贝叶斯推断在MCDB分布式平台上的实现

周志敏[1] 高申勇[2]
(浙江水利水电学院计算机与信息工程系 杭州 310018)[1] (浙江大学电气工程学院 杭州 310058)[2]

摘 要 提出了应用贝叶斯统计方法在分布式数据库MCDB上处理超大规模数据的实现方法,并以贝叶斯线性回归、话题模型的LDA和狄利克雷过程的聚类算法为例进行了论证。用户可以通过SQL语言定义变量之间的关系进行模拟。探索了一种使用简洁的SQL设计大规模统计学习系统的方法,其利用MCDB能够自动解决并行化和资源优化问题,以获得高性能的并行处理能力。

关键词 贝叶斯推断,并行算法,SQL,分布式系统
中图法分类号 TP302.7 **文献标识码** A

Implementation of Bayesian Inference on MCDB Distributed System

ZHOU Zhi-min[1] GAO Shen-yong[2]
(Department of Computer Science, Zhejiang University of Water Resources and Electric Power, Hangzhou 310018, China)[1]
(College of Electrical Engineering, Zhejiang University, Hangzhou 310058, China)[2]

Abstract This paper described how the Monte Carlo database system (MCDB) can be used to easily implement Bayesian inference via Markov chain Monte Carlo (MCMC) over very large datasets. Linear Bayesian regression, LDA and Dirichlet clustering were used as examples to demonstrate this task. To implement an MCMC simulation in MCDB, a programmer specifies dependencies among variables and how they parameterize one another using the SQL language. This paper devised a simple scheme for developing large scale machine learning systems with SQL, which with the help of MCDB, can automaticly deal with parallelization and optimization problems, to achieve high efficiency in computation.

Keywords Bayesian inference, Parallel algorithms, SQL, Distributed system

1 引言

在统计推断和机器学习领域,贝叶斯方法由于其整洁的数学意义、良好的计算性质、一致的判断准则与预测方法为人们所欢迎。其中的非参数方法则由于经常需要借助随机算法进行估计而效率很低,应用领域被大大限制。对马尔科夫蒙特卡罗方法(MCMC)进行优化和并行化,以及改进算法的可扩展性,将其应用于现有的优秀并行计算平台(例如Apache Hadoop[1]),将有助于很多工程应用领域的发展[2]。

2 蒙特卡罗数据库系统

蒙特卡罗数据库系统(MCDB)[3]是Rice大学设计的基于SQL语言的数据库系统,支持大规模的随机分析。MCDB的设计和实现在一系列的文章中都有描述(其中最完整的综述可以参见文献[3])。除一般数据库表之外,MCDB允许用户定义随机数据表单。在随机数据表单中,一些属性可以定义为满足特定概率分布的随机变量。这一过程可以重复多次以获得模拟的观测结果。

MCDB为处理大规模数据的机器学习算法的实现也带来了很多便利。MCDB允许根据马尔科夫链进行采样,即在每一次蒙特卡罗模拟中,MCDB可以生成一条以数据库实例为变量的马尔科夫链,亦即一个系列的数据库。因此,MCDB可以应用于马尔科夫蒙特卡罗(MCMC)模拟的实现,从而进行大规模数据集的贝叶斯推断。

3 MCDB的马尔科夫蒙特卡罗算法

采用MCDB进行MCMC,可以有效地并行运行多个独立的马尔科夫链,同时比较不同的马尔科夫链来分析描述后验概率分布的相关统计量,或进行最大后验估计[5]。

3.1 贝叶斯线性回归

为解释MCDB在大规模贝叶斯分析中的应用方式,以简单的贝叶斯线性回归为例进行分析[5]。首先描述模型和推断的过程,然后讨论它在并行系统中的实现方法。

3.1.1 模型和推断

假设有一个非常大的数据集 D,样本大小为 n。其中每个元组含有 d 个回归因子,数据的形式为 $\langle y_i, x_{i1}, x_{i2}, \cdots, x_{id}\rangle$,其中第 j 个回归因子的系数为 c_j。假设有如下生成过程:

1. $\sigma^2 \sim \mathcal{IG}(1,1)$;

2. 对每个 $i \in \{1,\cdots,n\}$: $c_i \sim \mathcal{N}(0,\alpha)$;

到稿日期:2012-09-25 返修日期:2012-12-14 本文受国家自然科学基金(61272539)资助。
周志敏(1966-),女,硕士,副教授,主要研究方向为信息系统分析与设计、分布式计算,E-mail:569378975@qq.com;高申勇(1981-),男,博士生,主要研究方向为并行计算处理。

3. 对每个 $i\in\{1,\cdots,n\}$，$j\in\{1,\cdots,d\}$：$r_{ij}\sim\mathcal{M}(x_{ij}\times c_j,\sigma)$；

4. 对每个 $i\in\{1,\cdots,n\}$：$y_i\leftarrow\sum_j r_{ij}$。

其中，$\mathcal{IG}$为逆伽马分布，$\mathcal{N}$为正态分布，α 为常数。这个过程并没有直接随机预测 y_i，而是将这个过程分解成了 d 项 r_{ij}。可以证明如果给定合适的先验概率，这个过程和直接采样具有相同的后验概率 $P(\{y_i\}|\{r_{ij}\})$。

给定一个数据集 D 和之前定义的过程，需要学习获得回归系数集合$\{c_j\}$的后验分布。首先，要导出上述过程的概率密度函数：

$$P(\{r_{ij}\}\cup\{c_{ij}\}|D)\propto\prod_{ij}\mathcal{M}(r_{ij}|x_{ij}c_j,\sigma)\times\prod_j\mathcal{M}(c_j|0,\alpha)\times\mathcal{IG}(\sigma^2|1,1)\times\begin{cases}1, & \text{若}\forall i,y_i=\sum_j r_{ij}\\ 0, & \text{其他}\end{cases}$$

根据该概率密度函数可以导出吉布斯采样器。在每次迭代中：

$$P(c_j|\cdot)\propto e^{\left[(2\sigma^2)^{-1}\sum_i(2r_{ij}x_{ij}c_j-x_{ij}^2c_j^2)\right]}\times\mathcal{M}(c_j|0,\alpha)$$

$$P(r_{i1},r_{i1},\cdots,r_{ik}|\cdot)\propto\prod_j\mathcal{N}(r_{ij}|x_{ij}\times c_j,\sigma)\times\begin{cases}1, & \text{若}\forall i,y_i=\sum_j r_{ij}\\ 0, & \text{其他}\end{cases}$$

$$P(\sigma^2|\cdot)=\mathcal{IG}(1+\frac{nd}{2},1+\frac{1}{2}\sum_{ij}(r_{ij}-x_{ij}c_j)^2)$$

3.1.2 并行处理吉布斯采样器的构建

本节描述如何采用 MCDB 的 SQL 语言定义吉布斯采样器。假设真实数据存在两张数据表中：

```
regressors(i,j,val)   %x_ij的取值
outcomes(i,val)   %y_i 的取值
```

此外，还需要建立 3 张随机数据表，分别用来存放 r_{ij}、回归系数和 σ。

初始化：首先初始化回归系数表，通过下面语句实现。

```
CREATE TABLE coefs[0] (j,val) AS
  FOR EACH dim IN (SELECT DISTINCT (j) FROM regressors)
    WITH sampledVal AS Normal (SELECT 0.0,100.0)
  SELECT dim.j,sampledVal.val
  FROM sampledVal
```

这段代码可以生成随机数表 coefs[0]，其中[0]表示是表格的初始模式。通过 SELECT DISTINCT (j) FROM regressors 选取一组两两不等的维数指标。随后遍历指标集，并生成满足正态分布的一组变量。这个过程可以通过 MCDB 中系统自带的 Normal VG 函数实现。Normal VG 函数的参数由 SELECT 0.0,100.0 提供。最后 SELECT 函数对每个维度选取生成的 sampledVal，并将结果存入 coefs[0]。

更新权重：更新 r_{ij} 数据表的代码如下：

```
CREATE TABLE contribs[iter] (i,j,val) AS
  FOR EACH outcome IN (SELECT i,val FROM outcomes)
    WITH sampledVals AS ContribsUpdate (
      (SELECT outcome.val)
      (SELECT regressors.j,regressors.val * curCoefs.val
      FROM regressors,coefs[iter-1] AS curCoefs
      WHERE regressors.i=outcome.i AND
            curCoefs.j=regressors.j)
      (SELECT val
      FROM standardDev[iter-1]))
    SELECT outcome.i,sampledVals.j,sampledVals.val
    FROM sampledVals
```

这段代码的 VG 函数接受的参数与吉布斯采样器中定义的相同。

此外需要生成用户定义的 VG 函数 ContribsUpdate。这个 VG 函数从 d 维协方差为 0 的多元正态分布空间中抽取样本，并使所有的维度加和等于 y_i。生成的方法类似 stick breaking 过程。首先通过拒绝采样生成 r_{i1} 和 $\sum_{j=2}^{d} r_{ij}$，并保证其加和为 y_i。随后依次生成 r_{ij}。

更新回归系数：生成回归系数的代码如下：

```
CREATE TABLE coefs[iter] (j,val) AS
  FOR EACH dim IN (SELECT DISTINCT (j) FROM regressors)
  WITH sampledVal AS CoefUpdate (
    (SELECT 1.0 / (2.0 * std.val * std.val)
    FROM standardDev[iter-1] AS std)
      (SELECT SUM (2.0 * contrib.val * regressor.val),
      SUM (-regressors.val * regressors.val)
      FROM regressors,contribs[iter] AS contrib
      WHERE regressors.j=dim.j AND
            contrib.j=dim.j AND
            regressors.i=contrib.i)
    (SELECT ALPHA))
  SELECT dim.j,sampledVal.val
  FROM sampledVal
```

和生成 r_{ij} 的代码一样，调用了用户定义的 VG 函数，这里是 CoefUpdate。我们将估计之前等式的概率密度函数的充分统计量当作参数。其中 3 段子查询分别计算 $(2\sigma^2)^{-1}$，$\sum_j 2r_{ij}x_{ij}$ 和 $\sum_j -x_{ij}^2$。ALPHA 为某用户定义的常数。VG 函数同样是通过拒绝采样的方法实现的，由于只需要对 c_j 采样，这一过程会简单很多。

更新标准差：最后，需要更新标准差 σ。可以采用 MCDB 中的逆伽玛分布的库来实现：

```
CREATE TABLE standardDev[iter] (val) AS
  WITH sampledVal AS InvGamma (
    (SELECT 1.0+0.5 * COUNT (DISTINCT regressors.i) *
            COUNT (DISTINCT regressors.j)
    FROM regressors)
    (SELECT 1.0+0.5 * SUM(pow(contrib.val-regressors.val *
            coef.val,2))
  FROM regressors,contribs[iter] AS contrib,coefs[iter] AS coef
  WHERE contrib.i=regressors.i AND contrib.j=regressors.j
        AND coef.j=regressors.j))
  SELECT SQRT (val)
  FROM sampledVal
```

其中两个子查询就对应了吉布斯采样器第三个等式的两个参数。

3.2 贝叶斯话题模型

LDA(Latent Dirichlet Allocation)模型[7]是贝叶斯话题模型中的流行算法。下面讨论如何使用 MCDB 实现 LDA 模型的贝叶斯推断。LDA 模型具有很广泛的应用范畴，比如文章分类、文章索引和降维问题。我们采用完整的吉布斯采样器来进行求解，而非采用常用的冲突采样器("collapsed" sampler)[8]。因为完整采样的信息量更有助于展示如何实现分布式贝叶斯推断方法。

3.2.1 模型和推断

在 LDA 中，文章代表一组隐含变量(话题)的混合模

型[8]。文章中的单词则是从这样的话题模型中选取的。假设我们的模型有 n 篇文章，共含有 m 个不同的单词，有 t 个话题，并且已知其中第 j 篇文章有 n_j 个词。给定参数 α 和 β，构造 LDA 蕴含的过程如下：

1. 对每个 $i\in\{1,\cdots,t\}$：$\Psi\sim D_m(\beta)$。

2. 对每个 $j\in\{1,\cdots,n\}$：

a) $\Theta_j\sim D_t(\alpha)$；

b) $z_i\sim\mathcal{M}(n_j,\Theta_j)$；

c)对每个 $k\in\{1,\cdots,t\}$：$w_{ij}\sim\mathcal{M}(z_{jk},\Psi_k)$

这个过程总结如下：在第 1 步中，首先生成 t 行 m 列的矩阵 Ψ。其中每一行 Ψ_i 是一张概率的表，每个元素 Ψ_{id} 为话题 i 产生单词 d 的概率。Ψ_i 从狄历克雷分布 $D_t(\alpha)$ 中采样。我们采用 Ψ_{*d} 来表示话题产生单词 d 的概率的列向量。

然后，对每篇文章 j 生成一个概率向量 Θ_j，这个向量控制了话题在文章中被选择的概率。在第 j 篇文章中，每个话题控制单词选择的次数满足参数为 n_j 和 Θ_j 的多项式分布 $\mathcal{M}(n_j,\Theta_j)$。$z_{jk}$ 表示通过话题 k 加入文章 j 的单词的个数。

最后，w_j 是一个 t 行 m 列的矩阵。w_{jkd} 表示文章 j 中通过话题 k 选入的单词 d 的个数。其中 w_{jk} 表示 w_j 中的第 k 列，则 w_{jk} 满足参数为 z_{jk}，Ψ_k 的多项式分布。在这个过程完成之后，可以计算出第 j 篇文章的语料(corpus)为 $d_j=\sum_k w_{jk}$。

要注意的是，虽然这个过程与原始 LDA 论文[9]中所描述的一致，但为了更方便地构建吉布斯采样器，我们对矩阵和向量进行了转置。吉布斯采样器的更新方程推导如下：

$$\Psi_i\sim D_m(\beta+\sum_j w_{ji})$$

$$\Theta_j\sim D_t(\alpha+z_j)$$

$$w_{j*d}\sim\mathcal{M}(d_{jd},\Theta_j\times\Psi_{*d})$$

其中，$\Theta_j\times\Psi_{*d}$ 代表矩阵相同位置元素对应的乘法。当矩阵的范数不是 1 时，需要进行伸缩以适合多项式分布的参数。由于 z_{jk} 可以通过 $\sum_d w_{jkd}$ 进行计算，我们没有必要对 z_j 进行单独的采样。

3.2.2 并行处理吉布斯采样器的构建

在 MCDB 中实现上述吉布斯采样器，需要保存 4 张数据表。前两张用来存放话题和文章的指示器，第三张存放狄利克雷分布的超参数 α 和 β，第四张用来记录每个单词在文章中出现的次数。

```
topics(topicID)
documents(docID)
hyperparameters(alpha,beta)  %超参数α和β
wordInDoc(docID,wordID,count)  %文章中的单词
```

除此之外，还需要 3 张随机数据表单，一张 psi 用来存放矩阵 Ψ，一张 theta 用来存放矩阵 Θ，一张 w 用来存放所有的 w 矩阵。

```
psi[i](topicID,wordID,prob)
theta[i](docID,topicID,prob)
w[i](docID,topicID,wordID,count)
```

初始化：首先，需要对 Θ 和 w 矩阵进行初始化。

更新 Θ：

```
CREATE TABLE theta[0] (docID,topicID,prob) AS
  FOR EACH d IN documents
    WITH Newprobs AS Dirichlet
      (SELECT topicID,alpha
      FROM topics,hyperparameters)
    SELECT d. docID,np. outID,np. probability
    FROM Newprobs AS np;
```

其中代码通过 FOR EACH d IN documents 遍历所有的文章。对每个 d，VG 函数 Dirichlet 通过 WITH 来调用。VG 函数的输入变量为每个维度的值都是 α 的向量，输出的结果为 outID(维度的指标)和 probability(对应该维度的概率取值)两个属性。最后的 SELECT 函数将 VG 函数的输出结果在临时的表单中整合起来。所有的表单连接起来组成了 theta[0]。

更新 w：

```
CREATE TABLE w[0] (docID,wordID,topicID,count) AS
  FOR EACH dw IN wordInDoc
    WITH TC AS Multinomial (
      (SELECT tm. topicID,tm. probability
      FROM theta[0] AS tm
      WHERE tm. docID=dw. docID),
      (SELECT dw. count))
    SELECT dw. docID,dw. wordID,tc. outID,tc. count
    FROM TC AS tc;
```

w 通过采样进行初始化，对每篇文章的每个不同的单词，通过 VG 函数 Multinomial 获得每个话题产生该词的次数。

更新 Ψ 向量：

```
CREATE TABLE psi[i] (topicID,wordID,prob) AS
  FOR EACH t IN topics
    WITH Newprobs AS Dirichlet
      (SELECT pw. wordID, sum(pw. count)+hyperparameters.
          beta
      FROM w[i] AS pw,hyperparameters
      WHERE pw. topicID=t. topicID
      GROUP BY pw. wordID,hyperparameters. beta)
    SELECT t. topicID,np. outID,np. probability
    FROM Newprobs AS np;
```

表单 topics 被遍历，对每个话题，通过 VG 函数 Dirichlet 生成相应的 Ψ 向量。生成的方法为吉布斯采样器第一个等式中的共轭狄利柯雷分布。

更新 Θ 向量：

```
CREATE TABLE theta[i] (docID,topicID,prob) AS
  FOR EACH d IN documents
    WITH Newprobs AS Dirichlet
      (SELECT pw. topicID,sum(pw. count)+hyperparameters. al-
          pha
      FROM w[i-1] as pw,topics AS t,hyperparameters
      WHERE pw. docID=d. docID AND pw. topicID=t. topicID
      GROUP BY pw. topicID,hyperparameters. alpha)
    SELECT d. docID,np. outID,np. prob
    FROM Newprobs AS np;
```

对每个文章，通过 VG 函数 Dirichlet 生成相应的 Θ 向量。生成的方法为吉布斯采样器第二个等式中的共轭狄利柯雷分布。

更新 w 向量：

```
CREATE TABLE w[i] (docID,wordID,topicID,count) AS
  FOR EACH dw IN wordInDoc
    WITH TC AS Multinomial(
      (SELECT tm. topicID,wpt. prob * tm. prob
      FROM psi[i-1] AS wpt,theta[i] AS tm
      WHERE wpt. wordID=dw. wordID AND
          wpt. topicID=tm. topicID AND
          tm. docID=dw. count),
      (SELECT dw. count))
    SELECT dw. docID,dw. wordID,tc. outID,tc. count
```

```
FROM TC AS tc;
```

更新 w 的方法与初始化 w 时类似，区别仅是生成文章中每个单词词频的多项式分布的参数概率的计算方法为吉布斯采样器的第三个等式：文章中出现该话题的概率乘以该话题影响出现该词的概率。

4 实验结果分析

当模拟的过程被定义之后，代码就可以被导入 MCDB。确定迭代次数之后，就可以进行计算了。当模拟完成之后，可以通过 SQL 语句来检查执行结果。比如，如果要查看不同维度对预测结果影响的重要性，可以执行这条语句：

```
SELECT j, AVG (regressors. val * coef. val * outcomes. val) -
AVG (regressors. val * coef. val) * AVG (outcomes. val)
FROM coefs[40] as coef, regressors, outcomes
WHERE regressors. i = outcomes. i AND
regressors. j = coef. j
GROUP BY j
```

这条查询计算了预测结果和每个维度变量的相关性。一般情况下，相关系数越高，则表示对结果的影响越大。

为了系统地评估基于 MCDB 的 LDA 算法，我们进行集合混杂度(Set Perplexity)的测试，计算方法为：$Perp(x^{test}) = \exp(-(1/N^{test})\log p(x^{test}))$[10]。对每篇文章，随机选取一半的词作为训练集，另一半作为测试集。文章的混合模型 θ_{kj} 通过训练集学习获得，指数概率通过这个混合模型和测试集的单词频率计算得到，以保证测试集中的单词在测试前不会被模型接收。

我们采用以下 3 个数据集比较 LDA(在一个处理器上的吉布斯采样器)和我们的分布式算法：KOS(dailykos. com)，NIPS(books. nips. cc)和 NYTIMES(ldc. upenn. edu)。每个数据集被分成训练集和测试集，集合的大小在表 1 中给出。

表 1 集合混杂度和执行速度数据集参数大小

	KOS	NIPS	NYTIMES
n_{train}	3000	1500	300000
m	6906	12419	102660
N	410000	1900000	100000000
n_{test}	430	184	34658

对每个语料库，m 是不重复的单词的个数，N 是总单词数。针对每个语料库，我们计算了两种算法在话题数为 K、处理器个数为 P 时分布式模型的集合混杂度，结果如图 1 和图 2所示。其中从左到右 3 列的处理器个数分别为 1,10,100。

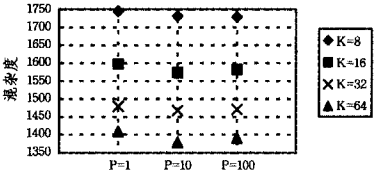


图 1 KOS 数据集不同处理器个数下的模型混杂度测试结果

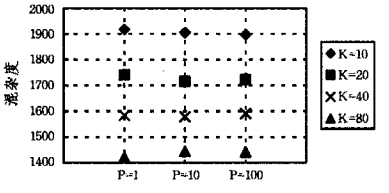


图 2 NIPS 数据集不同处理器个数下的模型混杂度测试结果

图 1、图 2 显示当样本的个数固定时，两种算法的混杂度的计算结果基本相同。图中显示了采用一个处理器和多个处理器下的测试集合混杂度。MCDB-LDA 的混杂度在 KOS 数据集处理器增加到 1000、NIPS 数据集增加到 500 的过程中基本保持为常数。值得一提的是，虽然没有正式的收敛性保证，每个测试数据集上分布式算法的实例都会收敛到满意的结果。

为了合理决定并行方法的可行性，还需要检查并行算法的收敛效率。如果并行算法的收敛慢于一般的方法，在并行化的过程中计算能力的提升会受到影响。然而实验结果显示并行的收敛速率与原始方法一致，如图 3、图 4 所示。

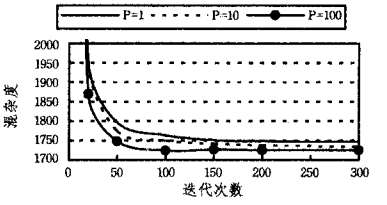


图 3 混杂度收敛速率

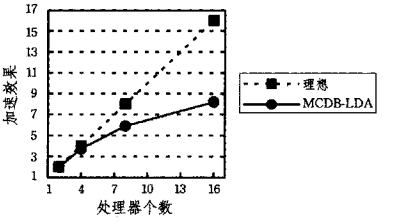


图 4 并行速率提升结果

考虑到非参数贝叶斯学习对随机算法的依赖性，当处理非常大的数据集时，每次迭代的时间会变得很长，进行 1000 次马尔科夫蒙特卡罗模拟将会变得非常困难，这的确是 MCDB 当前遇到的问题。然而，根据实际实验结果，采用吉布斯采样器，对于可以收敛的问题实例，一般的机器学习算法(LDA、混合高斯模型、图分解、贝叶斯矩阵分解等)会在 50 到 100 次迭代中收敛。

结束语 MCDB 已经在底层封装了并行算法。使用 MCDB 的 SQL 设计大规模统计学习系统，用户可以通过 SQL 语言定义变量之间的关系，不需要定义代码如何在多台机器上并行执行，MCDB 可以自动解决并行化和计算资源的优化问题。如此系统就可以简单地获得高性能并行处理能力。

参 考 文 献

[1] Drost I, Dunning T, Eastman J, et al. Introduction to Apache Mahout [Z]. mahout. apache. org. 2011

[2] Lunn D, Spiegelhalter D, Thomas A, et al. The BUGS project: Evolution, critique and future directions [J]. Statist. Med. , 2009, 28(25): 3049-3067

[3] Jampani R, Xu Fei, Wu Ming-xi, et al. The Monte Carlo Database System: Stochastic analysis close to the data [J]. ACM Trans. Database Syst. , 2011, 36(3): 18

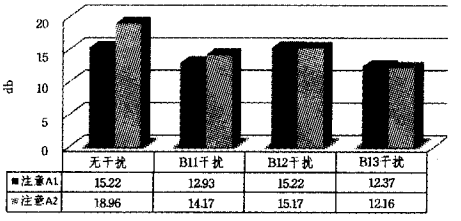
[4] Singh S, Subramanya A, Pereira F, et al. Distributed MAP inference for undirected graphical models [C]// Neural Information Processing Systems (NIPS), Workshop on Learning on Cores, Clusters and Clouds. 2010

(下转第 287 页)

图与注意对话显著图的输出信噪比 SNR 进行定量评价：

$$SNR(f,t)=10\log\frac{\sum_{t=1}^{T}\sum_{f=1}^{F}tf_0^2(f,t)}{\sum_{t=1}^{T}\sum_{f=1}^{F}[tf_0(f,t)-tf_i(f,t)]^2} \quad (12)$$

式中，$tf_0(f,t)$为理想的对象 i 的显著图，$tf_i(f,t)$为注意第 i 个对象的显著图。具体实验结果如图 11 所示。



	无干扰	B11干扰	B12干扰	B13干扰
■注意A1	15.22	12.93	15.22	12.37
※注意A2	18.96	14.17	15.17	12.16

图 11　音频选择性注意听觉显著图输出信噪比

从图 11 中可见，背景女声干扰 B11 对被注意男声 S2 显著性影响较小，而对被注意女声 S1 的显著性影响较大，分别降低约 4.8dB 和 2.3dB；背景纯音乐干扰 B12 对被注意男声 S2 和女声 S1 的显著性影响不大；背景男女声合唱音乐干扰 B13 对注意男声 S2 和女声 S1 的显著性影响较大，分别降低约 36%和 19%。整体背景干扰对显著性平均降低约 3.4dB，频谱相近的、音强较大的影响较大。实验结果与听觉的掩蔽特性非常近似。

结束语　本文提出的基于选择性注意的认知神经机制的听觉显著性计算模型，在结构和功能上模拟了听觉认知神经信息处理机制。该模型兼容了 BU 的数据驱动和 TD 的概念反馈两种听觉注意机制，很好地模拟了人类的听觉注意过程。在自然环境下，本模型能有效增强被注意音频的显著性，降低和抑制非注意背景混叠音的干扰。

尽管本文提出的听觉认知框架考虑了双耳定位因素，但考虑到篇幅的限制，本文所给的显著性模型的计算过程仅考虑了强度和频率在选择性注意中的贡献。基于选择性注意的认知神经机制的听觉显著性计算仍有大量尚未解决的问题，有很多的研究工作要做。在今后的研究中，将进一步考虑双耳时间差、强度差和耳廓等方位因素在听觉选择性注意的影响。

参 考 文 献

[1] Kalinli O. Tone and pitch accent classification using auditory attention cues[C]// Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on. Prague Congress Centre Prague, Czech Republic, 2011: 5208-5211

[2] Wang Lei, Pen Yuan, et al. The application of computational auditory peripheral model in underwater target classification[J]. Chinese journal of electronics, 2012(01): 199-203

[3] Yin Hui, Xie Xiang, Kuang Jing-ming. Acoustic features based on auditory model and adaptive fractional Fourier transform for speech recognition[J]. ACTA ACUSTICA (Chinese version), 2012, 1: 97-103

[4] Vaclav B, Rainer M, et al. A model-based auditory scene analysis approach and its application to speech source localization[C]// Acoustics, Speech and Signal Processing (ICASSP). Prague Congress Centre Prague, Czech Republic, 2011: 2624-2627

[5] De C B, Botteldooren D. A model of saliency-based auditory attention to environmental sound[C]// Proc. ICA. Sydney, Australia, 2010: 1-8

[6] Snyder J S, Pasinski A C, Devin M J. Listening strategy for auditory rhythms modulates neural correlates of expectancy and cognitive processing[J]. Psychophysiology, 2010, 48(2): 198-207

[7] Tabor K M, Coleman W L, et al. Tonotopic organization of the superior olivary nucleus in the chicken auditory brainstem[J]. Journal of comparative neurology, 2012, 520(7): 1493-1508

[8] Mutoh Y, Kashimori Y. Neural model of auditory cortex for binding sound intensity and frequency information in batys echolocation[C]// ICONIP'11 Proceedings of the 18th International Conference on Neural Information Processing-Volume Part I. Hangzhou, China, 2012: 62-69

[9] Rauschecker J P. An expanded role for the dorsal auditory pathway in sensorimotor control and integration[J]. Hearing research, 2011, 271(1/2): 16-25

[10] Chatterjee S, Kleijn W B. Auditory Model-Based Design and Optimization of Feature Vectors for Automatic Speech Recognition [J]. Audio, Speech, and Language Processing, IEEE Transactions on, 2011, 19(6): 1813-1825

[11] Zhao Ya-hui, Wang Hong-li, Cui Rong-yi. An Approach to Sound Feature Extraction Method Based on Gammatone Filter [J]. Advances in Multimedia, Software Engineering and Computing, 2012, 2: 371-376

[12] Zhu Jun-mei. A Multifactor Winner-Take-All Dynamics [J]. Neural computation, 2011, 23(7): 1835-1861

(上接第 259 页)

[5] Cai Z, Vagena Z, Jermaine C, et al. Very Large Scale Bayesian Inference Using MCDB [C]// Big Learn Workshop, Neural Information Processing Systems. 2011

[6] Blei D M, Ng A Y, Jordan M I. Latent Dirichlet Allocation [J]. Journal of Machine Learning Research, 2003, 3: 993-1022

[7] Porteous I, Newman D, Ihler A T, et al. Fast collapsed Gibbs sampling for Latent Dirichlet Allocation [C]// ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2008: 569-577

[8] Liu Zhi-yuan, Zhang Yu-zhou, Chang E Y, et al. Parallel Latent Dirichlet Allocation with Data Placement and Pipeline Processing [J]. ACM Transactions on Intelligent Systems and Technology, special issue on Large Scale Machine Learning, 2011, 2(3): 26

[9] Smola A J, Narayanamurthy S. An Architecture for Parallel Topic models [J]. The Proceedings of the VLDB Endowment, 2010, 3(1): 703-710

[10] Newman D, Asuncion A, Smyth P. et al, Distributed Inference for Latent Dirichlet Allocation [C]// Neural Information Processing Systems. 2007

[11] 张步良. 基于分类概率加权的朴素贝叶斯分类方法[J]. 重庆理工大学学报：自然科学版, 2012, 26(7): 81-83