目 录

[**一、EEG信号预处理程序说明** 2](#_Toc2782932)

[1、目的 2](#_Toc2782933)

[2、工程包含文件 2](#_Toc2782934)

[3、运行环境和所需函数库 2](#_Toc2782935)

[4、方法和算法 2](#_Toc2782936)

[5、算法流程 6](#_Toc2782937)

[6、主要函数详解 6](#_Toc2782938)

[**二、EEG分类模型程序说明** 8](#_Toc2782939)

[1、目的 8](#_Toc2782940)

[2、工程包含文件 8](#_Toc2782941)

[3、运行环境和所需函数库 8](#_Toc2782942)

[4、方法和算法 8](#_Toc2782943)

[5、算法流程 10](#_Toc2782944)

[6、主要函数详解 10](#_Toc2782945)

## 一、EEG信号预处理程序说明

### 1、目的

预处理算法的名称为：正则化共空间模式的短时傅里叶变换方法（RCSP-STFT）。正则化共空间模式（RCSP）克服了传统共空间模式（CSP）在样本数量较少时容易出现过拟合和对噪声太敏感的缺点，**其目的在于最大化不同类别信号的方差**，便于后续步骤的分类；短时傅里叶变换（STFT）算法是时频分析方法的一种，**其目的有两点，一是同时保留信号的时域和频域信息，将原始的一维时域信号转换为二维的时间-频域二维图像信号**，便于使用深度学习算法进行分类；**二是将信号从时域转换到频域后，可以对感兴趣的频域范围信号进行频带提取。**

### 2、工程包含文件

1. RCSP\_STFT.m：主程序
2. RCSP.m：RCSP算法的子函数程序
3. STFT.m：STFT算法的子函数程序
4. RegCsp.m：对应于RCSP算法的具体公式计算过程

### 3、运行环境和所需函数库

操作系统：Windows系统，MATLAB软件

运行语言：MATLAB

### 4、方法和算法

**4.1 CSP算法的基本原理**

CSP是一种在多通道信号中常用的处理方法，H. Ramoser将CSP方法首次用于手部运动想象的EEG信号分类问题中。由于基于国际10-20系统采集的EEG信号也是一种多通道信号，所以CSP方法在EEG信号处理中有着广泛的应用。CSP空间滤波通过信源分离的方法，基于对原始EEG信号协方差矩阵的同时对角化，求得最佳的空间投影矩阵，并利用该空间投影矩阵进行卷积滤波，使得处理之后的信号矩阵对于某一个类别方差最大，而对另一个类别方差最小，使得不同类别之间的类别方差更加明显。

本文基于左、右手运动想象EEG信号进行研究，原始的EEG信号可以表示为大小的矩阵，其中是EEG信号的通道数量，和采集实验时使用的设备以及选择的导联数目有关，*T*是每一个通道采集到的样本点数，和实验设定的采样频率和采样时长有关。左、右手原始EEG信号的标准空间协方差矩阵可以表示为：

(1)

其中表示矩阵的迹，表示矩阵的转置，和分别表示左手和右手的运动想象类别。

则左手和右手两类样本信号的平均空间协方差矩阵可以表示为：

(2)

其中和分别表示左手和右手运动想象的实验次数。

合成的协方差矩阵为：

(3)

接着对合成的协方差矩阵进行特征值分解：

(4)

其中是矩阵的特征向量，阶次为，是由非零特征值组成的对角阵，且特征值按降序排列。

定义白化矩阵为：

**P** (5)

利用白化矩阵，对左手和右手两类样本的平均空间协方差矩阵和进行变换：

(6)

和分别对应和变换后的矩阵，再将和进行特征值分解：

(7)

其中和两个特征向量矩阵满足条件，是单位矩阵。则对应特征值和满足关系：，所以当特征向量矩阵中的某一个特征向量对于有最大的特征值时，对于就有最小的特征值**。**

最终的空间投影矩阵可以用特征值矩阵和白化矩阵表示为：

(8)

利用空间投影矩阵，我们可以对原始的EEG信号矩阵进行空间投影，得到CSP方法处理的最终结果：

(9)

一般情况下，我们会将矩阵的前后各行作为原始EEG信号的特征，可以直接用于分类器的分类，或者基于此开展进一步的分析处理。

**4.2 RCSP算法的基本原理**

基于正则化准则和传统CSP算法，RCSP-STFT算法计算不同类别的正则化平均空间协方差矩阵过程如下：

(10)

其中，表示类别标记，和分别表示左手和右手的运动想象类别；和表示两个正则化的参数；表示矩阵的迹；在此处是一个大小为的单位阵，表示EEG信号的个通道。是一个包含了目标被试的实验数据和其他被试的通用实验数据在内的协方差矩阵，其定义如下：

(11)

其中，表示目标被试的类别所有次实验数据的协方差矩阵之和：

(12)

表示其他被试的类别所有次通用实验数据的协方差矩阵之和：

(13)

上述定义中所用到的符号含义如下：表示目标被试的每个类别选择的参与训练的实验次数，；因此对于目标被试的每个不同类别我们都可以计算协方差矩阵，表示类别的从到次训练实验次数的协方差矩阵之和；对于其他被试的每个不同类别我们可以计算协方差矩阵，表示类别的从到次训练实验次数的协方差矩阵之和；和是如公式(3-1)所计算的协方差矩阵。假设被试数量用表示，则可由计算得来：，则。

基于上述定义和本文的实验数据，得到RCSP算法对于左手和右手类别的正则化平均空间协方差矩阵：和，并可以结合成最终的RCSP算法的合成协方差矩阵：

(14)

至此，我们得到了RCSP算法的合成协方差矩阵，对应于传统CSP算法中公式(3)中的，之后的计算步骤与公式(4)到(9)相同，不再赘述。最终我们得到基于RCSP算法的空间投影矩阵和对原始EEG信号进行空间投影后的信号：

(15)

**4.3 STFT算法的基本原理**

我们将从矩阵筛选出的行向量记为，且，则非平稳信号的短时傅里叶变换定义如下：

(16)

其中，是时间的函数，表示窗函数，表示原始EEG信号的采样频率。

### 5、算法流程

|  |  |
| --- | --- |
| **Input：** | 原始EEG信号矩阵，EEG信号的通道数量，训练参数，被试数量，正则化参数和 |
| **Output：** | 表征图（The representation images） |
| 1 |  |
| 2 | *images* = RCSP-STFT（，，，，，） |
| 3 | **for** **do**： |
| 4 | **for** **do**： |
| 5 | 按照公式(10)~公式(14)计算； |
| 6 | 计算公式(4)~公式(8)计算投影矩阵； |
| 7 | 按照公式(15)对原始EEG信号进行投影； |
| 8 | 选择矩阵的第一行，并按照公式(16)进行STFT计算； |
| 9 | 筛选出mu和beta频带，并组合成表征图； |
| 10 | **return** *images* |

### 6、主要函数详解

1. [s, h] = sload(name, 0, 'OVERFLOWDETECTION:OFF');

功能：.gdf数据格式载入函数

作者：数据库提供

改写：无

输入：数据库全称name，后两项为默认值

返回：实验数据的数据矩阵s，记录实验的具体实验数值；实验数据的信息矩阵h，记录实验的通道、标签、时间标记等信息。

1. signalRCSP = RCSP(selCh, numCls, raw\_EEG, gnd, numTrn, gamma);

功能：RCSP数据处理函数

作者：未知

改写：罗杰

输入：selCh:RCSP后选择的通道数量

numCls:数据的类别数目，本文为2

raw\_EEG:原始EEG信号矩阵

gnd:数据对应的标签

numTrn:选择的训练实验数目，公式（11）的M值

返回：RCSP后的实验数据矩阵

1. Train = STFT(forSTFT);

功能：STFT数据处理函数

作者：未知

改写：罗杰

输入：选择的用于STFT的信号

返回：提取好mu频带与beta频带的二维输出图像。

## 二、EEG分类模型程序说明

### 1、目的

针对RCSP-STFT生成的数据进行模型的训练和识别，采用改进的CNN模型，称为并联卷积神经网络（PCNN）。

### 2、工程包含文件

主程序：pcnn.py

### 3、运行环境和所需函数库

操作系统：Ubuntu系统

运行语言：python

所需函数库：numpy，keras，time，sklearn，random

### 4、方法和算法

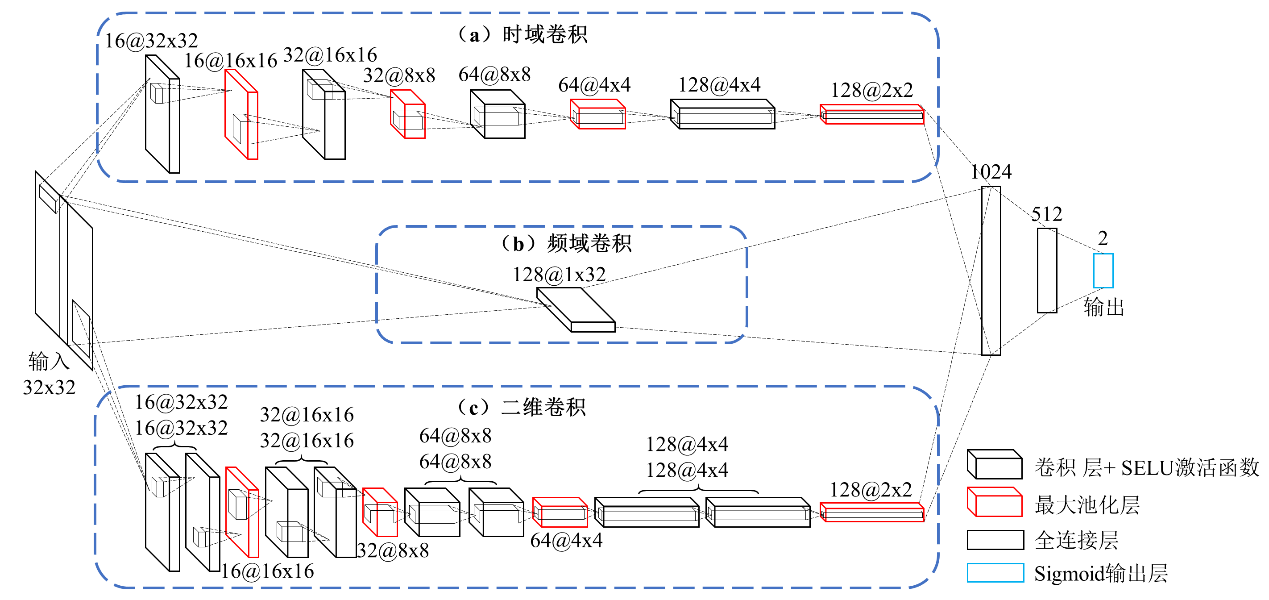


图1 PCNN的模型结构

PCNN的模型结构如图1所示。本文将每一个子模型中重复出现的单元称为一个“功能块”，每一个功能块中包含卷积和池化两种运算方式，但对卷积和池化运算的出现次数不做限制。图中黑色的立方体代表卷积和激活函数的运算，红色的立方体代表池化运算。经过RCSP-STFT的特征提取之后，得到的EEG信号表征图的大小为32×32，这也是PCNN模型输入的数据大小。

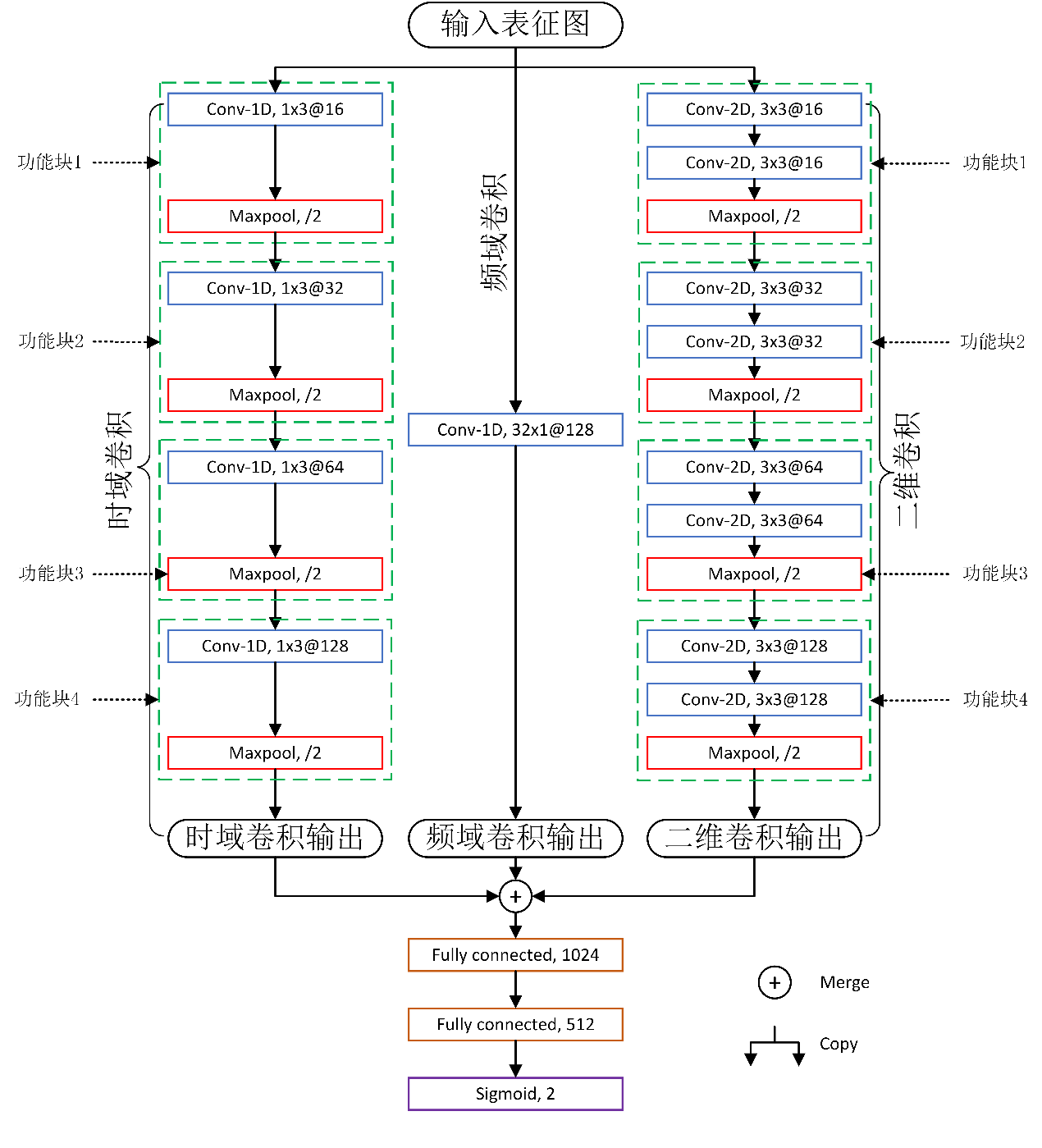


图2 PCNN的模型参数

对于时域一维卷积核，在进行全连接运算之前，我们在时域卷积运算的子模型中共设置了四个功能块，每一个功能块包含了一个卷积层和一个池化层，如图1(a)和图2所示。图2中更清楚的显示了PCNN的模型结构与参数。我们将时域卷积运算的卷积核大小设置为1×3。功能块1中的卷积层的卷积核数量为16个，经过卷积和激活函数的运算之后，使用了步长为2，大小为2×2的池化核进行池化运算；功能块2中卷积层的卷积核数量为32个，池化操作的参数与上一个功能块中的相同；功能块3和功能块4中卷积层的卷积核数量分别为64个和128个，且池化操作参数均相同。

对于频域一维卷积核，由于将该卷积核大小设置为小于输入表征图在频域通道上数目的做法是不被推荐的，因此将频域一维卷积核的大小设置为32×1，个数设置为128个。如图1(b)和图2所示，经过一次卷积运算之后，输入表征图上的所有频域通道都被混合为一个全新的特征图。

对于二维卷积核，在进行全连接运算之前，我们在二维卷积运算的子模型中共同样设置了四个功能块，不同的是每一个功能块中包含了两个卷积层和一个池化层，如图1(c)和图2所示。我们将二维卷积运算的卷积核大小设置为3×3。功能块1中的两个卷积层的卷积核数量都为16个；功能块2中两个卷积层的卷积核数量都为32个；功能块3和功能块4中的两个卷积层的卷积核数量分别为64个和128个。所有功能块中池化操作均与时域卷积中的参数设置相同，步长都为2，池化核的大小都为2×2。

经过时域卷积的运算之后，我们得到该子模型的输出矩阵维度是128×2×2；频域卷积运算和二维卷积运算子模型对应的输出矩阵维度分别是128×1×32和128×2×2。接着我们将所有的三维输出矩阵重新排列为一个向量，并合并三个子模型对应的向量，组成一个更大的向量之后，进行全连接运算。如图4.5和图4.6所示，在PCNN中，我们设置了两个全连接层，神经元的数量分别是1024个和512个，最后使用Sigmoid分类器进行最终的分类计算。

### 5、算法流程

如图2所示。

### 6、主要函数详解

1. Conv2D(32,(1,3), strides=2, padding='same', activation='selu')

功能：二维卷积函数

输入：filters：卷积核的数目（即输出的维度）

kernel\_size：单个整数或由两个整数构成的list/tuple，卷积核的宽度和长度。如为单个整数，则表示在各个空间维度的相同长度。

strides：单个整数或由两个整数构成的list/tuple，为卷积的步长。如为单个整数，则表示在各个空间维度的相同步长。任何不为1的strides均与任何不为1的dilation\_rate均不兼容

padding：补0策略，为“valid”, “same” 。“valid”代表只进行有效的卷积，即对边界数据不处理。“same”代表保留边界处的卷积结果，通常会导致输出shape与输入shape相同。

activation：激活函数，为预定义的激活函数名（参考激活函数），或逐元素（element-wise）的Theano函数。如果不指定该参数，将不会使用任何激活函数（即使用线性激活函数：a(x)=x）

返回：卷积后的数组。

1. BatchNormalization()

功能：批规范化，将前一层的激活值重新规范化，即使得其输出数据的均值接近0，其标准差接近1

输入：上一个卷积层的输出

返回：规范化后的输出。

1. Flatten()

功能：来将输入“压平”，即把多维的输入一维化

输入：上一个层的输出

返回：抹平后的一维向量。

1. MaxPooling2D(pool\_size=(2,2), strides=2, padding='same')

功能：二维池化函数

输入：

pool\_size：整数或长为2的整数tuple

strides：整数或长为2的整数tuple，或者None，步长值。

padding：‘valid’或者‘same’

返回：池化后的数组。

1. Dense()

功能：全连接层

输入：上一个层的输出

返回：全连接后的输出。