ایده و توضیحات کلی :

ابتدا داده ها را به دو قسمت زن و مرد تقسیم کردم تا در نهایت کار بتوانیم شبکه را بطور جداگانه با داده های

1-فقط زن

2-فقط مرد

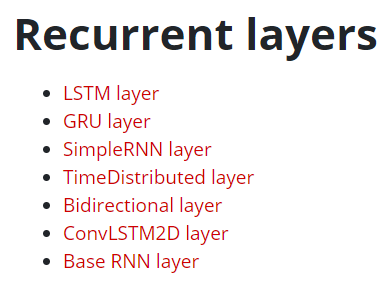
3-هر دو زن و مرد

آموزش داده و تفاوت های دقت آن ها را در کلاس های متفاوت بررسی کنیم

قابل توجه بود که توضیع کلاس ها در داده های زن و مرد کمی با هم تفاوت داشتند که در ادامه خواهیم دید

پس از تقسیم داده ها باید از آن ها با کتابخانه لیبروسا ویژگی هایی با سایز مشخص و فیکس استخراج شود تا بتوانیم آن ها را به شبکه بخورانیم ( برای اینکار سایز های 18 و 36 و 72 تست شده اند و بهترین آنها برای ادامه کار انتخاب شده است)

سپس داده های train را به دو بخش valid و train تقسیم میکنیم تا بتوانیم بعدا با داده های valid نشان دهیم هر کلاس با چه دقتی محاسبه شده است

سپس داده ها را به این شبکه ها میخورانیم تا مقایسه ای از آن ها را داشته باشیم .این عکس لیست تمام شبکه های sequential کتابخوانه کراس را نشان میدهد

برای این دیتا ست از شبکه های درون عکس فقط میتوانیم از LSTM و GRU و SimpleRNN و Bidirectional استفاده کنیم

که مقایسه آن ها را در جزییات خواهیم دید

بعد از بدست اوردن بهترین مدل که با داده های (هردو زن و مرد) بدست آمد. تاثیر داشتن سایز های مختلف ویژگی های استخراج شده و داده های جداگانه زن و مرد را بررسی خواهیم کرد

جزییات :

بعد از تست کردن و آزمایش هاپیر پارامتر های مختلف که شرح آنها در زیر آمده است به شبکه ی بهینه تر رسیدیم :

استفاده از batch\_normalization در بین لایه خروجی شبکه با حافظه و ورودی شبکه کلاس بندی کننده

* اما با توجه به این که از ابتدا داده های ما نرمال سازی شده بود نه چندان کمکی کرد و همچنین نوسانات بسیار شدیدی نیز در نمودار دقت دیده میشد پس از آن صرف نظر شد
* از دو regularization در لایه اول شبکه با حافظه استفاده شد باز هم بدلیل نرمال بودن ورودی تاثیر چندانی نداشت اما کمی و فقط کمی باعث بهبود دقت میشد و کمی نیز به آرام کردن نوسانات در نمودار دقت کمک میکرد پس این تکنیک را در لایه بعدی نیز قرار دادیم
* برای لایه (های) با حافظه سعی کردم لایه های فعال ساز مختلف را امتحان کنم. فعال ساز های خطی بشدت نتیجه بدی داشتند تا حدی که با استفاده از relu تقریبا هیچ پیشرفتی در دقت نبود (احتمالا به دلیل ساده بود ساختار شبکه ها)
* برای تابع بهینه ساز adam و sgd و rmsprop امتحان شدند. Sgd در مینیمم محلی بنظر میامد گیر میکند و پیشرفتی بعد از 37 درصد نشان نمیداد. از بین adam و rmsprop استفاده از rmsprop کمی نتیجه بهتری داشت و از آن در حالت های بعد استفاده شد
* بچ سایز های مختلفی نیز انتحان شد که بهترین آن ها 64 بود (احتمالا به دلیل مقدار کم داده ها بچ سایز های بزرگتر نواسانات بسیار شدید داشتند. بچ سایز های کوچک تر هم باعث اور فیتینگ میشدند)
* برای ویژگی های استخراجی از دو ویژگی پیشنهاد شده در ویدیو جلسه سوم تی ای mfcc و cens و هر دو با هم امتحان شد که نتیجه mfcc تنها از هر دو بهتر بود! (حتی از استفاده mfcc و cens بطور همزمان و ورودی 2 چنل نیز کمی بهتر بود احتمالا به دلیل متفاوت بودن زمان داده ها cens نمیتوانست آنگونه که باید شخصیت داده ها را حفظ کند)

پس از امتحان کردن حالات بالا برای شبکه به این هایپر پارامتر ها رسیدیم :

بهینه ساز : rmsProp

تابع محاسبه خطا : catagorialCrossentropy

توابع فعال ساز : tahn بجز لایه اخر که softmax بود

سایز بچ : 64

تعداد ایپاک ها : 50 (بجز شبکه های RNN که 10 بودند)

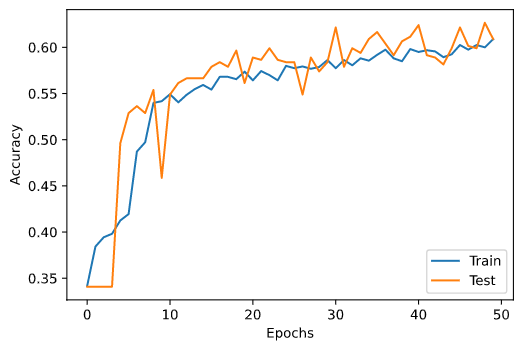
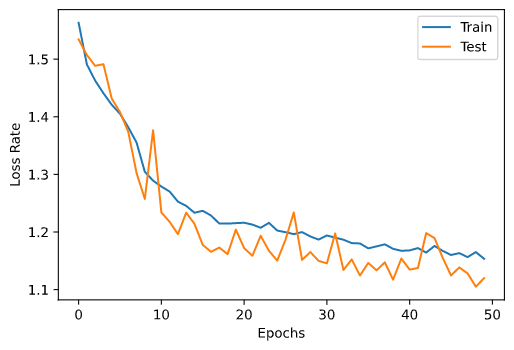
تکنیک regularization : kernelRegularization : l1(0.01) & activityRegularization : l2(0.01)هر دو این تکنیک ها رو لایه اول ساختار حافظه دار پیاده شدند

ویژگی استخراجی : mfcc

سایز ویژگی استخراجی : 36 ( البته انتظار میرفت سایز 18 برای ساختار SimpleRNN نتیجه بدی به همراه نداشته باشد)

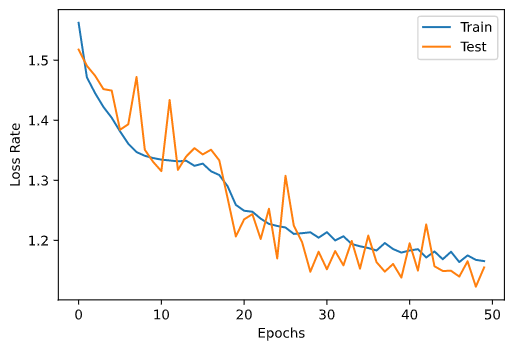
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

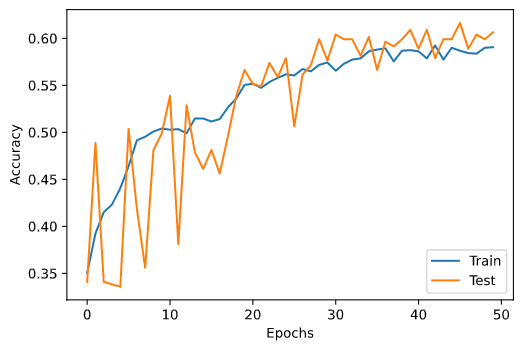
در زیر مقایسه انواع مختلف شبکه های با حافظه ای که کراس پشتیبانی میکند و مناسب داده های ما بود را میبینیم :

LSTM و GRU

LSTM : val\_acc = 65

Generally did better than GRU





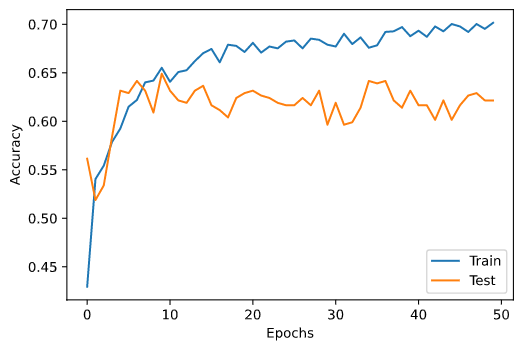
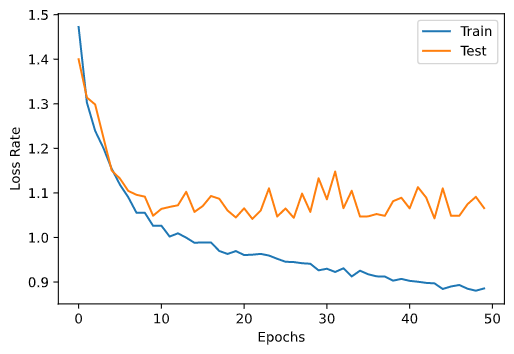
GRU : val\_acc = 62

and a lot of fluctuations on GRU comparing to LSTM

برای استفاده از SimpleRNN انتظار میرود نتیجه خوبی به بار نیاید چرا که با داشتن 36 گام زمانی انتظار میرود RNN ها به مشکل محو شدگی گرادیان بر بخورند (البته با داشتن 18 گام زمانی این مشکل کمتر و با داشتن 72 گام زمانی این مشکل بیشتر انتظار میرود)

اما به طرز شگفتی این اتفاق نیفتاد و SimpleRNN نه تنها نتیجه بدی نداشت (62 درصد دقت. تقریبا برابر GRU بلکه سرعت رسیدن به این دقت در آن بسیار بالا بود (10 ایپاک)

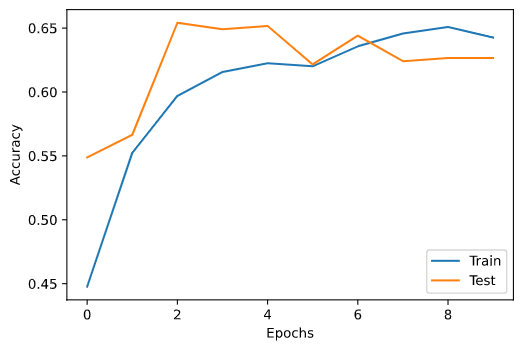
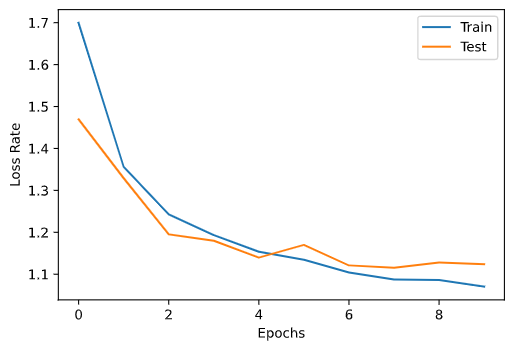
به همین منظور سعی کردم دو نوع SimpleRNN را امتحان کنم. اولی یک شبکه بسیار ساده با فقط یک لایه SimpleRNN و دومی داشتن چند SimpleRNN بطور متصل به هم (2 و 3 لایه امتحان شد که 2 لایه نتیجه بهتر و اورفیتتینگ کمتری ارایه میداد)

SimpleRNN : did so much better than expected.



این نمودار نشان میدهد با استفاده از این شبکه RNN چقدر از هر کلاس در داده های valid درست تشخیص داده شده اند

همانطور که انتظار میرود کلاس های اخر که تعداد نمونه های کمتری دارند درصد دقت بسیار پایینی نیز از خود نشان میدهند



2 SimpleRNNs

Connected together :

Overfitting reduced but not much improvement of accuracy

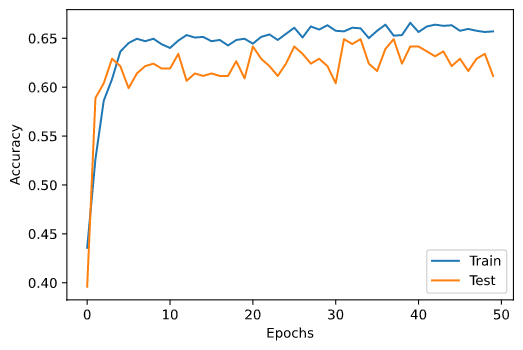
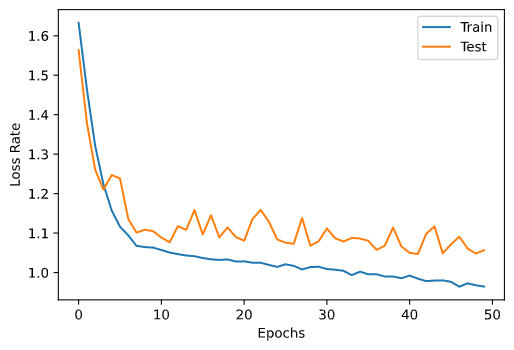
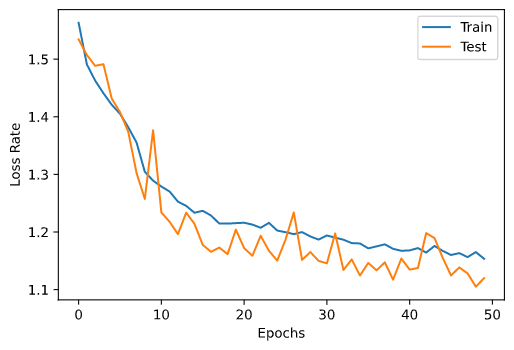
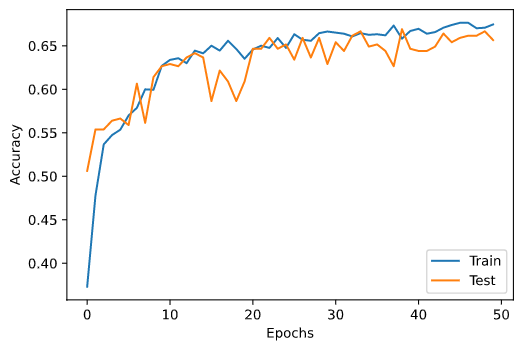


2 SimpleRNNs Connected Together Heatmap :

This version had better predictions on the first three class and yet worse results on the last 2 classes

با اینکه استفاده از RNN ها دقت قابل قبول و سرعت بسیار خوبی داشت هنوز Bidirectional ها میتوانند نتیجه کمی بهتری ارایه دهند

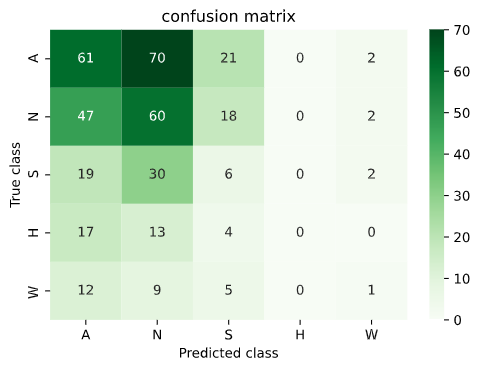
Bidirectional LSTM and GRU :

BidirectionalGRU :

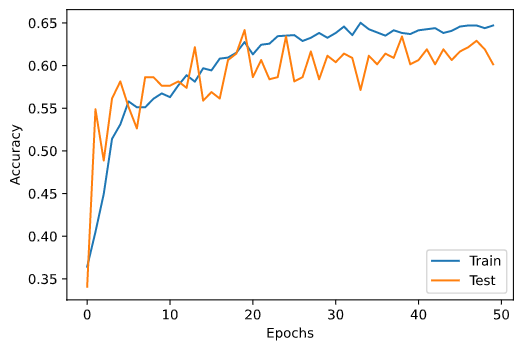
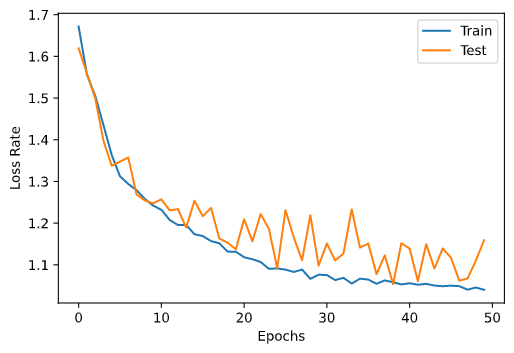
Extremely good rate of convergence though the accuracy a little less than BiLSTM

Bidirectional LSTM :

Less overfitting, and around 5 percent better than GRU

BidirectionalLSTM Heatmap :

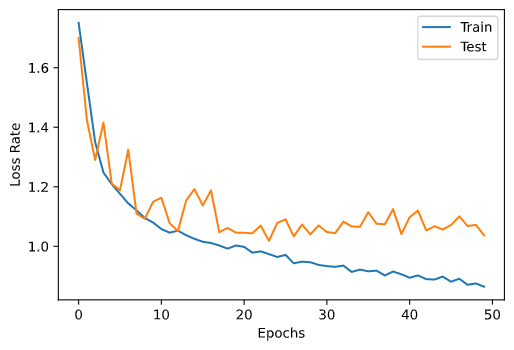
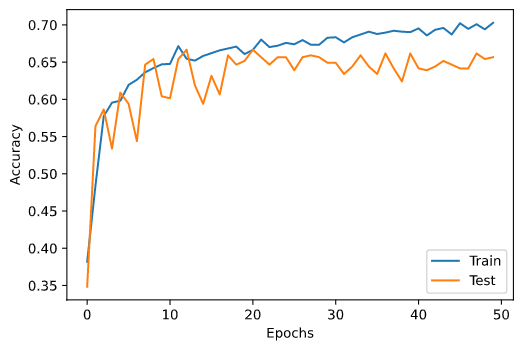
حال میتوانیم برای امتحان کردن سایز های مختلف ویژگی استخراجی و داده های ورودی زن و مرد بصورت جداگانه از BiDirectional LSTM که نتیجه بهتری داشت استفاده میکنیم



استفاده از سایز ویژگی های استخراجی برابر با 18 :

انقدری که انتظار میرفت کاهش دقت نداشتیم (افزایش سرعت و بهینگی حافظه در زمان های حجیم بودن دیتا ها میتواند انتخاب مناسب تری باشد تا افزایش کم دقت)

اما با توجه به داده ها ترجیه بر این است که سایز بالاتری را انتخاب کنیم.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



استفاده از سایز ویژگی های استخراجی برابر 72 نیز به اندازه ای که انتظار میرفت در افزایش دقت کمکی نکرد. البته نه تنها افزایش دقتی نداشتیم ( در مقایسه با سایز 26 که دقت 65.7 داشتیم و اینجا حدودا 65.2 بود ) بلکه به اورفیتینگ هم برخوردیم

اما حداقل "یک" داده از کلاس H درست تشخیص داده شد xD

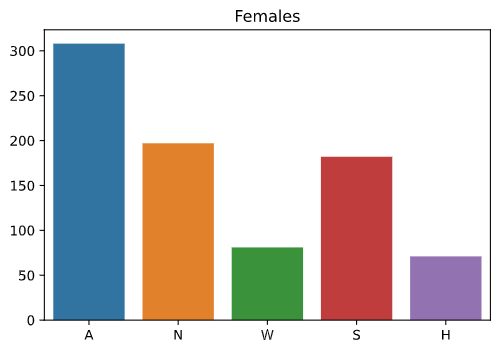
میتوان به این نتیجه رسید که استفاده از سیاز برابر 36 هم از نظر زمانی هم از نظر دقت و جلوگیری از اورفیتینگ انتخاب بهتر و مناسب تری برای شبکه های با حافظه ای مثل LSTM میباشد

البته سایز 18 برای شبکه ای با حافظه کوتاه مدت مانند SimpleRNN نیز امتحان شد و همانطور که انتظار میرفت دقت در برابر سایز 36 کمتر نشد (مقدار کمی نیز بهتر شد) چرا که محو شدگی گرادیان در این گونه شبکه با سایز های کمتر کاهش پیدا میکند که خود میتوامد باعث افزایش دقت شبکه شود

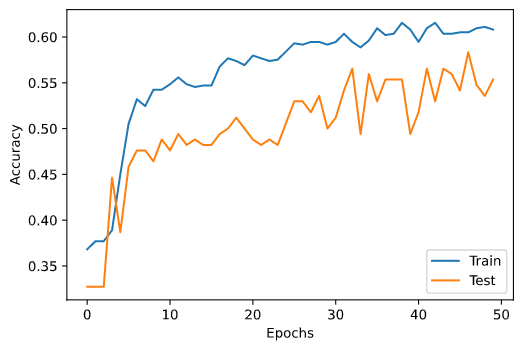
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

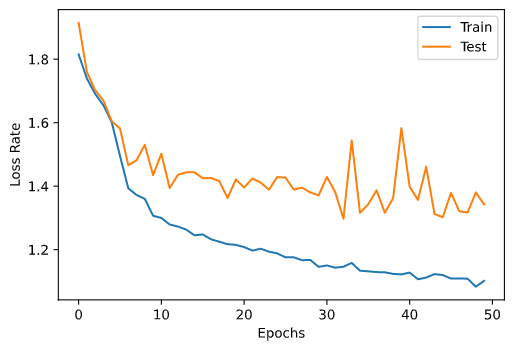
امتحان کردن داده های زن و مرد بصورت جداگنه :

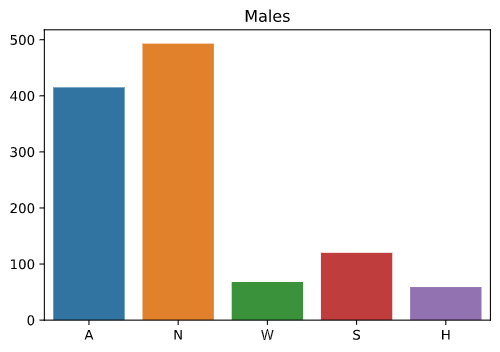
داده های زن :ابتدا توضیع این داده ها را میبینیم



این داده ها از نظر تعداد بسیار محدود بودند و به همین دلیل هم انتظار اورفیتینگ میرفت همچنین از نظر دقت نیز کاهش داشتیم که این نیز قابل انتظار بود

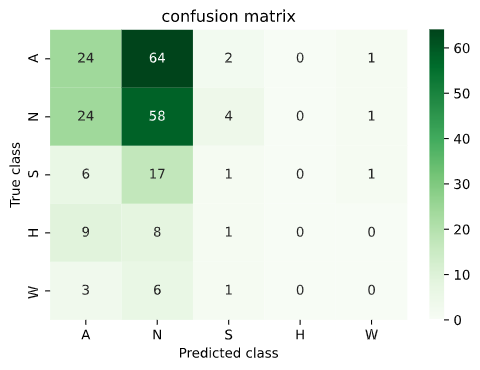




داده های مختص مرد :

این داده ها از نظر تعداد به نسبت داده های مختص زنان بیشتر هستند و همچنین توضیع کلاس ها نیز غیر متوازن تر است

در حالت منطقی باید انتظار داشت که در این داده ها به دلیل توازن کمتر دقت کمتری نیز داشته باشیم



اما همانطور که در چارت روبرو میبینیم با توجه به تعداد بسیاز کم داده ها و به تبع آن تعداد حتی کمتر داده های validation تقریبا همه داده های سه کلاس آخر غلط و نسبتا دقت بهتری در دو کلاس اول داریم که باعث میشود در مقیاس کلی به دقت بهتری حدود (68) درصد دقت برسیم

