پروژه درس سیگنال ها و سیستم ها – فاز دوم

امیرعلی رجایی

بهار 1403
دانشگاه صنعتی شریف
دانشکده مهندسی برق

به نام خدا

پروژه درس سیگنال ها و سیستم ها : فاز 2 — امیرعلی رجایی - 401101716

Epileptic Seizure Prediction Using Spectral Entropy-Based Features of EEG

1. بارگذاری دیتاست در متلب

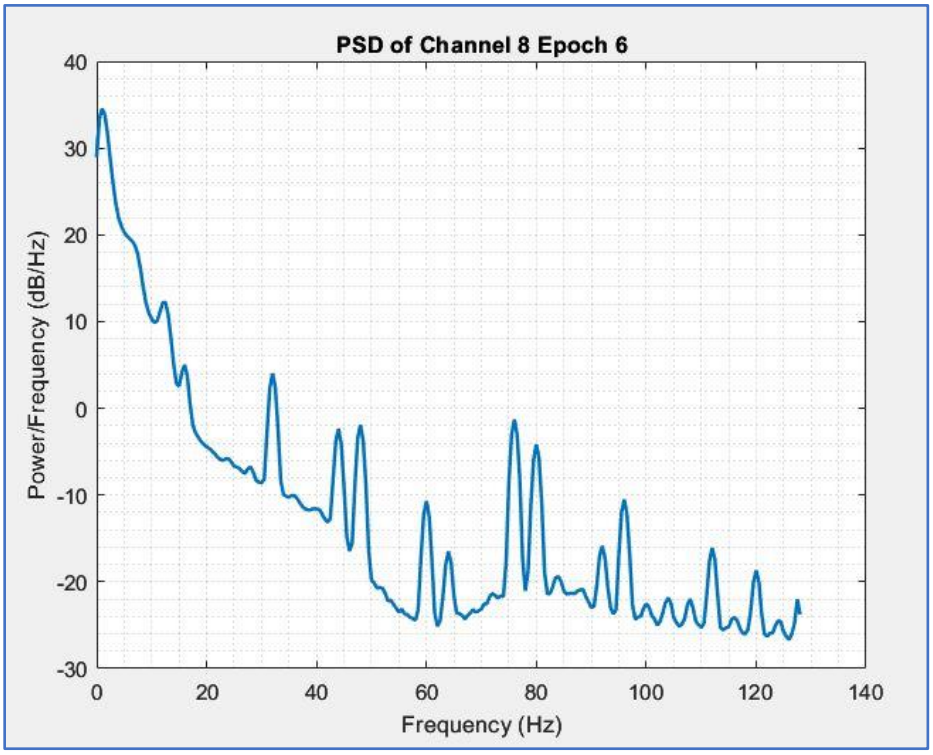
در این بخش باید دیتاست های موجود را درون متلب بارگذاری کنیم؛ برای این کار از توابع موجود در کتابخانه EEGLAB استفاده می کنیم. ابتدا با استفاده از تابع pop_biosig دیتاستی که در اختیار داریم را لود کرده و سپس با تابع eeg_checkset از صحت اطلاعات موجود در دیتاست اطمینان می یابیم.

2. محاسبه چگالی طیفی توان (PSD)

در این بخش با استفاده از دیتاست هایی که در بخش قبل در متلب لود کردیم، PSD آن ها را حساب می کنیم.

نحوه کار به این صورت است که هر سیگنال مغزی دریافت شده که مدت زمان 60 دقیقه دارد را به 6 بخش 10 دقیقه ای تقسیم کرده ایم. اصطلاحا به این بخش های 10 دقیقه ای epoch می گویند. سپس PSD را برای هرکدام از این epoch ها حساب کرده ایم. برای محاسبه PSD می توان از دو تابع pwelch و periodogram استفاده کرد که ما از pwelch استفاده کرده ایم. تابع pwelch مقادیر نرخ نمونه برداری، تعداد نمونه های دارای همپوشانی، تعداد نقاط برای محاسبه FFT و طوال پنجره تحت محاسبه را همراه با دیتاست اصلی دریافت کرده و PSD را محاسبه می کند.

در شکل زیر به طور دلخواه PSD دیتاست chb01_16 را برای کانال هشتم و epoch ششم آن محاسبه کرده و نمایش داده ایم:



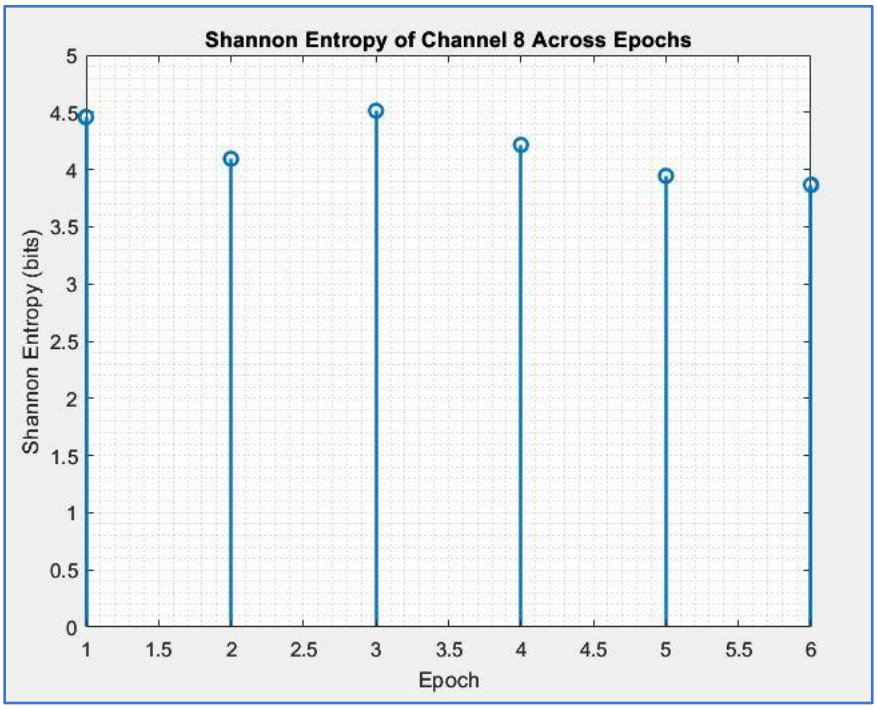
شکل 1 – چگالی طیفی توان برای کانال هشتم و epoch ششم از دیتاست chb01_16

کد این بخش به گونه ای زده شده که می توانیم، دیتاست، کانال و epoch را به صورت دلخواه انتخاب کنیم.

3. محاسبه Shannon Entropy

در این بخش با استفاده از فرمولی که در فایل پروژه آورده شده، مقدار Shannon Entropy را محاسبه می کنیم. این مقدار برای هر epoch به صورت یک عدد نمایش داده می شود.

در شکل زیر مقادیر Shannon Entropy برای epoch های 6 گانه ی کانال هشتم از دیتاست chb01_16 نمایش داده شده است:



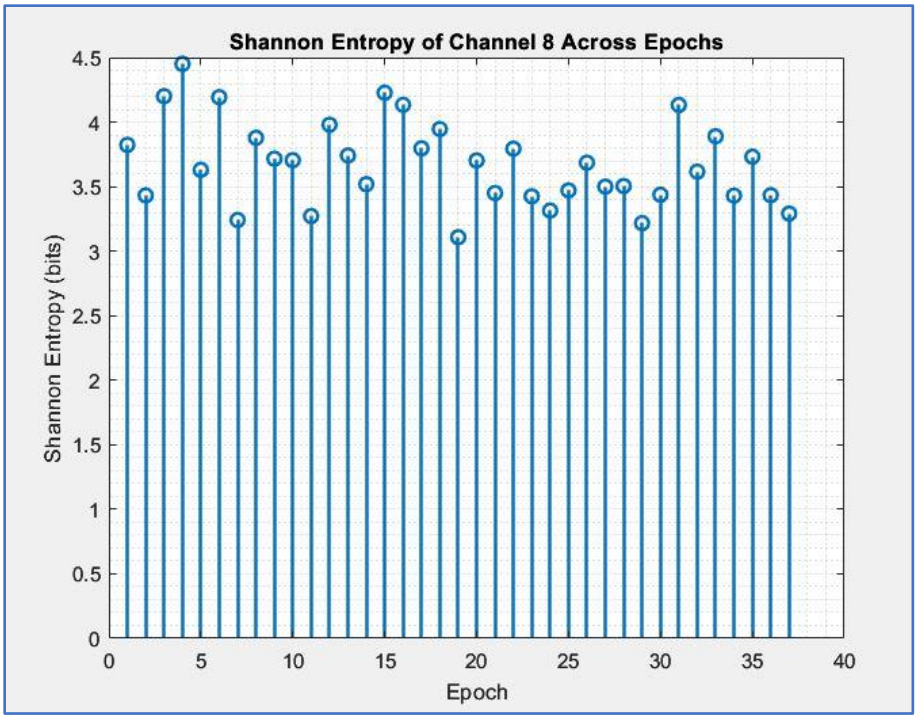
شکل 2 – مقادیر Shannon Entropy برای epoch های 6 گانه ی کانال هشتم از دیتاست chb01_16

4. استخراج ویژگی (Feature Extraction)

در این بخش با دیتا های شامل حمله صرعی سر و کار داریم. شیوه کار به این صورت است که بازۀ ده دقیقه ای قبل از هر حمله صرعی را به epoch های 16 ثانیه تقسیم می کنیم سپس برای هر کدام، متناظر با تعداد کانال های موجود، PSD را حساب کرده و از روی آن، Shannon Entropy را حساب می کنیم. علاوه بر این، مقادیر میانگین، انحراف معیار، مینیم و ماکسیمم را برای همه epoch ها حساب می کنیم. در نتیجه برای هر کدام از ویژگی های ذکر شده، یک ماتریس 37×23 خواهیم داشت. همه این پنج ماتریس در یک struct به نام features ذخیره می کنیم.

برای انجام این کار با استفاده از تابع getDataBeforeTime کل بازه ده دقیقه ای قبل از حمله را به صورت یک سیگنال کامل جدا می کنیم. سپس با استفاده از تابع getMatrix سیگنال جداسازی شده را به epoch های 16 ثانیه ای تقسیم می کنیم؛ در نهایت با استفاده از تابع getFeatures ویژگی های ذکر شده را با پیاده سازی الگوریتم ریاضیاتی آن ها محاسبه می کنیم.

در صفحه بعد تصویری از Shannon Entropy های محاسبه شده برای سی هفت epoch به دست آمده برای کانال هشتم از دیتاست chb01_16 را آورده ایم.



شکل 3 – Shannon Entropy های محاسبه شده برای سی هفت epoch به دست آمده برای کانال هشتم از دیتاست chb01_16

5. انتخاب دیتا های Train و Test (Train and Test Data Selection)

در این بخش با توجه به پیشنهادی که در بخش Additional Instructions در فایل پروژه داده شده، دیتا هایی که می خواهیم الگوریتم های لرنینگ را بر روی آن ها پیاده سازی کنیم، انتخاب می کنیم؛ یعنی:

- دیتای A : شش epoch از دیتای بدون حمله صرعی
- دیتای B : دو epoch بدون حمله عصبی از دیتای شامل حمله صرعی
- دیتای C : شش epoch حاوی حملات صرعی

دقت شود که epoch های در نظر گرفته شده، 10 دقیقه ای هستند.

6. انتخاب ویژگی (Feature Selection)

در این بخش با انجام t-test selection، مقدار p-value را برای ویژگی های پنج گانه ذکر شده در بخش 4، برای همه epoch های انتخاب شده در بخش قبل، بدست می آوریم. نمونه هایی که p-value آن ها کمتر از 0.005 بدست آمده را به عنوان نمونه های significant در نظر می گیریم.

7. جداسازی دیتا های Train از دیتا های Test

در این بخش مجددا با توجه به پیشنهادی که در بخش Additional Instructions داده شده عمل می کنیم؛ یعنی:

- دیتا های Train : چهار epoch از دیتای C، پنج epoch از دیتای A، یک epoch از دیتای B
- دیتا های Test : دو epoch از دیتای C، یک epoch از دیتای A، یک epoch از دیتای B

طبق بخش 6 مقادیر p-value محاسبه شده و نمونه های significant مشخص شده اند.

طبق بخش 7 دیتا های Train از دیتا های Test جداسازی شده اند.

اکنون، می توانیم الگوریتم های لرنینگ را بر روی دیتا های بدست آمده پیاده سازی و اجرا کنیم.

8. پیاده سازی SVM Classifier

در این بخش الگوریتم SVM Classifier را بر روی دیتا های موجود پیاده سازی می کنیم.

با محاسبه True Positive Rate و True Negative Rate مقادیر sensitivity و specificity را بدست می آوریم. در محاسبه این مقادیر، محاسبات تقسیم بر صفر که NaN هستند نیز لحاظ شده و مشکل آن بر طرف شده است.

لازم به ذکر است که الگوریتم SVM Classifier با استفاده از روش K-Fold Cross Validation پیاده سازی شده که در آن، طبق متن فایل پروژه، k را برابر با 5 فرض کرده ایم.

در شکل زیر نتایج Performance Measures را مشاهده می کنیم:

```
Average sensitivity across 5 folds is: 60.00%
Average specificity across 5 folds is: 90.00%
```

شکل4 – نتایج Performance Measures برای الگوریتم SVM Classifier

9. پیاده سازی KNN Classifier

در این بخش الگوریتم KNN Classifier را بر روی دیتا های موجود پیاده سازی می کنیم.

با محاسبه True Positive Rate و True Negative Rate مقادیر sensitivity و specificity را بدست می آوریم. در محاسبه این مقادیر، محاسبات تقسیم بر صفر که NaN هستند نیز لحاظ شده و مشکل آن بر طرف شده است.

لازم به ذکر است که الگوریتم KNN Classifier با استفاده از روش K-Fold Cross Validation پیاده سازی شده که در آن، طبق متن فایل پروژه، k را برابر با 5 فرض کرده ایم.

همچنین، تعداد همسایگی ها برای انجام محاسبات را برابر با 5 گرفته ایم.

در شکل زیر نتایج Performance Measures را مشاهده می کنیم:

```
Average sensitivity across 5 folds is: 90.00%
Average specificity across 5 folds is: 100.00%
```

شکل5 – نتایج Performance Measures برای الگوریتم KNN Classifier