计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.190100097

一种基于自注意力的句子情感分类方法

余珊珊[1]　苏锦钿[2]　李鹏飞[2]
1 广东药科大学医药信息工程学院　广州 510006
2 华南理工大学计算机科学与工程学院　广州 510640
(susyu@139.com)

摘　要　注意力机制近年来在多个自然语言任务中得到广泛应用,但在句子级别的情感分类任务中仍缺乏相应的研究。文中利用自注意力在学习句子中重要局部特征方面的优势,结合长短期记忆网络(Long Short-Term Model,LSTM),提出了一种基于注意力机制的神经网络模型(Attentional LSTM,AttLSTM),并将其应用于句子的情感分类。AttLSTM首先通过LSTM学习句子中词的上文信息;接着利用自注意力函数从句子中学习词的位置信息,并构造相应的位置权重向量矩阵;然后通过加权平均得到句子的最终语义表示;最后利用多层感知器进行分类和输出。实验结果表明,AttLSTM在公开的二元情感分类语料库Movie Reviews(MR),Stanford Sentiment Treebank(SSTb2)和Internet Movie Database(IMDB)上的准确率最高,分别为82.8%,88.3%和91.3%;在多元情感分类语料库SSTb5上取得50.6%的准确率。

关键词:深度学习;情感分类;自注意力;长短期记忆神经网络;自然语言处理
中图法分类号　TP183

Sentiment Classification Method for Sentences via Self-attention

YU Shan-shan[1],SU Jin-dian[2] and LI Peng-fei[2]
1 College of Medical Information Engineering,Guangdong Pharmaceutical University,Guangzhou 510006,China
2 College of Computer Science and Engineering,South China University of Technology,Guangzhou 510640,China

Abstract　Although attention mechanisms are widely used in many natural language processing tasks,there still lacks of related works about its applications in sentence-level sentiment classification. By taking advantage of self-attention mechanism in learning important local features of sentences,a multi-layer attentional neural network based on long-short term memory network (LSTM) and attention mechanism,named AttLSTM,was proposed and then applied into the fields of sentiment classification for sentences. AttLSTM firstly uses LSTM network to capture the contexts of sentences,and then takes self-attention functions to learn the position information about words in the sentences and builds the corresponding position weight matrix,which yields the final semantic representations of the sentences by weighted averaging. Finally,the results is classified and outputted via a multi-layer perceptron. The experiment results show that AttLSTM outperforms some relative works and achieves the highest accuracy of 82.8%,88.3% and 91.3% respectively on open two-class sentiment classification corpora,including Movie Reviews (MR),Stanford Sentiment Treebank (SSTb2) and Internet Movie Database (IMDB),as well as 50.6% for multi-class classification corpora SSTb5.

Keywords　Deep learning,Sentiment classification,Self-attention,Long-short term memory,Natural language processing

1　引言

句子情感分类一直是自然语言处理(Natural Language Processing,NLP)中的一个重要子任务和研究热点。近年来,不少学者利用深度学习中的长短期记忆模型在这方面展开研究,并在多个情感分类语料库上取得了最优的效果。例如,Wang[1]将LSTM应用于Twitter数据的二元情感分类预测,并在Stanford Twitter Sentiment(STS)语料库上取得了

到稿日期:2019-01-13　返修日期:2019-05-23　本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
基金项目:广东省自然科学基金(2015A030310318);广东省科技厅应用型科技研发专项资金(20168010124010);广东省医学科学技术研究基金项目(A2015065)
This work was supported by the Natural Science Foundation of Guangdong Province (2015A030310318),Applied Scientific and Technology Special Project of Department of Science and Technology of Guangdong Province (20168010124010),and Medical Scientific Research Foundation of Guangdong Province(A2015065)
通信作者:苏锦钿(SuJD@scut.edu.cn)

87.2%的准确率；Tai 等[2]将传统基于线性结构的 LSTM 扩展为基于树结构网络拓扑的 LSTM，提出了一致性 Tree-LSTM 模型，并通过实验得出该模型在 Stanford Sentiment Treebank(SSTb)语料库[3]的二元和五元分类上取得了 88.0%和 51.0%的准确率。这些研究一般将句子的语义表征为 LSTM 中最后一个时间步的隐藏状态，其优点是可通过保留句子中元素的长距离依赖关系刻画句子的语义，但无法体现不同元素对句子情感的不同影响程度。正如 Bahdanau 等[4]和 Ling 等[5]所指出，不同元素对上下文表示的贡献或结果的影响是不同的。在句子情感分类任务中，情感类的词汇对情感极性的影响力显然要比其他一般的单词大。例如，对于极性为 positive 的句子“I like natural language processing”，单词“like”对结果的影响程度显然更大。

为了更好地捕获模型在学习过程中的不同关注点，近 3 年来一些学者将图形图像中的注意力(Attention)机制引入 NLP，特别是结合经典的 Encoder-Decoder 框架提出了一系列的注意力模型(Attention model，AM)，如 Soft AM[4]，Global AM[6]和层次化 AM[7]等，并将其成功地应用于机器翻译[4,6]、文本摘要[8]、关系抽取[9]、阅读理解[10]和文本蕴含[11]等任务中。注意力机制使得 Decoder 端在生成目标序列时可综合考虑 Encoder 端的所有输出序列信息，而不是生成一个固定大小的中间语义向量。这不仅可以更好地体现输入源中各元素对目标结果的不同影响力，还可以减少由于句子过长而导致细节信息丢失的问题。

在机器翻译、词性标注和自动摘要等任务中，注意力向量的计算主要依赖于 Decoder 端的当前输入与 Encoder 端的输出序列之间的对齐关系。但对于句子情感分类任务来说，Decoder 端的当前输入也同样来自于 Encoder 端的输出序列，即注意力向量的计算完全依赖于 Encoder 端的输出。针对这一特点，Cheng[10]等在 Soft AM[4]和 Global[6]等研究的基础上进一步提出自注意力(Self-attention)机制，也称内注意力(Intra-attention)，其主要思路是在表征句子语义的同时考虑句子中元素的位置信息，即注意力向量的计算是依赖句子中各元素的位置信息，而不是元素间的对齐关系。随后，Parikh 等[11]在研究自然语言推断时将自注意力和 Soft AM 相结合，利用自注意提取句子中词之间的组合关系，最后证明了所提模型在 Stanford Natural Language Inference(SNLI)数据集上取得了最优效果。Paulus 等[8]在研究生成式自动摘要时分别将自注意力机制应用于 Encoder 端和 Decoder 端，取得了很好的效果。Yang 等[7]结合乘法自注意力提出了针对文档分类的层次化注意力机制，并将其分别应用于单词级别和句子级别的语义表征。Lin 等[12]利用多个注意力向量从句子中学习不同的局限特征，并通过惩罚项强化不同注意力向量间的分布。这些研究均证明了自注意力机制的有效性，但其不适合直接应用于句子的情感分类任务。Shen 等[13]提出一种基于有向和多维自注意力的神经网络用于学习句子的语义表示，并证明了该模型在不依赖循环神经网络或卷积神经网络的情况下仍然在 SSTb5 上取得了 51.72%的准确率。但该模型较为复杂，需要综合使用多种复杂的注意力机制。

本文在上述研究的基础上将自注意力与 LSTM 相结合，提出了一个针对句子情感分类的自注意力神经网络模型 AttLSTM，并通过公开的语料库进行验证和分析。本文第 2 节给出模型的总体结构及各层的实现细节；第 3 节描述模型的训练；第 4 节通过实验对模型进行验证和分析；最后总结并给出下一步的工作方向。

2 AttLSTM 模型

给定一个输入的句子 x，AttLSTM 模型可表示为一个函数 $f(\{y\},x)$，其中 $\{y\}=y_1,\cdots,y_m$ 为情感的极性类别集合，即模型的目的是计算 x 在每一个类别 $y_i(1\leqslant i\leqslant m)$ 上的似然概率。

2.1 总体结构

AttLSTM 模型的总体结构如图 1 所示。

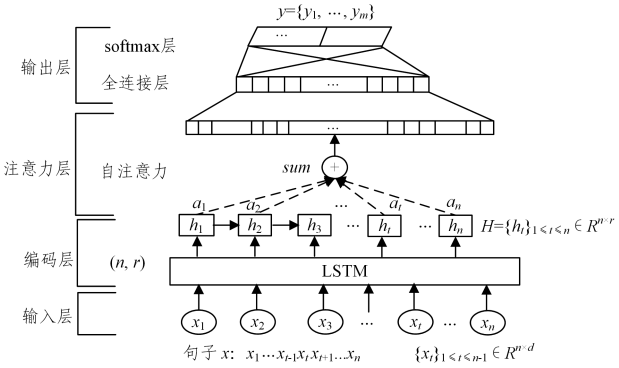

图 1 AttLSTM 模型的总体结构

Fig. 1 Architecture of AttLSTM model

AttLSTM 包括 4 个主要组成部分。

(1)输入层(Input)：主要对句子进行分词、符号过滤和补齐等操作，并将句子的符号表示转换为词向量的表示。

(2)Encoder 层(Encoder)：一个基于 LSTM 的编码器，依次读入句子中各个单词，并输出每一个时间步的词向量。

(3)注意力层(Attention)：一个基于自注意力机制的网络层，主要通过自注意力学习句子的重要局部特征，并对编码器层的输出进行加权求和形成新的句子语义表示。

(4)输出层(Output)：一个多层感知器，由一个全连接隐层和一个全连接 softmax 输出层构成，用于根据目标情感极性集合对结果进行预测和输出。

2.2 输入层

假设输入句子 x 共包含 n 个词，表示为 $x=x_1\oplus\cdots\oplus x_i\oplus\cdots\oplus x_k$，其中 $x_i(1\leqslant i\leqslant k)$ 为 x 中第 i 个词，$\oplus$ 为词之间的串联操作。简单起见，文中直接用 $x_{1:k}$ 表示句子 x。为了便于统一处理，需要对句子的长度进行补齐。假设句子长度阈值为 n(n 通常取训练语料库中的最大句子长度，也可指定某一具体值)，对于长度超过 n 的句子，只截取前 n 个词；对于长度少于 n 的句子，用特定标志(如〈PAD/〉)进行补齐。以句子 $x_{1:k}$ 为例，相应的补齐操作为：

$$x_{1:k}=\begin{cases}x_1\oplus\cdots\oplus x_n, & k\geqslant n\\ x_1\oplus\cdots\oplus x_k\oplus\{\langle\text{PAD}/\rangle\}_{n-k}, & k<n\end{cases}\tag{1}$$

每个词 $x_i(1\leqslant i\leqslant n)$ 被映射为一个连续且带语义信息的

低维稠密实数向量。假设 $\boldsymbol{E}\in R^{d\times|V|}$ 为一个预训练或随机初始化的实数词向量表，其中 $|\boldsymbol{V}|$ 表示词汇表 $\boldsymbol{V}$ 的大小，d 为词向量的维度(如 50，100 或 300 维)。$\boldsymbol{V}(x_i)$ 为词 x_i 在 $\boldsymbol{E}$ 中的位置向量，即对应 x_i 的位置为 1，其他位置均为 0。则利用 $\boldsymbol{E}$ 和矩阵向量积可将句子 $x_{1:n}$ 中的每一个词 x_i 转化为 $\boldsymbol{E}$ 中的一个 $1\times d$ 维向量 x_i^d：

$$x_i^d=\boldsymbol{V}(x_i)\times\boldsymbol{E} \tag{2}$$

由式(2)可将每一个句子 $x_{1:n}$ 转换成一个 $n\times d$ 维的实数向量矩阵表示 $\boldsymbol{x}^d=\{\boldsymbol{x}_1{}^d,\boldsymbol{x}_2{}^d,\cdots,\boldsymbol{x}_n^d\}$，或简记为 $x_{1:n}^d$。若训练集 $\boldsymbol{T}$ 中共包含 N 个句子，则由式(1)和式(2)可将 $\boldsymbol{T}$ 转换为一个 $N\times n\times d$ 的三维向量矩阵 $\boldsymbol{T}^d$。若句子中的词在 $\boldsymbol{E}$ 中不存在，则利用区间[−0.25，0.25]上的正态分布进行随机初始化。补齐标志〈PAD/〉表示 d 维全 0 向量，并在模型训练过程中自动忽略。

2.3 编码层

编码层利用一个正向的 LSTM 网络对句子进行编码，并学习句子中各个单词的长距离依赖关系，其结构如图 2 所示。

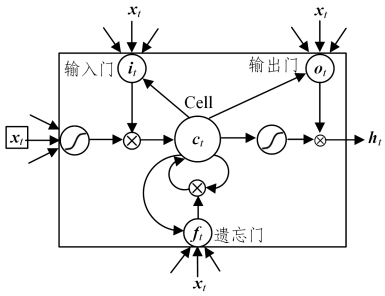

图 2 LSTM 的组成部分

Fig. 2 Components of LSTM

给定句子 $x_{1:n}$；t 为当前时间步($1\leqslant t\leqslant n$)；$\boldsymbol{c}_t$ 表示在输入 $\boldsymbol{x}_t$ 下所对应的神经元隐藏状态；$\boldsymbol{h}_{t-1}$ 为上一时刻的输出值；$\boldsymbol{o}_t$ 表示当前输入 $\boldsymbol{x}_t$ 所对应的输出；$\boldsymbol{W}_i$，$\boldsymbol{W}_f$，$\boldsymbol{W}_c$，$\boldsymbol{W}_o$，$\boldsymbol{U}_i$，$\boldsymbol{U}_f$，$\boldsymbol{U}_c$，$\boldsymbol{U}_o$ 和 $\boldsymbol{V}_o$ 均为相应的权重矩阵；$\boldsymbol{b}_i$，$\boldsymbol{b}_c$，$\boldsymbol{b}_f$ 和 $\boldsymbol{b}_o$ 为偏置向量；σ 为激活函数，一般为 sigmoid。则 LSTM 网络的计算过程如下：

$$\boldsymbol{i}_t=\sigma(\boldsymbol{W}_i\boldsymbol{x}_t+\boldsymbol{U}_i\boldsymbol{h}_{t-1}+\boldsymbol{b}_i) \tag{3}$$

$$\boldsymbol{c}_t=\tanh(\boldsymbol{W}_c\boldsymbol{x}_t+\boldsymbol{U}_c\boldsymbol{h}_{t-1}+\boldsymbol{b}_c) \tag{4}$$

$$\boldsymbol{f}_t=\sigma(\boldsymbol{W}_f\boldsymbol{x}_t+\boldsymbol{U}_f\boldsymbol{h}_{t-1}+\boldsymbol{b}_f) \tag{5}$$

$$\boldsymbol{c}_t=\boldsymbol{i}_t\otimes\boldsymbol{c}_t+\boldsymbol{f}_t\otimes\boldsymbol{c}_{t-1} \tag{6}$$

$$\boldsymbol{o}_t=\sigma(\boldsymbol{W}_o\boldsymbol{x}_t+\boldsymbol{U}_o\boldsymbol{h}_{t-1}+\boldsymbol{V}_o\boldsymbol{c}_t+\boldsymbol{b}_o) \tag{7}$$

$$\boldsymbol{h}_t=\boldsymbol{o}_t\otimes\boldsymbol{c}_t \tag{8}$$

显然，LSTM 的各时间步输出 $\boldsymbol{h}_t$ 不仅与当前的输入 $\boldsymbol{x}_t$ 有关，而且与前一阶段的隐藏状态 $\boldsymbol{c}_{t-1}$ 有关。假设 LSTM 网络的输出维度为 r，即对于每一个输入 $\boldsymbol{x}_t^d$，对应 $\boldsymbol{h}_t$ 的词向量维度为 r，则由 LSTM 网络可得到相应的 $1\times n\times r$ 矩阵 $\boldsymbol{H}=\{\boldsymbol{h}_t\}_{1\leqslant t\leqslant n}$，其中 $\boldsymbol{h}_t\in R^r$。

2.4 注意力层

注意力层的主要目的是学习每个位置上的权重值，使得句子中越重要的词所对应的注意力值越大，从而突出重要词对结果的影响。首先，利用自注意力计算句子中各个词的位置权重值，并构成注意力向量；接着，对编码层的输出矩阵进行加权求和，构成新的句子语义表示。

具体来说，对于每一个输入句子 $x\in R^{n\times d}$ 及相应的编码层输出 $\boldsymbol{H}\in R^{1\times n\times r}$，注意力向量 $\boldsymbol{\alpha}=\{\boldsymbol{\alpha}_t\}(1\leqslant t\leqslant n)$ 及句子的最终语义表示 $\boldsymbol{h}$ 的计算公式如下：

$$\boldsymbol{h}=\tanh(\boldsymbol{W}_t\boldsymbol{h}'+\boldsymbol{W}_n\boldsymbol{h}_n) \tag{9}$$

$$\boldsymbol{h}'=\sum_{t=1}^{n}\boldsymbol{\alpha}_t\times\boldsymbol{h}_t \tag{10}$$

$$\boldsymbol{\alpha}_t=\mathrm{softmax}(\boldsymbol{e}_t)=\frac{\exp(\boldsymbol{e}_t)}{\sum_{j=1}^{n}\exp(\boldsymbol{e}_j)+10^{-8}} \tag{11}$$

$$\boldsymbol{e}_t=\boldsymbol{V}^{\mathrm{T}}\tanh(\boldsymbol{W}_a\boldsymbol{h}_t+\boldsymbol{b}_a) \tag{12}$$

其中，$\boldsymbol{W}_t\in R^{n\times r}$，$\boldsymbol{W}_n\in R^{n\times r}$ 和 $\boldsymbol{W}_a\in R^{r\times r}$ 均为二维权重矩阵；$\boldsymbol{b}_a\in R^r$ 为偏置值向量；$\boldsymbol{\alpha}_t\in R^n$ 表示对 Encoder 端输出序列中第 t 个位置的注意力权重进行归一化后的结果；$\boldsymbol{e}_t\in R^n$ 利用一个隐藏层的全连接前向网络计算注意力的分配，然后利用神经元激活函数 tanh 将结果约束在区间[−1，1]上，最后通过二维矩阵 $\boldsymbol{M}\in R^{1\times r}$ 进一步提取特征。为了避免在对 $\boldsymbol{e}_t$ 进行归一化的过程中出现分母全为 0 而导致计算溢出的问题，本文将分母加一个非常小的值 10^{-8}；$\boldsymbol{h}'\in R^r$ 为利用注意力向量 $\boldsymbol{\alpha}_t$ 对 Encoder 端输出序列进行加权平均后所得到的向量。在 AttLSTM 中，我们没有将 $\boldsymbol{h}'$ 直接作为句子的最终语义表示，而是通过一个全连接隐层对 $\boldsymbol{h}'$ 和 $\boldsymbol{h}_n$ 进行学习，并将经过非线性激活函数 tanh 所得到 $\boldsymbol{h}$ 作为句子的最终语义表示。

2.5 输出层

输出层利用一个全连接隐层和一个全连接 softmax 输出层对注意力层的输出结果 $\boldsymbol{h}$ 进行分类预测。给定目标分类标签 $y=\{y_1,y_2,\cdots,y_m\}$，当 $m=2$ 时，y 为经典的二元分类(如 Positive/ Negative)，即基于 sigmoid 的逻辑回归；当 $m>2$，y 时为多元逻辑回归分类。

AttLSTM 的目标是计算每一个输入句子 x 在各个目标极性类别上的似然概率分布，具体可描述为：

$$f:x\rightarrow y=\{y_1,y_2,\cdots,y_m\} \tag{13}$$

假设符号 θ 表示 AttLSTM 模型中的全部参数，则对于给定的 x 及 θ，输出层将结果转换成集合 y 中各元素的条件概率分布 $P(y|x,\theta)$。给定训练集 $\boldsymbol{T}=\{(x^{(i)},t^{(i)},y^{(i)})\mid 1\leqslant i\leqslant|\boldsymbol{T}|\}$ 及 $y=\{y_1,y_2,\cdots,y_m\}$，假设 $y^{(i)}$ 为模型对输入 $x^{(i)}$ 的预测结果标签值，$t^{(i)}$ 为真实的结果，则 $f(x^{(i)},\theta)$ 针对每一个 y_j $(1\leqslant j\leqslant m)$ 分别估算其概率值 $P(y_j|x^{(i)},\theta)$，并输出一个归一化后的 m 维向量来表示模型在这 m 个标签值上的预测概率分布：

$$f(x^{(i)},\theta)=\frac{1}{\sum_{j=1}^{m}\exp(P(y_j|x^{(i)},\theta))}\begin{bmatrix}\exp(P(y_1|x^{(i)},\theta))\\ \exp(P(y_2|x^{(i)},\theta))\\ \vdots\\ \exp(P(y_m|x^{(i)},\theta))\end{bmatrix} \tag{14}$$

其中，等号右侧第一部分是对概率分布进行归一化。取该 m 维向量中值最大的 $\max(f(x^{(i)},\theta))$ 所对应的标签作为模型的最终预测结果，即：

$$y^{(i)}=\max(f(x^{(i)},\theta)) \tag{15}$$

3 模型训练

在 AttLSTM 模型中，采用交叉熵作为损失函数。对于

训练集 $\boldsymbol{T}=\{(x^{(i)},t^{(i)},y^{(i)})\mid 1\leqslant i\leqslant|\boldsymbol{T}|\}$ 中任意一个句子 x，其 softmax 损失函数的定义如下：

$$J(x^{(i)},\theta)=\begin{cases}-t^{(i)}\ln y^{(i)}-(1-t^{(i)})\ln(1-y^{(i)}), & m=2\\ -\sum_{i=1}^{m}(t^{(i)}\ln y^{(i)}+(1-t^{(i)})\ln(1-y^{(i)})), & m>2\end{cases} \tag{16}$$

整个训练集 $\boldsymbol{T}$ 的损失函数 $L(\boldsymbol{T},\theta)$ 定义为：

$$L(\boldsymbol{T},\theta)=\frac{1}{|\boldsymbol{T}|}\sum_{i=1}^{|\boldsymbol{T}|}J(x^{(i)},\theta) \tag{17}$$

在损失函数的基础上，模型通过迭代求解损失值和梯度下降来优化该问题，以使损失函数的值最小。为了提高效率，模型采用 mini-batch 的方式。每一批为一个 mini-batch，其数量(batch_size)K 一般远小于 $|\boldsymbol{T}|$，此时的损失函数为：

$$L(x^{(i:i+K)},\theta)=\frac{1}{K}\sum_{j=i}^{i+K}J(x^{(j)},\theta) \tag{18}$$

AttLSTM 采用基于 RMSProp 的优化器，其学习率 lr 为 0.001。为防止过拟合(Over-fitting)，模型采用基于 dropout[14] 的正则化策略，并分别将其应用在输入层和编码层的 LSTM 网络。对应的 dropout 值以及其他部分关键参数如表 1 所列。

表 1 AttLSTM 中的部分关键参数

Table 1 Key parameters of AttLSTM

层	参数	值
输入层	词向量维度 d	300
	Dropout	0.2
编码层	输出维度 r	128
	Dropout	0.5
注意力层	权值维度 s	128
输出层	隐层神经元数量	128
	Dropout	0.5
	激活函数	relu
训练器	分类器	rmsprop
	损失函数	多元交叉熵

4 实验及分析

4.1 实验准备

实验采用 4 个公开的著名语料库，分别为：1) Movie Reviews 语料库(MR)，来自于电影评论网站 Rotten Tomatoes 的 10462 个电影评论句子，为二元情感分类[15]；2) Internet Movie Database 语料库(IMDB)，共含 50000 个电影评论句子，为二元情感分类[16]；3) Stanford Sentiment Treebank 二元分类语料库(SSTb2)，针对电影评论的情感预测，共含 8732 个句子[3]；4) Stanford Sentiment Treebank 五元分类语料库(SSTb5)，共含 10744 个句子[3]。

在 SSTb2 中，为减少中性句子的干扰，参照文献[3]去掉标签值在区间(0.4,0.6)中的句子，只保留区间[0,0.4]的消极句子和区间[0.6,1]的积极句子。

由于 MR 语料库中没有区分训练集和测试集，因此随机取总数据集的 80% 作为训练集，20% 作为测试集，并将训练集的 10% 作为验证集。各二元分类语料库 MR，IMDB 和 SSTb2 的训练集、测试集及验证集中的句子统计信息如表 2 所列。

表 2 各二元分类语料库的句子统计信息

Table 2 Sentences statistics information in various binary-classification corpora

语料库	训练集		测试集		验证集
	积极	消极	积极	消极	
MR	4264	4264	1067	1067	853
	共 8528		共 2134		
IMDB	12500	12500	12500	12500	2500
	共 25000		共 25000		
SSTb2	3606	3605	909	912	692
	共 6911		共 1821		

从表 2 可以看出，各二元分类语料库中积极和消极语料分布较为均衡，不存在明显的类别不均衡现象。

多元情感分类语料库 SSTb5 的训练集、测试集和验证集的相关统计信息如表 3 所列。

表 3 多元分类语料库中各集合的句子统计信息

Table 3 Sentences statistics information in datasets of various multiple classification corpus

语料库	训练集	测试集	验证集
SSTb5	8534	2210	853

各语料库中句子的相关统计信息如表 4 所列。

表 4 各语料库中的句子统计信息

Table 4 Sentences statistics information in various corpora

语料库	最大句子长度	平均长度	标准差	词汇量
MR	56	20.38	9.48	18761
IMDB	2494	234.17	173.51	100204
SSTb2	53	18.51	9.27	15476
SSTb5	53	18.38	9.27	17061

从表 4 可看出，MR，SSTb2 和 SSTb5 语料库中的句子均为短句子，因此取最大句子长度作为长度阈值。而 IMDB 语料库中大部分为长句子，若取最大句子长度 2494 为长度阈值，则会因维度太大而增加模型的训练时间。图 3 给出了 IMDB 的各个句子的长度分布情况，由于 88.6% 的句子长度都落在区间(1,300]内，因此为兼顾性能将 IMDB 语料库的最大句子长度阈值定为 300，并采用批量训练和验证的方式。

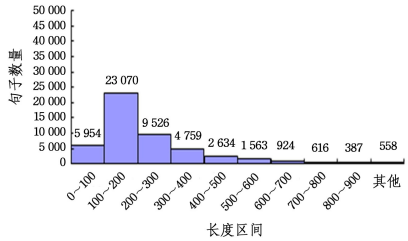

图 3 IMDB 中句子的长度分布情况

Fig. 3 Length distribution of sentences in IMDB

为便于与相关工作进行对比，实验采用以下 4 种不同的预训练词向量对输入句子进行初始化。

(1)google300[1)]：基于 word2vec 工具及 1000 亿个 Google 新闻单词训练所得的公开词向量表。

(2)glove300[2)]：基于 GloVe 算法及 Common Crawl 网页数据训练所得的公开词向量表，区分单词大小写。

1) google300: http://code.google.com/p/word2vec

2) glove300: http://nlp.stanford.edu/projects/glove/

(3)fast300[1]:基于 Facebook 的 fastText[17]算法及 Common Crawl 新闻数据训练所得的公开词向量表。

(4)em300:随机初始化,并在训练模型的过程中同时训练词向量。

所有词向量的维度 d 统一为 300。各个预训练好的词向量信息如表 5 所列。

表 5 各个预训练好的词向量信息

Table 5 Information of pre-trained word embeddings

词向量	维度 d	训练语料	符号量/十亿	词汇量/百万
google300	300	Google News	100	3
glove300	300	Common Crawl 数据,区分大小写	840	2.2
fast300	300	Common Crawl 数据	600	2

4.2 不同词向量表的比较(实验 1)

本实验的目的是比较和分析基于不同词向量的 AttLSTM 模型在各语料库上的表现。实验过程中,除使用的词向量表不同外,模型的所有参数均保持一致。实验结果如表 6 所列。

表 6 AttLSTM 模型在各词向量下的结果

Table 6 Results of AttLSTM based on various word embeddings

语料库	词向量	准确率	召回率	$F1$ 值
MR	em300	0.694	0.693	0.693
	google300	0.813	0.813	0.813
	glove300	0.826	0.826	0.826
	fast300	**0.828**	**0.827**	**0.827**
IMDB	em300	0.837	0.837	0.837
	google300	0.908	0.908	0.908
	glove300	0.907	0.907	0.907
	fast300	**0.913**	**0.913**	**0.913**
SSTb2	em300	0.801	0.801	0.801
	google300	0.879	0.879	0.879
	glove300	**0.882**	**0.882**	**0.882**
	fast300	**0.882**	**0.882**	**0.882**
SSTb5	em300	0.390	0.388	0.388
	google300	0.504	0.499	0.499
	glove300	0.494	0.494	0.494
	fast300	**0.506**	**0.500**	**0.501**

从表 6 可知:

(1)采用 em300 时,模型在各语料库上均取得一定的分类准确率,这说明直接从语料库中学习词向量的方式是可行的,且这种学习方式是无监督的,不需要依赖于其他的先验知识。当采用其他预训练好的词向量时,模型的准确率有明显的提升。这说明基于大规模语料训练所得的词向量表确实有助于提升模型在句子情感分类方面的效果,其主要原因在于预训练的词向量中包含了更丰富的语法和语义信息。

(2)在同一语料库上,相同的模型采用不同的预训练词向量时效果有一定的差别。总体来看,fast300 的准确率和 $F1$ 值最高。我们认为这一方面与训练词向量的语料数据有关,另一方面与训练词向量的算法有关。从训练的语料来看,google300 基于新闻数据,其句子语法比较严谨规范,而 glove300 和 fast300 是基于网页数据。由于各语料库中的句子主要来自于用户评论,其句子语法特点与 glove300 和 fast300 的训练数据更加接近。从 word2vec 和 GloVe 各自的训练算法来看,google300 词向量是利用 word2vec 及基于一定上下文窗口的 CBOW 模型训练所得;GloVe 算法综合考虑了全局词频共现统计信息;fastText 算法的模型架构与 word2vec 中的 CBOW 模型很类似,但增加了 N-grams 信息。从实验的结果来看,GloVe 算法在 SSTb2 语料上与 fastText 算法表现相当,但在 MR,IMDB 和 SSTb5 上不如 fastText。

4.3 与其他相关研究工作的比较(实验 2)

表 7 列出了相关工作的比较,其中第二列主要给出各模型所使用的句子表示方式及相应的特征或词向量维度等。相关工作如下:(1)N-grams+RbF[19],融合评分等级特征和 N-grams 特征的 SVM 模型;(2)Paragraph Vector[18],基于 PV-DBOW 和 PVDM 的训练模型;(3)Constituency Tree-LSTM[2],基于树结构网络拓扑的 LSTM;(4)CNN-static 和 CNN-multichannel[20],均为卷积神经网络(Convolutional Neural Network,CNN),共有 4 层,即词嵌入层、卷积层(3,4,5)、池化层(最大值池化)和输出层;(5)LSTM+MLP,由一层单向的 LSTM 和一个多层感知器组成,其中句子的语义表示为 LSTM 的最后一个隐藏状态。LSTM+MLP 模型除没有采用自注意力外,其他参数均与 AttLSTM 相同。

表 7 相关研究工作的比较

Table 7 Comparisons of related works

模型名称	句子表示及特征	MR/%	IMDB/%	SSTb2/%	SSTb5/%
N-grams+RbF[19]	bag-of-words,一元词、二元词、三元词、基于评分等级的特征	—	89.87	—	—
Paragraph Vector[18]	静态词向量和段落向量,由目标语料库训练所得,d=800	—	92.3	87.8	48.7
Constituency Tree-LSTM[2]	静态词向量,glove300,d=300	—	—	87.5	49.7
CNN-static[20]	静态词向量,google300,d=300	81.0	—	86.8	45.5
CNN-multichannel[20]	双通道词向量,google300,d=300	81.1	—	88.1	47.4
LSTM+MLP	静态词向量,google300,d=300	81.2	90.5	85.6	48.7
AttLSTM	静态词向量,google300,d=300	81.3	90.8	87.9	50.4
LSTM+MLP	静态词向量,glove300,d=300	81.8	90.2	87.0	47.6
AttLSTM	静态词向量,glove300,d=300	82.6	90.7	88.2	49.4

从表 7 可以看出:

(1)相对于 SVM 模型 N-grams+RbF,各个 AttLSTM 和 LSTM+MLP 在 IMDB 语料库上的准确率均有一定的提升。这说明基于 SVM 分类器虽然有较好的表现,但需要结合一些外部特征(如评分等级和 N-grams 信息等),而深度学习在减少外部特征及人工规则等方面则更具优势。

(2)对于 Paragraph Vector 来说,其在 IMDB 上准确率最高为 92.3%,在 SSTb2 和 SSTb5 上的准确率分别为 87.8%

[1] fast300:https://fasttext.cc

和 48.7%。基于 google300 和 glove300 的 AttLSTM 在 IMDB 上的准确率分别为 90.8% 和 90.7%,不如 Paragraph Vector,但在 SSTb2 和 SSTb5 上的准确率比 Paragraph Vector 略高。我们认为:Paragraph Vector 是基于目标语料库进行词向量的训练,并且综合考虑了段落向量,因此对于训练数据多且句子较长的语料(如 IMDB)来说具有一定的优势,但对于训练数据较少且以短句子为主的语料(如 SSTb2 和 SSTb5)效果相对一般;而 AttLSTM 采用了公开的预训练词向量表,且只考虑了句子的前 300 个单词,因此虽然在 IMDB 上的准确率不如 Paragraph Vector,但 AttLSTM 的性能更好,而且更适合短句子。

(3)对于 Constituency Tree-LSTM 来说,其在 SSTb2 语料上的准确率比同样基于 glove300 的 AttLSTM 低 1%,但在 SSTb5 上的准确率比 AttLSTM 略高。这说明结合语法树和自注意力的 LSTM 各有优势,但 Tree-LSTM 需要使用较为复杂的树结构,并且依赖于句子的语法结构分析和解释。

(4)对于 CNN-static 和 CNN-multichannel 来说,当采用相同的 google300 词向量和维度时,AttLSTM 在 MR 和 SSTb2 上的准确率均较高,特别是在 SSTb5 上有明显的提升,但在 SSTb2 上比 CNN-multichannel 略低。其主要原因在于 CNN-multichannel 中结合了动态词,因此在一定程度上能较好地利用语料库本身的信息。

(5)对于 AttLSTM 和 LSTM+MLP 来说,在相同的词向量和参数条件下,AttLSTM 在各个语料上均有一定的提升,这说明自注意力机制确实能够利用编码端的所有输出序列信息来提升模型的分类效果。

4.4 注意力矩阵可视化

利用自注意力机制还可以直观地展现模型在训练及验证过程中对句子中不同位置的关注点。

下面以基于 glove300 的 AttLSTM 模型及 SSTb2 语料库为例,分别选择测试集中的 4 个句子绘制相应的 Heatmap 图,如图 4 所示。

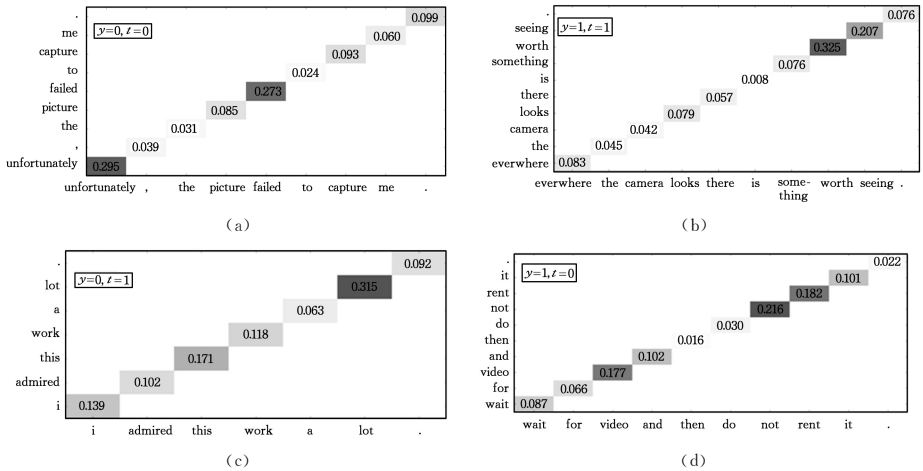

图 4 注意力向量可视化

Fig. 4 Visulization of attention vectors

图 4(a)、图 4(b)均为预测正确的句子,显然自注意力机制能够很好地关注到句子中的情感词。而图 4(c)、图 4(d)均为预测错误的句子,可以看出自注意力并没有学习到句子中的关键词。

结束语 自注意力机制近两年来在基于深度学习的 NLP 中得到应用。本文的主要贡献是利用自注意机制提出一种简单的句子情感分类模型,并通过实验证明了自注意力在情感分类任务上的有效性。虽然与一些相关工作相比,本文所提出的模型并未取得最优的结果,但其优点是模型结构简单,在短句子方面表现较好,而且不需要依赖于其他的语法树、外部特征或知识库等,具有较好的泛化性。值得注意的是,AttLSTM 模型由于在计算注意力向量时需要考虑编码端的所有输出序列信息,因此增加了额外的参数和计算量。这意味着随句子长度的增加,模型的训练时间也相应增加。

后续将进一步完善本文工作,特别是将 Hard AM 和 Soft AM 相结合,以解决长句子情感分类的计算量大的问题。

参 考 文 献

[1] WAGN X,LIU Y C,SUNET C J,et al. Predicting Polarities of Tweets by Composing Word Embeddings with Long Short-Term Memory[C]// Proc. of the 53rd Annual Meeting of the Ass. for Computation Linguistics and the 7th Int. Joint Conf. on Natural Language Processing. Stroudsburg: Association for Computational Linguistics,2015:1343-1353.

[2] TAI K S,SOCHER R,MANNING C D. Improved Semantic Representations from Tree Structured Long Short-term Memory Networks[C]// Proc. of the 53rd Annual Meeting of the Ass. for Computational Linguistics and the 7th Int. Joint Conf. on Natural Language Processing. Stroudsburg: Association for Computational Linguistics,2015:1556-1566.

[3] SOCHER R,PERELYGIN A,WU J,et al. Recursive Deep Models for Semantic Compositionality over a Sentiment Treebank [C]// Proc. of the 2013 Conf. on Empirical Methods in Natural Language Processing. California: Stanford Press, 2013: 1631-1642.

[4] BAHDANAU D,CHO K Y,BENGIO Y. Neural Machine Translation By Jointly Learning to Align and Translate [C]// ICLR 2015. New York:Cornell University Press,2015.

[5] LING W,TSVEYKOV Y,AMIR S,et al. Not All Contexts Are

Created Equal: Better Word Representations with Variable Attention[C]// Conf. on Empirical Methods in Natural Language Processing. Stroudsburg: Ass. for Computational Linguistics, 2015: 1367-1372.

[6] LUONG M T, PHAM H, MANNING C D. Effective Approaches to Attention-based Neural Machine Translation[C]// EMNLP 2015. Stroudsburg: Association for Computational Linguistics, 2015: 1412.

[7] YANG Z C, YANG D Y, DYER C, et al. Hierarchical Attention Networks for Document Classification[C]// Association for Computational Linguistics, NACCL 2016. Stroudsburg, 2017: 1480-1489.

[8] PAULUS R, XIONG C M, SOCHER R. A Deep Reinforced Model for Abstractive Summarization[C]// International Conference on Learning Representations (ICLR 2018). 2017.

[9] LI L F, NIE Y P, HAN W H, et al. A Multi-attention-Based Bidirectional Long Short-Term Memory Network for Relation Extraction[C]// ICONIP 2017. Berlin: Springer, 2017: 216-227.

[10] CHENG J P, LI D, LAPATA M. Long Short-Term Memory-Networks for Machine Reading[C]// Association for Computational Linguistics, EMNLP 2016. Stroudsburg, 2016: 551-561.

[11] PARIKH A, TACKSTROM O, DAS D, et al. A Decomposable Attention Model for Natural Language Inference[C]// Association for Computational Linguistics, EMNLP 2016. Stroudsburg, 2016: 2249-2255.

[12] LIN Z H, FENG M W, SANTOS C N, et al. A Structured Self-Attentive Sentence Embedding[C]// ICLR 2017. New York: Cornell University Press, 2017.

[13] SHEN T, JIANG J, ZHOU T Y, et al. DiSAN: Directional Self-Attention Network for RNN/CNN-Free Language Understanding[C]// AAAI-18. 2018: 5446-5455.

[14] SRIVASTAVA N, HINTON G, KRIZHEVSKV A. Dropout: A Simple Way to Prevent Neural Networks from Overfitting[J]. Journal of Machine Learning Research, 2014, 15(1): 1929-1958.

[15] PANG B, LEE L. Seeing Starts: Exploiting Class Relationships for Sentiment Categorization with Respect to Rating Scales[C]// ACL 2005. NY: ACM Press, 2005: 115-124.

[16] MAAS A L, DALY R E, PHAM P T, et al. Learning Word Vectors for Sentiment Analysis[C]// The 49th Annual Meeting of the Ass. for Computational Linguistics. Stroudsburg: Association for Computational Linguistics, 2011: 142-150.

[17] JOULIN A, GRAVE E, BOJANOWSKI P, et al. Bag of Tricks for Efficient Text Classification[C]// Association for Computational Linguistics, EACL 2017. Stroudsburg. 2017: 427-431.

[18] LE Q V, MIKOLOV T. Distributed Representations of Sentences and Documents[C]// 31st International Conference on Machine Learning. Beijing: International Machine Learning Society, 2014: 1188-1196.

[19] NGUYEN D Q, VU T, PHAM S B. Sentiment Classification on Polarity Reviews: An Empirical Study Using Rating-based Features[C]// Proc. of the 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis. Stroudsburg: Association for Computational Linguistics, 2014: 128-135.

[20] KIM Y. Convolutional Neural Networks for Sentence Classification[C]// Association for Computational Linguistics, EMNLP 2014. Stroudsburg, 2014: 1746-1751.

YU Shan-shan, born in 1980, Ph. D, is senior member of China Computer Federation. Her main research interests include machine learning, big data and semantic Web.

SU Jin-dian, born in 1980, Ph. D, associate professor. His main research interests include natural language processing, artifical intelligence, machine learning.