基于并行化递归神经网络的中文短文本情感分类

谢 铁 郑 啸 张 雷 王修君
(安徽工业大学计算机科学与技术学院 安徽 马鞍山 243002)

摘 要 情感分析的一个重要应用是判断用户对于产品评论的情感倾向,这些用户评论一般都是字数较少的短文本。传统方法多利用词袋模型获取单词的浅层特征来进行情感分析,利用这些简单特征训练的模型在短文本,尤其是在复杂语法问题上效果并不理想。通过利用深度递归神经网络算法来捕获句子语义信息,并引入中文"情感训练树库"作为训练数据来发现词语情感信息,在短文本情感五分类的问题上取得了较高的准确率。针对复杂模型在海量数据训练上的时间效率问题,通过在 Spark 并行框架下实现了模型的并行化处理,使得模型的可扩展性和时间效率得到提升。

关键词 深度学习 情感分析 文本分类 Spark

中图分类号 TP3 **文献标识码** A DOI:10.3969/j.issn.1000-386x.2017.03.037

SENTIMENT CLASSIFICATION OF CHINESE SHORT TEXT BASED ON PARALLELIZED RECURSIVE NEURAL NETWORK

Xie Tie Zheng Xiao Zhang Lei Wang Xiujun
(*School of Computer Science and Technology, Anhui University of Technology, Maanshan 243002, Anhui, China*)

Abstract A significant application of sentiment analysis is to determine the user's semantic orientation in product reviews which are generally short texts. Traditional methods often acquire the shallow characteristics of words for sentiment analysis through bag-of-words model. However, the model trained through these simple characteristics doesn't have a good performance in short text, especially complex syntax context. Through using deep recursive neural network to capture the semantic information and introducing a Chinese sentiment training treebank as the training set to find the sentiment information, a relatively higher accuracy on five-class short text sentiment analysis is achieved. Aiming at the problem of training time efficiency in large scale data, the parallelization is implemented through Spark, which can enhance the scalability and time efficiency of the model.

Keywords Deep learning Sentiment analysis Text classification Spark

0 引 言

在自然语言中,除了反映客观事实的信息之外,通常也包含着反映人类主观感情色彩的信息。情感分析的目的就是让计算机通过规则或统计学习等手段,来标注出人类自然语言文本里表达的情感、观点。

情感分析的对象最初是篇章级的文本。随着移动互联网的发展,互联网上出现了更多的短文本数据,情感分析的研究对象也开始转为短文本。短文本相对于篇章级的长文本而言,其提供的信息量较少,且口语化严重,提升了分析和分类难度。在基于机器学习的情感分类算法中,传统方法主要采用了词袋模型,词袋模型无法适应短文本中更加复杂的语言环境,其容错率大为降低。神经概率语言模型出现以后,抽取词语的特征并进行向量化表示的方法得到了较大认可。于是在短文本的情感分类算法中,特征选取也开始从学习数据集的统计特征转向每个词语的语义特征。Soch-

收稿日期:2016-03-28。国家自然科学基金项目(61402008,61402009);安徽省科技重大专项(16030901060);安徽高校省级自然科学研究重大项目(KJ2014ZD 05);安徽省高校优秀青年人才支持计划。**谢铁**,硕士生,主研领域:自然语言处理。**郑啸**,教授。**张雷**,教授。**王修君**,讲师。

er[1]使用的递归自编码模型正是利用词向量自底向上结合的方式,来计算一个句子的特征向量。这种语义合成的方式在情感分类任务中取得了不错的效果,但无法准确地捕获一个词语对周围词语的影响。同时,随着以微博、社交评论为代表的短文本的大量涌现,当前算法也面临了短文本大数据带来的计算性能下降的挑战。

本文使用递归张量神经网络作为模型,该模型用一个张量参数捕获词语对其邻近词语的语义影响。为了使模型能够运用于中文文本,本文建立了中文情感训练树库。为了更好满足算法对大数据处理实时性的需求,本文对该模型在 Spark 框架下做了并行化处理的优化。

1 相关工作

情感分析目前主要有两类方法:基于词典的方法和基于机器学习的方法。前者需要构建情感词典,通过统计文本中情感词的条目来计算情感倾向,其局限性在于无法解决未登录词问题。Pang[2]较早运用机器学习方法处理情感分类任务,通过 n 元模型(n-gram)作为特征,再运用朴素贝叶斯、最大熵分类器、支持向量机进行分类,取得了较好的效果。随后研究者开始尝试各种方法进行情感分析任务:Mulle[3]把传统的特征与句法关系相结合。之后 Kennedy[4]把上下文语境和情感倾向性转移考虑到特征选择中。但以上方法本质上还是基于词袋模型,词袋模型其忽视语序的特点导致某些情形下的分类效果很差。比如,像这样两个句子:“这部电影优点大于缺点”和“这部电影缺点大于优点”,对于词袋模型来说,这两个句子具有相同的词袋模型的向量表示,意味着在词袋模型中,这两个句子处理结果是一样的。然而,事实上,这两个句子表达的观点截然相反。Nakagawa [5]运用加依存树的条件随机场模型(CRF)对词袋模型进行改进,该模型可以适用于一些情感极性转移的情况。但一方面该方法需要大量人工构建的语料,另一方面以上方法在建模过程中使用的函数简单,特征的选取也较为简单,导致其对复杂语句的表达能力有限。

中文文本的情感分析基本沿用英文情感分析的方法。如 Zhao[6]等人在 CRF 的基础上加入“冗余特征”来进行情感分析。李寿山等人[7]认为不同领域需要选择不同的分类方法,因此他采用一种基于 Stacking 的组合分类方法用以组合不同的分类。谢丽星[8]等人使用了三种不同的方法,包括表情符号规则方法,情感词典规则方法,基于 SVM 的层次结构多策略方法,进行情感分析,并指出基于 SVM 的层次结构多策略方法效果最好。目前针对中文的情感分析的研究进展相对英文的情感分析有所滞后,主要在于中文情感分析存在以下难题。(1) 中文需要分词,分词错误会对情感分析产生影响。(2) 由于中文一词多义现象繁多,给中文情感词典的构建造成了难度。如“水分”,一般认为是物理意义上的水,为中性词。引申义为夹杂着不真实的成分,为贬义词。(3) 中文环境下,尤其是语言较为随意的网络上,“反讽”的用法较多。这给情感分析造成很大困难。

上述方法多采用语言的浅层特征,如词频,这难以应对短文本复杂的语法环境。因此情感分析任务开始从这些浅层特征,转向更加抽象的高层特征,如语义。使用深度学习可以通过一种深层的非线性网络结构来实现复杂函数的逼近,从而捕获数据的本质特征。深度学习的网络结构本质是一个多层神经网络[9]。Bengio[10]提出神经概率语言模型,使用二元语言模型把单词映射为一组向量,这种词向量可以抽象出单词的语义特征,这为后面深度学习算法在自然语言处理领域里的应用奠定了基础。此后,Socher [11]认为,文本跟图像一样也具有类似的树形递归结构,在进行单词的组合时,往往不是按照单词的顺序组合,而是先组合语法或语义相近的单词或者短语。这样一个句子的每个单词作为一个节点按照相应的语法规则可以自底向上结合为一个树形结构。由于一个词可以映射为一个词向量,因此按照这种树形结合以后,树的根节点则代表着整个文本的向量。随后,Socher[1]就利用该方法提出递归自编码器模型 RAE(Recursive Auto Encoder)应用于情感分类任务。但是 RAE 模型无法完全捕捉复杂句子结构下长短语的合成语义[12],于是在矩阵向量神经网络模型(Matrix-Vector RNN)[12]中,在沿用 RAE 模型中向量的同时为每个节点赋予一个矩阵。向量描述的是该词语本身的含义,矩阵则用于描述该词语是如何改变其相邻词语或短句的含义。但 MV-RNN 的问题在于,参数太多,而且参数规模依赖于词表大小,这将会导致计算量非常大,学习也会不够充分。递归张量神经网络 RNTN(Recursive Neural Tensor Network)[13]解决了这一问题,但该模型依赖于特殊的数据集——情感树库。如果要在中文文本分析中运用此模型,需要建立一套中文情感训练树库。

2 基于递归神经网络的情感分类模型

基于递归神经网络的情感分类模型的核心思想是通过叶子节点(单词)自底向上计算父节点而得到的

句子向量来进行情感分类。图 1 展示了这一过程,当输入一个句子时,它把该句子解析为一棵二叉树,每个叶子节点是一个词,并以向量表示。接着自底向上计算每个叶子节点的父节点的向量。父节点同样也将作为下次的输入接着向上计算父节点的向量,直到计算到顶点为止。

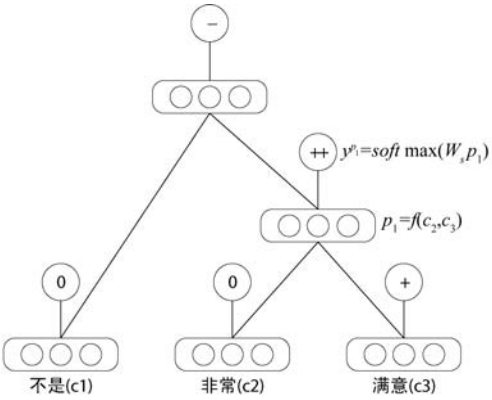

图 1　节点利用 f 函数来自底向上的合成
父节点向量,通过 softmax 函数来预测情感

图 1 中每个单词是一个 d 维的向量,并进行随机初始化。所有的向量存于一个矩阵 L。其中 $L \in R^{d\times|V|}$,$|V|$ 是数据集中所有词语的个数,即词表的大小。

每个节点还包含一个 *softmax* 分类器:

$$y^a = softmax(W_s a) \tag{1}$$

其中,$W_S \in R^{k\times d}$(k 是情感标签的数量)。a 表示节点的向量。

不同递归神经网络模型最大的区别在于模型的组合函数 f 可以根据应用需求来定义。正如前面所说,递归张量神经网络模型(RNTN)既要捕获单词组合时的组合含义,又要保证参数的数目不能过多。因此,在递归张量神经网络模型中,使用张量来表示组合参数,所有节点共用一套张量参数。这样通过张量进行坐标变换使得某些分量可以依照某些规则作线性变换,来降低整体的参数数量。因此组合函数为:

$$p = f([c_1;c_2]^{\mathrm{T}} V^{[1:d]}[c_1;c_2] + w[c_1;c_2]) \tag{2}$$

其中,$V^{[1:d]} \in R^{2d\times 2d\times d}$ 是张量参数。

实际上 RNN 模型可视为 RNTN 模型在张量 V 为 0 时的特例。在这种情况下 V 不会对输入的词向量造成任何影响。反过来,当 V 不为 0 时,V 显然会改变词向量的特征。在充分训练以后,可视为张量 V 捕获了语义合成时的信息。

模型的任务是最大化正确预测情感标签的概率,也就是最小化每个节点的预测值 $y^i \in R^{c\times 1}$ 与目标值 $t^i \in R^{c\times 1}$ 的交叉熵(C 表示类别的数目)。

RNTN 模型使用如下交叉熵函数:

$$E(\theta) = \sum_i \sum_j t_j^i \log y_j^i + \lambda \| \theta \|^2 \tag{3}$$

其中 j 表示第 j 个节点。θ 表示 RNTN 模型参数,$\theta = (V, W, W_S, L)$。

由于 *softmax* 分类中对权重 W_S 求导具有一般性,仅仅是简单地把各个节点的误差相加而得到。因此定义 x_i 作为节点的一个向量。忽略权重 W_S 的微分表达式。每一个节点都是通过权重 V, W 递归反向传导修正其误差。定义 $\delta^{i,s} \in R^{d\times 1}$ 为节点 i 本身的 *softmax* 误差:

$$\delta^{i,s} = (W_S(y^i - t^i)) \otimes f'(x^i) \tag{4}$$

其中⊗是哈达姆算子。取 tanh( ),f' 是其导数。某一节点的求导必须是通过树的自顶向下的计算过程来完成。参数 W, V 的完整求导过程是各个非叶子节点导数的总和。下面举例来说明这一问题。对于节点 i,将这个节点反馈的误差定义为 $\delta^{i,com}$。显然,对于顶点 $p2$ 来说,它只接收来自自己的 *softmax* 误差。因此 $\delta^{p_2,com} = \delta^{p_2,s}$。$\delta^{p_2,com}$ 可用于计算对 V 的求导:

$$\frac{\partial E^{p_2}}{\partial V^{[k]}} = \delta_k^{p_2,com}[a;p_1][a;p_1]^{\mathrm{T}} \tag{5}$$

其中 $V^{[k]}$ 是 V 中第 k 个片。$\delta_k^{p_2,com}$ 是向量第 k 个元素。现在可以计算 $p2$ 两个子节点的误差了:

$$\delta^{p_2,down} = (W^{\mathrm{T}}\delta^{p_2,com} + S) \otimes f'([a;p_1]) \tag{6}$$

其中定义:

$$S = \sum_{k=1}^{d} \delta_k^{p_2,com}(V^{[k]} + (V^{[k]})^{\mathrm{T}})[a;p_1] \tag{7}$$

上述中 $p1$ 节点的反馈误差等于该节点本身误差加上 $p2$ 节点向下传给 $p1$ 节点的误差 $\delta^{p_2,down}$。其中 $\delta^{p_2,down}$ 分为两个部分,$\delta^{p_2,down}[1:d]$ 和 $\delta^{p_2,down}[1+d:2d]$,前者传给左孩子 a,后者传给右孩子 $p1$,因此 $p1$ 的误差为:

$$\delta^{p_2,com} = \delta^{p_1,s} + \delta^{p_2,down}[1+d;2d] \tag{8}$$

$V^{[k]}$ 完整的导数计算过程就是每个非叶子节点的导数相加,公式如下:

$$\frac{\partial E}{\partial V^{[k]}} = \frac{\partial E^{p_2}}{\partial V^{[k]}} + \delta_k^{p_2,com}[a;p_2][a;p_2]^{\mathrm{T}} \tag{9}$$

对 W 的求导方式也是类似的,不再赘述。

RNTN 模型的训练是通过最小化交叉熵函数来实现的,通过对上述参数求导不难实现梯度下降算法寻找函数的最优值。但传统的随机梯度下降(SGD)算法本质上是顺序性的,这种完全串行都得方式使得参数更新非常耗时,也因此模型的并行较为困难。

考虑到模型参数更新过程的独立性,本文对 RNTN 模型做如下修改:(1) 将参数更新部分从 RNTN 模型中提出,单独置于参数服务器上运行,参数服务器由一个或多个节点组成。(2) 将训练集划分多个子集,并对每个子集运行一个单独的 RNTN 模型。(3) 每个模型副本计算出参数的最优值后,与参数服务器交换

参数,并获得参数服务器提供的最新参数进行迭代更新,而参数服务器对不同的节点传来的参数进行求平均以更新参数。

3 中文情感训练树库的构建

由于模型非常依赖于情感树库,语言之间的语法差异使得树库无法跨语言,因此这也给模型的跨语言问题上增加的难度。目前中文环境下并没有标注好的情感树库,因此本文为了使 RNTN 模型能够适用于中文环境建立了一套情感训练树库。

本文利用 python 爬虫在大众点评网对某一餐饮业的评论数据进行爬取。共收集了 11 256 条数据。首先需要对这些数据去除 HTML 标签以及去除标点符号,然后人工去除与评价内容无关的评论数据。接下来使用分词软件进行分词处理,本文采用了 python 下的开源分词工具 Jieba 分词。构建语法树的过程同样基于上述的递归神经网络模型,但去掉 softmax 层。为了增加训练结果中树结构的可靠性,可以在一个更大规模的语料中进行训练。在得到语法分析树后,利用本文建立的标注系统对树的每个节点人工进行情感标注,这样生成一套含有 11 256 棵标注好情感的语法树库,即情感训练树库。

考虑到普通两类(褒贬义)或者三类(褒贬意加中性)情感划分无法准确把握句子情感,比如“喜欢”和“非常喜欢”的区别,或“不是特别满意”和“特别不满意”的区别。此外由于非常极端的情感出现率很低,一般情况下五类足以覆盖用户的情感观点,过多的情感分类也会提升模型训练的难度。综合以上,本文选择五类情感划分——分别为非常消极、消极、中性、积极、非常积极五类。

RNTN 模型一方面需要靠情感训练树库来决定节点合成顺序,另一方面,通过节点标签的变化来捕获句子情感极性的转移。如图 2 所示,模型的输入值是一棵每个节点都被标记了情感标签的句法分析树。

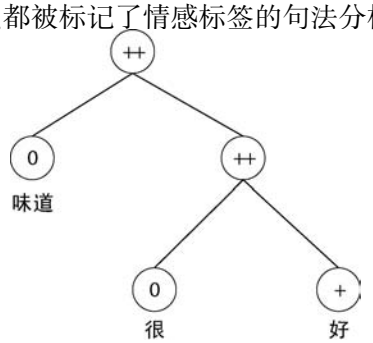

图 2 “味道很好”这个句子组成的一棵情感训练树

模型首先会通过一个映射层把每个单词赋予一个向量,向量通过 softmax 层,自底向上地根据语法树的合成顺序来计算出情感极性。在图 2 中,注意到“好”字被标记为积极类情感,在与“很”字结合以后,它们的父节点被标记为非常积极类的情感。通过这种极性的变化,来对向量和张量的值进行调优,使其捕获到这种类型的语义变化。

4 基于 Spark 的并行化 RNTN 模型

RNTN 模型涉及到大量迭代操作,因此本文选择基于内存计算的 Spark 云计算平台进行并行化处理。如第 2 节所述,算法训练的迭代过程主要集中在梯度下降算法更新参数部分,通过对这种串行方式的改进,并行化的基本操作主要是将数据集划分为若干分片,并在每个分片上运行一个单独的模型副本,模型副本之间的通信均需要通过参数服务器,该服务器用于维护各模型副本的参数状态。但是,传统的并行方式通常是同步模式,比如 Spark 的 Mllib 库以及 Mahout。这一做法的缺点在于参数服务器总要等待所有节点提交完参数误差以后,再统一进行参数的更新。由于不同节点上的处理时间不同,这样会导致已经计算完毕的节点需要等待其他未完成本轮计算的节点。而本模型由于计算过程相对独立,计算结果并不依赖于其他模型副本的计算结果,因此最佳方式是异步模式,即:在参数服务器上一旦有节点提交参数,立即更新本地参数,并分配新的参数给该节点,无需等待其他节点计算完毕。在这个过程中一个模型副本产生的最新结果不会立刻体现在其他模型副本中,直到他们进行下次参数交换。参数服务器也可以是由多个节点组成,这时每个参数服务器只与和自己通信的计算节点交换参数。如图 3 所示。

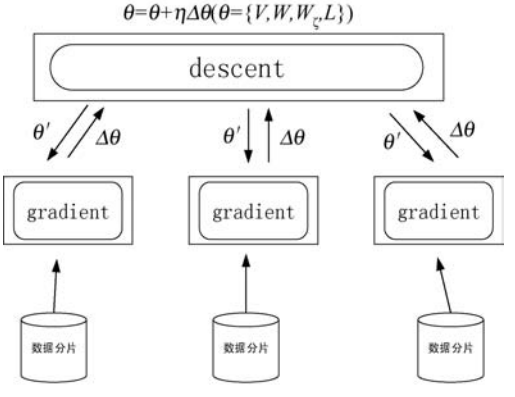

图 3 异步模式的并行方式

由于异步模式的一个特点是计算节点与参数服务器交换完参数需要进行下一轮迭代,因此必须把数据

集切分成若干子集,一方面如果参数服务器是由多台计算机组成的,可以分配给不同的参数服务器,另一方面把大数据集切分成多个小批量的数据集以便在计算节点完成计算后多次迭代。需要指出的是,这里和 Spark 的 parallelize 函数进行数据切分的目的不同。如果直接使用 parallelize 函数把数据切分为计算节点数量对应的份数并分配给计算节点,这样整个模型计算完成仍然需要等待最后一个计算节点完成计算,就谈不上异步模式了。因此异步模式的一个前提就是需要把数据切分为非常多个小的子集,然后进行多次迭代实现。

在 Spark 环境下进行异步模式的另外一个难点在于,Spark 提供的函数是基于同步并行模式的,那么参数服务器在迭代过程中,必须等待计算节点计算完成才会进行下一轮迭代。所以为了实现异步模式,需要一些改进。例如:在通常做法中,Spark 通过 mappatations 方法把某个计算过程广播到计算节点,然后对分配给节点的数据进行一系列的计算,得到结果后进行下一轮迭代,这和同步模式无异。因此,如果进行异步模式,不能直接广播计算过程。本文对其修改如下:mappatations 传给计算节点的函数作用是用于向参数服务器发出 url 请求,当参数服务器接收到请求会返回给该节点一个模型的副本。计算节点拿到模型副本完成计算后再向参数服务器 post 出自己的计算结果。这样,只要 mappatations 能够成功把该函数分配给某一计算节点就完成此次迭代,而无需等待计算节点反馈结果。

由上分析可知,对 RNTN 模型并行化的一般步骤如下:首先在参数服务器上需要对数据切分成若干子集。每次迭代分配给计算节点一个子集。同时,参数服务器上定义 gradient 函数和 descent 函数。前者用于运行模型的梯度下降过程,该函数会在计算节点请求模型副本时发送给计算节点执行。后者运行于参数服务器用于在计算节点返回梯度时更新参数。在计算节点处理每个分片之前,都会向参数服务器请求最新的模型副本(包含最新参数的模型副本),在计算节点收到模型副本后会执行 descent 函数,运行梯度下降过程。该过程执行完毕会向参数服务器发送梯度,此时参数服务器更新参数,等待计算节点发出下一轮请求。

下面的算法伪代码中,getModel( ) 的作用是向参数服务器请求一个模型的副本。sendGradient( ) 的作用是把误差返回给参数服务器。decent( ) 是在参数服务器上更新参数。gradent( ) 主要是在计算节点运行模型副本,更新参数。SparkRNTN( ) 是并行化 RNTN 的主函数。

算法 1　getModel,向服务器请求最新的模型副本

```
input:ParamServerIP                    //参数服务器的 IP 地址
output:model
                //model 为包含了最新参数的 RNTN 模型副本
getModel(ParamserverIP)
    model = getModelFormParamserver(ParamserverIP)
    return model
```

算法 2　计算节点向参数服务器返回参数误差

```
input: ParamserverIP,update            //update 为参数误差
sendGradient(update. ParamserverIP)
    sendGradientToParamserver(update,ParamserverIP)
```

算法 3　descent,更新参数服务器上的参数

```
input:model,update
descent(model,update)
    model. paramater + = update * α
```

算法 4　在计算节点训练模型

```
input:model,data
output:update
gradient(model,data)
    update = model. train(data)
    return update
```

算法 5　程序的主函数

```
input:trees      //输入数据为情感语法树的括号表示形式
SparkRNTN (trees)
    sc = SparkContext( )
    for i = 0 to 数据集总和 step = minibatch
        data = sc. parallelize(trees[i:i + minibatch])
        //把 sendGradient、gradient、getModel 发送到计算节点执行
        rdd. mapPartitions(sendGradient(gradient(getModel(ParamserverIP), data),ParamserverIP )). collect( )
        while true:
            if 有节点发出更新参数的请求:
            update = 获取计算节点 post 来的参数误差
            descent (model,update)  //在本地更新模型参数
        if 有节点请求模型:
            将最新的 model 发送给计算节点
    return model
```

5　实　验

本文设计两套不同的实验。第一个实验是验证模型在中文数据下的准确性。第二个实验主要是验证并行 RNTN 模型的性能。在实验 1 中,所用数据为 11 256条美团网上用户对新石器烧烤店的评论数据。实验 2 在 Spark 云平台环境中进行,所用环境配置为:4 台 CPU2.8 GHz、8 GB 内存的台式机作为计算节点,1 台 CPU2.2 GHz,内存 8 GB 的台式机作为参数服务器,

Spark 版本为 2.4.1,python 版本 2.7.3。

5.1 RNTN 模型在中文语言环境下的应用

首先对每条数据经过如下处理:去重,去标点,分词,构建语法树。其中语法树结构用树的括号形式表示,并对每个节点从非常消极到非常积极用 1 到 5 之间的数字进行标记。整理后的数据如表 1 所示。

表 1 情感树的括号表示形式

1	(4 (3 (2 一如既往)(3 (2 的)(3 不错)))(4 (2 感觉)(4 (2 相当)(3 (2 的)(3 赞)))))
2	(4 (4 (4 (2 服务)(4 (2 很)(3 好)))(3 (2 味道)(3 (2 也)(3 不错))))(3 (2 人均)(3 (2 消费)(3 (2 很)(3 实惠)))))
3	(1 (2 (2 生菜)(2 (2 居然)(2 (2 (2 要)(2 自己))(2 买))))(1 (2 不)(3 开心)))

为验证 RNTN 模型在中文环境各类复杂句式下的情感分类准确率,本文针对转折句式、否定句式和一般句式几种不同句式分别做了实验。其中采用的典型转折句式 251 条,否定句式 198 条,一般句式 11 256 条。

从表 2 可以看出:首先,单纯的 word2vec 模型能够捕获语义信息但无法捕获情感信息,因此在三类测试中的准确率均较低。其次,由于词袋模型忽略语序,无法分辨哪个情感词在转折之前哪个之后,也就无法区分情感往哪一方偏移,在否定句式中也只是记录否定词出现的次数,因此误差也较大。RAE 在使用语义做特征的基础上捕获情感信息,相对其他模型有一定的优势。而 RNTN 模型由于加入情感树库作为训练集,节点上的标签能够非常清楚地指示情感转移或否定现象,加之张量对某些修饰词语的捕获,因此获得了最高的准确率。同时也验证了引入了中文情感树库的 RNTN 模型能够很好地应用于中文语言环境。

表 2 情感 5 分类的准确率

模型	转折句式准确率	否定句式准确率	一般句式准确率
Word2vec	27.8	46.2	61.8
BOW + Svm	31.3	52.7	66.5
Doc2vec	35.4	56.1	70.8
RAE	46.6	65.1	74.2
RNTN	60.8	74.3	77.6

5.2 RNTN 模型在 Spark 云平台的并行效果

为了衡量 RNTN 模型在并行优化后的性能,使用如下指标[17]:

(1) Scaleup:度量不同处理器规模下,处理不同规模数据的性能。该指标计算式为:

$$Scaleup = \frac{runTime(1 \times Data, 1)}{runTime(n \times Data, n)} \tag{10}$$

其中 $runTime(n \times Data, m)$ 表示模型在 m 个计算节点上处理 n 份数据的时间,下同。

(2) Sizeup:度量了在平台固定的情况下,依次增加数据量时算法的性能。该指标计算式为:

$$Sizeup = \frac{runTime(1 \times Data, n)}{runTime(n \times Data, n)} \tag{11}$$

(3) Speedup:度量了在数据规模相同下,并行计算比串行计算运算速度加快的程度。该指标计算式为:

$$Speedup = \frac{runTime(Data, 1)}{runTime(Data, n)} \tag{12}$$

首先是 Scaleup,分别在 1 台至 4 台机器上处理一份完整的数据,所得到的 Scaleup 性能如 4 图所示。

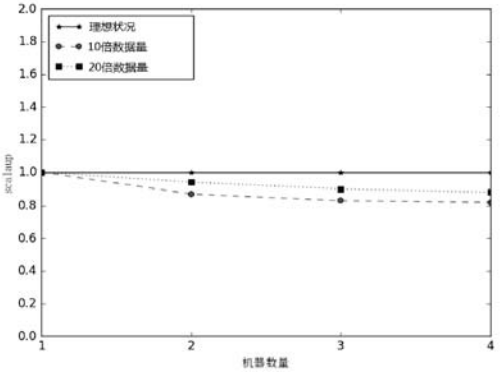

图 4 Scaleup 的性能展示

从图 4 中可以看出,随着节点的规模增大,所得到的 Scaleup 曲线趋于稳定,显示出了很好的扩展性。此外数据量规模变大,节点扩展性变高,这是因为数据量越大越能较好利用节点的计算能力。

为验证 Sizeup 的性能,本文分别考虑 1 至 4 台计算节点分别在 10 倍、20 倍和 40 倍的原始数据集上运行的结果。

从图 5 可以看出,除了增加节点性能会得到提升外。数据规模的增大,也会导致性能的明显增大,这是因为随着数据规模增大,节点通信的时间在整个算法的运行时间的比例会减小。因此获得了较好的 Sizeup 性能。

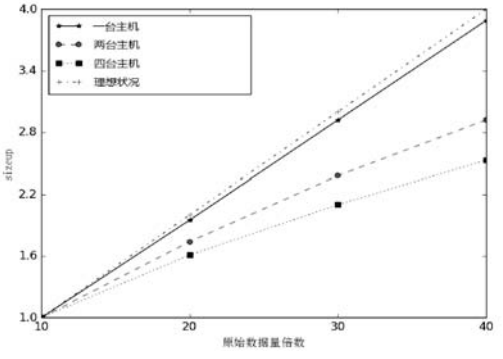

图 5 Sizeup 的性能展示

在验证 Speedup 性能的实验中,使用与验证 Sizeup

性能一样的实验环境。如图 6 所示。

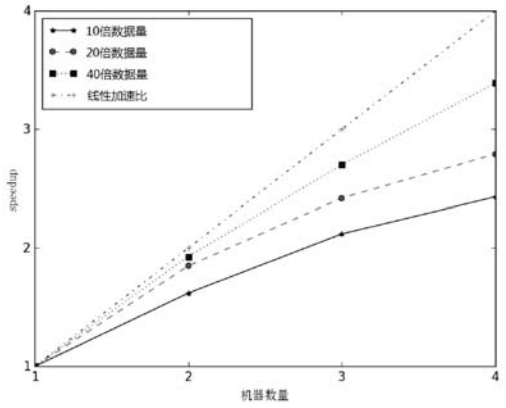

图 6　Speedup 的性能展示

可以发现图 6 中随着节点数目变多,线性加速比下降。原因在于节点数增加会增加节点的通信时间。另外在规模越大的数据集上,模型的线性加速比效果越好。因此并行算法可以有效地应对大数据集。

6　结　语

本文主要工作一是通过构建一套中文情感训练树库并引入递归张量神经网络模型进行了中文环境下的 5 分类短文本情感识别,取得了较好的效果。二是对递归张量神经网络模型实现了并行化处理,通过实验证明,该并行化的模型有良好的并行性能。

但是并行化的 RNTN 模型依然有很多问题需要解决。主要在:(1) 在较长文本中,句子的树形结构会非常壮大,导致性能下降,而且节点越多,实际上干扰信息也就越多。因此下一步工作解决句子节点过多对模型产生的负影响。(2) 在并行化过程中,本文对 Spark 异步模式改进只是通过建立一个 Web 服务器在计算节点完成任务之后以 Http 通信方式来代替 Spark 内部的通信,更好的方式显然是针对 Spark 的内部通信机制加以改进,未来将会在这方面进行研究。

参　考　文　献

[1] Socher R, Pennington J, Huang E H, et al. Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions[C]//Proceedings of the Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2011: 151 - 161.

[2] Pang B, Lee L, Vaithyanathan S. Thumbs up? Sentiment Classification using Machine Learning Techniques[C]//Proceedings of EMNLP-2002, 2002: 79 - 86.

[3] Mullen, Tony, Nigel Collier. Sentiment analysis using support vector machines with diverse information sources[C]//Proceedings of EMNLP-2004, 2004: 412 - 418.

[4] Kennedy, Alistair, Diana Inkpen. Sentiment classification of movie reviews using contextual valence shifters[J]. Computational Intelligence, 2006, 22(2): 110 - 125.

[5] Nakagawa T, Inui K, Kurohashi S. Dependency Tree-based Sentiment Classification using CRFs with Hidden Variables[C]//Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics. Association for Computational Linguistics, 2010: 786 - 794.

[6] Zhao Jun, Liu Kang, Wang Gen. Adding Redundant Features for CRFs-based Sentence Sentiment Clasification[C]//Proceedings of EMNLP-2008, 2008: 117 - 126.

[7] 李寿山,黄居仁. 基于 Stacking 组合分类方法的中文情感分类研究[J]. 中文信息学报,2010,24(5):56 - 61.

[8] 谢丽星,周明,孙茂松. 基于层次结构的多策略中文微博情感分析和特征抽取[J]. 中文信息学报,2012,26(1):73 - 83.

[9] Hinton G E, Osindero S, Yw T. A fast learning algorithm for deep belief nets[J]. Neural Computation, 2006, 18(7): 1527 - 54.

[10] Bengio Y, Schwenk H, Senécal J S, et al. A neural probabilistic language model[J]. Journal of Machine Learning Research, 2003, 3(6): 1137 - 1155.

[11] Socher R, Lin C Y, Ng A Y, et al. Parsing Natural Scenes and Natural Language with Recursive Neural Networks[C]//ICML. 2011: 129 - 136.

[12] Socher R, Huval B, Manning C D, et al. Semantic compositionality through recursive matrix-vector spaces[C]//Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. Association for Computational Linguistics, 2012: 1201 - 1211.

[13] Socher R, Perelygin A, Wu J Y, et al. Recursive deep models for semantic compositionality over a sentiment treebank[C]//Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2013: 1631 - 1642.

[14] 梁军,柴玉梅,原慧斌,等. 基于深度学习的微博情感分析[J]. 中文信息学报,2014,28(5):155 - 161.

[15] 李宁,罗文娟,庄福振,等. 基于 MapReduce 的并行 PLSA 算法及在文本挖掘中的应用[J]. 中文信息学报,2015,29(2):79 - 86.

[16] Dean J, Corrado G S, Monga R, et al. Large Scale Distributed Deep Networks[J]. Advances in Neural Information Processing Systems, 2012: 1232 - 1240.

[17] Xu X, Jäger J, Kriegel H P. A Fast Parallel Clustering Algorithm for Large Spatial Databases[J]. Data Mining & Knowledge Discovery, 1999, 3(3): 263 - 290.

(下转第 232 页)

可以从 ROC 曲线和 AUC 值看出,融合之后的 AUC 值为0.906 8,明显比在每个蛋白质相似度网络的预测效果高。因此,相似度网络融合算法可以减少每个矩阵中的噪音,可以有效地提高预测算法的准确性。

4 结 语

本文在 HeteSim 算法的基础上,通过改进自相关关系相似度的计算方式,提出了 LPHeteSim 算法。在单个蛋白质相似度网络上的实验表明,改进之后的算法的有效性要好于未改进的 HeteSim 算法。

为了进一步提高预测准确性,本文采用融合蛋白质相似度网络的方法,构造出置信度更高的蛋白质相似度矩阵。实验结果表明,在融合之后的蛋白质相似度网络执行预测算法,要好于在单个蛋白质相似度网络上的预测结果。因此,结合相似度融合算法和 LPHeteSim 算法,可以有效地提高预测新的长非编码 RNA 和蛋白质相互作用的有效性。

参 考 文 献

[1] Bonasio R,Shiekhattar R. Regulation of transcription by long noncoding RNAs[J]. Annual Review of Genetics,2014,48:433.

[2] International Human Genome Sequencing Consortium. Finishing the euchromatic sequence of the human genome[J]. Nature,2004,431(7011):931-945.

[3] Geisler S,Coller J. RNA in unexpected places:long non-coding RNA functions in diverse cellular contexts[J]. Nature reviews Molecular cell biology,2013,14(11):699-712.

[4] Engreitz J M,Pandya-Jones A,McDonel P,et al. The Xist lncRNA exploits three-dimensional genome architecture to spread across the X chromosome[J]. Science,2013,341(6147):1237973.

[5] Zhu J J,Fu H J,Wu Y G,et al. Function of lncRNAs and approaches to lncRNA-protein interactions[J]. Science China Life Sciences,2013,56(10):876-885.

[6] Bellucci M,Agostini F,Masin M,et al. Predicting protein associations with long noncoding RNAs[J]. Nature Methods,2011,8(6):444-445.

[7] Pancaldi V,Bähler J. In silico characterization and prediction of global protein-mRNA interactions in yeast[J]. Nucleic Acids Research,2011:gkr160.

[8] Muppirala U K,Honavar V G,Dobbs D. Predicting RNA-protein interactions using only sequence information[J]. BMC Bioinformatics,2011,12(1):1.

[9] Wang Y,Chen X,Liu Z P,et al. De novo prediction of RNA-protein interactions from sequence information[J]. Molecular BioSystems,2013,9(1):133-142.

[10] Lu Q,Ren S,Lu M,et al. Computational prediction of associations between long non-coding RNAs and proteins[J]. BMC Genomics,2013,14(1):1.

[11] Pang K C,Frith M C,Mattick J S. Rapid evolution of noncoding RNAs:lack of conservation does not mean lack of function[J]. Trends in Genetics,2006,22(1):1-5.

[12] Li A,Ge M,Zhang Y,et al. Predicting Long Noncoding RNA and Protein Interactions Using Heterogeneous Network Model[J]. BioMed Research International,2015,2015.

[13] Yang J,Li A,Ge M,et al. Prediction of interactions between lncRNA and protein by using relevance search in a heterogeneous lncRNA-protein network[C]//Control Conference(CCC),2015 34th Chinese. IEEE,2015:8540-8544.

[14] Shi C,Kong X,Huang Y,et al. Hetesim:A general framework for relevance measure in heterogeneous networks[J]. Knowledge and Data Engineering,IEEE Transactions on,2014,26(10):2479-2492.

[15] Wang B,Mezlini A M,Demir F,et al. Similarity network fusion for aggregating data types on a genomic scale[J]. Nature Methods,2014,11(3):333-337.

[16] Ashburner M,Ball C A,Blake J A,et al. Gene Ontology:tool for the unification of biology[J]. Nature Genetics,2000,25(1):25-29.

[17] Jcquart P. Nouvelles recherches sur la distribution florale[J]. Bull. Soc. Vand. Sci. Nat,1908(0):44.

[18] Finn R D. Pfam:the protein families database[J]. Encyclopedia of Genetics,Genomics,Proteomics and Bioinformatics,2012.

[19] Apweiler R,Martin M J,O'Donovan C,et al. Update on activities at the Universal Protein Resource(UniProt) in 2013[J]. Nucleic acids research,2013,41(D1):D43-D47.

[20] Smith T F,Waterman M S. Identification of common molecular subsequences[J]. Journal of Molecular Biology,1981,147(1):195-197.

[21] Szklarczyk D,Franceschini A,Wyder S,et al. STRING v10:protein-protein interaction networks,integrated over the tree of life[J]. Nucleic Acids Research,2014:gku1003.

[22] Yuan J,Wu W,Xie C,et al. NPInter v2. 0:an updated database of ncRNA interactions[J]. Nucleic Acids Research,2014,42(D1):D104-D108.

[23] Xie C,Yuan J,Li H,et al. NONCODEv4:exploring the world of long non-coding RNA genes[J]. Nucleic Acids Research,2014,42(D1):D98-D103.

(上接第 211 页)

[18] 孙志军,薛磊,许阳明,等. 深度学习研究综述[J]. 计算机应用研究,2012,29(8):2806-2810.

[19] 陈钊,徐睿峰,桂林,等. 结合卷积神经网络和词语情感序列特征的中文情感分析[J]. 中文信息学报,2015,29(6):172-178.