به نام خدا

درس: علوم اعصاب محاسباتی مقدماتی

استاد: دکتر کربلایی

گزارش پروژه شماره ۳

سیّدمحمّدامین منصوری طهرانی

۹۴۱۰۵۱۷۴

علی شیرالی

۹۴۱۰۹۱۶۵

توجه: لطفا تمام s1.mat تا s10.mat را در فولدری با نام Data در کنار فولدر Function و کد اصلی و توابع ضمیمه شده قرار دهید.

قسمت اول: آشنایی با مقاله پژوهش اصلی

۱. هدف به کارگیری یک سیستم BCI بر مبنای EEG است. از نکات ویژه این بررسی این است که دقت این روش به خصوص BCI(P300) برای spelling بسیار زیاد بوده و همچنین این دقت، با تعداد نسبتاً کمی داده training در مقایسه با روشهای دیگر که از داده گیری بسیار طولانی تری بدست می آیند حاصل می شود. از ولی نکات اصلی این تحقیق ثابت نگه داشتن پارامترهایی مانند سایز ماتریس و فاصله بین تحریکها ولی بر روی تعداد سابجکتهای زیاد است.

۲. آزمایش به این صورت است که هر شخص یکی از دو متد SC یا RC را انتخاب میکند. در هر مدل ابتدا به صفحه نگاه میکند و باید داده training بدست آوریم. در متد SC هر خانه برای مدتی روشن شده و در متد RC هر سطر/ستون برای مدتی روشن می شوند. شخص باید به کاراکتر مورد نظر خود نگاه کند و تعداد دفعات روشن شدن آن را بشمارد(برای افزایش تمرکز بر روی آن کاراکتر). هر شخص در ابتدا کلمه WATER را حرف به حرف نگاه میکند و داده های training را بدست می آوریم.(در دیتای در دسترس به نظر نمیرسد این کارها منظم و مطابق پیپر انجام شده باشند و ناهماهنگی هایی بین متن و دیتاست موجود است.) به وسیله این داده ها مدل LDA ساخته میشود. سپس کاربر باید تلاش کند LUCAS را تایپ کند. پس از 15 بار روشن شدن یک خانه یا سطر/ستون در هر روش، بلوک signal processing وارد عمل شده و با استفاده از مدل LDA و گرفتن داده های مشاهده(test) نتیجه classifier را ارائه میدهد.

۳. به جای اینکه 36 کلاس مختلف داشته باشیم و هر مشاهده را در یکی از این کلاس ها قرار دهیم، مشاهده ها را به دو دسته target و nontarget تقسیم بندی میکنیم. اگر خانه روشن شده یا سطر/ستون روشن شده حاوی حرف مورد نظر ما باشد به عنوان target شناخته شده و label یک را میگیرد و در غیر این صورت label آن صفر میشود.(nontarget)

۴. از نتایج مهم این پژوهش تاییدی بر optimal بودن روش P300 برای spelling است. زیرا همانطور که اشاره شد با صرف مدت زمان اندکی قادریم مدل بسیار قوی ای بسازیم و برای درصد زیادی از سابجکتها نتیجه مطلوب بدست آوریم. از مزایای این روش طی این پژوهش عدم وابستگی نتیجه به جنسیت و مصرف سیگار/قهوه و تحصیلات و تجربه کار بود که عمومیت زیادی به روش می بخشد. تنها پارامتر موثر در آن به طور معنا دار میزان خواب بود که کمتر بودن آن اثر مثبتی در نتیجه میگذارد. همچنین در این روش برتری متد RC بر SC نیز مشخص شده است.

قسمت دوم: آشنایی با دیتاست

۱. دیتای هر subject دو ماتریس است (train و test) که بعد دوم آنها بعد زمان است. در سطر اوّل زمان‌های متناظر با هر ستون نوشته شده است. سطر دوم تا نهم سیگنال EEG مربوط به 8 کانال مختلف را در بر دارد. در سطح 10 ام اعداد صفر و ناصفر داریم. اعداد ناصفر همواره به فرم 4 عدد یکسان کنار هم می‌آیند که خانه اوّل متناظر با زمان تحریک است و عدد آن مربوط به سطر، ستون یا خانه فلاش زده شده است (بسته به پروتکل استفاده شده). سطر یازدهم نیز آرایه‌ای از صفر و یک است که یک نشان می‌دهد آن تحریک target بوده است.

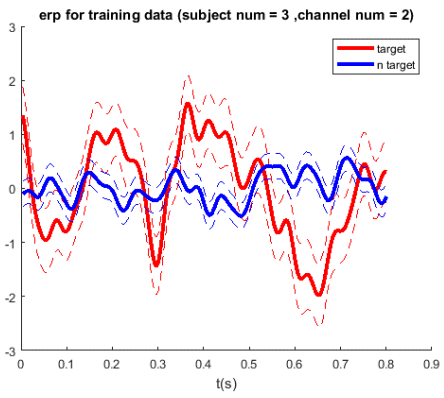
۲. با توجه به اعداد سطر یازدهم، سابجکتهای 1 و 2 از پروتکل SC استفاده کرده‌اند و بقیه از پروتکل RC.

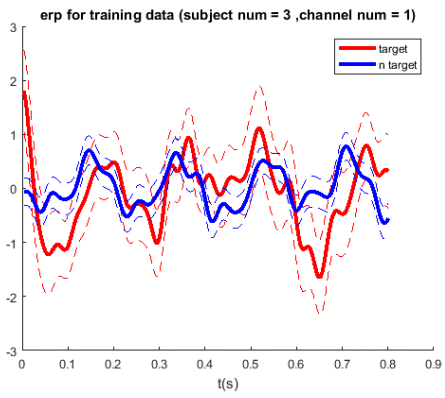
قسمت سوم: بررسی ERP به روش سنتی

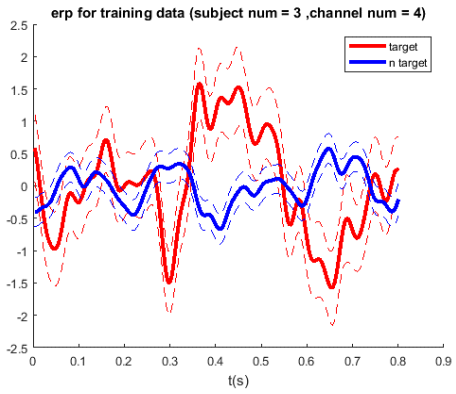
۱. از توابع پیوست (BPF و FilterDFT) برای فیلترکردن سیگنال‌های EEG در بازه 0.5 تا 30 هرتز استفاده می‌کنیم. از خروجی‌های FilterDFT به نظر می‌سد group delay فیلتر جبران شده است پس نیازی به تغییر اندیس شروع تحریک‌ها نداریم. همچنین واضح است ابتدا فیلترینگ انجام می‌گیرد و سپس trialها جدا می‌شوند.

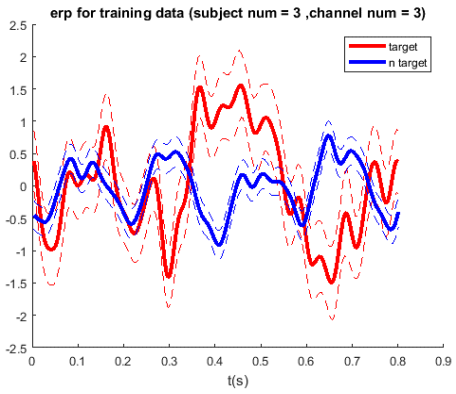
۲. subject 3 را برای این بخش (و قسمت‌های آتی) انتخاب می‌کنیم که از پروتکل RC استفاده کرده است. erp را برای هر کانال با تفکیک target از non-target و به طور جداگانه برای داده‌های train و test محاسبه و رسم می‌کنیم. نتیجه در ادامه ارائه شده است. شایان ذکر است بازه اطمینان $\pm \frac{\sigma}{\sqrt{n-1}}$ حول میانگین (erp) درنظر گرفته شده که در آن n تعداد trialهای مربوطه است.

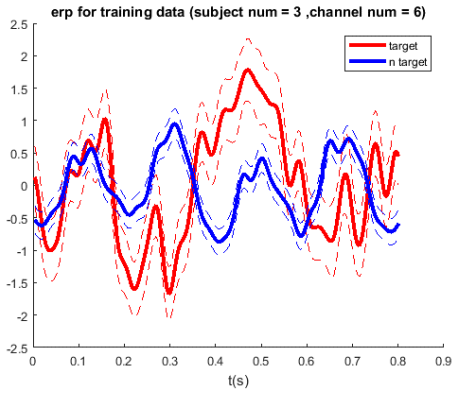
برای داده‌های training:

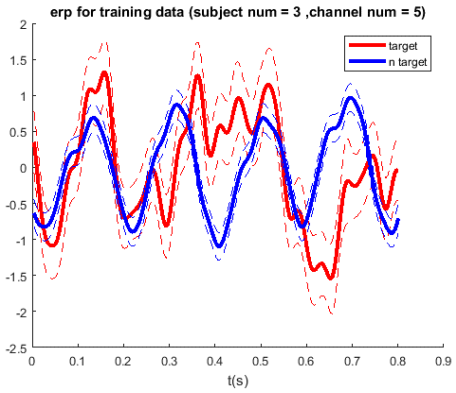


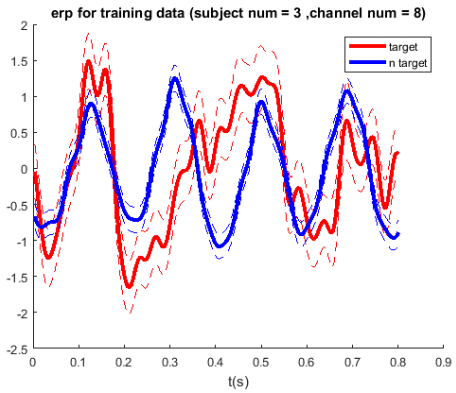


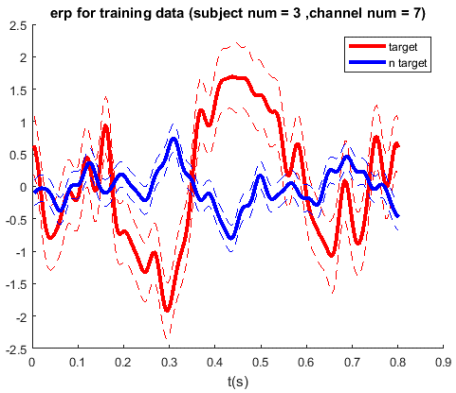




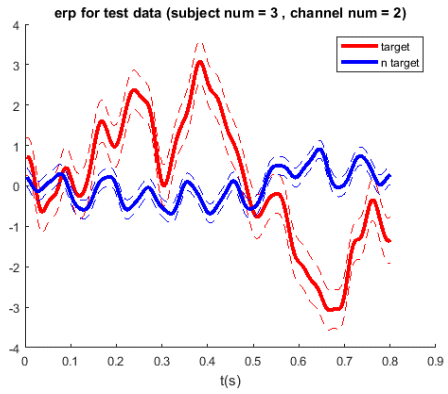


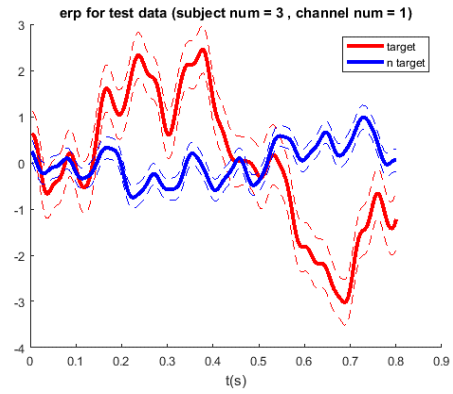


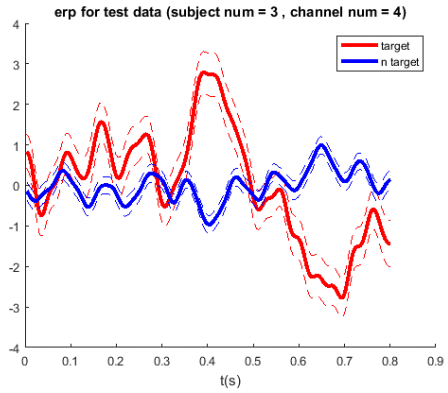


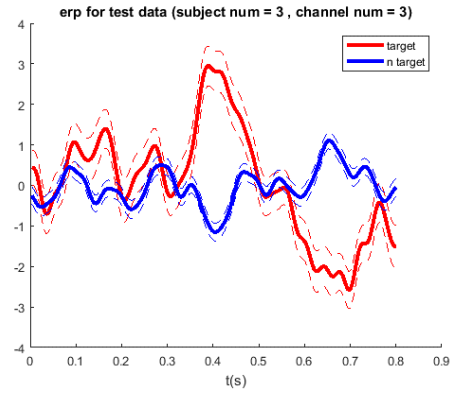


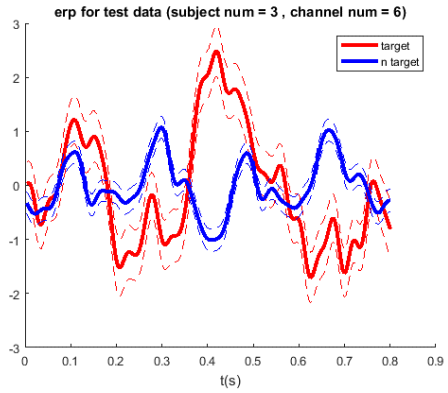
برای داده‌های test:

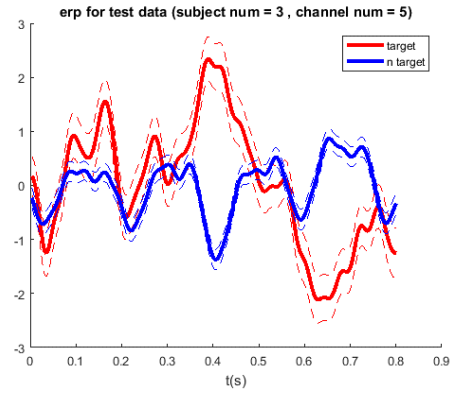

erp for test data (subject num = 3 , channel num = 2)
target
n target
t(s)

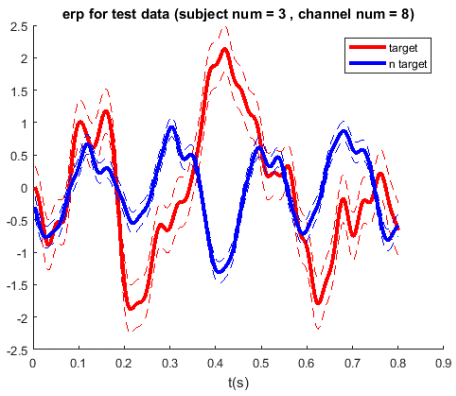

erp for test data (subject num = 3 , channel num = 1)
target
n target
t(s)

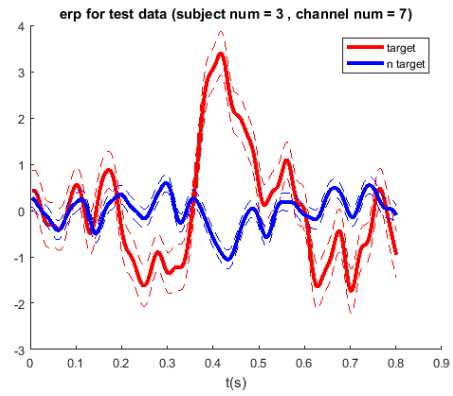

erp for test data (subject num = 3 , channel num = 4)
target
n target
t(s)


erp for test data (subject num = 3 , channel num = 3)
target
n target
t(s)

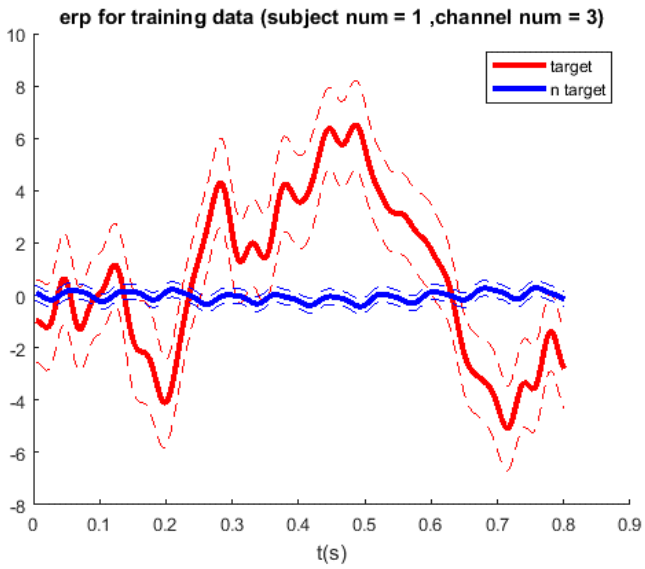

erp for test data (subject num = 3 , channel num = 6)
target
n target
t(s)


erp for test data (subject num = 3 , channel num = 5)
target
n target
t(s)


erp for test data (subject num = 3 , channel num = 8)
target
n target
t(s)


erp for test data (subject num = 3 , channel num = 7)
target
n target
t(s)

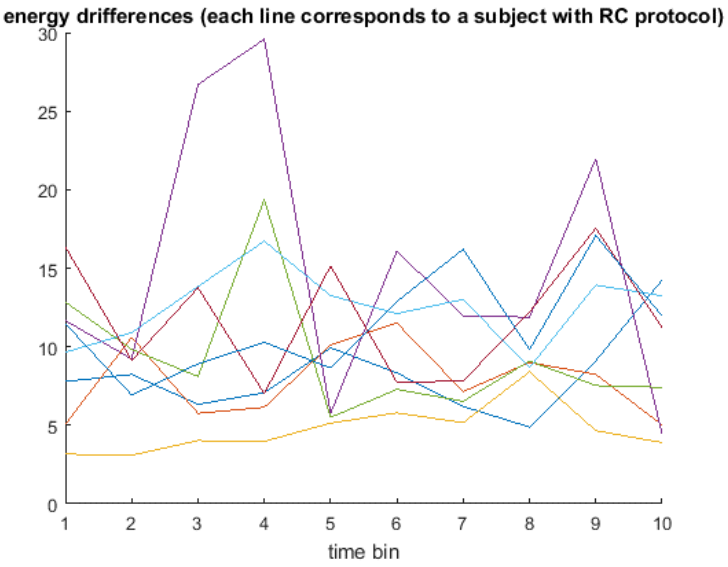
۳. به طور خاص برای پروتکل RC در یک 800 epoch میلی ثانیه‌ای، حدود 5 بار سطر یا ستونی فلش زده می‌شود (معادلا subject تحریک می‌شود) که زمان شروع epoch ها بر شروع اوّلین تحریک منطبق است. بنابراین منطقی است انتظار داشته باشیم که در erp non-target ها که تفاوتی بین تحریک‌های قرارگرفته در یک epoch نیست، خروجی متناوب باشد و حدود 5 دوره تناوب (یا مضاربی از آن) را دربر گرفته باشد. اما در epoch های target، تحریک اوّل target است که انتظار داریم پاسخ متفاوتی را نسبت به تحریک‌های بعدی موجب شود پس منطقی است erp target پترن منظم non-target را نداشته باشد. این مسئله به طور خاص در پروتوکل SC که targetها کم احتمال‌ترند و پاسخ‌های بزرگتری ایجاد می‌کنند محسوس تر است (شکل زیر را ببینید).



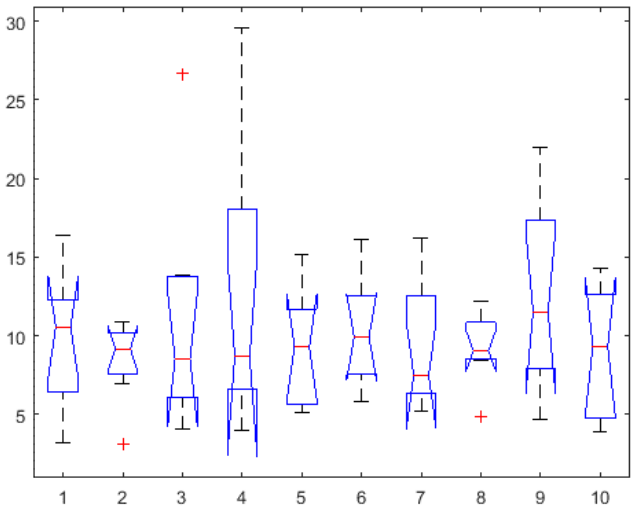
۴. پاسخ به این سوال این که چه کانال یا بازه زمانی بهتر از همه می‌تواند متمایز کننده target ها و non-target ها باشد شاید تا حدی به این بستگی داشته باشد که بعدا از کدام فیچرهای آن کانال یا بازه زمانی می‌خواهیم برای کلاسیفای کردن استفاده کنیم. به نظر می‌رسد بهترین راه تشخیص کانال‌ها یا بازه‌های تمایز دهنده استفاده از کلاسفایری با ورودی همه نقاط و همه کانال‌ها و بعد بررسی تاثیرگذاری فیچرهای مختلف باشد که در قسمت چهارم انجام خواهد شد. اما اگر تقریبی با مسئله برخورد کنیم بنابرتجربه‌های قبلی استفاده از چند نوع فیچر در داده‌های EEG معقول است که یکی از آن‌ها انرژی است (این فیچر اگر از راه‌حلی مانند matched filter برای erp استفاده می‌کردیم می‌توانست از نظر SNR اپتیموم باشد). برای هر کانال ما بازه زمانی erp را به تعدادی بازه زمانی کوچکتر تقسیم می‌کنیم و در هر بازه‌ی $[a_k\ b_k]$

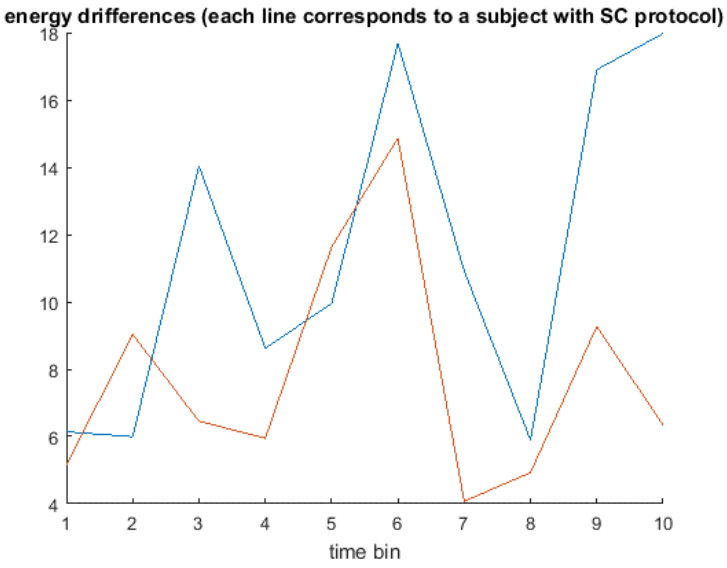
$$Ed[k] = \frac{\sum_{i=a_k}^{b_k} \left(s_{target}[i] - s_{ntarget}[i]\right)^2}{var\left(s_{ntarget}\right)}$$

را محاسبه می‌کنیم که Ed[k] نشانگر این است در بازه زمانی k ام، سیگنال‌های target و ntarget (s) تا چه اندازه به معنای انرژی از هم دورند. این کار را برای هر subject و کانال انجام می‌دهیم. برای تشخیص کانال‌های متمایز کننده، Edهای بازه‌های مختلف آن کانال را با هم جمع می‌کنیم. اگرچه بدین ترتیب کانال ارجح برای هر subject بدست آمد اما در مجموع هیچ کانالی که به طور معنی‌داری برای همه subjectها معنی‌دار باشد نیافتیم. اما برای تشخیص بازه زمانی متمایز کننده، Edهای آن بازه را در کانال‌های مختلف با هم جمع کردیم. نتیجه برای subjectهای مختلف در ادامه قابل مشاهده است.

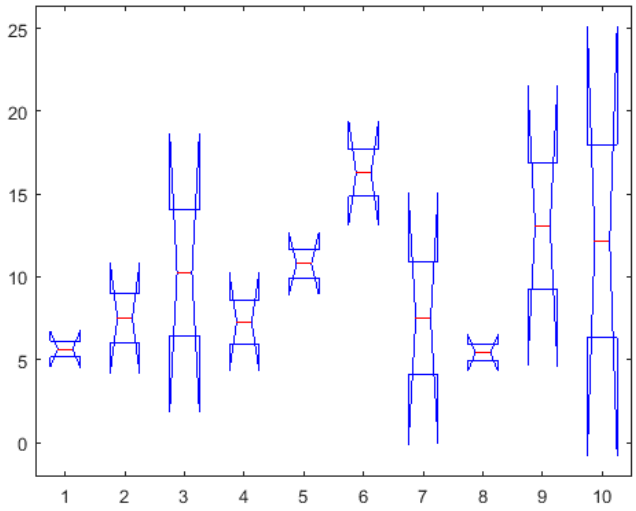


توزیع Ed در بازه‌های زمانی مختلف:





توزیع Ed در بازه‌های زمانی مختلف:



همانطور که از شکل های بالا پیدا است، به نظر می‌رسد بازه‌های زمانی ششم و نهم (حدود 400 و 640 میلی ثانیه)، تمایز دهنده‌ترین بازه‌ها از نظر Ed هستند. البته برای پروتوکل SC بهترین بازه با دامنه بسیار بزرگتری بازه 6 ام هست (حدود 400 میلی ثانیه) که با توجه به کم احتمال‌تر بودن targetها در انتظارش را داشتیم.

۵.

a- نوسانی بودن non-target erp ها و نامنظم‌تر بودن target erp ها تقریبا در تمامی subjectها برقرار هست. البته همانطور که قبلا اشاره شد مشاهده این اثر در پروتکل SC راحت‌تر است (subject 1 , 2). البته استثناهایی هم وجود دارد. برای مثال کانال 5 سابجکت 5 به نظر در داده‌گیری اشکال داشته یا

کانال 6 سابجکت 8 فرکانس نوساناتش بسیار بالاست. همچنین فرکانس نوسان erp سابجکت‌های 6 تا 10 از 3 تا 5 بیشتر است که تا حدی در تناقض با استدلال سوال 3 این قسمت است.

b- در سوال 4 به این سوال جواب داده‌ایم. به طور خلاصه با تعریفی که از بازه‌زمانی یا کانال متمایز کننده ارائه دادیم، هیچ کانال متمایز کننده مشترک معنی داری بین subject ها نبود اما بازه‌های زمانی تمایز دهنده قابل تعریف است.

c- همانطور که در سوال 4 دیدیم، در مقایسه target erp ها بین SC و RC، در SC دامنه بیشتری (با تقریبا همان تاخیر) در قله بازه زمانی مهم اوّل (حدود 400 میلی ثانیه) دیده می‌شود که انتظارش را داشتیم چرا که احتمال رخ دادن target در SC کمتر است پس فرد بیشتر تعجب خواهد کرد! (با دانش قبلی می‌دانیم دامنه و lateny، P300 با احتمال رویداد رابطه عکس دارد). در مقاله نیز به همین موضوع اشاره شده است و نتایج سازگار است.

d- از آنجا که تفاوت target و non-target در SC محسوس‌تر است انتظار داریم کلاسیفایر مبتنی بر پروتوکل SC بهتر کار کند هر چند به دلایلی که در مقاله ذکر شده است (از جمله bit rate بیشتر و بالانس‌تر بودن کلاس‌ها) RC در عمل نتیجه بهتری داده است.

۶. در این قسمت برای هر کانال و subject، target erp و non-target erp را محاسبه کردیم و درباره تفاوت کلی در پترن‌های آنها (نوسانی یا نوسان برهم‌خورده) بحث کردیم. تفاوت target و non-target در اکثر کانال‌ها و subjectها محسوس بود. سپس تلاش کردیم با تعریف معیاری وابسته به انرژی، کانال‌ها و بازه‌های زمانی‌ای را که بهتر می‌توانند target را از non-target تشخیص دهند، پیدا کنیم. کانال تمایزدهنده وابسته به subject بود اما بازه زمانی مشترک تمایزدهنده قابل تعریف بود که اتفاقا حوالی 400 میلی ثانیه اتفاق می‌افتد. در همین جا ملاحظه کردیم که سیگنال‌های SC با تعریف ما از نظر انرژی به تفاوت بین target و non-target بیشتر حساس هستند که با توجه به دانش قبلی ما از دامنه و latency P300، امری محتمل بود.

قسمت چهارم: پیاده‌سازی الگوریتم P300-Speller

۱. به همان روش قسمت قبل داده‌ها فیلتر شده(از آرایه فیلترشده استفاده میکنیم) و عملیات کم‌نمونه‌برداری برای هر کدام از حالتهای train و test و همچنین target و nontarget با برداشتن ۴ تا در میان داده‌های کانال‌ها انجام شده‌است. سپس این 4 حالت در یک struct ذخیره میشوند.

۲. کانالهای 2 تا 9 برای هریک از حالتهای target/ntarget-train/test جدا شده و کنار هم قرار داده میشوند. در نهایت طوری مرتب میشوند که ستونها متغیرها یا فیچرها باشند و سطرها trial ها یا observation ها. به این ترتیب آماده وارد شدن به classifier میشوند. ضمناً همه این 4 حالت در یک struct به صورت منظم مانند قبل ذخیره میشوند. برای ساخت مدل به predictor ها و label ها نیاز داریم. Predictor ها همان قسمت داده های train استراکت اخیر هستند و label آنهایی که target هستند برابر 1 و label آنهایی که nontarget هستند برابر 0 صفر است. به این ترتیب label ها را نیز ساخته و به همراه predictor ها با آنها مدل را میسازیم. با دستور predict، label پیشبینی شده توسط مدل برای داده های train را محاسبه کرده و تفاوت آن را با آنچه واقعا هست یافته و صحت مدل بر داده های train را تعیین میکنیم. در مرحله بعدی آن همین کار را با داده های test میکنیم و صحت را بر روی آن ها بدست می آوریم. نتیجه ها در زیر آورده شده اند.(پس از اجرای کد در بردارهای trainaccuracy و testaccuracy قابل مشاهده اند.)

برای سابجکت 3:

$$train\ accuracy = \%95.67\ , test\ accuarcy = \%74.78$$

برای 5fold cross validation میدانیم که باید داده های train را به 5 قسمت مساوی تقسیم کرد و سپس با 4 قسمت training را انجام داد و نتیجه را روی آخرین fold تست کرد. با 5 بار تغییر دادن این fold مربوط به تست و میانگین گرفتن صحت مدل روی هر تست(که خود یکی fold های داده های train است.)، به صحتی برای مدل می رسیم.

برای این کار داده ها و طبیعتاً متناظر با آنها label های آنها به طور رندم جابجا شد و به بازه های مساوی از اولین اندیس جدا میکنیم و جلو میرویم. داده ها و label ها تقسیم میشود. برای هر fold ای که به عنوان تست انتخاب میشود مدل LDA را روی بقیه 4 fold باقی مانده train کرده و صحت مدل بر روی این تست مشخص شده است. در نهایت نیز میانگین گزارش شده است:

$$kfold\ accuracy = \%71.00$$

مساله در حالت high variance است. به این معنی که یا تعداد مشاهده ها کم است و یا تعداد فیچرها زیاد است. این نتیجه گیری از این حقیقت ناشی میشود که مشاهده میکنیم مدل با داده های train تطابق بسیار بالایی دارد. اما به خوبی قادر نیست داده های جدید را classify کند.(test accuracy و kfold accuracy).

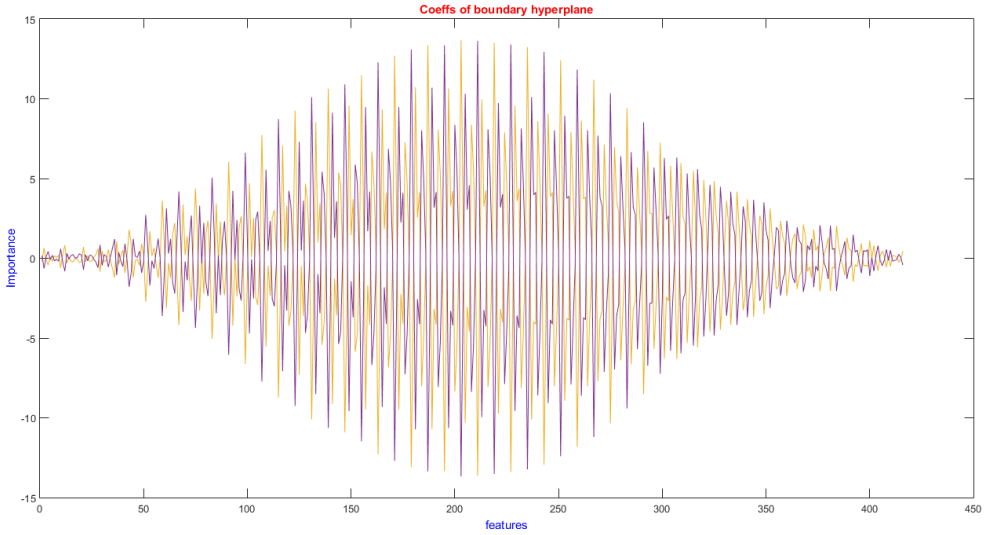
۳. برای این کار تابع IndExtraction و TrialExtraction را اندکی تغییر میدهیم و به همین علت به نام تابع های جدید متناظر این ها modified اضافه شده است. علت این تغییر این است که در این حالت target و nontarget برای ما مشخص نیستند. بنابراین ما کل داده های test را با هم خواهیم داشت. در ادامه آن ها را downsample میکنیم. سپس مانند قبل برای این که بتوانیم آن ها را به مدل بدهیم، کانال های EEG آن را کنار هم میگذاریم و از مدل label های پیش بینی شده برای آن را بدست می آوریم. همچنین با تابع اصلاح شده استخراج اندیس، اندیس مکان هایی که یک فلش زدن شروع شده است را می یابیم. Label پیش بینی شده برای این اندیس ها را بررسی میکنیم و آن هایی که این label برای آن ها یک بوده است را جدا میکنیم. این ها همان target ها هستند. عدد سطر یا ستون در متد RC و یا خانه در متد SC با این target ها بدست می آید. قدم بعدی یافتن حرف متناظر با این کدها است. دقت میکنیم که از 5 حرفی بودن کلمه آگاهی داریم. اگر متد SC باشد که در سابجکت های ما معادل دوتای اول است، آن خانه هایی که به target مربوط است را به 5 بازه مساوی تقسیم میکنیم(به ترتیب زمانی تشخیص دهیم هر حرف چه بوده است.) و در هر بازه مد داده ها را محاسبه میکنیم. سپس با کمک جدول lookup ای که نوشته ایم حرف را از بیشترین خانه ای که در هر بازه target شده است، دیکود میکنیم. اگر متد RC باشد پس از تقسیم کردن به 5 بازه، داده ها را به دو دسته آنهایی که عددشان از 6 کمتر و بیشتر است تقسیم میکنیم و مد هر دو دسته را در هر یک از 5 بازه بدست می آوریم. در هر بازه ابتدا ستون و سپس سطر با بیشترین target شدن را بدست می آوریم و با یک lookup table دیگر حرف متناظر با این دو عدد را دیکود میکنیم.

چون سابجکت ما 3 است و این سابجکت از متد RC استفاده کرده است داریم(که دو حرف آن دقیق است،حرف اول و سوم نزدیک حرف مورد نظر بر روی شکل ماتریس است.):

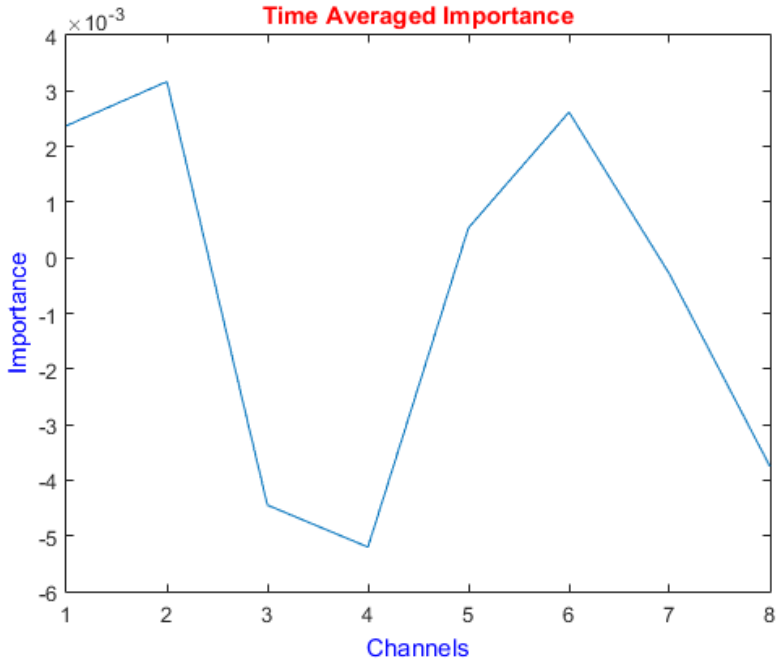
$$decoded\ word = FCQAS$$

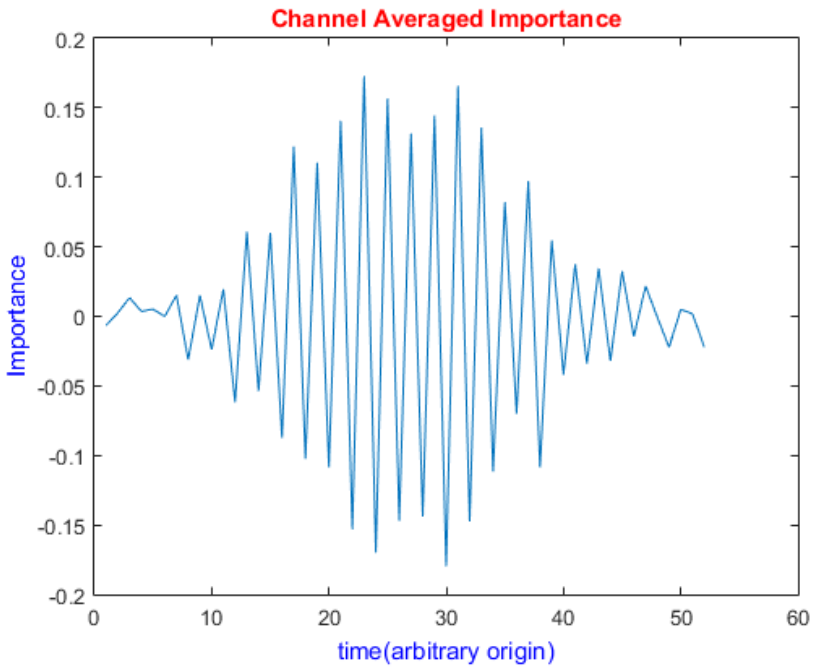
۴. پس از جستجو مشخص شد استراکت‌های coeff در مدل obj، به تعداد ۲*۲ (تعداد کلاس*تعداد کلاس) هستند و coeff(i,j) شامل مرز کلاس i با کلاس j است. به همین علت coeff(1,1) و coeff(2,2) خالی هستند و می‌توانید آن‌ها را مشاهده کنید. هم‌چنین در هر کدام از coeff(1,2) یا coeff(2,1) نیز ضرایب ابرصفحه‌ی مرز که معادله آن‌را تشکیل می‌دهند نیز وجود دارد. این ابرصفحه در معادله‌ای به فرم $Lx = const$ صدق می‌کند و هرچه L بیشتر باشد، به ازای یک x خارج مرز، از مرز بیشتر دور شده و در

واقع بهتر کلاس‌بندی می‌شود پس ضرایب بزرگتر معادل «مهم‌تر» بودن است. پس از رسم coeff(1,2) و coeff(2,1) به تصویر زیر می‌رسیم:



مشخص است که کانال های میانی اهمیت بیشتری دارند.(چون کانالها را پشت سر هم گذاشته بودیم) اما با توجه به نوسانات به نظر می رسد در هر کانال نیز زمان هایی دارای ارجحیت باشند و باعث افت و خیز اهمیت در کانالهای کم اهمیت شده باشند. اما برای بررسی دقیق تر، باید آن را از حالت پشت سر هم در بیاوریم.





مشاهده می‌شود اگر روی زمان‌ها میانگین گرفته شود و قدر مطلق اهمیت کانال‌ها مد نظر قرار داده شود کانال خاصی نسبت به بقیه برتری ندارد. اما از لحاظ زمانی اگر روی کانال‌های مختلف میانگین بگیریم، در زمان‌های میانی اهمیت بیشتری خواهیم داشت. با توجه به ۸۰۰ میلی‌ثانیه بودن کل بازه، این زمان با اهمیت‌تر حدود ۴۰۰ میلی‌ثانیه خواهد بود.

نتایج تا حد مطلوبی با سوال ۴ قسمت سوم هم‌خوانی دارد و آن‌جا هم نتوانستیم از کانال‌ها اطلاع مفیدی بدست آوریم اما همین حدود بازه زمانی مهم برای کانال‌ها بدست آمد.

قابل انتظار بودن نتیجه با توجه به الگوریتم LDA: LDA در واقع یک احتمال posteriori را ماکزیمم می‌کند که یک مشاهده در یک کلاس برود. از طرفی می‌توان نشان داد این ماکزیمم کردن احتمال، هم‌ارز ماکزیمم کردن SNR برای matched filter یا همان ERP هاست. در قسمت قبل هم زیاد بودن نسبت انرژی‌ها یا معادلاً SNR به همین نتایج این بخش منجر شد.

۵. عملیات سوال 1 برای همه سابجکت ها: این کار به سادگی در کد مانند قسمت قبل انجام شده است.

عملیات سوال 2 برای همه سابجکت ها: این کار از قسمت قبل تعمیم یافته و نتیجه در زیر آورده شده است.

Train accuracy	Test Accuracy	5-fold CV
98.7	96.15	95.26
99.19	97.19	95.78
95.67	74.78	67.00
98.56	79.22	66.11
98.22	78.78	69.00
94.67	73.44	66.00

96.78	77.33	66.22
98.11	83.11	66.56
98.44	83.00	67.56
97.78	82.56	66.00

برای سابجکت های 3 تا 10 مشکلی که قبلا به آن اشاره شد وجود دارد(high variance) و علت آن هم همان است. تعداد داده ها نسبت به فیچرها کم است و مدل فقط بر روی داده های train خوب عمل می کند ولی در مورد داده های جدید(test) نتیجه خیلی خوبی نمی دهد.

اما در مورد دو سابجکت اول مشکل چیز دیگری است. داده ها در این سابجکت skew هستند به این معنی که یک کلاس($label = 1$) خیلی کمتر از کلاس دیگر رخ میدهد(نادر است) و تعریف accuracy لزوماً معیار خوبی نیست. برای مثال اگر همه را صفر پیش بینی کنیم هم درصد صحت زیادی خواهیم داشت که بدیهی است غلط است! پس دو معیار دیگر تعریف میکنیم. نخست به جدول زیر توجه میکنیم:

	Actual value:1	Actual value:0
Predicted value:1	True positive	False positive
Predicted value:0	False negative	True negative

معیاری به نام precision تعریف میکنیم و آن را نسبت $\frac{true\ positive}{true\ positive+false\ negative}$ قرار میدهیم. تعبیر آن تعداد یک هایی است که درست تشخیص داده ایم به کل یک هایی که واقعا بوده است. با این کار اثر کم بودن یک ها در نظر گرفته میشود.

معیاری به نام recall تعریف میکنیم و آن را نسبت $\frac{true\ positive}{true\ positive+false\ positive}$ قرار میدهیم. تعبیر آن تعداد یک هایی است که درست تشخیص داده ایم به کل یک هایی که تشخیص داده ایم. در این حالت اگر مثلا همه را یک تشخیص دهیم و یک های موجود را شامل شویم صحت مسلماً 100 درصد نیست! این معیار با کم بودنش صحت واقعی را نشان میدهد. نتیجه این دو معیار برای دو سابجکت اول در زیر نشان داده شده است.

	Precision	Recall
Subject 1	5.33	10.81
Subject 2	24.00	48.65

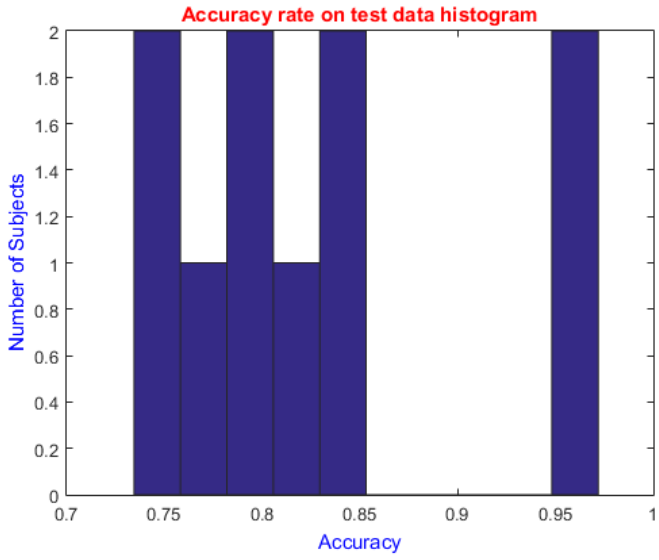
مشاهده میشود نتیجه اصلا مطلوب نیست و نباید در وهله اول فریب accuracy زیاد را خورد!

عملیات سوال 3 برای همه سابجکت ها: در یک حلقه برای هر سابجکت، مانند قبل دیتای فیلترشده downsample شده و سپس به فرمت ورودی classifier در می آید. از قسمت train آن برای هر سابجکت مدل LDA ساخته شده و سپس label های پیش بینی شده برای همه trial های هر سابجکت

تعیین میشوند. بعد از آن مقادیر عددی سطر/ستون یا خانه روشن شده که به label های 1 مربوط است ذخیره میشود. مثل قبل، برای دو سابجکت اول که متد SC دارند از روشی که در قسمت 3 توضیح داده شد برای تشخیص حرف و برای سایر سابجکت ها نیز از روش دیگر که باز در قسمت 3 توضیح داده شد برای تشخیص استفاده می شود. دقت می کنیم که فرض بر این است که ما از 5 حرفی بودن کلمه آگاهی داریم. نکته مهم این است که در هر بار اجرا شدن حلقه یک مدل LDA متناسب با سابجکت ساخته شده و بر روی داده test همان سابجکت امتحان میشود. نتیجه تشخیص حروف در جدول زیر مشخص شده است.(در workspace در متغیر decodedcharacter تشخیص متناظر با همه سابجکت ها قابل مشاهده است.

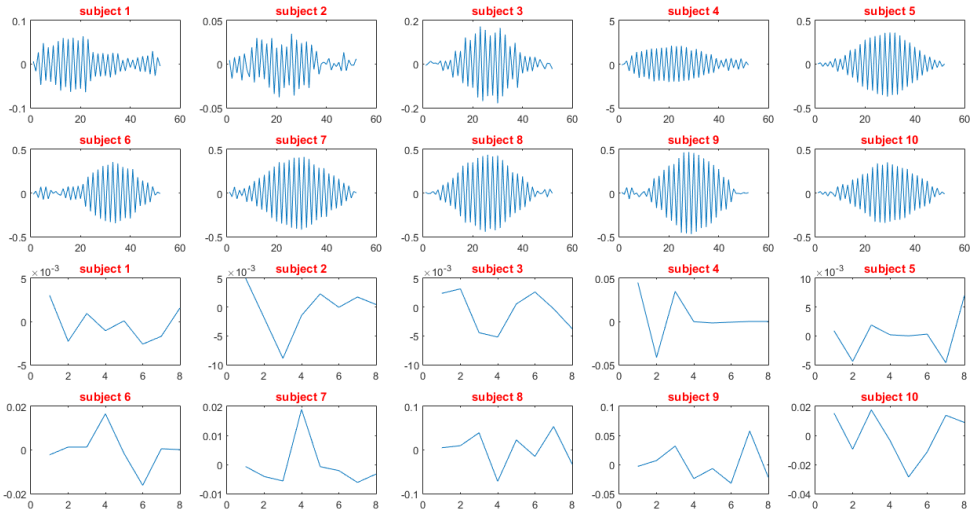
Subjects	Decoded Words
1(SC)	1K6AO
2(SC)	UKIAA
3(RC)	FCQAS
4(RC)	LUIAS
5(RC)	WATER
6(RC)	WAIZU
7(RC)	EATER
8(RC)	WATER
9(RC)	WATER
10(RC)	WGTER

الف)



ب) برای 3 نفر دقیق(5 و 8 و 9)، برای 3 نفر با یک حرف خطا(سابجکت های 4 و 7 و 10)، برای دو نفر با 3 حرف خطا(3 و 6) و برای دو سابجکت اول نیز فقط با یک حرف درست توانسته است تایپ کند.

ج) در همان حلقه مربوط به سوال سه که برای همه سابجکت ها انجام میشود، عملیات سوال 4 را نیز برای همه انجام میدهیم. نتیجه در تصویر زیر مشاهده میشود:



مشاهده میشود که دو سابجکت اول که از روش SC استفاده کرده اند به مقدار اندکی در زمان اهمیت با بقیه فرق دارند. به غیر از این حدود زمانی مناسب برای همه سابجکت ها در یک مقدار است. بهترین الکترود مانند قسمت قبل قابل تشخیص نیست. همانطور که در 10 تصویر پایین مشاهده میشود هیچ الگوی ساده و بارزی که قابل تعمیم به همه سابجکتها باشد نمیتوان یافت و به نظر نمیرسد اثر کانال آنچنان اهمیتی داشته باشد.

د) بله. تفاوت فاحشی هم وجود دارد. هم از روی معیارهای precision و recall که تعریف کردیم و هم از روی خروجی classifier مشخص است که روش SC اصلا نتیجه خوبی نداده حال آنکه روش RC نه تنها 3 نتیجه دقیق داده بلکه 3 نتیجه با 80 درصد صحت(4 حرف درست) نیز داشته است و می توان نتیجه گرفت که این روش مناسب تر است.

ه) با انتظار اولیه مان سازگاری نداشت اما با استدلال و نتیجه گیری مقاله سازگاری بسیار زیادی داشت. RC ها با توجه به دلایلی که در قسمت سوم اشاره کردیم توانستند با درصد موفقیت خیلی بیشتری کلمه مورد نظر را تایپ کنند و این الگوریتم نسبت به SC به طور کلی ارجحیت دارد.

۶. دو نتیجه مهمی که می توان گرفت این است که این الگوریتم که بر مبنای P300 کلمات را تشخیص میدهد الگوریتم بسیار قوی ای بوده و با همین 10 سابجکت نیز نتایج بسیار قابل توجه هستند و خطا کم است. همچنین برتری روش RC بر SC مشخص میشود چرا که همانطور که در قسمتهای قبل به آن اشاره شد، این سیستم برخلاف مناسب بودن بر روی داده train، بر روی داده هایی که از آن ها اطلاعات ندارد اصلا خوب عمل نمیکند. علاوه بر 5-fold accuracy در نتیجه هم مشخص است که هیچکدام از سابجکت 1 یا 2 تشخیص خوبی نداشته اند. این به دلیل skew شدن توزیع داده ها در این روش نیز هست که باعث

بی اعتباری accuracy بر روی train نیز می شود.(precision و recall معیارهای برملا کننده واقعیت بودند!) همچنین بازه زمانی مهم نیز برای همه سابجکتها تقریبا مشخص و حدود 400 میلی ثانیه پس از ابتدای بازه است. اما کانال ها از لحاظ اهمیت نسبت به هم ارجحیتی ندارند.

قسمت پنجم: سوال دلخواه

می‌دانیم LDA کلاسیفایر اپتیموم است از نظر احتمال خطای اشتباه کلاسیفای کردن داده‌های جدید که از یک توزیع آمده‌اند ([1] را ببینید.) این قضیه با فرض برقراری سه شرط است:

- توزیع فیچرها در هر کلاس گوسی است.
- ماتریس کوواریانس همه کلاس‌ها مانند هم است.
- میانگین و ماتریس کواریانس داده‌ها را می‌دانیم.

به نظر می‌رسد از این سه شرط، شرط سوم است که باعث می‌شود LDA در برخی موارد اپتیموم نباشد یا خوب عمل نکند. به خصوص این مسئله وقتی تعداد داده‌ها از تعداد فیچرها به طور قابل ملاحظه‌ای بزرگتر نباشد دیده می‌شود. به نظر می‌رسد مسئله ما در این‌جا از همین جنس است. به خصوص انتظار داریم در پروتوکل SC این اشکال بیشتر جلوه کند. لذا تلاش خواهیم کرد با راه‌حلی که در [1] تحت عنوان shrinkage ارائه شده است مرحله کلاسیفاینگ را بهبود دهیم.

در این‌جا به طور خلاصه روش shrinkage را شرح می‌دهیم.

تاکنون ماتریس کواریانس ورودی را این‌گونه تخمین می‌زدیم:

$$\Sigma = \frac{1}{\mathrm{n}-1}\sum_{i=1}^{n}(x_i-\mu)(x_i-\mu)^T$$

علی‌رغم این‌که این تخمین unbiased است، [1] ادعا می‌کند وقتی نسبت داده‌ها به فیچرها به اندازه کافی بزرگ نیست باعث می‌شود مقدارویژه‌های بزرگتر Σ بزرگتر و مقدارویژه‌های کوچکتر آن کوچکتر تخمین زده شوند. لذا می‌توان Σ را به شکل زیر tune کرد:

$$\Sigma' = (1-\gamma)\Sigma' + \gamma\nu I$$

که $\nu = \frac{trace(\Sigma')}{\mathrm{n}}$ است و $\gamma \in [0,1]$. از طرفی چون می‌توان نوشت $\Sigma = \mathrm{VDV^T}$ داریم:

$$\Sigma' = (1-\gamma)VDV^T + \gamma\nu I = V\big((1-\gamma)D + \gamma\nu I\big)V^T$$

پس می‌توان گفت Σ و Σ' برداریژه‌های یکسانی دارند اما برداریژه‌های Σ' را می‌توان regularized کرد.

راه‌های مختلفی برای انتخاب γ وجود دارد اما کاری که ما در اینجا می‌کنیم انتخاب γ برای رسیدن به بهترین هزینه در 4 fold-cross validation است. هزینه‌ای که در این‌جا تعریف می‌کنیم accuracy نیست بلکه با توجه به skewed بودن داده‌ها از precision(P) و recall(R) استفاده می‌کنیم. چون لازم است یک figure of merit برای حداکثر داشته باشیم از ترکیب فیشر آنها یعنی $\frac{2PR}{P+R}$ استفاده می‌کنیم.

لازم است که کد جدیدی برای محاسبه LDA با Σ' بنویسم. بدین منظور کد [2] را ویرایش کرده‌ایم. همچنین از توابع آماده متلب برای min کردن استفاده می‌کنیم. بدین منظور fminunc، fmincg، fminsearch و fminbnd را بررسی کردیم که نهایتا به نظر می‌رسد fminbnd مناسب‌ترین گزینه باشد زیرا می‌توانیم در پروتکل RC از ابتدا در محدوده کوچکتری جست‌وجو کنیم. لازم به ذکر است برای استفاده از این توابع، لازم است cost (و بعضا مشتق آن) را محاسبه کنیم که در توابع جدیدی انجام پذیرفته است. برای کاهش خاصیت تصادفی cost نیز میانگین 500 بار تکرار رندوم k-fold میزان قرار داده شده است!

بعد از tune شدن γ برای هر subject، کلاسیفایرها روی داده test (که از ابتدا از داده‌های train و validation جدا بود) بررسی می‌کنیم. نتایج در ادامه ارائه شده است. به نظر می‌رسد عملکرد shrinkage LDA به خصوص در پروتکل SC در مقایسه با حالت بدون آن (قسمت چهارم گزارش) قابل قبول بوده است. شایان ذکر است اجرای این بخش از کد نیاز به زمان بسیار زیادی دارد.

num	Predicted text
1-SC	'0KWVT'
2-SC	'UKAAS'
3-RC	'LUKAS'
4-RC	'LUIAA'
5-RC	'WATDR'
6-RC	'WDDLC'
7-RC	'EAZER'
8-RC	'WATER'
9-RC	'WATER'
10-RC	'WATER'

[1] Blankertz2011," Single-trial analysis and classification of ERP components — A tutorial"

[2] https://www.mathworks.com/matlabcentral/fileexchange/29673-lda--linear-discriminant-analysis