علوم اعصاب محاسباتی

گزارش تمرین ۳

فرآیند لرنینگ در نورون ها

عرشیا حسینمردی

شماره دانشجویی : 98222030

هدف از این پروژه پیاده سازی STDP و R-STDP است .

فایل کد با نام CN3 در کنار همین فایل گزارش قرار دارد.

بخش اول تمرین :

در بخش اول هدف ایجاد 3 نورون است که هر سه به همدیگر متصل هستند و از ما میخواهد که تغییرات وزن ها ا از روش stdp بررسی کنیم ، و ما از حالت pair-based استفاده میکنیم ؛ چون ما در اینجا 3 نورون داریم هر نورون میتواند نقش pre یا post را بازی کند و بستگی به حالت ممکن است مثلا 3 نورون به شکل pre,pre,post باشند. در ادامه به توضیح کد میپردازیم:

در بخش اول کلاس های lif و تابع های آن است که در تمرین اول و دوم دیده ایم یک سری تغییر به تابع پتانسیل داده ایم تا به هدفمان در صورت سوال برسیم.

سپس ما تابع ()STDPfunc را داریم که هدف کلی آن محاسبه وزن ها با توجه به نورون ها است.

و همینطور اجرای الگوریتم STDP است که در قسمت کد مشخص شده است.

چندین تابع plot داریم currentPlot که به ما جریان را نشان میدهد ، STDPplot که برای نشان دادن ΔW بر روی ΔT است و تابع rastorPlots که برای نمایش fire کردن است که در تکلیف سری 2 نیز با این تابع کار کردیم و weightPlot که برای نمایش وزن ها به کار میرود.

در ادامه به بررسی مثال هایی با جریان های متفاوت از عملکرد STDP را میبینیم:

شروع حل مثال از صفحه بعدی

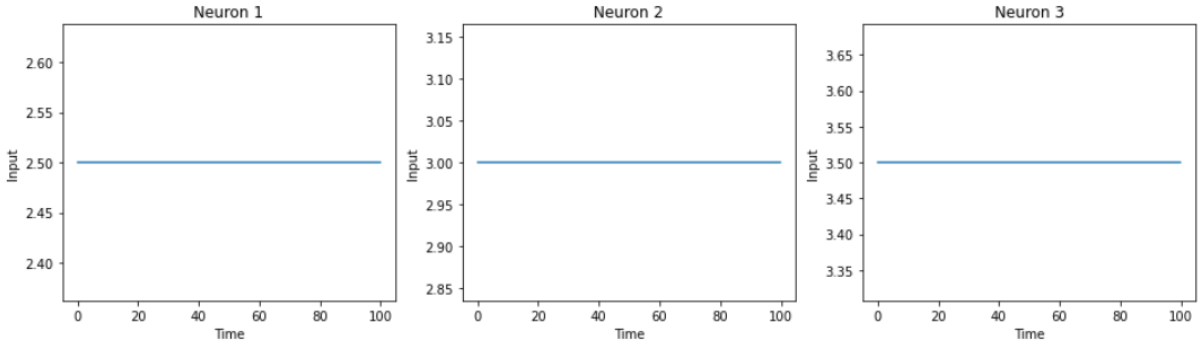
مثال 1)

در مرحله اول ما 3 نورون داریم که به آنها جریان های ثابت 2.5 ، 3 و 3.5 میدهیم و تابع STDPfunc را روی آنها اجرا میکنیم ،وزن های به دست آمده به شکل زیر هستند :

برای خواندن وزن ها توجه شود که وزن های روی قطر که 0.3333 هستند بی نیاز هستند چون مربوط به رابطه نورون 1 با نورون 1 ، نورون 2 با نورون 2 و ... است که در واقع وزن های مورد نیاز ما نیستند ؛ در ادامه باید بگوییم که شکل خواندن وزن ها مثل ماتریس است یعنی وزن نورون post 1 و نورون pre، 2 باشد برابر 0.57923055 و به همین شکل برای نورون 3 و نورون 1 برابر 0.38837 است و ....

```
weights:
 [[0.33333333 0.57923055 0.37873947]
 [0.16593919 0.33333333 0.66132504]
 [0.38837907 0.1621324  0.33333333]]
```

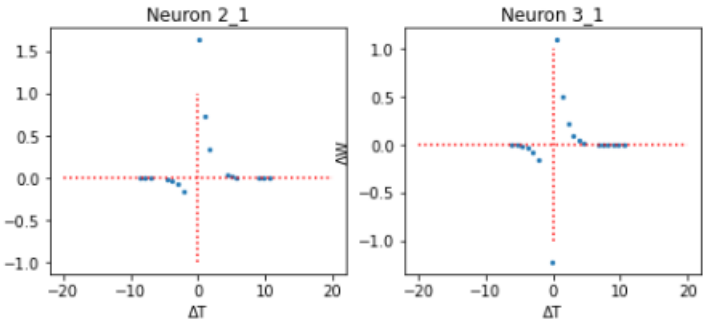
تابع جریان نورون ها را میتوانید ببینید :



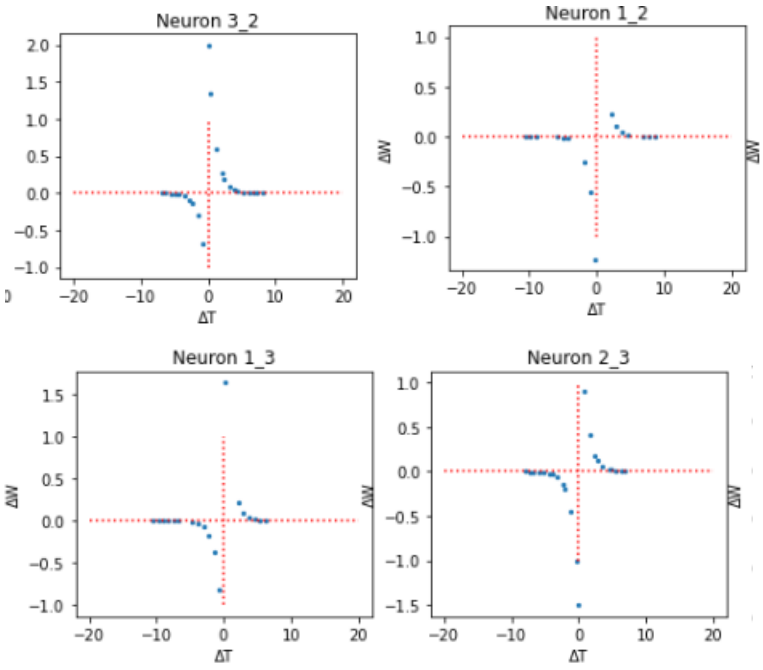
تابع STDPplot :

را که توضیحش را دادیم نیز میتوانید

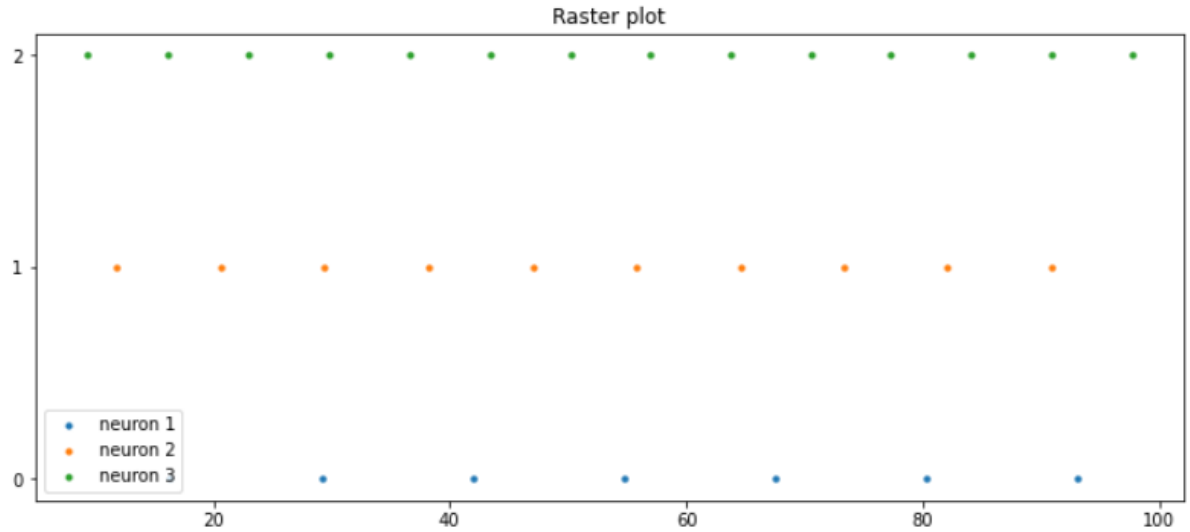
در اینجا مشاهده کنید



ادامه اشکال در ابتدای صفحه بعدی



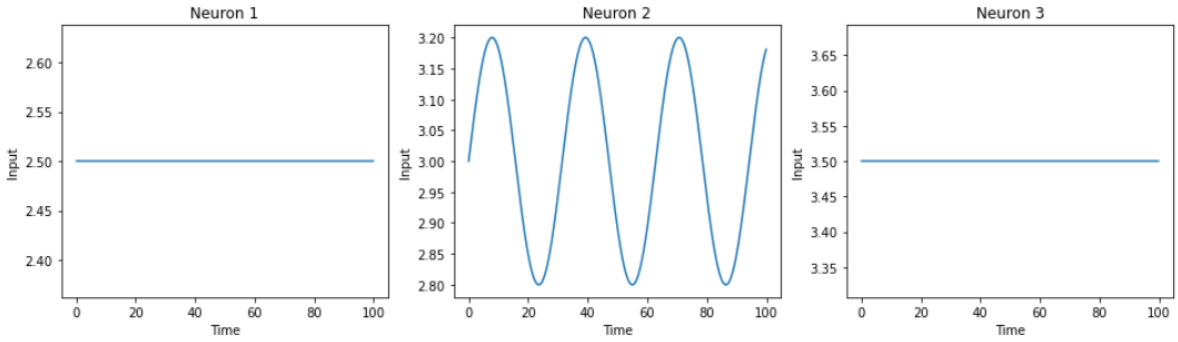
تابع rastorplot :



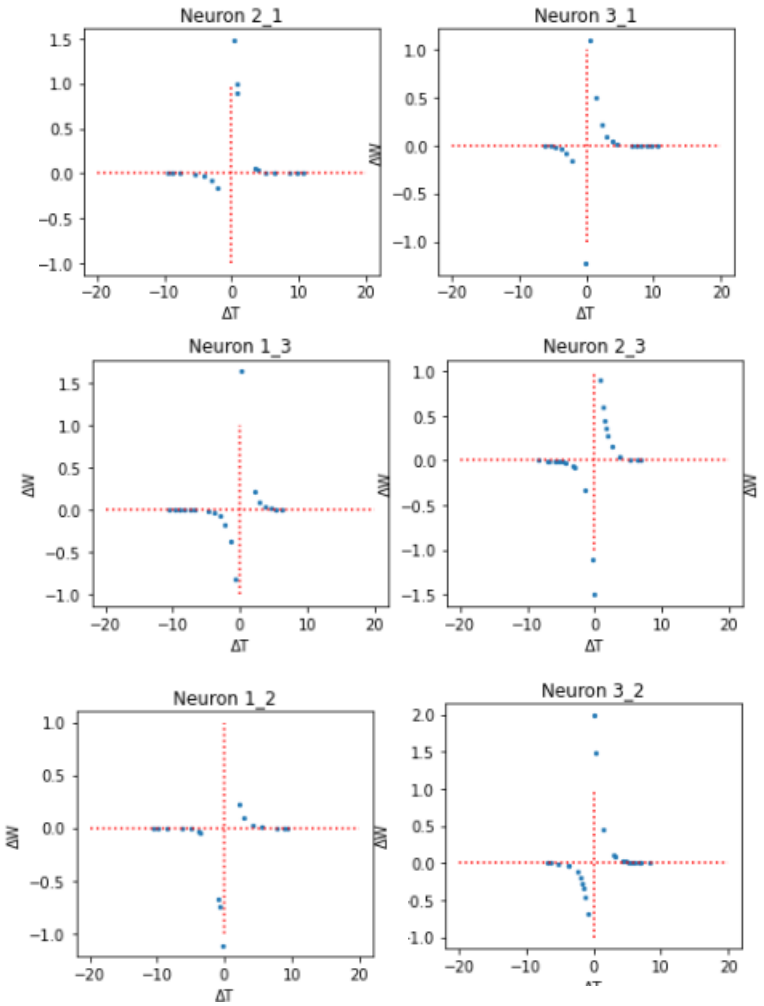
تابع شکل وزن ها را نیز در کد مشخص است ولی چون در گزارش شکل تغییرات آنرا و همینطور شکل fire کردن نورون ها را قرار دادیم به دلیل اینکه گزارش شلوغ نشود ، درون گزارش قرار نداده ایم ولی در کد میتوانید مشاهده کنید.

مثال دوم)

در مثال دوم تابع جریان به شکل زیر است:

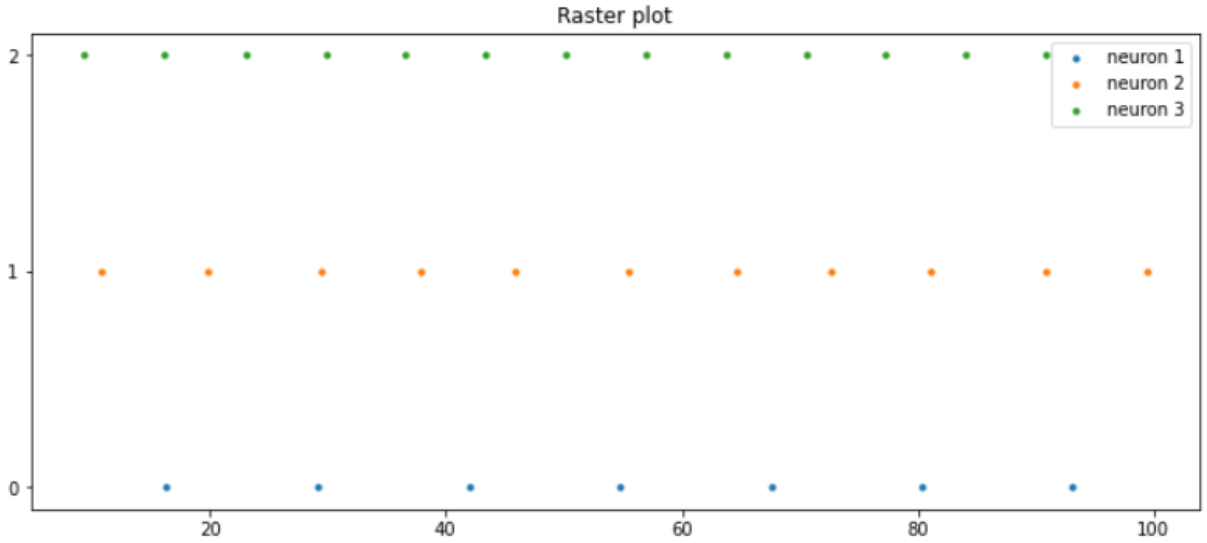


همچنین وزن ها و تغییرات آن به شکل زیر میشود :




```
weights:
 [[0.33333333 0.65548376 0.37873947]
 [0.10750066 0.33333333 0.53733478]
 [0.38837907 0.30522506 0.33333333]]
```

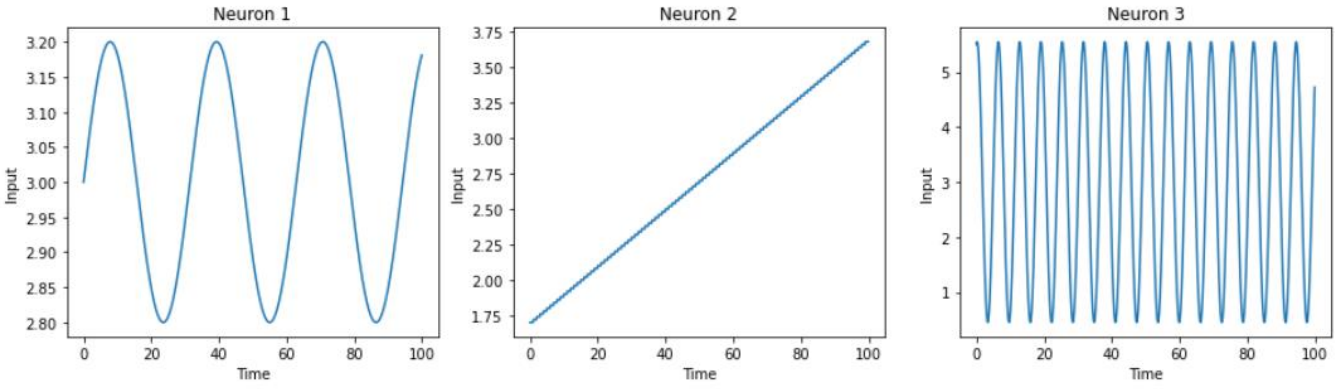
همچنین شکل rastorplot آن به شکل زیر است:



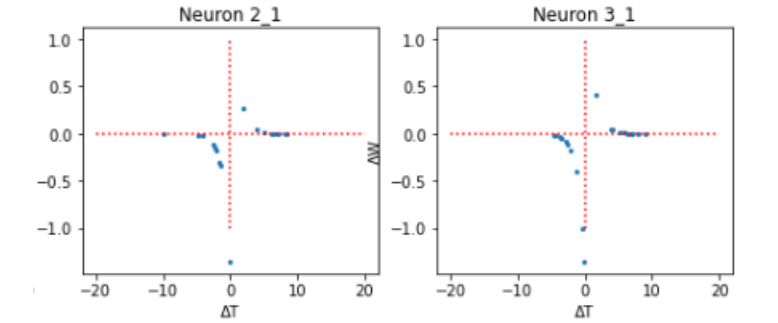
مثال سوم)

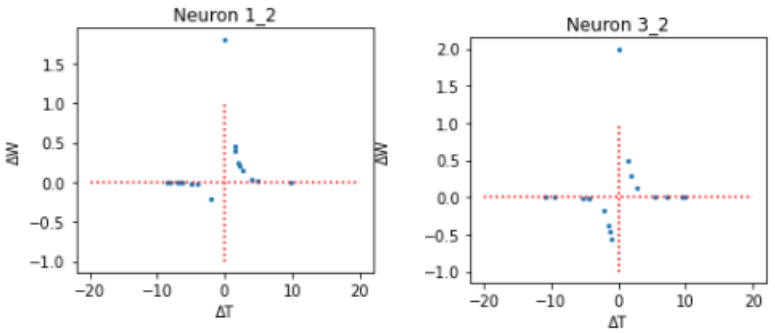
و در مثال آخر که به همه تابع جریان های غیر ثابت میدهیم:

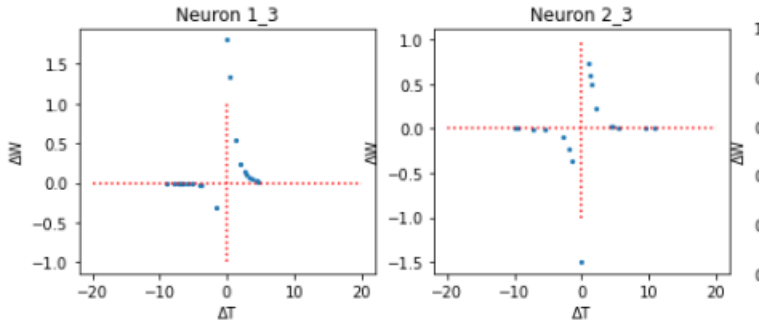
(جریان های سینوسی ،پله ای ، و ترکیب سینوسی و کسینوسی)



وزن ها و تغییرات مثال سوم به شکل زیر میشود :

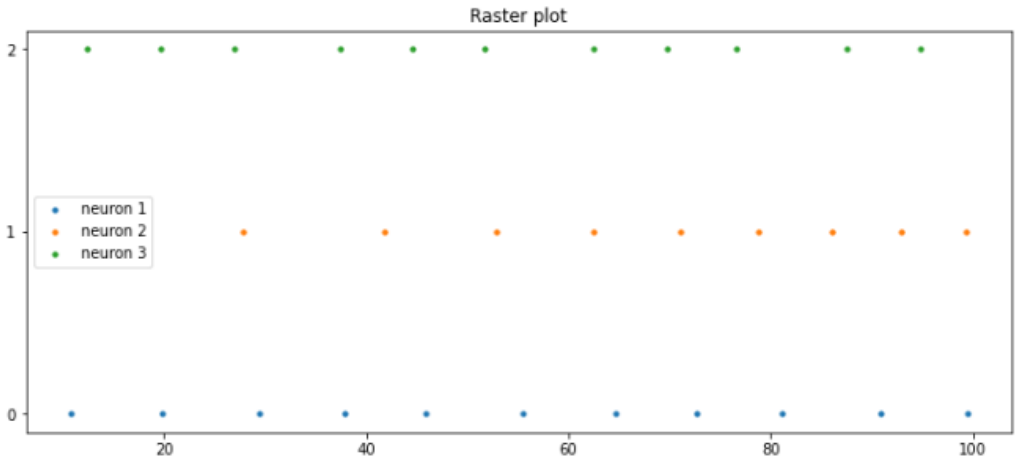







```
weights:
 [[0.33333333 0.65548376 0.37873947]
 [0.10750066 0.33333333 0.53733478]
 [0.38837907 0.30522506 0.33333333]]
```

و شکل rastor plot آن:



با حل مثال بالا به درک خوبی از روش stdp رسیدیم. و تغییرات وزن ها بین نورون ها را بررسی کردیم .

بخش دوم تمرین:

در بخش دوم برای اتصال بین این 5 نورن که 5 تا ورودی و 2 تا خروجی است از پکیج های آماده موجود در brain2 استفاده میکنیم و همینطور از تابع learnProcess که کار آن هندل کردن نورون ها و آموزش آنها با استفاده از داده های train است یک سری تابع دیگر نیز هستند که در واقع بخشی از روابطی است که ما در reward-base-stdp به آن نیاز داریم که میتوانیم در قسمت کد مشاهده کنیم.

همچنین با استفاده از پکیج pandas فایل حاوی داده های train و test را میخوانیم و این دو قسمت را از یکدیگر جدا میکنیم. و در learnProcess از آنها برای آموزش استفاده میکنیم.

در پایین میتوانید دقت روش r-stdp ای که ما پیاده سازی کرده ایم را در دو قسمت test و train مشاهده کنید.

```
accuracy of SNN on train data:  78.1 %
accuracy of SNN on test data:  69.3 %
```