پروژه هوش محاسباتی

علیرضا خیراندیش

شماره دانشجویی :
۹۷۱۰۱۶۰۴

فاز ۱ :

در این قسمت ما سعی کرده ایم تا جایی که میشود دیتا استخراج کنیم و برای این منظور یک تابع feature extracture نوشتیم که ما دیتای ورودی و ورودی های دیگر مثل فرکانس و غیره را میدهیم و این تابع به ما به ازای تعداد لیبل ها ویژگی استخراج میکند و برای بعضی از این ویژگی ها ما به ازای هر کانال یک ویژگی به دست می آوریم برای مثال توان سیگنال های باند پایه های تتا و الفا و غیره را میتوان در نظر گرفت و بعضی وقت ها توان کل سیگنال باند پایه برای همه ی کانال های EEG به دست می آید که میتوانیم آن را در نظر بگیریم.در این تابع تنها ویژگی هایی که ربطی به label نداشتن به دست آمده اند و ویژگی هایی همچون واریانس و form factor و correlation و hist و غیره و در حوزه ی فرکانس نیز ویژگی های مهم همچون توان باند الفا بتا گاما و توان فرکانسی های مختلف را میتوان محاسبه کرد که میتوان در function دید.

کار مهمی که ما برای محسابات انجام دادیم نرمالایز کردن(هم کم کردن میانگین و یک کردن واریانس) کل داده (هم داده ی ترین و هم داده ی تست) بود و همچنین برای ویژگی ها ما آنها را به محدوده ی یک و منفی یک محدود میکنیم به این ترتیب داده های ما مقایسه پذیر میشوند.و همچنین روی داده ها کاهش بعد یا pca نیز زده شده که بدین ترتیب دیگر داده ها روی بعدی رفته اند که تقریبا بر هم عمودند و جدا سازی آنها شاید بهتر انجام شود.

برای قسمت بعد متناسب با معیار فیشر یک بعدی داده ها را sort میکنیم بدین صورت که داده هایی که بیشترین جدا پذیری از طریق معیار فیشر داشتند را به دست می آوریم و ۱۰۰ تای اول آن را انتخاب میکنیم (این ۱۰۰ تا به این دلیل است که ویژگی های دیگر شاید در گروه بهتر باشند ولی در این جا تنها تکی ها را انتخاب میکنیم.)

برای به دست آوردن ویژگی ما کار های دیگری نیز انجام دادیم و یک ویژگی های دیگری که به لیبل ها مرتبط بودند را نیز به دست آوردیم که از جمله ی آنها میتوان به CSP اشاره کرد و البته ویژگی دیگری که از طریق به دست آوردن پایه های تجربی بنابر دیتا میباشد نیز به دست آوردیم که این کار به روش EMD پیاده سازی شد که از این طریق توانستیم چند پایه برای هر لیبل به دست آوریم که این روش را میتوان در جلسه ی توضیح شفاهی اگر اهمیت داشت توضیح دهم.

بدین ترتیب از دیتا هایمان ویژگی های به دست آمده بدون لیبل و با لیبل آن را به دست آوردیم و با استفاده از معیار فیشر آن را اهمیت بندی کردیم و حال برای قسمت بعد میتوانیم از ویژگی های مهمتر استفاده کنیم.

البته معیار CSP چون در مقالات مختلف همیشه برای دسته بندی دیده شده است بنابر این این معیار را به وسیله ی فیشر نسنجیده ایم و آن را مستقیما استفاده کرده ایم.

در این قسمت گفته شده است که به ازای تعداد لایه های مختلف و به ازای شعاع های مختلف نتیجه ی کار خود را گزارش کنیم که برای اینکار چون تعداد پارامتر های قابل tune زیاد بوده است ما نتیجه ی زیاد جالب نگرفته ایم و نتیجه ی اصلی در فاز دو آمده است ولی در حالت کلی برای این قسمت باید تعداد ویژگی های مد نظر را انتخاب کنیم که من ۴۵ انتخاب برای ویژگی را مناسب میدانم و تعداد نورون ۲۵ تا را نیز مناسب میدانم بنابر امتحان های مختلف و اجرای for بر روی شبکه های مختلف و همچنین مقدار شعاع 19.5 را نیز برای شبکه ی RBF مناسب میدانم و البته برای بهترین نتیجه باید به قسمت دوم ارجاع شود.

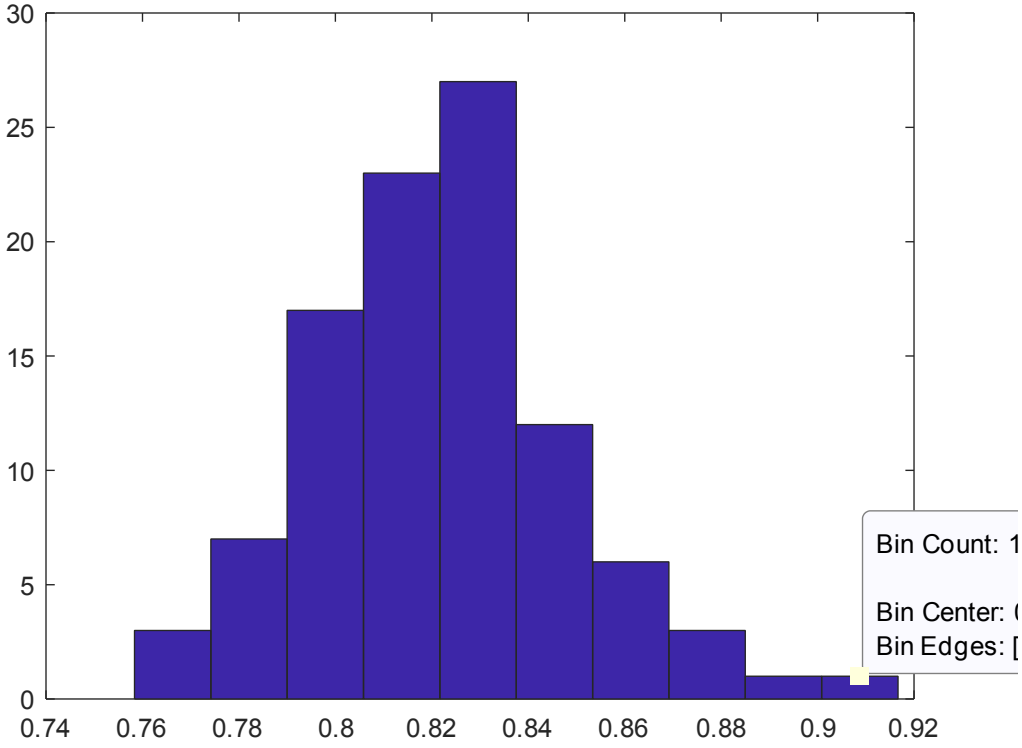
فاز ۲ :

در این فاز ما یک کروموزوم به طول ۴۵ میسازیم که آلل های آن میتواند از ۱۰۰ مشخصه ی انتخاب شده از ویژگی های فیچر با بیشترین مقدار به دست آمده است(این مقدار ۱۰۰ قابل تغییر است و معمولا بین ۱۰۰ تا ۱۵۰ جواب خوبی را میداد.) و ۲۸ تا ویژگی نیز از CSP به دست آوردیم که کلا میشود ۱۲۸ (البته در هنگام ارائه شاید مقدار متفاوتی که نتیجه بهتر میدهد را قرار دهیم ولی تا اکنون اینگونه پیش رفته ایم.) سپس معیار انتخاب را roulette wheel انتخاب میکنیم و معیار تابع هزینه را acc قرار میدهیم که سعی میکنیم آن را بهینه کنیم و تابع های mutation را تعریف و تابع 1 point cross over را نیز تعریف میکنیم.

نکته ی مهمی که در پیاده سازی به آن برخوردم این بود که اگر ما به ازای ۲۴ تا ولیدیشن و ما بقی دیتای ترین شبکه را آموزش میدادیم acc به دست آمده جنرالایز نبود و دیتا خیلی overfit میشد(دقت ۱۰۰ به دست می آمد ولی وقتی 5-fold استفاده میکردم دقت به ۷۰ میرسید که نشان میداد این تابع هزینه باید بهبود یابد.) برای همین و این که بتوان تست کرد بر روی ۱۲۰ دیتا کارمان درست است یا نه ما از ایده ی 5-fold استفاده کرده ام ولی هر بار ۲۴ دیتای اول را ولید در نظر میگیرم ولی هر بار دیتای اصلی randperm میکنم که دیتای من بر روی این ۱۲۰ دیتا فیت نشود و سعی شود همه ی داده ها را نبیند و البته مقدار acc آن با مقدار واقعی شبیه باشد.

بدین منظور اجرای برنامه خیلی کند شده است زیرا که در الگوریتم تکاملی که ما با جمعیت اولیه ی ۲۰ تایی شروع میکنم به ازای هر بار اجرا ۲۰ بار شبکه ترین میشود و همچنین تعداد جنریشن ها را نیز برابر با ۵۰ تا ۲۰۰ بنابر پردازش کامپیوتر میتوان قرار داد (برای مثال به ازای ۲۰۰ در کامپیوتر های آزمایشگاه انجام شد که توان پردازشی بیشتری داشتند.)

من به ازای آزمایش های مختلف معمولا جواب بهینه که پیدا میکردم و با همان ایده ی گفته شده حدودا در acc برابر با 0.933 بهترین شبکه ای بود که میتوانسته است بسازد بوده است.



حال اگر این شبکه را ۱۰۰ بار بسازیم و هر بار بر روی دیتای 5-fold شده اجرا کنیم به نتیجه ی acc های آمده شده در بالا میرسیم که همانطور که میبینیم میانگین حدودی 0.82 دقت میرسیم و جواب های معقولی را نیز میگیریم.

جواب به دست آمده در بالا نتیجه ی انتخاب های ویژگی زیر میباشد.

Feature_index =
[498,1200,1814,1058,65,1806,444,999,1231,113,402,1246,1059,255,189,1408,538,1775,925,1230,1101
,179,1822,896,188,1237,1239,1787,489,409,234, 1701,51,461,34,146,
1051,138,719,1754,1752,1751,1750,1749 ,1748];

شبکه ی MLP را به ازای تعداد نورون ۲۵ برای لایه ی پنهان و به ازای انتخاب ۴۵ ویژگی به دست آمده از طریق شبکه ی ژنتیکی به دست آوردیم .

حال برای به دست آوردن شبکه ی RBF با استفاده از این ویژگی های به دست آمده(۴۵ ویژگی که در قسمت قبل به دست آوردیم) میبینیم که به ازای ۱۹ نورون و شعاع معادل با spread 19.5 به دست می آید که میبینیم که دقت 0.8667 به دست می آورد.

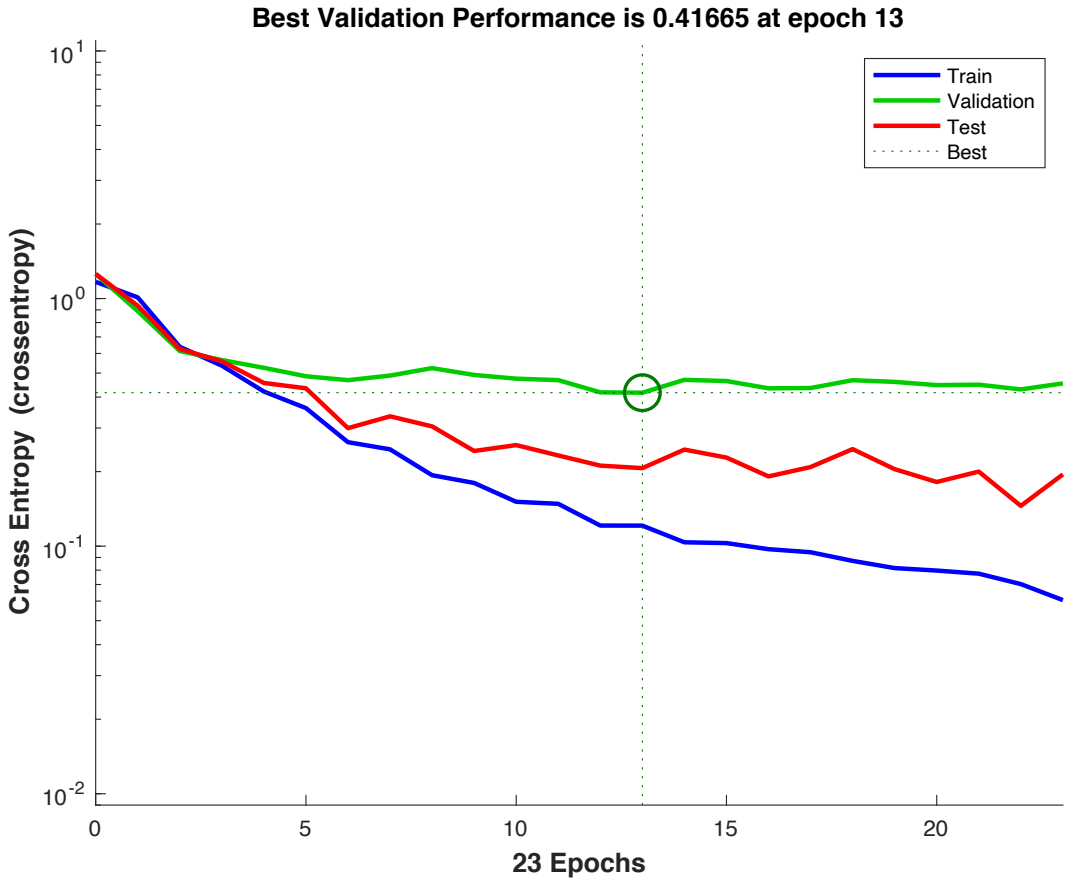
```
spread = 19.5;
Maxnumber = 19;
err = 0 ;
% using 5-fold cross-validation
% acc_1 = zeros(100,1);
% for j = 1 :100
%     total_err_1 = 0 ;
for k = 1:5
    train_indices = [1:(k-1)*24,k*24+1:120] ;
    valid_indices = (k-1)*24+1:k*24 ;
%     TrainX = ChosenFeatures(ind2,train_indices) ;
%     ValX = ChosenFeatures(ind2,valid_indices) ;
    TrainX = Normalized_Train_Features(s,train_indices) ;
    ValX = Normalized_Train_Features(s,valid_indices) ;
    TrainY = TrainLabel(:,train_indices)-1 ;
    ValY = TrainLabel(:,valid_indices)-1 ;

    net = newrb(TrainX,TrainY,10^-6,spread,Maxnumber,1) ;

    predict_y = net(ValX);
```

```
Command Window
NEWRB, neurons = 13, MSE = 0.0812046
NEWRB, neurons = 14, MSE = 0.0795311
NEWRB, neurons = 15, MSE = 0.0783629
NEWRB, neurons = 16, MSE = 0.0745638
NEWRB, neurons = 17, MSE = 0.0717991
NEWRB, neurons = 18, MSE = 0.0682047
NEWRB, neurons = 19, MSE = 0.0665379
    0.8667
```

حال که محاسبه ها را کردیم و با این مقادیر سعی میکنیم برای test ها لیبل بگذاریم و برای لیبل گذاری MLP ما آزمایش را ۱۰۰ بار تکرار میکنیم و از آنجایی که تعداد لیبل های یک و صفر برابر میباشد بنابر این ما نیز ۲۰ تا لیبل ۱ و ۲۰ تا لیبل ۲ میگذاریم و به ازای لیبل های به دست آمده در این صد تکرار آنهایی که بیشترین لیبل یک داشتن را sort میکنیم.



یک نمونه از اجرا و cross entropy آمده است.

نتیجه در فایل های mat. آپلود شده آمده است.

توضیحات بیشتری اگر لازم بود در توضیحات شفاهی داده میشود.