# 并行化的情感分类算法的研究

余永红[1,2] 向小军[2] 商 琳[2]
(南京邮电大学通达学院 南京 210003)[1] (南京大学计算机科学与技术系 南京 210093)[2]

**摘 要** 在海量数据集上执行情感分类任务时,传统的单机情感分类算法的扩展性成为系统的瓶颈。在云计算平台Hadoop上,实现了情感分类任务中特征提取、特征向量加权和情感分类等算法的MapReduce化。在情感语料数据集上,对各种子步骤组合下情感分类算法的精度及每种算法的时间开销进行了对比分析。实验结果验证了实现的并行化情感分类算法的有效性,同时它为用户选择合适算法实现情感分类任务提供了有价值的参考信息。

**关键词** 情感分类,Hadoop,云计算,MapReduce
**中图法分类号** N532 **文献标识码** A

## Research on Parallelized Sentiment Classification Algorithms

YU Yong-hong[1,2] XIANG Xiao-jun[2] SHANG Lin[2]
(College of Tongda, Nanjing University of Posts&Telecommunications, Nanjing 210003, China)[1]
(Department of Computer Science and Technology, Nanjing University, Nanjing 210093, China)[2]

**Abstract** The scalability problem becomes a bottleneck for traditional stand-alone sentiment classification algorithms due to the massive data. We implemented feature extraction, feature weighting and classification algorithms involved in sentiment classification task by using MapReduce technique on Hadoop platform. We evaluated our proposed parallelized sentiment classification algorithms on real data sets in terms of precision and time costs. Experimental results show the effectiveness of these parallelized sentiment classification algorithms and also provide valuable references for users to select suitable sentiment classification algorithms according to user requirements.

**Keywords** Sentiment classification, Hadoop, Cloud computing, MapReduce

## 1 引言

互联网技术的迅速发展使用户可以通过微博、博客、评论等方式对一些热点事件、产品信息等进行反馈,表达自己的观点。这些反馈信息中隐藏着用户对某些事件或者产品的情感倾向,如"赞成"或"反对"等,挖掘反馈信息中隐藏的情感倾向能更好地了解用户的消费习惯,分析热点事件的舆情,为商家和政府机构提供决策支持。情感分类的研究内容就是分析和挖掘用户对商品的评论、热点事件发表的观点以及博客、微博等,识别用户的情感倾向以及情感随时间变化的规律。

目前情感分类技术主要分为:非监督学习、半监督学习和监督学习3类。由于很难找到合适的情感词典用于非监督学习,而所有的半监督学习方法需要很强的假设性,这些局限性限制了非监督学习和半监督学习方法在情感分类任务中的应用,使得情感分类任务大量采用监督学习的方法。近年来,研究者针对情感分类中的特征提取、特征向量加权和分类等任务,提出了一些方法和原型系统。

2002年,Bo Pang[1]等首先利用Naive Bayes、Max Entropy、SVM等分类技术,在文档级别上对文档进行自动的情感分类;2004年,Bo Pang[2]等又提出使用图最小割的方法对文档的句子进行主观性判断;Dave等[3]首先测试和训练结构化的商品评论信息,确定特征词及用来判断评论信息是积极还是消极的积分方法,然后使用分类器对来自互联网的商品评论进行分类;Mullen等[4]使用SVM技术在多种相关的信息源上进行情感分析,这些信息源包括短语和形容词的偏好度量方式以及文本主题知识;另外,Jun Li等[5]研究了影响情感分类算法的性能因素:特征表示、特征加权方式和特征维度。实验结果表明,使用WBB(Word-Based Bigram)、CBT(Chinese Character-Based Trigram)表示特征,利用布尔加权的方式,Naive Bayes(NB)分类算法在不同特征维度下取得最好的平均性能;由于监督学习方法在情感分类任务上的应用趋于成熟,因此近年来的研究工作都聚焦在特征选择方法上[6-9]。

然而随着移动互联网应用的不断普及,用户发表的微博、博客、产品评论和电影评论等信息总量呈指数级增长,达到TB(TetaBytes)级甚至PB(PetaBytes)级,而且有不断增长的趋势。面对海量数据,传统的单机情感分类算法难以快速地完成情感分类的任务,需要研究适合于海量数据的情感分类任务的计算模式。云计算的兴起和发展为解决海量数据下的情感分类任务提供了新的解决方案。本文的主要工作是基于开源的云计算平台-Hadoop,利用Hadoop平台中MapReduce

到稿日期:2012-09-12 返修日期:2012-12-24 本文受国家自然科学基金项目(61035003),科技部国际科技合作计划项目(2010DFA11030),江苏省自然科学基金项目(BK2010054)资助。

余永红(1978-),男,博士生,讲师,主要研究方向为数据挖掘、云计算,E-mail:yuyh.nju@gmail.com;向小军(1986-),女,硕士生,主要研究方向为数据挖掘;商 琳(1973-),女,硕士生导师,主要研究方向为数据挖掘、粗糙集。

和 HDFS 技术为情感分类任务提供了分布式并行化的处理能力和海量数据的存储空间，实现了情感分类任务中的特征提取、特征向量加权和分类 3 个子任务并行化，加快了各个子任务的运行速度。实验结果验证了并行化情感分类算法的有效性和扩展性。

本文第 2 节详细描述情感分类任务中特征提取、特征加权和分类算法的并行化实现；第 3 节在各种并行情感分类子任务组合下，分析情感分类任务在真实数据集上的实验结果；最后总结全文，并描述进一步的研究工作。

## 2 情感分类任务的并行化实现

基于监督学习的情感分类任务一般有 3 个重要的步骤：(1)特征提取：从文本中提取用来表征文本的特征词；(2)特征向量加权：对特征向量进行量化，计算每一个特征对文档的表征度；(3)分类：基于第 2 步的特征向量，用某种监督机器学习方法来二分类文档，把待预测的文档分成正类或负类。

针对情感分类任务的 3 个子任务，本文在 Hadoop 平台上实现了 3 种特征提取算法：基于情感词典的特征提取、基于 Bigram 的特征提取和基于 Substring 的特征提取，两种特征向量加权算法：Bool 特征向量加权算法和 TF-IDF 算法，最后实现了 Rocchio 和 KNN 分类算法。下面详细介绍各种算法 Mapreduce 化的实现细节。

### 2.1 特征提取算法的并行化

#### 2.1.1 基于情感词典的特征提取

基于情感词典的特征提取把情感词典中的词作为特征来表征文档。基于情感词典的特征提取算法的 MapReduce 化过程如下：

(1)使用分词工具对原始数据集进行分词，得到分词后的数据集。

(2)读入情感词典，将所有情感词保存在 List lexicon 中。

(3)在每个 Map 中读入分词后的数据集块，对块中每个分词后的文档提取 Label 和文档 ID，作为 key 值，然后判断分词后文档的每个词是否在 List lexicon 中，如果是，则加入到 value 中，最后发送 key/value 对到 Reduce 函数。

(4)在 Reduce 函数中将相同 key 值的所有 value 写入到 SequenceFile feature 中。

#### 2.1.2 基于 Bigram 的特征提取

Zhongwu Zhai 等[9]的实验结果表明基于 Bigram 的特征提取算法是 Ngram 特征提取算法中效果最佳的算法。Ngram 以短句为单位进行特征提取。Ngram 特征提取算法的伪代码如表 1 所列。

表 1 Ngram 特征提取算法

```
Input:S[1,…,L]
Output:F[1,…,P],P=L−N+1
for i=1 to L−N+1 do
    F. insert(S[i,i+N−1])
end for
return F
```

在 Ngram 特征提取算法中，$N$ 定义 Ngram 的大小，$N=2$ 时为 Bigram 特征提取算法。基于 Bigram 的特征提取算法的 Mapreduce 化过程如下：

(1)在 Map 函数中以文档为单位读取文档的 Label 和 ID，存入 key 中。

(2)以常见的标点符号对文档进行分割，得到短句集合。

(3)对于短句集合中的每个短句，按照表 1 中描述的 Ngram 特征提取算法提取短句中的 Bigram 特征，存入 value 中。

(4)将 key/value 对发送给 Reduce 函数。在 Reduce 函数中将相同 key 值的所有 value 写入到 SequenceFile feature 中。

#### 2.1.3 基于 Substring 的特征提取

基于 Substring 的特征提取可以提取文档中的子词组和超词组，这对于非主题的文本分类很有意义。基于 Substring 的特征提取算法的伪代码如表 2 所列。

表 2 Substring 特征提取算法

```
Input:S[1,…,L]
Output:F[1,…,P],P=L(L+1)/2
for N=1 to L do
  for i=1 to L−N+1 do
    F. insert(S[i,i+N−1])
  end for
end for
return F
```

基于 Substring 特征提取算法的 Mapreduce 化过程与基于 Bigram 特征提取算法的 Mapreduce 化过程类似，只需要在第 3 步按照表 2 中描述的 Substring 特征提权算法提取短句中的 substring 特征即可。

由于 Substring 特征提取算法产生的特征维度很高，并且存在很多冗余和无意义的特征，因此本文在 Substring 特征提取算法的基础上对特征维度进行了约简，筛选掉无意义的特征和冗余特征。定义两个参数 $L$ 和 $H$。$L$ 是最低频度，$H$ 是最高频度，频度低于 $L$ 的特征去掉，频度高于 $H$ 的特征也去掉。筛选冗余特征时，如果特征 $r$ 是特征 $f$ 的冗余特征，那么必定存在如下性质：

(1)$r \subseteq f$，即 $r$ 是 $f$ 的字串；

(2)$TF(r)=TF(f)$，$TF$ 表示字符串的频度，$r$ 和 $f$ 总是同时出现，所以频度一样。

根据冗余性质，基于 Substring 特征约简算法的伪代码如表 3 所列。

表 3 Substring 特征提取上的特征约简算法

```
Input:F[1,…,P],L,H
Output:reduced F
1. For all features F[1,…,P],compute their frequency TF[1,…,P]
2. For all features F[1,…,P],if (TF[i] <L or TF[i]>H) remove F[i]
3. for the remaining feature F[1,…,Q],if F[i]⊆ F[j] and TF[i]=TF[j],
   remove F[i]
```

基于 Substring 特征约简算法的 Mapreduce 化过程主要由两个 MapReduce Job 组成，第一个 MapReduce Job 以 Mapreduce 化的 Substring 特征提取算法的输出数据作为输入数据，针对每个文档，在 Reduce 函数中计算文档中每个特征的频次。第二个 MapReduce Job 的 Map 函数以第一个 MapReduce Job 的输出数据作为输入数据，根据冗余性质和参数 $L$ 和 $H$，对特征值进行筛选。

### 2.2 特征向量加权算法的并行化

特征向量加权就是对特征进行数值化，用量化后的特征值表征特征与文档的相关程度。本文在 Hadoop 平台上实现了两种特征向量加权算法：Bool 算法和 TF-IDF 算法。

### 2.2.1 Bool 算法的并行化

假设 $t$ 表示文档 $d$ 的一个特征，Bool 特征加权算法的公式如下：

$$bool(t,d)=\begin{cases}1, & tf(t,d)>0\\ 0, & tf(t,d)=0\end{cases} \tag{1}$$

式中，$tf(t,d)$ 表示文档 $t$ 中 $d$ 的频度。Bool 算法的 Mapreduce 化伪代码如表 4 所列。

表 4 Bool 算法的 MapReduce 化伪代码

```
Map(Text key,Text value)//key:label@文档 ID,value:特征数据
  HashSet list
  for each index in value
    list. add(index)
  Vector vector=new Vector(dimension);
  for i=0 to dimension do
      if(list. contain(i))
        vector. set(i,1)
      else vector. set(i,0)
  end for
  write(key,vector)
```

其中，dimension 表示特征维度的大小。在 Map 函数中以 lable@文档 ID 为 key，vector 为 value，将 key/value 对发送给 Reduce 函数，Reduce 函数仅做 key/value 值的存储工作。

### 2.2.2 TF-IDF 算法的并行化

TF-IDF 是一种用于信息检索与文本挖掘的加权技术。对于某文档 $d$ 中词语 $t$，词频 TF 的计算公式如下：

$$tf(t,d)=\frac{n(t,d)}{\sum_i n(t_i,d)} \tag{2}$$

式中，$n(t,d)$ 是词语 $t$ 在文档 $d$ 中出现的次数，分母是文档 $d$ 中所有词语出现次数的总和。文档中词语 $t$ 的 IDF(Inverse Document Frequency)值由总文档数目除以包含该词语的文件数目，再取对数得到。IDF 计算公式如下：

$$idf(t)=\log\left(\frac{|D|}{|D_t|}\right) \tag{3}$$

式中，$|D|$ 表示文档总个数，$|D_t|$ 表示包含词语 $t$ 的文档个数。词的 TF-IDF 值的计算公式如下：

$$tf\text{-}idf(t,d)=tf(t,d)*idf(t) \tag{4}$$

TF-IDF 倾向于保留重要的词语和过滤掉常见的词语。

TF-IDF 算法的 MapReduce 化伪代码如表 5 所列。

表 5 TF-IDF 算法的 MapReduce 化伪代码

```
Job1
Mapper(Text,Text,TextPair,DoubleWritable)
map(Text key,Text value)// key:类标@文档 id value:特征索引数据
  HashMap<String,Double) map;
  TextPair tp;
  for eachindex in value do
    compute count of index in the document;//count 是 index 在该文档中的次数
    map. put(index,count);
    tp. set(index,key)
    write(tp,count/length);//length 是文档的长度
  end for
Reducer(TextPair,DoubleWritable,Text,Text)
reduce(TextPair key,Iterable<DoubleWritable>value)
//key. first:index key. second:类标@文档 id values:与 index 对应的 tf 列表
  Double df =0;
  for each tf in values do
    map. put(key. second,tf);
    df=df+1;
  end for
  for each lableId in map. key
    double idf=log(NDoc/df);
    double tfidf=map. get(lableId) * idf;
    write(lableId,key. first@tfidf);
  end for
```

```
Job2
Mapper(Text,Text,Text,Text)
map(Textkey,Textvalue)//key:类标@文档 id value:index@tfidf
  write(key,value);
Reducer(Text,Text,Text,VectorWritable)
reduce(Textkey,Iterable<Text>values)// key:类标@文档 id value:index@tfidf
Vector vector=new Vector(dimension);//dimension 是特征维度
for each value in values do
  int index=values,index;
  double tfidf=values,tfidf;
end for
write(key,vector);
```

在上面的算法中，Job1 主要针对每个文档中的每个特征计算它的 TF-IDF 值。Job2 主要实现将(文档 id，特征@tfidf)形式的数据转换为(文档 id，vector)的形式。

## 2.3 情感分类算法的并行化

本文实现了基于空间向量模型 Rocchio 算法[12] 和 KNN 算法的 Mapreduce 化。

### 2.3.1 Rocchio 情感分类算法的并行化

Rocchio 算法的基本思想是首先对文档特征向量加权，然后为每一类文档 $C_j$ 建立一个原型向量 $\vec{c}_j$。

$$\vec{c}_j=\alpha\frac{1}{|C_j|}\sum_{\vec{d}\in C_j}\frac{\vec{d}}{\|\vec{d}\|}-\beta\frac{1}{|D-C_j|}\sum_{\vec{b}\in D-C_j}\frac{\vec{b}}{\|\vec{b}\|} \tag{5}$$

式中，$\alpha,\beta$ 为调节因子，用来协调正例和负例的相对影响程度，$|C_j|$ 是类别 $j$ 中文档的个数，$\|\vec{d}\|$ 是 $\vec{d}$ 的欧式长度。根据测试文档特征向量 $\vec{t}$ 与每个 $\vec{c}_j$ 的余弦相似度判定测试文档特征向量 $\vec{t}$ 的类别，即：

$$H_{TFIDF}(\vec{t})=\underset{\vec{c}_j\in C}{\operatorname{argmax}}\cos(\vec{t},\vec{c}_j)=\underset{\vec{c}_j\in C}{\operatorname{argmax}}\frac{\vec{t}\cdot\vec{c}_j}{\|\vec{c}_j\|\cdot\|\vec{t}\|} \tag{6}$$

在 MapReduce 化 Rocchio 情感分类算法时，考虑算法实现的复杂度，本文取 $\alpha=1,\beta=0$。Rocchio 情感分类算法的 MapReduce 化伪代码如表 6 所列。

表 6 Rocchio 算法的 MapReduce 化伪代码

```
Job1 Training Model
Mapper(Text,VectorWritable,Text,VectorWritable)
map (Text key,VectorWritable value)//key:类标@文档 id value:特征加权向量
  String label=key. split("@")[0];
  write(label,label value);
Reducer(Text,VectorWritable,Text VectorWritable)
reduce(Text key,Iterable<VectorWritable> values)
  Vector vector =new Vector(dimension);
  Int count=0;
  for each value in values do
    vector=vector+value;
    count++;
  end for
  vector =vector/count ;
  write(key,vector);
```

```
Job 2:classification
Mapper(Text,VectorWritable,Text,Text)
Matrix matrix;
String label[2];
```

```
setup()
  读取模型文件,把模型中的向量存储在 matrix 中,把 key 存储在 label 中;
map (Text key, VectorWritable value )//待分类数据,key:类标@文档 id
    value:特征加权向量
  double max=0;
  String predictLabel;
  for each vector in matrix do
    double cos=cosine(value,vector);
    if( max< cos)
    max=cos;
    predictLabel =label[vector];
end for
write(predictLabel,key );// predictLabel:预测类标
```

其中 Job1 用于训练模型,计算类别文档的原型向量;Job2 计算测试文档的特征向量与每个原型向量的余弦相似度,判定测试文档的类标。

### 2.3.2 KNN 情感分类算法的并行化

KNN 分类算法依据最邻近的 $K$ 个样本的类别来决定待分类样本所属类别。本文采用余弦相似度度量文本特征向量之间的距离。KNN 情感分类算法的 Mapreduce 化伪代码如表 7 所列。

表 7 KNN 算法的 Mapreduce 化伪代码

```
Mapper(Text,VectorWritable,Text,Text)
Matrix matrix;
String[ ] label;
map(Text key,VectorWritable value )//训练数据
  存储 key 到 label 中,存储 value 到 matrix 中;
cleanup( )
  计算测试向量与所有的训练数据的余弦相似度,选择前 k 个相似度最大的
  训练数据和测试向量一起发送
Reducer(Text,Text,Text,Text)
reduce(Text key,Iterable(Text) values )
  (1)从 values 中提取出距离,选择 k 个距离最小的训练数据;
  (2)对于 k 个数据样本,把多数样本所属的类别作为该测试数据的类别;
write( label,key );
```

上述 reduce 函数中的 key 为测试样本的 key,values 为 map 函数发送的 $k$ 个与测试样本距离最近的样本类标和对应的相似度。

## 3 实验及其结果分析

本文在情感语料数据集上进行了实验,数据集为 ChnSentiCorp-Book-del-4000(书籍领域)和 ChnSentiCorp-NB-del-4000(笔记本领域)。对于每个情感语料数据集,设置训练数据与测试数据集的大小比例为 9:1,使用交叉验证,取分类精度的平均值为分类算法最后的精度。

实验环境具有 6 个节点,每个节点的配置为内存 4G,8 核处理器 Intel(R) Core™ i7 CPU 860@2.80GHz。每个节点上操作系统为 Ubuntu 10.04,Hadoop 版本为 0.20.2。

对于每个情感语料数据集,在特征向量提取、特征向量加权和分类等步骤上分别选择不同的算法进行组合,对分类精度和每个算法的时间代价进行了统计。在 KNN 分类算法中,取 $K=5$。对于基于 Substring 特征提取算法,在约简后的特征向量上进行特征向量加权和分类。为了使特征词具有区分功能以及在同等特征规模上对比分类算法,在实验中设定 $L$ 为 3,$H$ 设置为文档总数的一半。在不同情感语料数据集上,不同算法组合得到的情感分类精度如表 8、表 9 所列。

表 8 在 ChnSentiCorp-Book-del-4000 上的精度

| 特征提取方法 | 特征向量加权 | 分类方法 | 精度 |
|---|---|---|---|
| Bigram | Bool | Knn | 46% |
| Bigram | Bool | Rocchio | 89.5% |
| Bigram | Tfidf | Knn | 68% |
| Bigram | Tfidf | Rocchio | 91.0% |
| Sentiment Lexicon | Bool | Knn | 70.8% |
| Sentiment Lexicon | Bool | Rocchio | 71.3% |
| Sentiment Lexicon | Tfidf | Knn | 64.7% |
| Sentiment Lexicon | Tfidf | Rocchio | 65.0% |
| Substring | Bool | Knn | 86.1% |
| Substring | Bool | Rocchio | 87.1% |
| Substring | Tfidf | Knn | 74.9% |
| Substring | Tfidf | Rocchio | 90.0% |

表 9 在 ChnSentiCorp-NB-del-4000 上的精度

| 特征提取方法 | 特征向量加权 | 分类方法 | 精度 |
|---|---|---|---|
| Bigram | Bool | Knn | 84% |
| Bigram | Bool | Rocchio | 82.9% |
| Bigram | Tfidf | Knn | 78.4% |
| Bigram | Tfidf | Rocchio | 87.7% |
| Sentiment Lexicon | Bool | Knn | 78.9% |
| Sentiment Lexicon | Bool | Rocchio | 78.4% |
| Sentiment Lexicon | Tfidf | Knn | 74.1% |
| Sentiment Lexicon | Tfidf | Rocchio | 81.6% |
| Substring | Bool | Knn | 79.5% |
| Substring | Bool | Rocchio | 68.8% |
| Substring | Tfidf | Knn | 82.9% |
| Substring | Tfidf | Rocchio | 85.3% |

表 8、表 9 的实验结果表明,尽管基于情感词典的特征提取算法很直观,但是取得的精度并不高,这是由于一般情况下评论本身就很短,出现情感词的几率相对较小,并且随着新词的出现,导致情感词典中的情感词并不全面。Bigram 和 Substring 特征提取方法整体上优于情感词典特征提取方法,验证了其被广泛应用的原因。

另外,在各种组合下 Rocchio 算法都优于 KNN 算法,显示了 Rocchio 算法分类精度上的优势。对比情感分类任务在不同情感语料库上的精度,笔记本语料数据集上的精度高于书籍语料库上的精度,这是由于书籍语料库上抽取的特征数量高于笔记本语料数据集,因而文档特征词分布稀疏,影响了分类精度。

不同情感语料数据集上每个算法的运行时间(单位:s)如表 10、表 11 所列。

表 10 各语料库在各个特征提取算法上的时间代价

| 语料库 \ 特征提取算法 | Sentiment Lexicon | Bigram | Substring+特征约简 |
|---|---|---|---|
| ChnSentiCorp-Book-del-4000 | 33 | 30 | 2682 |
| ChnSentiCorp-NB-del-4000 | 30 | 23 | 2323 |

表 11 各语料库在各个特征向量加权算法上的时间代价

| 语料库 \ 特征向量加权算法 | Bool | Tfidf |
|---|---|---|
| ChnSentiCorp-Book-del-4000 | 90 | 107 |
| ChnSentiCorp-NB-del-4000 | 80 | 104 |

表 10 实验结果显示 Bigram 特征提取在时间上是最优的,基于情感词典的特征提取次之,Substring 特征提取算法时间开销最大。表 11 的实验结果表明,Bool 特征向量加权算法在时间上略优于 TF-IDF,但是从情感分类的角度看,如果要兼顾精度和时间代价,在选择加权算法时应使用 TF-IDF 加权算法。表 12 的实验结果表明,KNN 时间开销小于 Roc-

chio 算法，但是分类精度低于 Rocchio 算法。

表 12　各语料库在各个分类算法上的时间代价

| 语料库 \ 分类算法 | Knn | Rocchio |
|---|---|---|
| ChnSentiCorp-Book-del-4000 | 59 | 83 |
| ChnSentiCorp-NB-del-4000 | 60 | 83 |

以上的实验结果表明，在选择特征提取、特征向量加权和情感分类算法时，需要根据用户的需求和数据集特征来进行算法组合，在分类精度和时间代价之间进行权衡。数据量大应该选择时间代价小的算法；如果数据量小，就选择精度高的算法，因为数据越小，各种算法的时间代价的差异越不明显。

**结束语**　云计算这一新型计算模型的兴起，为解决海量数据情感分类任务提供了可能性。本文利用 Hadoop 的 MapReduce 和 HDFS 技术，针对情感分类任务中特征提取、特征向量加权和分类等 3 个子任务，实现了 3 个子任务中涉及算法的 MapReduce 化。在情感语料数据集上进行了实验，针对每个子任务，选择不同的并行化算法进行组合，对每种组合下的情感分类精度及其每种算法的时间开销进行了对比分析，验证了本文实现算法的有效性，同时为用户选择不同算法实现情感分类任务提供了有价值的参考信息。

在本文工作的基础上，我们下一步将研究云平台下智能化的数据挖掘服务技术，根据用户的需求，考虑同类型不同算法的各种特征，如性能、精度等，自动化、智能地实现云平台下数据挖掘算法的组合，为用户提供智慧数据挖掘服务。

## 参 考 文 献


[1] Pang B, Lee L, Vaithyanathan S. Thumbs up? Sentiment Classification Using Machine Learning Techniques[C]// Proceedings of the EMNLP'02. 2002:79-86

[2] Pang B, Lee L. A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts[C]// Proceeding of ACL. 2004:271-278

[3] Dave K, Lawrence S, Pennock D. Mining the peanut gallery: opinion extraction and semantic classification of product reviews [C]// Proceedings of WWW2003. 2003

[4] Mullen T, Collier N. Sentiment analysis using support vector machines with diverse information sources[C]// Proceedings of EMNLP'2004. 2004

[5] Li J, Sun M. Experimental Study on Sentiment Classification of Chinese Review using Machine Learning Techniques[C]// Proceedings of IEEE NLP-KE'2007. 2007

[6] Zhai Zhong-wu, Xu Hua, Li Juu, et al. Sentiment Classification for Chinese Reviews Based on Key Substring Features[C]// Proceedings of Natural Language Processing and Knowledge Engineering. 2009:24-27

[7] Devitt A, Ahmad K. Sentiment polarity identification in financial news: a cohesion-based approach[C]// Proceedings of ACL. 2007:984-991

[8] Shein K P P, Nyunt T T S. Sentiment classification based on Ontology and SVM Classifier[C]// Proceedings of ICCSN. 2010:169-172

[9] Zhai Zhong-wu, Xu Hua, Kang Ba-da, et al. Exploiting effective features for Chinese sentiment classification[J]. Expert Syst. Appl., 2011, 38(8):9139-9146

[10] Jeffrey D, Sanjay G. MapReduce: simplified data processing on large clusters[J]. Commun. ACM, 2008, 51(1):107-113

[11] Rocchio J. Relevance Feedback in Information Retrieval[M]. The SMART Retrieval System: Experiments in Automatic Document Processing, Chapter 14, Prentice-Hall, 1971:313-323


(上接第 205 页)


[2] Papageorgiou Elpiniki I, Wojciech F. Application of Evolutionary Fuzzy Cognitive Maps for Prediction of Pulmonary Infections [J]. IEEE Transactions on Information Technology in Biomedicine, 2012, 16(1):143-149

[3] Michael G. Fuzzy Cognitive Strategic Maps in Business Process Performance Measurement[J]. Expert Systems with Applications, 2013, 40(1):1-14

[4] Kontogianni Areti D, Papageorgiou Elpiniki I, Christos T. How do you perceive environmental change? Fuzzy Cognitive Mapping informing stakeholder analysis for environmental policy making and non-market valuation[J]. Applied Soft Computing, 2012, 12(12):3725-3735

[5] Kumar M V, Laurens B, Papageorgiou Elpiniki I, et al. Fuzzy cognitive maps and cellular automata: An evolutionary approach for social systems modeling[J]. Applied Soft Computing, 2012, 12(12):3771-3784

[6] Peng Zhen, Yang Bing-ru, Wu Li-feng. The Construction of Fuzzy Cognitive Map Classification Model based on Relational Database[J]. International Journal of Digital Content Technology and its Applications, 2011, 5(5):33-41

[7] 张桂云，马希荣，杨炳儒. 复杂模糊认知图的分解研究[J]. 计算机科学，2007，34(4)：129-132

[8] Peng Zhen, Wu Li-feng. Two-level Fuzzy Cognitive Map Mining for Text Categorization[J]. International Journal of Digital Content Technology and its Applications, 2012, 6(2):296-302

[9] Lin Chun-mei. Combination study of multi fuzzy cognitive map [A]// Knowledge-Based Intelligent Information and Engineering Systems, 2008 [C]. Zagreb, 2008:332-339

[10] Wojciech S, Lukasz K, Witold P. A Divide and Conquer Method for Learning Large Fuzzy Cognitive Maps[J]. Fuzzy Sets and Systems, 2010, 161:2515-2532

[11] 骆祥峰. 认知图理论及其在图像分析与理解中的应用[D]. 合肥：合肥工业大学，2003

[12] 胡可云，陆玉昌，石纯一. 基于概念格的分类与关联规则的集成挖掘方法[J]. 软件学报，2000，11(11)：1478-1484

[13] Papageorgiou E I, Stylios C D, Groumpos P P. Active Hebbian Learning Algorithm to Train Fuzzy Cognitive Maps[J]. International Journal of Approximate Reasoning, 2004, 37(3):219-249

[14] Papageorgiou E I, Stylios C D. Fuzzy Cognitive Map Learning based on Nonlinear Hebbian Rule[A]// Artificial Intelligence, 2003[C]. Australia, 2003:256-268

[15] Sudjianto A, Hassoun M H. Statistical Basis of Nonlinear Hebbian Learning and Application to Clustering[J]. Neural Networks, 1995, 8(5):707-715

[16] Wojciech S, Lukasz K, Witold P, et al. Genetic Learning of Fuzzy Cognitive Maps[J]. Fuzzy Sets and Systems, 2005, 153(3):371-401

[17] Peng Zhen, Yang Bing-ru. Research on Balance of Fuzzy Cognitive Map[J]. Scientific Research and Essays, 2011, 6(31):6444-6447