的数据一般是离散的。

具体来说，给定d_x维的观测向量$\boldsymbol{x}\in\{0,1\}^{d_x}$，以及$d_h$维的隐向量$h\in\{0,1\}^{d_h}$，RBM 建立它们的联合概率分布

$$P(x,h)=\frac{e^{-\mathrm{Energy}(x,h)}}{Z_\theta} \tag{14}$$

其中 Energy(x,h)是变量x和h的能量方程，定义为 Energy$(x,h)=-x'\boldsymbol{W}h-b'x-d'h$；$Z_\theta$是确保概率形式的归一化项，定义为$Z_\theta=\sum_{x_1}\cdots\sum_{x_{d_x}}\sum_{h_1}\cdots\sum_{h_{d_h}}e^{-\mathrm{Energy}(x,h)}$；模型的参数集合是$\theta=\{\boldsymbol{W},\boldsymbol{b},\boldsymbol{d}\}$，分别代表观测变量$\boldsymbol{x}$和隐变量$h$的交互、观测变量的偏置以及隐变量的偏置。

在以上的概率参数表达下，RBM 的推断问题变得相对简单，数学上可以证明[16]，对于 RBM，有下式成立

$$\begin{aligned}&P(h\mid x)=\prod_{i=1}^{d_h}P(h_i\mid x)\\&P(h_i=1\mid x)=\mathrm{sigmoid}(d_i+\boldsymbol{W}_{\cdot i}'x)\end{aligned} \tag{15}$$

其中$W_{\cdot i}$是矩阵$\boldsymbol{W}$的第i列。这样，计算给定观测变量$\boldsymbol{x}$的情况下隐变量的条件概率可以通过计算d_h个条件概率得到，即给定$\boldsymbol{x}$，对于任意的$i,j\in\{1,2,\cdots,d_h\}$，$i\neq j$，h_i，h_j是独立的。类似地，有

$$\begin{aligned}&P(x\mid h)=\prod_{i=1}^{d_x}P(x_i\mid h);\\&P(x_i=1\mid h)=\mathrm{sigmoid}(b_i+\boldsymbol{W}_{i\cdot}h)\end{aligned} \tag{16}$$

其中$W_{i\cdot}$是矩阵$\boldsymbol{W}$的第i行。训练 RBM 的过程即是极大化观测数据似然的过程。为此，求得在x_0处的概率$P(x_0)=\sum_h P(x_0,h)$关于 RBM 参数θ的梯度为[16]

$$\frac{\partial\log P(x_0)}{\partial\theta}=-\sum_h P(h\mid x_0)\frac{\partial\mathrm{Energy}(x_0,h)}{\partial\theta}+\sum_{x,h}P(x,h)\frac{\partial\mathrm{Energy}(x,h)}{\partial\theta} \tag{17}$$

因为$P(h|x)$可以很方便地求出，故而上式中的第一求和项$\sum_h P(h\mid x_0)\frac{\partial\mathrm{Energy}(x_0,h)}{\partial\theta}$可以在多项式时间内求得；第二求和项$\sum_{x,h}\frac{\partial\mathrm{Energy}(x,h)}{\partial\theta}$却因为牵扯到指数项求和，并且没有条件独立性而无法有效计算。注意到该求和项实际上是随机变量$\frac{\partial\mathrm{Energy}(x,h)}{\partial\theta}$在当前模型指定的概率下的期望，故而可以采取马尔可夫链蒙特卡洛抽样的方法近似计算。具体来说，从训练样本中选取一初始值，然后从条件概率$P(h|x_0)$中抽样出h_1，再从$P(x|h_1)$中抽样x_1，如此循环，得到一串序列：$x_0\to h_1\to x_1\to h_2\to\cdots$，使用$\frac{\partial\mathrm{Energy}(x,h)}{\partial\theta}$在这串序列上的均值来作为第二项期望的估计。近似得到对数似然的梯度后，沿梯度方向更新参数θ即可。

基于上述的 RBM 模型，Mnih 和 Hinton 在文献[8]中提出了 3 种概率图模型来求得单词嵌入。3 种模型的关键在于如何定义不同的、以单词嵌入

向量为参数的能量函数。其中前两个模型包含了隐变量，与 RBM 的能量函数形式很相似，第三个模型（对数线性模型）没有隐变量。

3.2　两种隐变量模型

Mnih 等人的第一个模型——分解 RBM 模型的能量函数如下定义：$\text{Energy}(w_n, w_{1:n-1}, h) = -(\sum_{i=1}^{n} \boldsymbol{C}'_{w_i} W_i)h - \boldsymbol{d}'h - \boldsymbol{b}'\boldsymbol{C}_{w_n} - \boldsymbol{r}'\boldsymbol{v}_{w_n}$。其中 $\boldsymbol{C}_{w_i}$ 是单词 w_i 的嵌入向量，$\boldsymbol{v}_{w_n}$ 是对应单词 w_n 的 0-1 向量（one-hot representation）。矩阵 $\boldsymbol{W}_i$ 指定了第 i 个位置的单词的嵌入向量 $\boldsymbol{C}_{w_i}$ 与隐变量 h 的交互，向量 $\boldsymbol{d}$ 指定隐变量 h 的偏置，向量 $\boldsymbol{b}$ 指定最后一个单词 w_n 嵌入向量的偏置，向量 $\boldsymbol{r}$ 指定 w_n 本身的偏置。模型参数集合为 $\theta = \{\boldsymbol{C}, \boldsymbol{W}, \boldsymbol{d}, \boldsymbol{b}, \boldsymbol{r}\}$。注意到，本模型与前述的各个基于神经网络的模型最大的区别在于每个位置的单词指定了不同的参数 $\boldsymbol{W}_i$，这代表不同位置的单词对单词的 w_n 影响是不同的。

第二个模型——时序分解 RBM 模型采用了与递归神经网络类似的思想，注意到事实上单词 w_t 之前所有的单词（$w_1, w_2, \cdots, w_{t-1}$）对 w_t 均有影响，然而大部分工作都做了一个很强的假设：w_t 仅与其之前的 $n-1$ 个单词（$w_{t-n+1}, \cdots, w_{t-1}$）有关。时序 RBM 模型的目标就在于克服这种假设，即用给定的单词序列（$w_1, \cdots, w_{t+n-1}$）预测单词 w_{t+n}。

为了达到这个目的，在预测单词 w_{t+n} 时，对数线性模型维持了 t 个与分解 RBM 模型非常类似的模型，即 t 个 n 单词序列（$w_1, \cdots, w_n$），（$w_2, \cdots, w_{n+1}$），…，（$w_t, \cdots, w_{t+n-1}$）中的每一个均作为一个独立的分解 RBM 模型的可见变量。这样得到了 t 个隐层表示，使用 h_τ 来代表第 τ 个模型的隐层变量。

为了将上文中的信息不断后传至需要预测的单词 w_{t+n}，在第 $\tau+1$ 个模型中，除了第 $\tau+1$ 个 n 单词序列，上一个模型的隐变量 h_τ 也作为第 $\tau+1$ 个模型的输入可见变量参与训练，h_τ 与 $h_{\tau+1}$ 之间的交互矩阵 $\boldsymbol{A}$ 是多层之间共享的，即 $\boldsymbol{A}$ 不随着层数 τ 而改变。时序分解 RBM 模型通过这种“组合本层原有输入与上一层输出作为本层真正输入”的方式克服了之前 n 单词窗口方法的局限，不难看出该方式与递归神经网络的思想是非常类似的。

两个模型的推断和学习与经典的受限玻尔兹曼机推断和学习的方式类似，此处不再赘述。

3.3　对数线性模型

Mnih 提出的第三个模型——对数线性模型（Log-bilinear model，LBL）与前两个基于 RBM 的模型不同的地方在于该模型没有加入隐变量的建模。事实上，该模型的能量函数定义与分解 RBM 模型非常类似：$\text{Energy}(w_n, w_{1:n-1}, h) = -(\sum_{i=1}^{n-1} \boldsymbol{C}'_{w_i} \boldsymbol{W}_i)\boldsymbol{C}_{w_n} - \boldsymbol{b}'\boldsymbol{C}_{w_n} - \boldsymbol{r}'\boldsymbol{v}_{w_n}$。不同在于 LBL 模型中，使用前 $n-1$ 个单词的嵌入与第 n 个单词 w_n 的嵌入向量作用，而并非与隐层变量作用，矩阵 $\boldsymbol{W}_i$ 仍然指定了不同位置单词的不同作用。

舍去隐变量使得模型的物理意义更清楚，同时使得计算更加简单。事实上，该模型与 Word2Vec 模型非常类似，除了以下两点不同：

（1）LBL 对不同位置的单词指定了不同的参数矩阵 $\boldsymbol{W}_i$，而 Word2Vec 中不存在这样的矩阵，Word2Vec 直接使用上下文单词与预测单词的嵌入向量作用，即相当于 $\boldsymbol{W}_1 = W_2 = \cdots = W_{n-1} = \boldsymbol{I}$，$\boldsymbol{I}$ 为单位矩阵。

（2）做预测时，Word2Vec 考虑被预测单词的上下文，即左窗口和右窗口内的单词均考虑，HBL 则仍是只考虑上文（左窗口）。

在文献[9]中，Mnin 和 Hinton 采用上文所述的树状加速方法对对数线性模型进行加速，取得了良好的实验效果。他们不使用任何外部知识数据（例如 WordNet）构建的单词树，而是采用一种基于训练语料的类似自助法的方式：首先构建一棵随机的单词树，然后基于该树训练得到单词嵌入向量，最后基于得到的单词嵌入向量进行层次聚类产生最终使用的单词树。

4　基于单词与上下文共生矩阵分解的方法

用 $\boldsymbol{F}$ 来代表单词与上下文的共生矩阵。一般来讲，是一种刻画单词与其上下文的一种计数模型，$\boldsymbol{F}$ 的行空间是单词空间，列空间是上下文空间，$\boldsymbol{F}$ 的行列元素 $\boldsymbol{F}_{i,j}$ 代表第 i 个单词和第 j 个上下文环境的关系，一般是单词 i 在环境 j 中出现的次数或者频率。构建了共生矩阵 $\boldsymbol{F}$ 之后，对 $\boldsymbol{F}$ 做矩阵分解，即可得到单词在隐空间上的一个表示，即为单词的嵌入向量。基于不同的上下文可以构建不同的矩阵。

4.1 潜在语义分析

潜在语义分析(Latent semantic analysis, LSA)[18]是一种分析单词与文档相关性的常用方法。在 LSA 中,上下文是所有的文档,假定词典大小为 p,文档数为 q,则 $\boldsymbol{F}\in\mathbf{R}^{p\times q}$,$F_{i,j}$ 代表第 i 个单词在第 j 个文档中频率,或者 tf·idf 值。LSA 对 $\boldsymbol{F}$ 矩阵实施奇异值分解

$$\boldsymbol{F}=\boldsymbol{U}\Sigma\boldsymbol{V}' \tag{18}$$

式中:$\boldsymbol{U}\in\mathbf{R}^{p\times r}$,$\boldsymbol{V}\in\mathbf{R}^{q\times r}$,$\boldsymbol{U}\boldsymbol{U}'=\boldsymbol{I}$,$\boldsymbol{V}\boldsymbol{V}'=\boldsymbol{I}$,$r$ 是矩阵 $\boldsymbol{F}$ 的秩,$\Sigma=\mathrm{diag}(\sigma_1,\cdots,\sigma_r)$ 记录了 $\boldsymbol{F}$ 的所有奇异值。注意到矩阵 $\boldsymbol{U}$ 记录了每个单词在 r 维隐空间下的表示,类似的矩阵 $\boldsymbol{V}$ 记录了每个文档在维隐空间下的表示,σ_i 指定 r 维隐空间中第 i 维的权重。这里的隐空间可以代表主题空间,则由 $F_{i,j}=\sum_{k=1}^{r}U_{i,k}\sigma_k V_{j,k}$ 可见,单词 i 在文档 j 中的词频(或者 tf·idf 值),是由单词 i 在每个主题下的权重与对应主题在文档 j 中的权重的乘积的加权平均,这里加权的权重是每个主题的权重($\sigma_1,\cdots,\sigma_r$)。

可见,可以使用矩阵 $\boldsymbol{U}$ 作为单词的嵌入向量矩阵,$\boldsymbol{U}$ 的第 i 行即是单词的嵌入向量,该嵌入向量是 r 维的。

4.2 基于典型相关分析的方法

基于典型相关分析(Canonical correlation analysis,CCA)的方法[19,20]构建的上下文矩阵有两个:一是每个单词左窗口内的所有单词;二是每个单词右窗口内的所有单词。该方法的基本思想是极大化两个矩阵的协方差。

典型相关分析(CCA)和主成分分析(Principle component analysis,PCA)类似:给定一个矩阵,PCA 计算一个投影方向,使得矩阵的行向量投影到该方向之后方差最大;CCA 与此不同的是它处理两个矩阵:以本文的问题为例,假设语料中共有 n 个单词($w_1,\cdots,w_n$),词典大小为 $|V|$,用 $\boldsymbol{W}$ 代表所有单词的矩阵。用 $\boldsymbol{L}$ 和 $\boldsymbol{R}$ 分别代表单词的左窗口矩阵和右窗口矩阵,$\boldsymbol{L}\in\mathbf{R}^{n\times|V|h}$,$\mathbf{R}\in\mathbf{R}^{n\times|V|h}$,这里 h 是指定的窗口大小。希望计算得到两个方向 ϕ_l 和 ϕ_r,使得 L 在 ϕ_l 方向的投影与 $\mathbf{R}$ 在 ϕ_r 方向的投影的协方差最大,可以认为这两个方向保持了 $\boldsymbol{L}$ 和 $\mathbf{R}$ 中最"有用"的一维信息。该最大化目标可以用下式来表达

$$\max_{\phi_l,\phi_r}\frac{E[\langle\boldsymbol{L},\phi_l\rangle\langle\mathbf{R},\phi_r\rangle]}{\sqrt{E[\langle\boldsymbol{L},\phi_l\rangle^2E[\langle\boldsymbol{R},\phi_r\rangle^2]}}$$

其中 E 代表经验期望。用 C_{lr}(C_{ll})代表 $\boldsymbol{L}$ 和 $\boldsymbol{R}$(L 和 L 之间)之间的协方差矩阵,易知 $\boldsymbol{C}_{lr}=\boldsymbol{L}'\boldsymbol{R}$,$C_{ll}=\boldsymbol{L}'\boldsymbol{L}$。那么可证明上式的解$\langle\phi_l,\phi_r\rangle$由下面的方程给出

$$\boldsymbol{C}_{ll}^{-1}\boldsymbol{C}_{rl}\boldsymbol{C}_{rr}^{-1}\boldsymbol{C}_{rl}\phi_l=\lambda\phi_l$$
$$C_{rr}^{-1}C_{rl}C_{ll}^{-1}C_{lr}\phi_r=\lambda\phi_r$$

用$\langle\phi_{\boldsymbol{L}},\phi_{\boldsymbol{R}}\rangle$来代表 m 个最大的左特征向量和右特征向量集合,这里的"最大"指对应的特征值最大,可见 $\boldsymbol{\phi}_{\boldsymbol{L}}\in\mathbf{R}^{|V|h\times m}$,$\boldsymbol{\phi}_{\boldsymbol{R}}\in\mathbf{R}^{|V|h\times m}$。基于上述符号,文献[20]中给出的两步典型相关分析算法如下:

(1)输入:矩阵 $\boldsymbol{L},\boldsymbol{W},\boldsymbol{R}$。

(2)对 $\boldsymbol{L}$ 和 $\boldsymbol{R}$ 做典型相关分析,$\boldsymbol{CCA}(\boldsymbol{L},\boldsymbol{R})\rightarrow(\boldsymbol{\phi}_{\boldsymbol{L}},\boldsymbol{\phi}_{\boldsymbol{R}})$。

(3)计算矩阵 $\boldsymbol{S}=[\boldsymbol{L}\boldsymbol{\phi}_{\boldsymbol{L}}\boldsymbol{R}\boldsymbol{\phi}_{\boldsymbol{R}}]$。

(4)对矩阵 $\boldsymbol{S}$ 和 $\boldsymbol{W}$ 做典型相关分析,$\boldsymbol{CCA}(\boldsymbol{S},\boldsymbol{W})\rightarrow(\boldsymbol{\phi}_s,\boldsymbol{\phi}_w)$。

(5)输出:矩阵 $\boldsymbol{\phi}_w$,即为最终的单词嵌入矩阵。

该算法中,第一步是求左右窗口矩阵 $\boldsymbol{L},\boldsymbol{R}$ 的 $\boldsymbol{CCA}$,在求得后$\langle\boldsymbol{\phi}_{\boldsymbol{L}},\boldsymbol{\phi}_{\boldsymbol{R}}\rangle$,分别将 $\boldsymbol{L}$ 投影到 $\boldsymbol{\phi}_{\boldsymbol{L}}$ 方向,将 $\boldsymbol{R}$ 投影到 $\boldsymbol{\phi}_{\boldsymbol{R}}$ 方向(算法第三行),得到每个单词的一个隐状态 $\boldsymbol{S}$,$\boldsymbol{S}$ 记录了左右窗口矩阵中最相关的 m 个成分。在第二步中,对矩阵 S 和原始矩阵 $\boldsymbol{W}$ 做 $\boldsymbol{CCA}$,以求得的 m 个关于 $\boldsymbol{W}$ 的投影矩阵 $\boldsymbol{\phi}_w\in\mathbf{R}^{|V|\times m}$作为最终的的单词嵌入矩阵。事实上,可以证明,第二步中求单词嵌入矩阵的过程就是对隐变量矩阵取平均的过程,有 $\boldsymbol{\phi}_w(w)=\mathrm{avg}(\boldsymbol{S}_t:w_t=\boldsymbol{W})$。

5 模型比较

各种方法的数学模型和实现细节的比较结果总结在表 1。具体来说,比较的方面包括:(1)方法属于的类型,包括神经网络模型、受限玻尔兹曼机模型和共生矩阵分解模型。(2)训练的目标,包括极大化数据条件似然($\max P(w_t|\mathrm{contest})$、极大化数据观测似然($\max P(w_t,\mathrm{context})$)、区分正确数据和错误数据(使观测数据的得分大于随机生成的错误数据得分)以及基于相关性的降维。(3)是否含有隐层。(4)模型考虑的上下文:包括窗口大小(固定大小、可变大小)、窗口位置(左窗口、右窗口)、单词所在句子以及所有文档;可变大小的窗口和左右窗口代表更多的信息被考虑进去,从而使得模型有更好的性能。(4)是否有加速方法。这里主要指树状加速方法。

各个方法的优缺点总结在表 2。主要从 3 个

表 1　不同单词嵌入方法的模型细节比较

Table 1　Comparison of different word embedding models

方法名称	类型	训练目标	是否含隐层	上下文	加速方法
NNLM[5]	神经网络	极大化数据条件似然	是	固定大小的左窗口内单词	树状加速:基于 WordNet 构建单词树[6]
Word2Vec[7,10]	神经网络	(1)极大化数据条件似然 (2)区分正确数据和错误数据	否	固定大小的左、右窗口内单词	树状加速:基于哈夫曼编码构建单词树[10]
CW08[12,13]	神经网络	(1)极大化数据条件似然 (2)区分正确数据和错误数据	是	单词所在句子	无
RNNLM[14]	神经网络	极大化数据观测似然	是	可变长度的左窗口内单词	无
FRBM[8]	受限玻尔兹曼机	极大化数据观测似然	是	固定大小的左窗口内单词	无
TFRBM[8]	受限玻尔兹曼机	极大化数据观测似然	是	可变长度的左窗口内单词	无
LBL[8]	受限玻尔兹曼机/神经网络	极大化数据观测似然	否	固定大小的左窗口内单词	树状加速:基于数据的自助算法构建单词树[9]
LSA[18]	共生矩阵分解	基于相关性降维		所有文档	无
CCA[19,20]	共生矩阵分解	基于相关性降维		固定大小的左、右窗口内单词	无

表 2　不同单词嵌入方法的优缺点比较

Table 2　Pros and Cons of different word embedding models

方法名称	模型表达能力	训练效率	有无理论保证
NNLM[5]	隐层加入使得神经网络可表达的函数空间变大,表达能力较强	可使用树状加速方法加速,但隐层的加入使得参数集合变大,从而训练效率一般	神经网络的优化目标非凸,很难找到全局最优解,从而没有理论保证
Word2Vec[7,10]	没有隐层,同时上下文窗口大小固定,而且直接使用输入单词和输出单词的交互来计算概率,没有参数矩阵,从而表达能力很差	树状加速方法以及较小的参数空间使得训练效率很高	没有理论保证,理由同上
CW08[12,13]	含有多层隐层,同时处理的输入对象是一个句子,而并非固定的单词窗口,因而表达能力很强	没有合适的加速方法,训练效率很低	没有理论保证,理由同上
RNNLM[14]	有隐层,并且通过递归的结构使得窗口大小没有限制,从而表达能力很强	会遭遇梯度爆炸和梯度消损,训练效率很低	没有理论保证,理由同上
FRBM[8]	有隐层,表达能力较强	训练 RBM 需要采用马尔可夫链蒙特卡洛抽样的方法,收敛速率一般很慢,从而训练效率很低	RBM 的训练目标非凸,同时抽样的方法难以确定是否收敛,从而没有理论保证
TFRBM[8]	有隐层,同时采用了类似 RNN 的无限制窗口,从而表达能力很强	训练效率很低,理由同上	没有理论保证,理由同上
LBL[8]	没有隐层,上下文窗口固定,但使用了参数矩阵来指定输入单词向量与输出单词向量的交互,表达能力较差	训练效率很低,理由同上	没有理论保证,理由同上
LSA[18]	非概率模型,表达能力较差	训练过程需要实施复杂度为矩阵奇异值分解,效率很低	优化目标是凸的,奇异值分解唯一确定了最优解,从而有理论保证
CCA[19,20]	非概率模型,表达能力较差	典型相关分析(CCA)需要计算矩阵特征向量,复杂度同样为,训练效率很低	典型相关分析的优化目标是凸的,最优解唯一确定,从而有理论保证

方面比较它们的优缺点:模型的表达能力强弱、训练效率高低以及优化的方法有没有理论保证(即可达到最优解)。

表 2 中,几个缩写词的意义如下:RNNLM(Recurrent neural network language model,递归神经网络语言模型);FRBM(Factored RBM,受限玻尔兹曼机),TFRBM(Temporal factored RBM,时序分解受限玻尔兹曼机),其他缩写词全称已在文中全出。

6 结束语

单词嵌入是当今非常流行的用于文本处理任务的一种技术。本文综述了当下流行的各种求取单词嵌入向量的方法,包括基于神经网络的方法、基于受限玻尔兹曼机的方法以及基于单词与上下文共生矩阵分解的方法。本文阐述了各个方法的具体数学模型和实现细节,同时给出了各个方法优缺点的比较,以期让相关研究者更加熟悉单词嵌入这一技术,并将该技术应用到各种新的文本处理问题中。未来的单词嵌入向量工作中,一个重要的方向将外部知识库中的知识考虑进来,结合当前深度神经网络技术的飞速发展,产生更好的单词嵌入表示。这里的知识库可以包含单词形态相似度、句法相似度以及语义相似度等方面的知识,通过这样一些外部知识的辅助,深度神经网络将会得到对文本处理任务更有用的信息,从而有可能获得更好的单词嵌入表示。

参考文献:

[1] Rie Kubota Ando, Tong Zhang. A high-performance semi-supervised learning method for text chunking [C]//Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics (ACL' 05). Stroudsburg, PA, USA: Association for Computational Linguistics, 2005:1-9.

[2] Suzuki J, Isozaki H. Semi-supervised sequential labeling and segmentation using giga-word scale unlabeled data[C]//Proceedings of the 46th Annual Meeting on Association for Computational Linguistics (ACL' 08). Columbus, Ohio, USA: Association for Computational Linguistics,2008: 665-673.

[3] Suzuki J, Isozaki H, Carreras X, et al. An empirical study of semi-supervised structured conditional models for dependency parsing[C]//Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2. [S. l.]:Association for Computational Linguistics, 2009: 551-560.

[4] Bengio Y, Ducharme R, Vincent P. A neural probabilistic language model[C]//Advances in Neural Information Processing Systems. Vancouver, British Columbia, Canada: Neural Information Processing Systems Foundation, 2001: 933-938.

[5] Bengio Y, Ducharme R, Vincent P, et al. A Neural Probabilistic Language Model[J]. Journal of Machine Learning Research, 2003, 3: 1137-1155.

[6] Morin F, Bengio Y. Hierarchical probabilistic neural network language model [C]//Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics. Barbados: ACM, 2005:246-252.

[7] Mikolov T, Chen K, Corrado G, et al. Efficient estimation of word representations in vector space[EB/OL]. arXiv preprint arXiv, 2013:1301:3781.

[8] Mnih A, Hinton G. Three new graphical models for statistical language modelling[C]//Proceedings of the 24th International Conference on Machine Learning. Corvallis, USA: ACM, 2007: 641-648.

[9] Mnih A, Hinton G E. A scalable hierarchical distributed language model[C]//Advances in Neural Information Processing Systems. Vancouver, B C, Canada: Neural Information Processing Systems Foundation, 2008: 1081-1088.

[10] Mikolov T, Sutskever I, Chen K, et al. Distributed representations of words and phrases and their compositionality [C]//Advances in Neural Information Processing Systems. Nevada, United States: Neural Information Processing Systems Foundation, 2013: 3111-3119.

[11] Larochelle H, Lauly S. A neural autoregressive topic model[C]// Advances in Neural Information Processing Systems. Nevada, United States: Neural Information Processing Systems Foundation, 2012: 2717-2725.

[12] Collobert R, Weston J. A unified architecture for natural language processing: Deep neural networks with multitask learning[C]//Proceedings of the 25th International Conference on Machine Learning. Helsinki, Finland: ACM,2008: 160-167.

[13] Ronan Collobert, Jason Weston, Léon Bottouet, et al. Natural language processing (almost) from Scratch[J]. J Mach Learn Res, 2011,12:2493-2537.

[14] Mikolov T, Zweig G. Context dependent recurrent

neural network language model[C]//Proceedings of the 4th IEEE Workshop on Spoken Language Technology. Florida, United States: IEEE, 2012: 234-239.

[15] Blei D M, Ng A Y, Jordan M I. Latent dirichlet allocation[J]. The Journal of Machine Learning Research,2003,3:993-1022.

[16] Bengio Y. Learning deep architectures for AI[J]. Foundations and Trends in Machine Learning, 2009, 2(1): 1-127.

[17] Hinton G E, Salakhutdinov R R. Reducing the dimensionality of data with neural networks[J]. Science, 2006,313(5786): 504-507.

[18] Dumais S T, Furnas G W, Landauer T K, et al. Using latent semantic analysis to improve access to textual information[C]//Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. [S.l.]: ACM, 988:281-285.

[19] Dhillon P S, Foster D P, Ungar L H. Multi-View learning of word embeddings via CCA[C]//Advances in Neural Information Processing Systems. Granada, Spain: Neural Information Processing Systems Foundation, 2011, 24: 199-207.

[20] Dhillon P, Rodu J, Foster D, et al. Two step CCA: A new spectral method for estimating vector models of words[EB/OL]. arXiv preprint arXiv, 2012: 1206.6403.

作者简介:陈恩红(1968-),男,教授,研究方向:机器学习、数据挖掘、社会网络、个性化推荐系统;田飞(1990-),男,博士研究生,研究方向:机器学习、信息检索、自然语言处理,E-mail:tianfei@mail.ustc.edu.cn;邱思语(1992-),女,学士,研究方向:机器学习、深度学习、神经网络;许畅(1991-),女,学士,研究方向:机器学习、自然语言处理;刘铁岩(1976-),男,高级研究员/教授,研究方向:机器学习、信息检索、数据挖掘、计算经济学。