### 本周完成的内容：

1. 基于脑电信号的疲劳检测调研

基于EEG的疲劳检测，调研中发现在2019年中发表在IEEE的基于时空卷积神经网络的驾驶员疲劳评估作何提出了ESTCNN网络，其分类精确率可以达到97.37%，**该方法的优点是考虑到脑电信号的时间和空间性**。而在其他方法中，也有比其准确率更高的传统算法，如基于SVM的疲劳脑电信号分类，作者在其实验数据中得到的分类结果准确率达到了99.3%，但是不考虑脑电信号的其他相关性质，并且数据是进行预处理过的。

2019年在IEEE上，EEG-Based Spatio–Temporal Convolutional Neural Network for Driver Fatigue Evaluation （基于时空卷积神经网络的驾驶员疲劳评估），作者基于多通道的脑电图信号的时空结构，开发了一种ESTCNN，用于检测驾驶员疲劳。分类精度达到97.37%，且计算效率也具有优势。

2008年发表在sciencedirect上的Can SVM be used for automatic EEG detection of drowsiness during car driving?

使用脑电信号的4个主要频段上的4个频率特征作为判据，将两种状态（机敏和疲劳）EEG数据样本用于训练SVM分类器，经过培训的SVM在未分类的EEG数据上进行了测试，分类准确率达到99.3%。

2017年发表在plos one上的Driver fatigue detection through multiple entropy fusion analysis in an EEG-based system（通过基于EEG的系统中的多个熵融合分析来检测驾驶员疲劳），作者将多个熵（谱熵、近似熵、样本熵和模糊熵）融合为特征，并与四个自回归模型进行了比较。通过来自四通道的EEG数据，使用四个分类器建立了疲劳检测的模型，经验证，准确率98.3%。

在2017年发表在Neural上A multimodal approach to estimating vigilance using EEG and forehead EOG，**在特征提取过程中不能忽略EOG和其他生理信号噪声的干扰**，提出了一种多模式方法，该方法通过将EEG和前额EOG的采集电极放置相结合来估计驾驶员疲劳，其目的是整合两者的时间依赖并获得改进的性能。

2017年发表在IEEE上的Driver fatigue classification with independent component by entropy rate bound minimization analysis in an EEG-based system，提出了一种基于两级脑电图的分类法对疲劳驾驶进行分类，该系统使用独立的熵率边界最小化分析进行源分离，使用自回归建模进行特征提取，并使用贝叶斯神经网络进行分类。分类结果表明，准确率为88.2%。

2019年发表在IEEE上的A Semi-Supervised Wasserstein Generative Adversarial Network for Classifying Driving Fatigue from EEG signals，作者为了解决有限的标记训练样本问题，提出了一种具有去梯度惩罚的半监督wasserstein生成对抗网络（sWGAN-GP）。所提出的是WGAN-GP包含一个分类器，该分类器具有GAN中判别器的体系结构，它的损失函数使训练过程中生成的EEG样本能够扩展有限的训练样本，从而提高分类性能

2017年发表在frontiers上的Improving EEG-Based Driver Fatigue Classification Using Sparse-Deep Belief Networks，作者采用自回归建模作为特征提取算法，并采用稀疏-深度置信网络（sparse-DBN）作为分类算法。与其他分类器相比，稀疏DBN是一种半监督学习方法，它将预训练层中用于建模特征的无监督学习与下一层中用于分类的监督学习相结合。作者并用其他网络进行了实验对比，人工神经网络（ANN）、贝叶斯神经网络（BNN）、和原始深度信念网络（DBN）分类器。分类结果中使用AR特征提取器和DBN分类器，分类性能得到提高，分类准确率为90.6%，而与ANN(准确性为79.3%，roc为0.83)和BNN分类器（准确性为83.6%，AUROC为0.87)相比，接受者工作曲线下面积(AUROC)为0.94。

1. EEGNet网络的实现，并进行测试，使用随机生成100个0-1之间数据样本进行分类训练，并进行测试。

Epoch 0  
[‘acc’, ‘auc’, ‘fmeasure’]

Training Loss 1.6107637286186218  
Train - [0.52, 0.5280448717948718, 0.6470588235294118]  
Validation - [0.55, 0.450328407224959, 0.693877551020408]  
Test - [0.54, 0.578926282051282, 0.6617647058823529]

Epoch 1  
[‘acc’, ‘auc’, ‘fmeasure’]  
Training Loss 1.5536684393882751  
Train - [0.45, 0.41145833333333337, 0.5454545454545454]  
Validation - [0.55, 0.4823481116584565, 0.6564885496183207]  
Test - [0.65, 0.6530448717948717, 0.7107438016528926]

Epoch 2  
[‘acc’, ‘auc’, ‘fmeasure’]  
Training Loss 1.5197088718414307  
Train - [0.49, 0.5524839743589743, 0.5565217391304348]  
Validation - [0.53, 0.5870279146141215, 0.5436893203883495]  
Test - [0.57, 0.5428685897435898, 0.5567010309278351]

Epoch 3  
[‘acc’, ‘auc’, ‘fmeasure’]  
Training Loss 1.4534167051315308  
Train - [0.53, 0.5228365384615385, 0.4597701149425287]  
Validation - [0.5, 0.48152709359605916, 0.46808510638297873]  
Test - [0.61, 0.6502403846153847, 0.5517241379310345]

Epoch 4  
[‘acc’, ‘auc’, ‘fmeasure’]  
Training Loss 1.3821702003479004  
Train - [0.46, 0.4651442307692308, 0.3076923076923077]  
Validation - [0.47, 0.5977011494252874, 0.29333333333333333]  
Test - [0.52, 0.5268429487179488, 0.35135135135135137]

Epoch 5  
[‘acc’, ‘auc’, ‘fmeasure’]  
Training Loss 1.440490186214447  
Train - [0.56, 0.516025641025641, 0.35294117647058826]  
Validation - [0.36, 0.3801313628899836, 0.2]  
Test - [0.53, 0.6113782051282052, 0.27692307692307694]

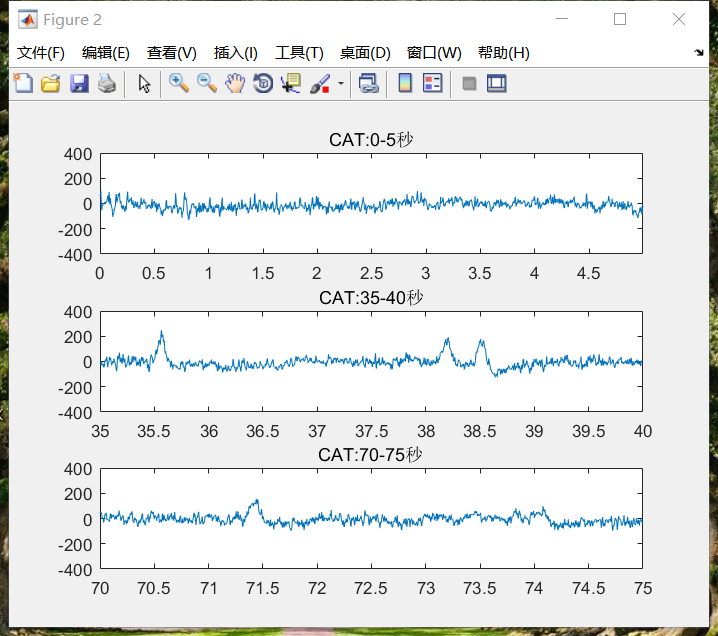
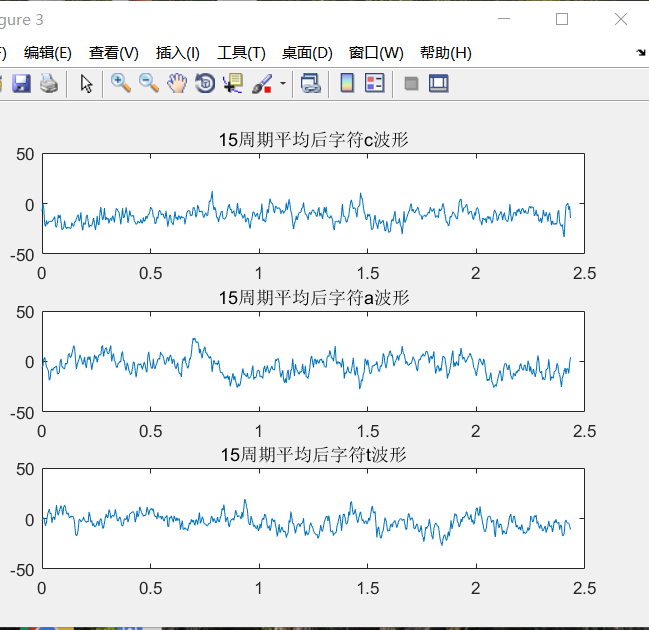
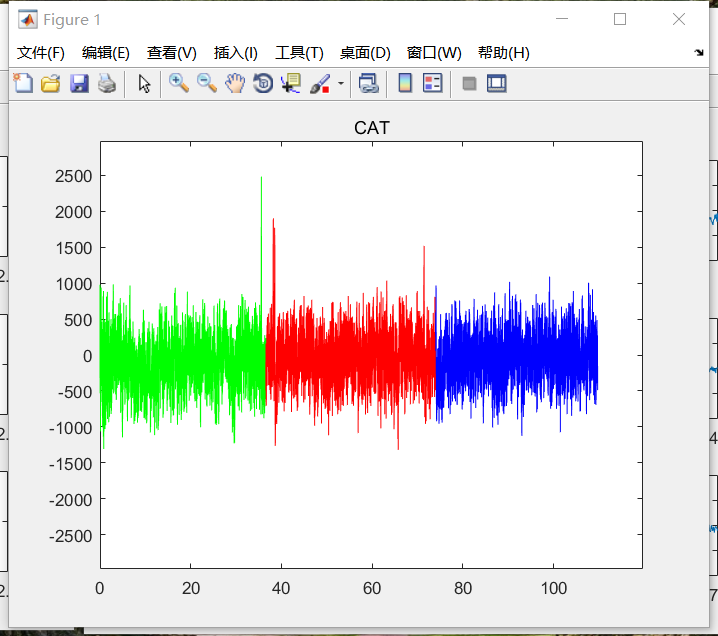
Epoch 6  
[‘acc’, ‘auc’, ‘fmeasure’]  
Training Loss 1.4722238183021545  
Train - [0.47, 0.4194711538461539, 0.13114754098360656]  
Validation - [0.46, 0.5648604269293925, 0.2285714285714286]  
Test - [0.5, 0.5348557692307693, 0.10714285714285714]

Epoch 7  
[‘acc’, ‘auc’, ‘fmeasure’]  
Training Loss 1.3460421562194824  
Train - [0.51, 0.44871794871794873, 0.1694915254237288]  
Validation - [0.44, 0.4490968801313629, 0.2]  
Test - [0.53, 0.4803685897435898, 0.14545454545454545]

Epoch 8  
[‘acc’, ‘auc’, ‘fmeasure’]  
Training Loss 1.3336675763130188  
Train - [0.54, 0.4130608974358974, 0.20689655172413793]  
Validation - [0.39, 0.40394088669950734, 0.14084507042253522]  
Test - [0.51, 0.5400641025641025, 0.19672131147540983]

Epoch 9  
[‘acc’, ‘auc’, ‘fmeasure’]  
Training Loss 1.438510239124298  
Train - [0.53, 0.5392628205128205, 0.22950819672131148]  
Validation - [0.42, 0.4848111658456486, 0.09375]  
Test - [0.56, 0.5420673076923076, 0.2413793103448276]

1. 对下载的p300数据用MATLAB进行预处理



### 下周目标：

实现对P300数据使用EEGNet进行二分类。