文章编号：2095-2163(2020)02-0301-07 中图分类号：TP391.1 文献标志码：A

面向社交媒体的反讽识别

罗观柱，赵妍妍，秦 兵，刘 挺
（哈尔滨工业大学 计算机科学与技术学院，哈尔滨 150001）

摘 要：反讽是社交媒体中常用的一种修辞方法，反讽的存在对传统的情感分析或观点挖掘带来了挑战。反讽修辞中一种常用的表达形式为使用极性相反的情感词来表达“前后情感矛盾”。本文针对该形式的反讽，提出了一种基于注意力机制的神经网络模型，该模型可以捕捉一句话中的前后情感矛盾的两个词从而推断是否为反讽。该模型不考虑句子的上下文，仅从句子本身的结构出发，计算任意两个词之间的注意力分数从而发现导致反讽的关键词。本模型在多个数据集上取到了很好的效果，并且该模型有较好的可解释性。

关键词：反讽识别；注意力机制；情感分析；社交媒体；神经网络

Social media-oriented sarcasm detection

LUO Guanzhu, ZHAO Yanyan, QIN Bing, LIU Ting
(School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China)

【Abstract】 Sarcasm is a rhetorical method commonly used in social media. The existence of sarcasm poses a challenge to traditional sentiment analysis or opinion mining. A common form of expression in sarcasm rhetoric is the use of emotional words with opposite polarities to express " inconsistency." In this paper, a neural network model based on attention mechanism is proposed for this form of sarcasm. This model can capture the two inconsistent words in a sentence to infer whether it is sarcasm. The model does not consider the context of the sentence, only from the structure of the sentence itself, calculates the attention score between any two words to find the keyword that leads to sarcasm. This model has achieved good results on multiple data sets, and has good interpretability.

【Key words】 sarcasm detection; attention mechanism; sentiment analysis; social media; neural network

0 引 言

随着社交媒体(Social Media)的高速发展，如Twitter、Reddit、微博等已经是人们日常生活中的一部分，网民倾向并擅长在社交媒体中使用某些修辞方法来宣泄情感，比如使用幽默、讽刺或反语等表达方式来强调个人情感。在社交网络上这种发表观点或表达情感的修辞方法丰富着人类语言，但同时修辞方法的添加，给多项自然语言处理(NLP)任务带来了明显的困难。比如在情感分析任务中，以往传统的技术则难以正确检测含反语讽刺文本等修辞成分的真实情感。修辞方法的广泛运用随之带来的副作用会严重影响社交媒体文本的情感计算或观点挖掘的检测准确性，故而研究反语、讽刺或幽默等修辞方法，对于多项自然语言处理任务，尤其是情感分析、观点挖掘等具有重要意义。在社会媒体的常用修辞方法中，反语(可译作 Irony)或者讽刺(可译作 Sarcasm)的应用较为普遍。具体来讲，前者常使用跟作者心中原意不一致的词来强调情感，往往隐含有否定、反对、讽刺或者自嘲等情感，是一种带有强烈感情色彩的修辞方法，以“I just love being ignored □”为例，作者显然想表达一种被人忽视的负面情感，但字面上却使用了强烈的褒义词“love”。后者常用夸张或比喻等修辞对某对象进行一种揭露，或者批评嘲笑等，以“Good thing Trump is going to bring back all those low education high paying jobs.”为例，作者通过这种文字来表达对 Trump 的批判。反语与讽刺的关系[1]，从某种意义上可以认为讽刺是包含作者个人情绪(比如包含攻击抨击等情绪)的反语形式的一种。为了方便表述，在本文中会将反语和讽刺统一表达为反讽一词，反语和讽刺的区别在此忽略。

反讽的类别可以进一步划分，在 SemEval2018 任务 3[2] 中将反讽分为 3 类，即：前后情感矛盾(ironic by clash)、场景反讽(situation Irony)和其他反讽(other irony)。前后情感矛盾反讽比如“I just ***love*** being ***ignored*** □”中的 {***love***, ***ignored***} 为极性

作者简介：罗观柱(1991-)，男，硕士研究生，主要研究方向：自然语言处理；赵妍妍(1983-)，女，博士，副教授，主要研究方向：自然语言处理、信息抽取、篇章语义等；秦 兵(1968-)，女，博士，教授，博士生导师，主要研究方向：自然语言处理、信息抽取、篇章语义等；刘挺(1972-)，男，博士，教授，博士生导师，主要研究方向：自然语言处理、文本挖掘、信息检索等。

收稿日期：2019-06-10

相反的两词,正因为这两个词的使用导致了该句话为反讽修辞;场景反讽比如“Just saw a ***non - smoking sign*** in the lobby of a ***tobacco company***”,“***non - smoking sign***”在“***tobacco company***”这种场景下才是一种反讽的说法;其他反讽为不含明显极性相反词的类型,比如“Human brains disappear every day. Some of them have never even appeared”。经统计前后情感矛盾反讽约占 69.9%。本文针对反讽中的前后情感矛盾形式,设计了一种词对注意力(word pairs attention)模型,可以捕捉{***love***, ***ignored***}这种极性相反词,从而可以推断一句话是否是反讽修辞。

1 相关工作

反讽是一种常用的修辞方法,国内外众多研究者对反讽理论及其识别方法做了很多工作。国内外学者对反讽检测提出了若干算法,大多数的研究都将反讽识别看作一种文本分类任务。这些算法可分为 3 类,分别是:基于规则的反讽识别、基于传统机器学习的反讽识别和基于深度学习的反讽识别。对此可做阐释分述如下。

1.1 基于规则的反讽识别

Tsur 等人[3]提出的讽刺检测算法用到了少量标注的种子句子,但没有使用未标注数据,通过网页搜索自动扩展种子集作为训练集合(train set),然后使用拓展后的训练集合来学习并分类,学习时使用的特征包括 2 类:基于模板的特征和基于标点的特征。对于前者,每个模板是一个高频词的有序列表,类似于数据挖掘技术中的列表模板。后者则包括叹号、问号和引号等标点符号的数量,以及句中首字母大写和全大写的单词数量,最后使用 k-近邻进行分类。

1.2 基于传统机器学习的反讽识别

Gonzalez-Ibanez 等人[4]用 tweets 数据研究了直接表达正负面观点的讽刺和非讽刺的推文。过程中采用了基于 SVM 和逻辑回归的监督学习方法。特征为 unigram 和一些基于词典的信息,包括词类、感叹词和标点符号等。其中也用到了表情符号和恢复标记。

1.3 基于深度学习的反讽识别

基于深度学习的方法最近在 NLP 研究中的众多领域引起了轰动并成果显著。在反讽识别任务上,Bamman 等人[5]使用待检测文本的上下文信息,并进一步挖掘社交用户的行为信息,设计基于深度学习的讽刺识别模型。Zhang 等人[6]使用双向递归神经网络捕捉目标推特文本的句法和语义信息,同时利用与目标推文相关的历史推文自动学习特征进行讽刺识别,并取得较好的性能。Chen 等人[7]和 Gui 等人[8]从表示学习的角度切入,提高文本分类情感分析模型的效果。Ghosh 等人[9]提出的一种卷积长短时记忆网络(CNN-LSTM-DNN)取得了最好的性能。

2 词对注意力模型

针对前后情感矛盾式反讽(占比 69.9%),研究提出了一种上下文无关的反讽识别模型。例如“I just ***love*** being ***ignored*** □”,“***Shitty*** drivers are always ***fun***.”,对于词对(word pairs){***love***, ***ignored***}与{***Shitty***, ***fun***}在情感、状态或行为上“相反”,从这一点出发研究可以构造一种基于注意力机制[10](Attention Mechanism)的深度学习网络模型,以此用来查找矛盾词对。对此拟展开研究论述如下。

2.1 词对矛盾模型

一般来说,已有的反讽识别算法往往依靠较深的序列上下文神经网络模型来对要检测的反讽句子进行表述,较常用的序列模型有 GRU[11]门控循环单元、LSTM[12]长短时间记忆网络等模型提取上下文信息特征。这类网络模型共同的不足是使用中常常难以准确地捕捉目标反讽句子的“词对不一致(word pairs clash)”或者称之为“词对矛盾”这一鲜明特征,因为直接使用 GRU、LSTM 表征句子意味着忽略了目标反讽句中的明显特征,该特征的缺失可能会影响模型的效果。针对反讽识别的模型应该能够关注到前后不一致(矛盾)的词对,而且模型的准确性也会得到提高,更重要的是通过这种思路模型还会具有一定的可解释性。本文提出的模型使用了注意力机制来实现上述目标。该模型的整体框架如图 1 所示。

由图 1 可知,为了捕捉两词之间的“矛盾性”,研究构造了一种 word pairs attention 模型(WPA),即再将句子经过 BiLSTM 层表示后,任意两词的隐层向量做 attention,这样对于一个长度为 L 的句子可得到 $L \times L$ 的注意力分数矩阵,然后使用某种方案利用该注意力分数矩阵得到句子的向量表示,最后使用 softmax 概率归一化函数对句子表示向量进行二元分类可得相应的类别。其中,d_e 为 word embedding 的维度,L 为句子长度,d_h 为隐层向量维度,d_s 为句子表示向量的维度。为此,研究还提出了 2 种利用注意力分数矩阵的方案,一种是使用 max

pooling[13]，对应的模型称为 WPA-P；另一种是二次 attention[14]，对应的模型称为 WPA-A。前者是对矩阵的每一行进行 max pooling 操作得到 $L \times 1$ 向量，在此基础上进一步得到句子向量；后者是将 $L \times L$ 的注意力分数矩阵看作 L 个 $L \times 1$ 向量，进一步得到 L 个以词为基准的句子表示，再对其做 self-attention 得到句子向量。

2.1.1　word pairs attention 计算

简单来讲，word pairs attention 模型（WPA）是一种基于词对（word pairs）关系的模型，可以引导模型训练中刻意关注{***love***，***ignored***}这种不一致的词对关系，WPA 模型框架如图 2 所示。

想要计算任意词对的注意力分数，研究使用的是线性感知机来计算注意力分数，具体的计算公式为：

$$s_{ij} = \boldsymbol{W}_a([w_i; w_j]) + b_a. \tag{1}$$

其中，$w_i, w_j \in R^{d_h}$ 为句子中的任意两个词的 BiLSTM 隐层表示；符号“;”代表两向量拼接；$\boldsymbol{W}_a \in R^{1\times 2d_h}$ 为感知机的系数矩阵；标量 b_a 为感知机的偏置；标量 s_{ij} 即为这两个词的注意力分数。

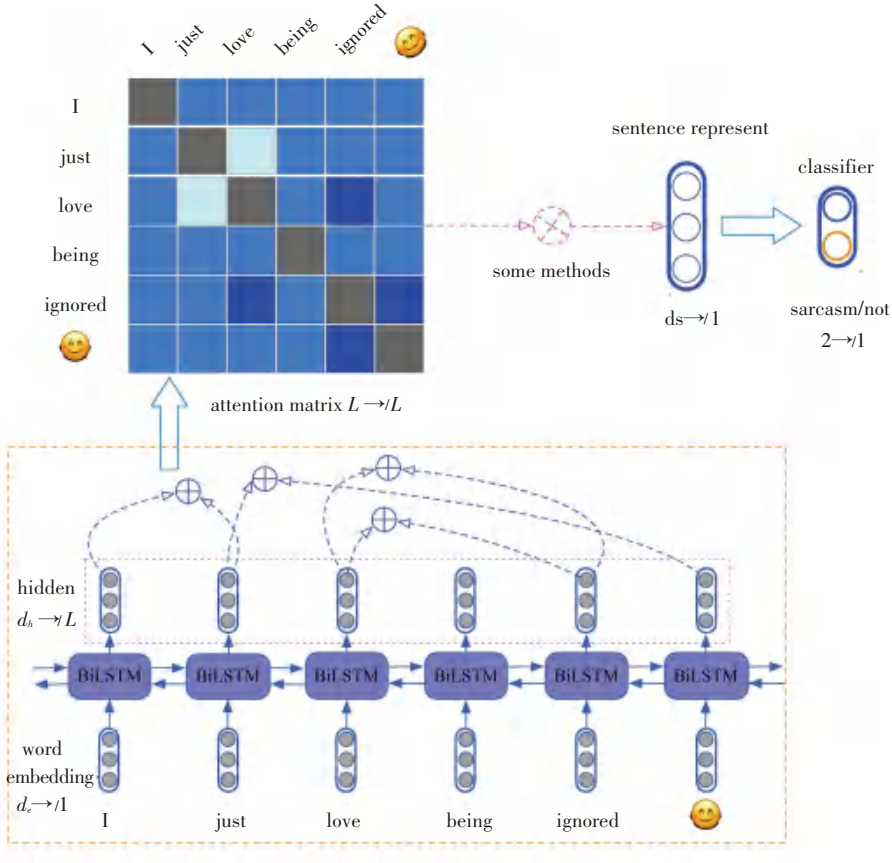

图 1　词对矛盾模型

Fig. 1　Word pairs clash model

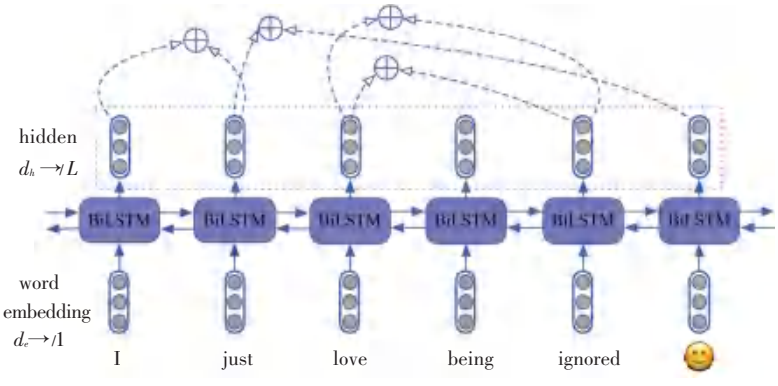

图 2　Word pairs attention 计算

Fig. 2　Calculate word pairs attention

显然,一句话中任意两个词做 attention 操作可得到 $L \times L$ 个注意力分数,这里需要注意的是,其中一词与该词本身的注意力分数手动设置为 0,考虑到词与自身不可能存在词对矛盾关系,因此无需计算注意力分数。

2.1.2 注意力分数矩阵的池化处理

由 2.1.1 节研究得到了一个 $L \times L$ 的注意力分数矩阵,接下来考虑如何利用该矩阵。一种最简单的思路是借鉴计算机视觉中的卷积神经网络(CNN or ConvNet)的池化(Pooling)技术。Pooling 操作常用于 CNN 网络中,是对卷积操作后产生的特征图(feature map)的一种降维操作,常用的有最大化池化(max pooling)、平均化池化(average pooling)等。Pooling 操作可以极大地减少参数数量和计算量,减小内存消耗,保持平移不变性,增大感受视野。对于二维 $L \times L$ 注意力分数矩阵,可以使用 Pooling 技术将其降为 $L \times 1$ 的一维向量。这里研究采用的是 max pooling 技术,因为相对于 average pooling 而言,max pooling 更适合捕捉矛盾词对,比如对于"I just ***love*** being ***ignored*** 🙂"来说,"I"与某个词的注意力分数越大,可以认为"I"与该词相对于其他词而言更具有矛盾性。max pooling 技术,即对于每一行的 attention 值取其最大作为该行的 attention。从直观上来讲,研究只关心与当前词最相关的另一个词,这就是使用 max pooling 的原因。具体见图 3。

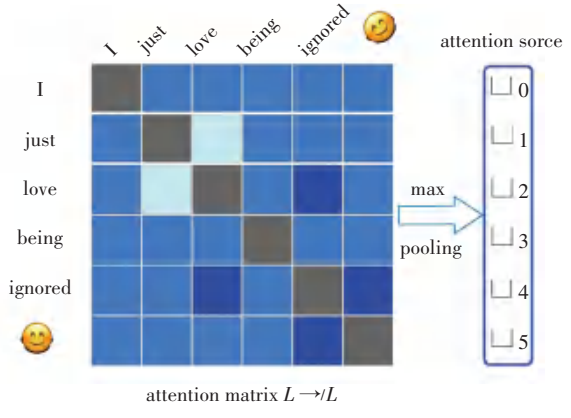

图 3 按行取 max pooling

Fig. 3 Max pooling by row

由图 3 可知,对任意的两词做注意力操作,从而得到了注意力分数矩阵(attention score matrix)。在该矩阵中,每一行取最大的分数,这样即可得到 $L \times 1$ 的一维向量,接下去还要使用 softmax 函数做归一化处理,如此一来就可以得到注意力分数的概率形式,具体公式如下:

$$\boldsymbol{a} = softmax(\max_{row} \boldsymbol{s}) , \tag{2}$$

其中,$\boldsymbol{s}$ 为注意力分数矩阵,$\boldsymbol{a} \in R^L$。

由此,研究得到了注意力分数向量 $\boldsymbol{a} \in R^L$,与原始的 BiLSTM 隐层向量相乘即可得到本句的向量表示,即:

$$\boldsymbol{v}_a = \sum_{i=0}^{L-1} \boldsymbol{w}_i \, a_i . \tag{3}$$

其中,L 为句子总长度,标量 a_i 表示输入句中第 $i(0 \leqslant i < L)$ 个词的注意力分数,$w_i \in R^{d_h}$ 为第 i 个词的隐层向量。v_a 可以理解为每个词的隐层进行加权(注意力分数)求和。综上研究过程即如图 4 所示。

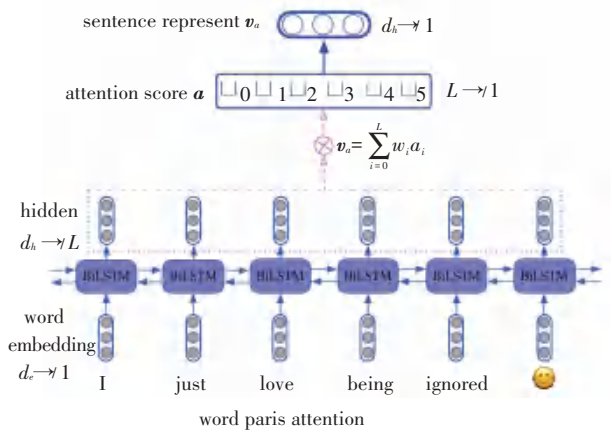

图 4 句子表示

Fig. 4 Sentence representation

2.1.3 注意力分数矩阵的二次 attention 处理

在 2.1.2 节中使用了 max pooling 技术将 $L \times L$ 注意力分数矩阵降为 $L \times 1$ 的一维向量,在本节将使用另一种方式来处理注意力分数矩阵,如图 5 所示。

由图 5 可知,$L \times L$ 注意力分数矩阵 $\boldsymbol{s}$ 中的第 i 行($0 £ i < L$)是 $L \times 1$ 的向量,可以看作是以句中第 i 个词为参考点的注意力分数向量,对 $L \times 1$ 的向量做 softmax 概率归一化处理后,L 个 $L \times 1$ 的向量可计算得到 L 个句子 attenion 向量表示,即:

$$\boldsymbol{v}_{sa}^{\mathrm{T}} = \boldsymbol{v}_{hs} softmax(\boldsymbol{s}) , \tag{4}$$

其中,$\boldsymbol{s} \in R^{L \times L}$ 是二维注意力分数矩阵;$\boldsymbol{v}_{hs} \in R^{d_h \times L}$ 是 BiLSTM 的隐层向量组成的矩阵;$\boldsymbol{v}_{sa} \in R^{L \times d_h}$ 是 L 个句子 attenion 向量表示。

然后将 $\boldsymbol{v}_{sa}^{\mathrm{T}}$ 使用 self attention 机制(使用感知机算法加 tanh 激活函数),计算出二次 attention 的注意力分数 $\boldsymbol{s}_a$,可将其转为 $d_h \times 1$ 的二次 attention 向量表示 $\boldsymbol{v}_{saa}$。至此,可推得 $\boldsymbol{v}_{saa}$ 计算公式为:

$$\boldsymbol{s}_a = softmax(\boldsymbol{\omega}^{\mathrm{T}} \tanh(\boldsymbol{W}_{aa} \boldsymbol{v}_{sa}^{\mathrm{T}})) , \tag{5}$$

$$\boldsymbol{v}_{saa} = \boldsymbol{v}_{hs} \boldsymbol{s}_a^{\mathrm{T}} . \tag{6}$$

其中，$\boldsymbol{W}_{aa} \in R^{d_h \times d_h}$，$\boldsymbol{\omega} \in R^{d_h}$ 是权重矩阵；$\boldsymbol{s}_a \in R^{1 \times L}$ 是二次 attention 的注意力分数；$\boldsymbol{v}_{saa} \in R^{d_h \times 1}$ 就是经过二次 attention 后的句子向量表示。

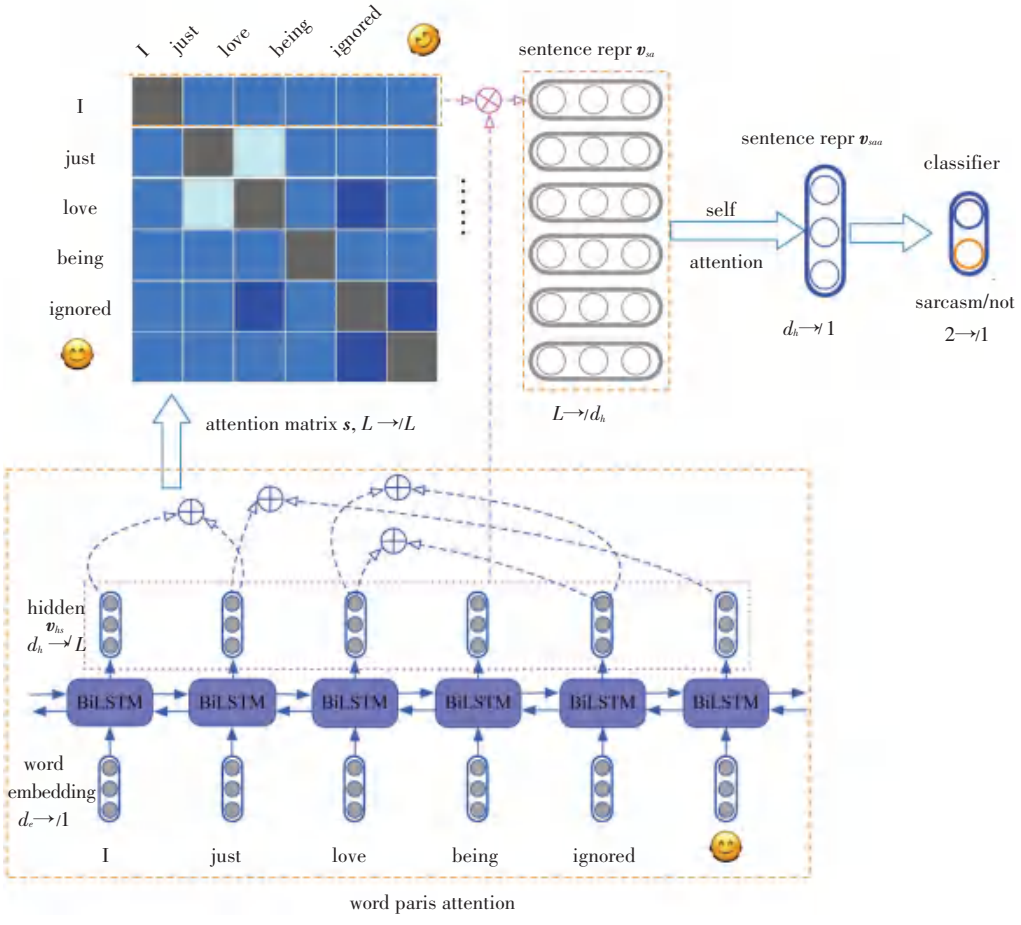

图 5　注意力分数矩阵的二次 attention 处理

Fig. 5　Attention matrix attented sencondly

2.1.4　句子向量的分类

研究使用了 2 种方式（见 2.1.2 节，2.1.3 节）获得句子表示。前者使用 max pooling 生成句子表示 $\boldsymbol{v}_a$，后者使用二次 attention 生成句子表示 $\boldsymbol{v}_{saa}$，后续将分别用这两种表示进行分类任务，见图 6。

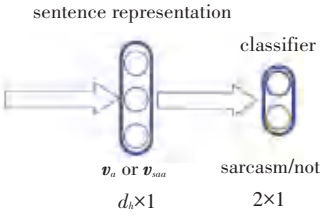

图 6　句子表示做分类

Fig. 6　Classify by sentence representation

通过线性变换将 $\boldsymbol{v}_a$ 或 $\boldsymbol{v}_{saa}$ 映射为二维向量，而后使用 *softmax* 进行概率归一化处理，得到相应标签的置信度。其公式为：

$$\hat{y} = softmax(\boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{v}_a + b)\ ,$$

或者

$$\hat{y} = softmax(\boldsymbol{\omega}^{\mathrm{T}} \boldsymbol{v}_{saa} + b)\ . \tag{7}$$

其中，$\hat{y}$ 为网络的输出标签（置信度最大的类别）。

因为该模型是端到端（End-to-End）训练的，就使得交叉熵（Cross Entropy）损失函数或者对数似然（log-likelihood）损失函数可以用作训练时的优化目标（两个函数在二分类情况下具有一致性），即：

$$J(\theta) = -\sum_{i=1}^{N} [y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)\ ] + \frac{\lambda}{2} |\theta|^2. \tag{8}$$

其中，$J(\theta)$ 为损失函数；$\hat{y}$ 为网络的标签输出；y 为真实标签；$\frac{\lambda}{2} |\theta|^2$ 为 L_2 正则项。

2.2　结合 LSTM 的词对矛盾模型

2.1 节中基于词对注意力得到了句子表示，该

句子表示含有矛盾词对信息,这些信息则是判断反讽的重要特征。此外,还应利用原始的序列信息,在这里使用了 BiLSTM 做句子的序列表示,并将该表示与 2.1 节中的句子表示组合作为新的句子表示。因此用作句子分类的句子向量由 WPA-P / WPA-A 和 BiLSTM 的句子表示组成。这样前者可以发现句子内的矛盾词, 比如例句中的 ***love*** 与 ***ignored***;后者可以表征句子的序列信息, 如原始的上下文信息等。

BiLSTM 的句子表示可以看作是普通句子序列化的一种建模表示,模型如图 7 所示。

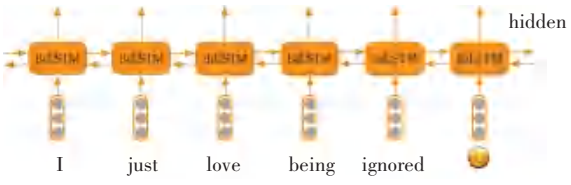

图 7 BiLSTM 做句子表示

Fig. 7 BiLSTM sentence representation

BiLSTM 句子表示用最后一个隐层输出表示该句的语义信息。该部分的输入为 word embedding, 输出为各个词的隐层向量,这里取隐层中最后一个词的隐层作为 BiLSTM 句子表示。

将 BiLSTM 句子表示 v_{hlast} 和 WPA-P/WPA-A 句子表示拼接得到最终的句子向量表示 $\boldsymbol{v}_c \in R^{2d\times 1}$, 以此可以做分类预测,这里是二分类任务,正例是反讽,负例是非反讽。研究得到的 WPA-BiLSTM 模型如图 8 所示。

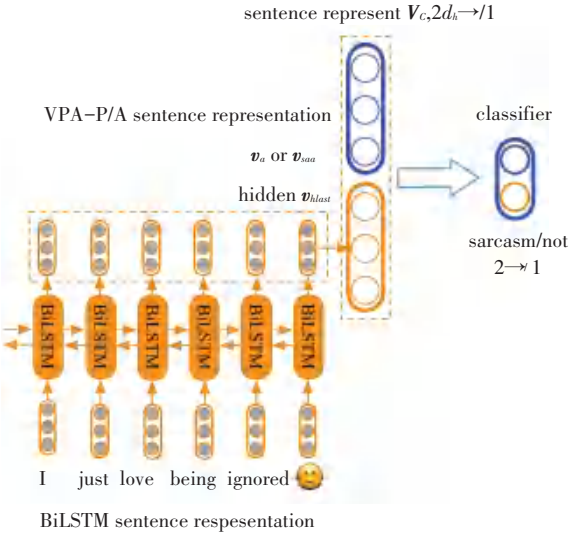

图 8 WPA-BiLSTM 模型

Fig. 8 WPA-BiLSTM model

2.3 结合 CNN 的词对矛盾模型

在 2.1 节中使用了 BiLSTM 作为原始句子的序列信息,本节将使用卷积神经网络(CNN)捕获句子的 N-gram 信息,并与 2.1 节中的 WPA-P/A 句子向量拼接作为分类器输入。与 BiLSTM 的序列建模不同,CNN 在 NLP 任务中常被认为会捕获句子的 N-gram[15] 特征,而 N-gram 特征是自然语言处理中的一项极其重要的特征,广泛应用于各项文本的分类任务中。CNN 与 WPA-P/A 结合的示意图如图 9 所示。

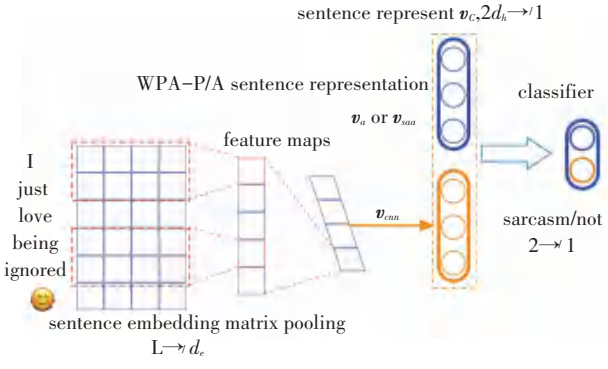

图 9 WPA-CNN 模型

Fig. 9 WPA-CNN model

3 实验

研究中,采用 5 个模型(WPA-P, WPA-A, WPA-A-BiLSTM,WPA-A-CNN,WPA-A-BiLSTM-CNN)在 3 个数据集上做了对比实验,分别是 Rilff、Ptacek、SemEval-2018,数据集规模见表 1。

表 1 数据集规模

Tab. 1 Dataset size

数据集	总和	反讽	非反讽
Riloff et al. 2013	1 650	309	1 341
Ptacek et al. 2014	50 033	21 023	29 010
SemEval-2018	4 618	2 222	2 396

研究中又采用了 3 个基线模型做对比,分别是 CNN、LSTM、Attention based LSTM,实验结果见表 2。

由表 2 可知,无论是 WPA-P 模型、还是 WPA-A 模型都要好于 3 个基线模型,WPA-A 效果要略好于 WPA-P。再加入额外信息(BiLSTM,CNN)后,效果均有提高,尤其是同时加入 BiLSTM 和 CNN 后, P,R,F_1 提升明显。最终,研究得到的 WPA-A-BiLSTM-CNN 模型的 word pairs 注意力分数如图 10 所示。

由图 10 可以看到,WPA-A-BiLSTM-CNN 中的 WPA 部分确实捕获到了“矛盾词对”,比如{love, ignored},{sore, fun}等。

表 2　实验结果

Tab. 2　Experiment results

模型	Riloff			Ptacek			SemEval-2018		
	P	R	F_1	P	R	F_1	P	R	F_1
CNN	64.01	63.98	63.99	67.01	69.30	68.14	52.40	61.03	56.39
LSTM	65.00	66.30	65.64	68.21	70.60	69.86	52.68	60.34	56.25
Attn LSTM	64.47	64.39	64.43	70.59	71.06	70.82	53.33	60.56	56.72
WPA-P	65.30	66.87	66.08	71.10	72.32	71.70	53.41	61.87	57.33
WPA-A	65.76	67.45	66.59	71.43	72.71	72.04	54.08	61.45	57.53
WPA-A-BiLSTM	66.59	68.31	67.44	72.29	73.25	72.77	54.10	61.31	57.48
WPA-A-CNN	65.98	67.98	66.97	71.33	74.08	72.68	55.23	61.98	58.41
WPA-A-BiLSTM-CNN	67.02	68.30	67.65	72.10	74.28	73.17	55.41	62.00	58.52

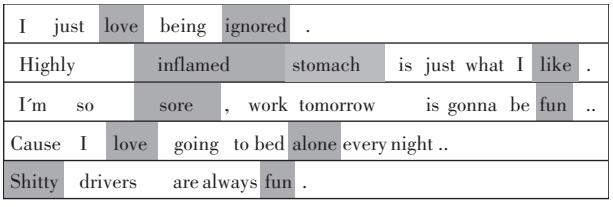

图 10　WPA-A-BiLSTM-CNN 模型的 word pairs 注意力分数

Fig. 10　word pairs attention score of WPA-A-BiLSTM-CNN model

4　结束语

反讽修辞方法在社交媒体中应用广泛，这同时给情感分析和观点挖掘等带来了挑战。针对前后矛盾形式的反讽修辞，本文提出了一种 word pairs attention 模型(WPA)，其主要思想为计算句中任意两词的注意力分数，这样可以助推模型在训练中着重关注某重点词对，因此该模型可以捕捉文本中的前后矛盾词对，也正是该词对是导致反讽的重要原因。除此之外，还使用了 BiLSTM 来做句子的序列表示，使用 CNN 提取句子 N-gram 特征，实验证明，WPA 与 BiLSTM 或 CNN 结合可以提升模型的整体性能。

参考文献

[1] JOSHI A, BHATTACHARYYA P, CARMAN M J. Automatic sarcasm detection: A survey[J]. ACM Computing Surveys (CSUR), 2017, 50(5): 1.

[2] Van HEE C, LEFEVER E, HOSTE V. SemEval-2018 task 3: Irony detection in English Tweets[C]//Proceedings of The 12th International Workshop on Semantic Evaluation. New Orleans, Louisiana: Association for Computational Linguistics, 2018: 39.

[3] TSUR O, DAVIDOV D, RAPPOPORT A. Semi-supervised recognition of sarcastic sentences in Twitter and Amazon[C]// Proceeding of the Conference on Computational Linguistics. Uppsala, Sweden: ACL, 2010 : 107.

[4] GONZALEZ-IBANEZ R, MURESAN S, WACHOLDER N. Identifying sarcasm in Twitter: A closer look[M]//LIN D, MATSUMOTO Y, MIHALCEA R. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers. Portland: Association for Computational Linguistics, 2011, 2: 581

[5] BAMMAN D, SMITH N A. Contextualized sarcasm detection on Twitter[C]//Proceedings of the 9th International Conference on Web and Social Media (ICWSM'15). Oxford, UK: AAAI, 2015: 574.

[6] ZHANG Meishan, ZHANG Yue, FU Guohong. Tweet sarcasm detection using Deep Neural Network[C]//26th International Conference on Computational Linguistics (COLING 2016). Osaka, Japan: dblp, 2016 : 2449.

[7] CHEN T, XU R, HE Y, et al. Learning user and product distributed representations using a sequence model for sentiment analysis[J]. IEEE Computational Intelligence Magazine, 2016, 11(3): 34.

[8] GUI L, ZHOU Y, XU R F, et al. Learning representations from heterogeneous network for sentiment classification of product reviews[J]. Knowledge-Based Systems, 2017, 124: 34.

[9] GHOSH A, VEALE D T. Fracking sarcasm using neural network[C]//Proceedings of the 7th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis. San Diego, California: Association for Computational Linguistics, 2016: 161.

[10] BAHDANAU D, CHO K, BENGIO Y. Neural machine translation by jointly learning to align and translate[J]. arXiv preprint arXiv: 1409.0473, 2014.

[11] CHUNG J, GULCEHRE C, CHO K, et al. Empirical evaluation of gated recurrent neural networks on sequence modeling[J]. arXiv preprint arXiv: 1412.3555, 2014.

[12] GERS F A, SCHMIDHUBER J, CUMMINS F. Learning to forget: Continual prediction with LSTM[J]. Neural Computation, 2000, 12(10): 2451.

[13] KIM Y. Convolutional neural networks for sentence classification[J]. arXiv preprint arXiv: 1408.5882 , 2014.

[14] LIN Zhouhan, FENG Minwei, SANTOS N D S, et al. A structured self-attentive sentence embedding[J]. arXiv preprint arXiv: 1703.03130, 2017.

[15] TRIPATHY A, AGRAWAL A, RATH S K. Classification of sentiment reviews using n-gram machine learning approach[J]. Expert Systems with Applications, 2016, 57: 117.