گزارش پروژه ۴

مهرانه مقتدایی فر ۹۷۲۲۲۰۸۶ و محمدرضا صیدگر ۹۷۲۲۲۰۵۵

تشخیص احساسات:

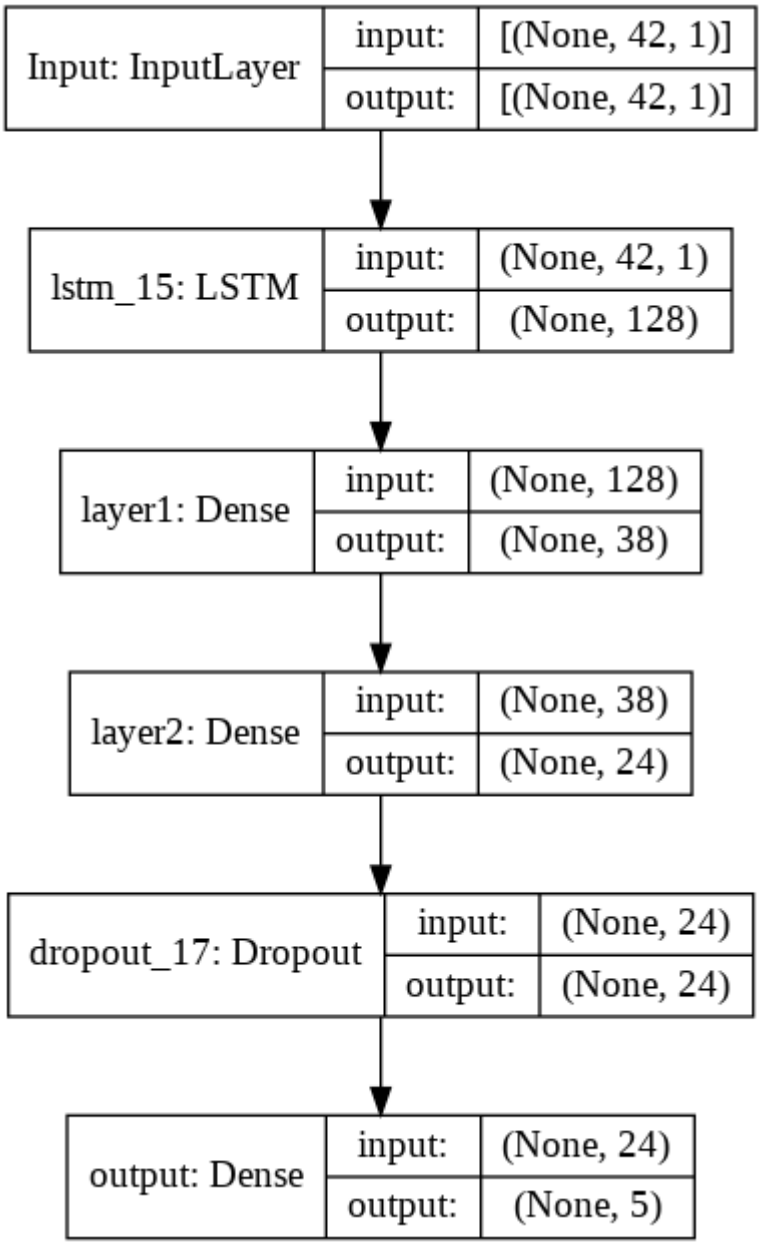
همانطور که میدانیم در برای تشخیص احساسات در صدا ما نیاز به شبکه های دارای حافظه داریم مثل شبکه های بازگشتی اما RNN میدانیم که حافظه طولانی مدتی آنچنان ندارد و همینطور مشکل vanishing gradient که برای حل این مشکلات شبکه های جدید تری مثل LSTM و GRUو ... آمدند که دارای حافظه طولانی تر هستند . روی داده های صدا عملکرد بهتری دارند پس ما برای تشخیص احساسات در صدا بیشتر از LTMS استفاده کردیم و در یکی از مدل ها هم عملکرد شبکه GRU را مورد بررسی قرار دادیم.

همینطور میدانیم که ما برای حل این مسئله باید یه سری ویژگی ها از صدا ها دربیاریم و اون ویژگی ها را به شبکه میدیم تا فرایند یادگیری انجام شود.برای این کار از کتابخانه librosa استفاده کردیم که نتایج استفاده از برخی از آن ها در بررسی مدل ها مشهود است ولی در کل عملکرد mfcc از بقیه با توجه به مطالعات ما بنظر باید بهتر باشد.

https://ieeexplore.ieee.org/document/8300161

ما توسط mfcc ۴۲ ویژگی از هر صوت در میاوریم و اون رو به ورودی شبکه حافظه دارمان میدهیم از cens هم استفاده میکنیم باز هم با ۴۲ ویژگی و همینطور از poly که خروجیش فقط ۲ ویژگی دارد.

بررسی مدل های مختلف:

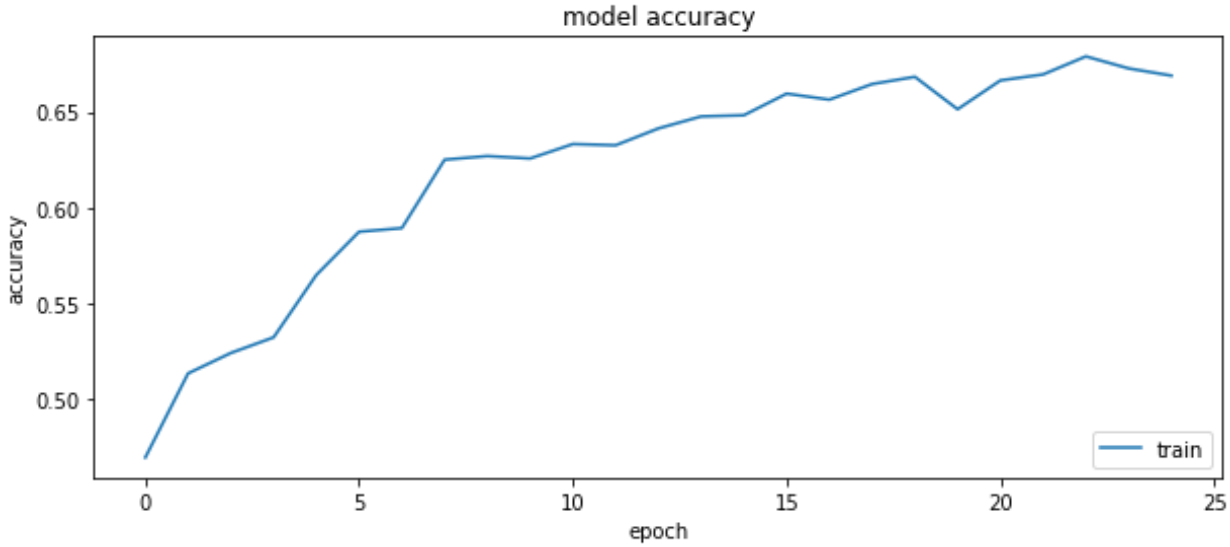


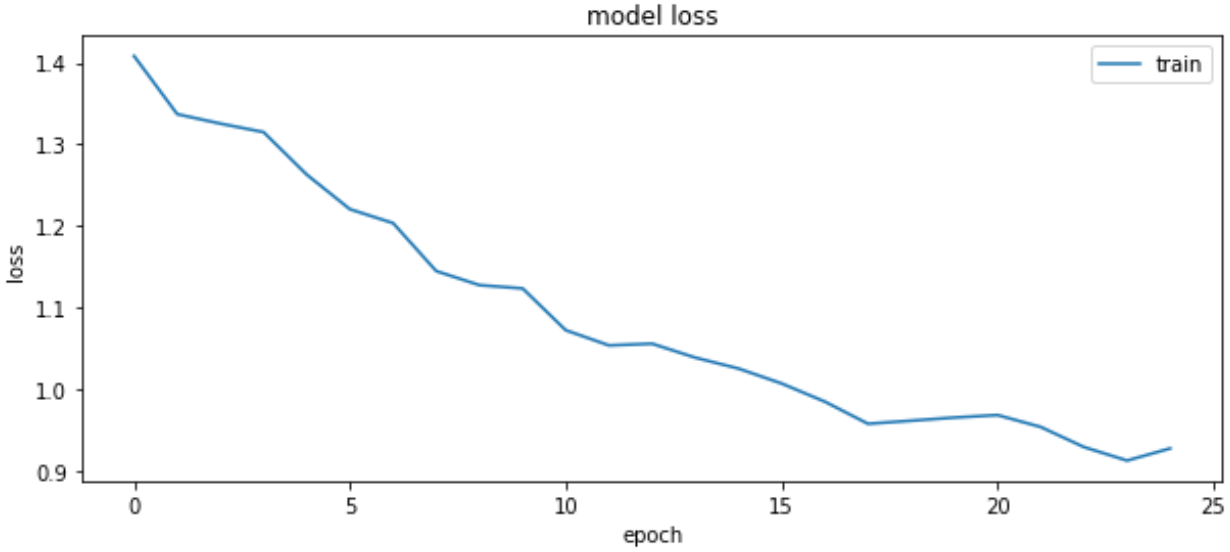
طبق شکل بالا قالب کلی شبکه های ما به شکل بالا است که در هر مدل خاص اگر تفاوتی بود پایین تر ذکر خواهد شد.

همانطور که میبینیم خروجی ما ۵ نورون دارد که نشان دهنده ۵ کلاس متفاوت احساسات است.

مدل ۱:

در مدل اول از LTMS استفاده کردیم و با استخراج کننده mfcc و از optimizer ادام استفاده شده که درستی و خطای ما به شکل زیر است:



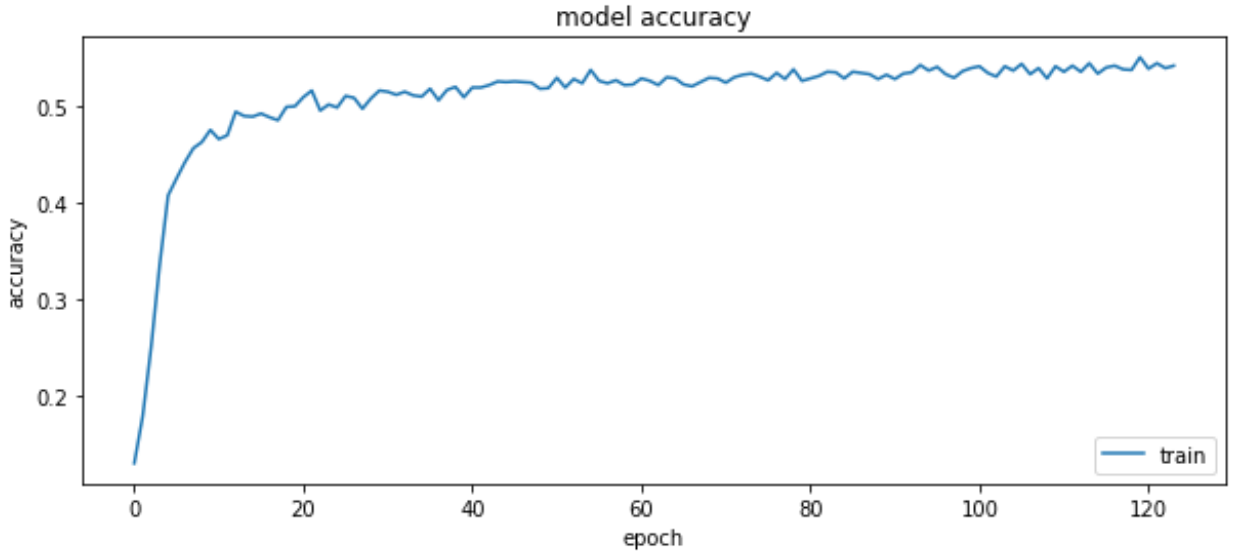


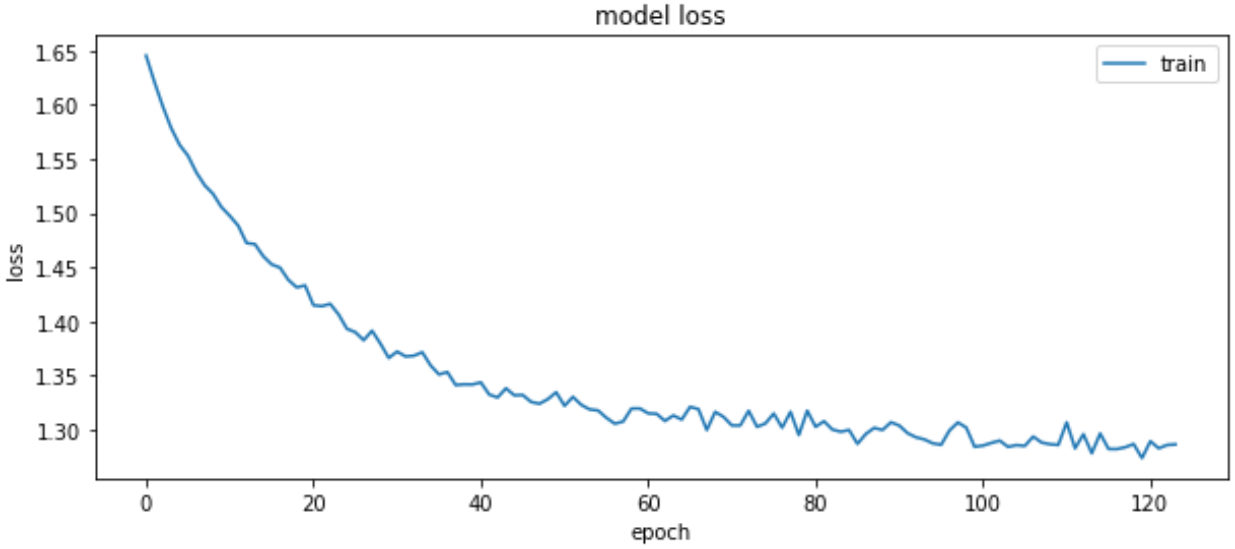
همانطور که میبینیم درستی روی trainتا حدود ۰,٦٥ رفت که درستی خوبی میباشد و روی validationهم تا حدود ۰/۶ و میبینیم که در مرحله ۲۵ ام شبکه ما برایش توقف زودرس اتفاق افتاده

پس از یادگیری این شبکه رو داده های train شبکه را روی داده های test اجرا کردیم و حدودا درستی ۰/۵۶ داشت که در کل خوب است.

مدل ۲:

در مدل دوم جزییات شبکه دقیقا مثل بالاست با این تفاوت که ایندفعه از optimizer sgd استفاده شد با نرخ یادگیری ۰/۰۰۰۲ و شتاب دهنده ۰/۹

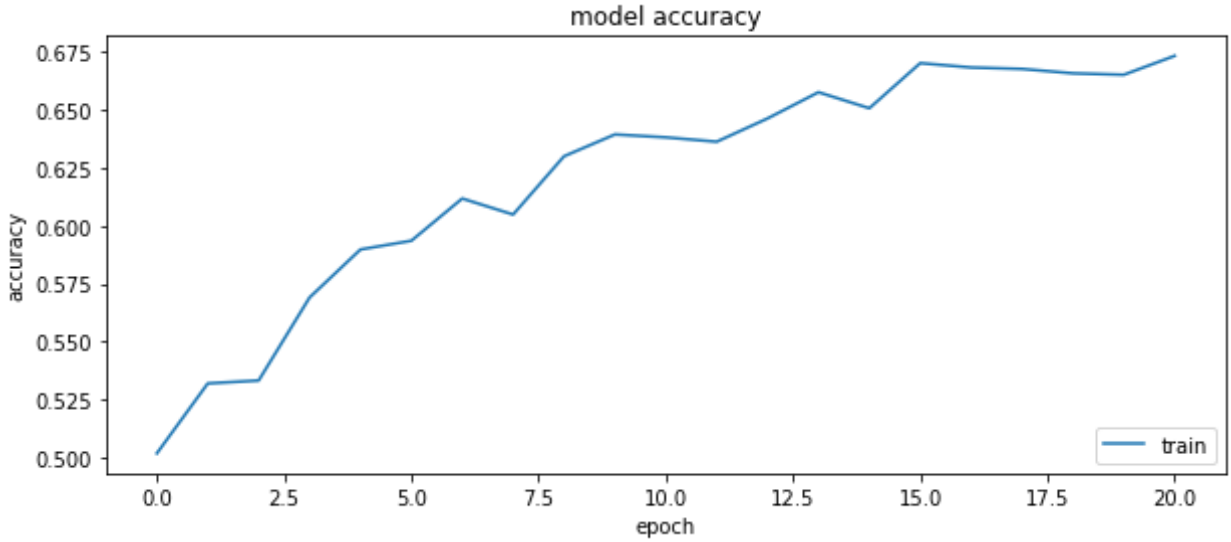


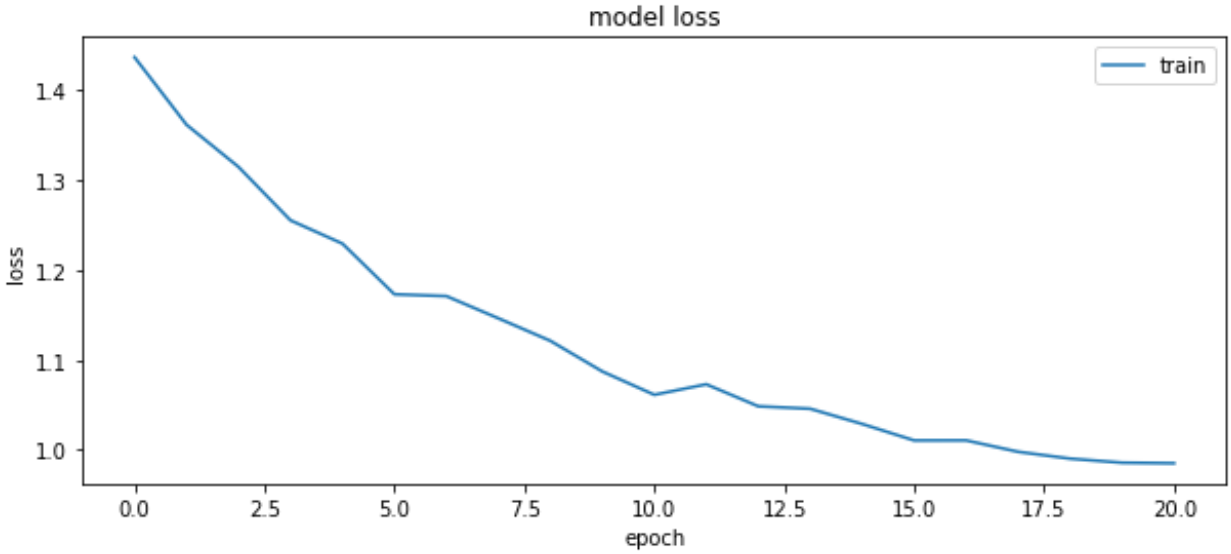


میبینیم طبق تصاویر که روی داده های train به درستی ۰/۵ رسیدیم و روی validationهم به درستی حدود ۰/۵۳ رسید و سپس شبکه را روی داده های تست اجرا کردیم که درستی ۰/۵ داشته که نسبت به مدل اول ضعیف تر بوده ولی درکل همچنان خوب است.

مدل ۳:

مدل سوم هم تفاوتش با مدل های قبلی باز در optimizerاست که از RMSprop استفاده شده که در درستی تغیرات جزئی بوجود میاید

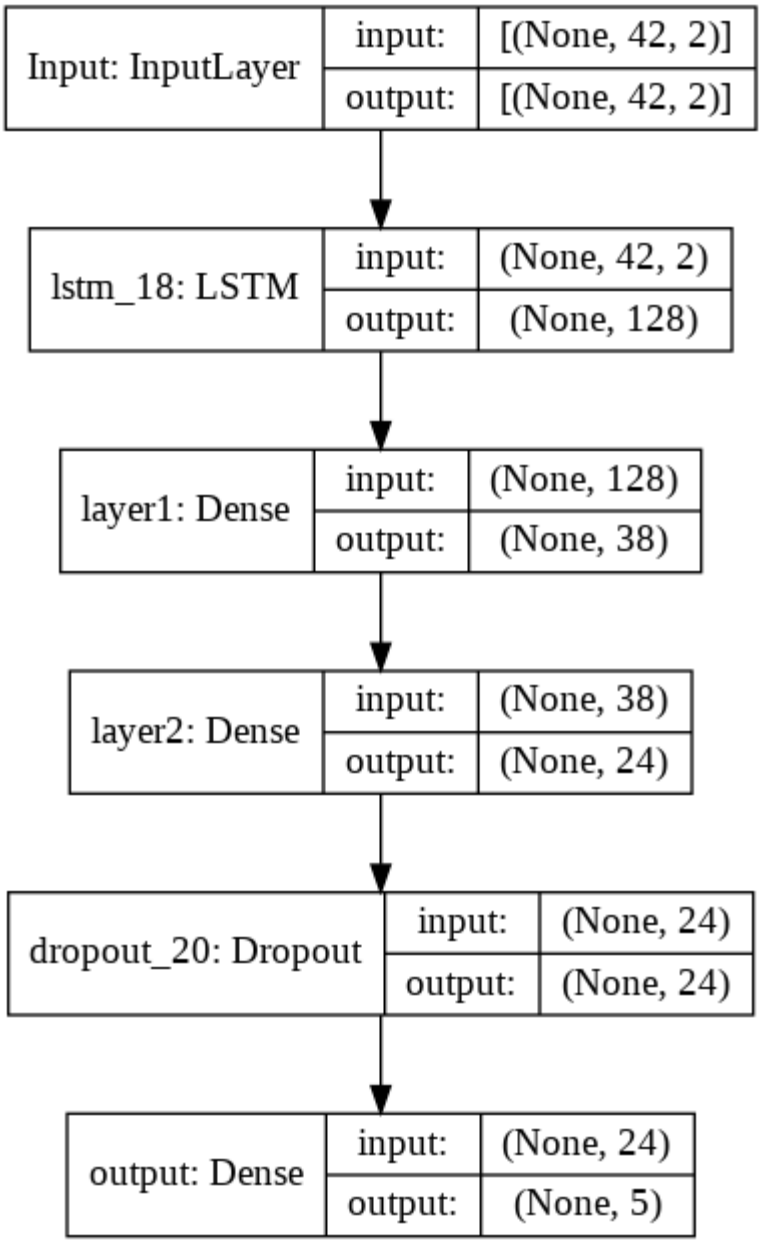




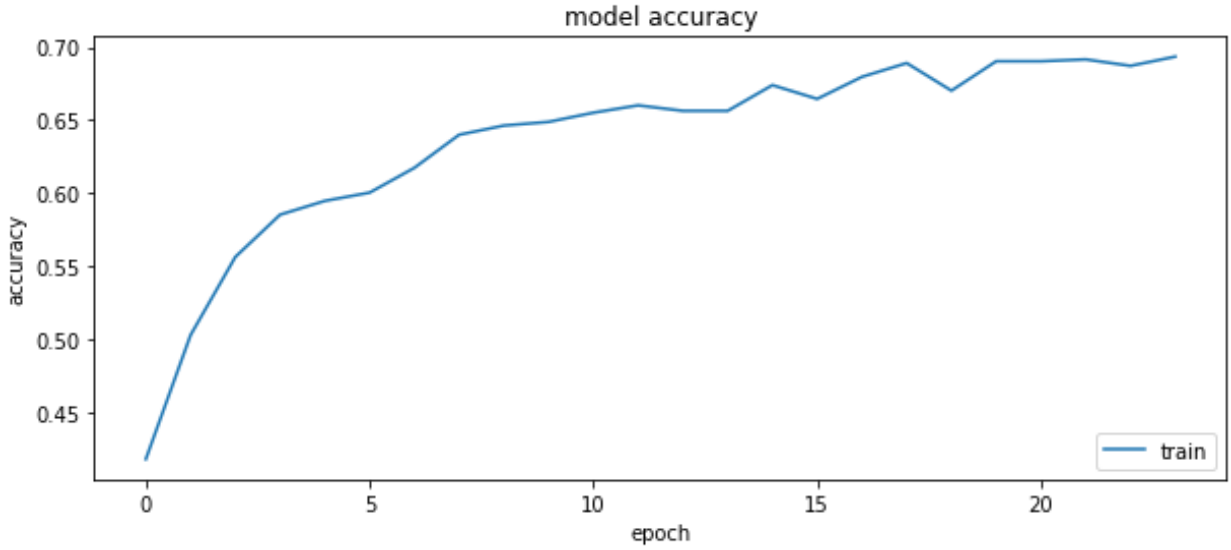
طبق نمودار ها به درستی ۰/۶۷ روی train و حدودا ۰/۶۳ روی validation رسیدیم و روی داده های تست درستی ۰/۵۴ داشته است که تقریبا با مدل ۱ معادل است ولی روی تست مقداری ضعیف تر بوده.

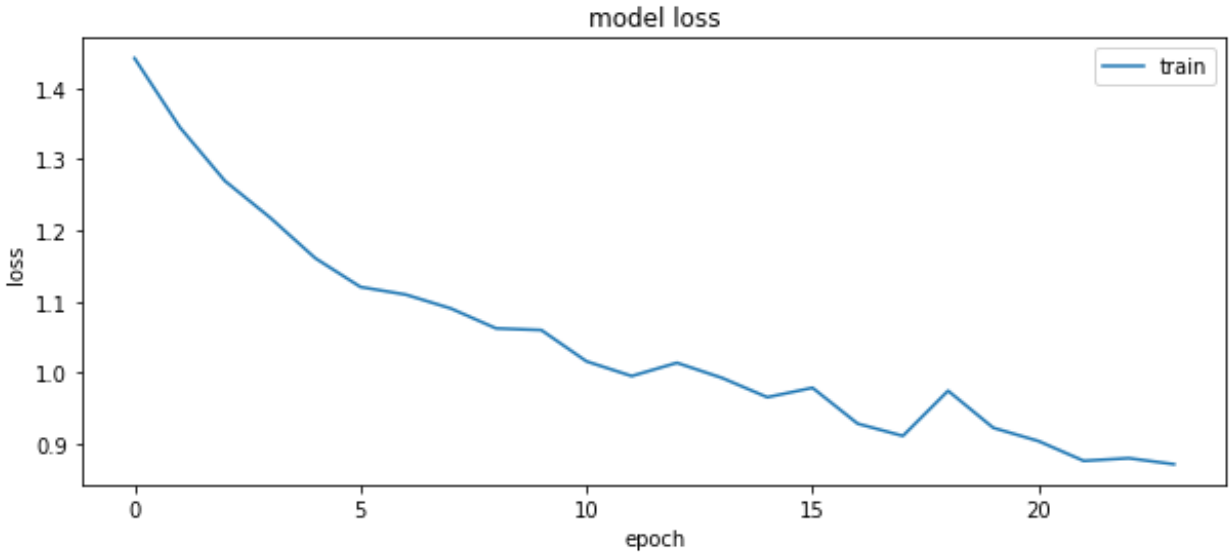
مدل ۴:

اما در این مدل نسبت به مدل های قبلی تفاوت ها بنیادی تر میشوند به این صورت که ما ایندفعه از صدا هایمان ویژگی های mfcc و هم cens را استخراج کرده و به شبکه میدهیم یعنی ایندفعه بجای دادن یک بردار ۴۲ تایی به شبکه یک ماتریس ۲ * ۴۲ به او میدهیم پس شبکه ما به شکل زیر است



در این مدل باز از optimizer adam استفاده کردیم

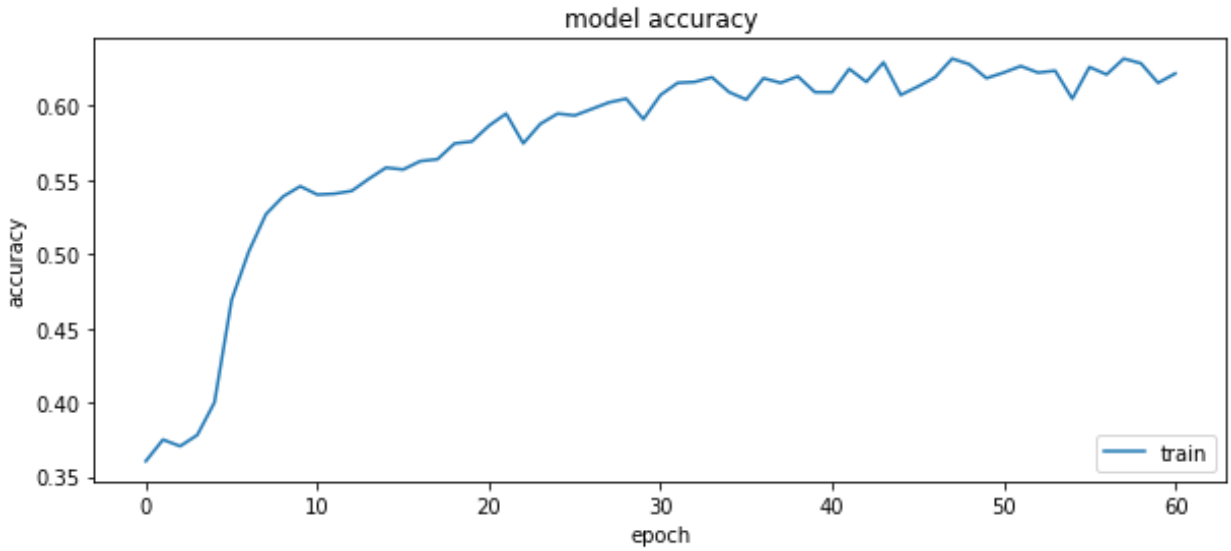


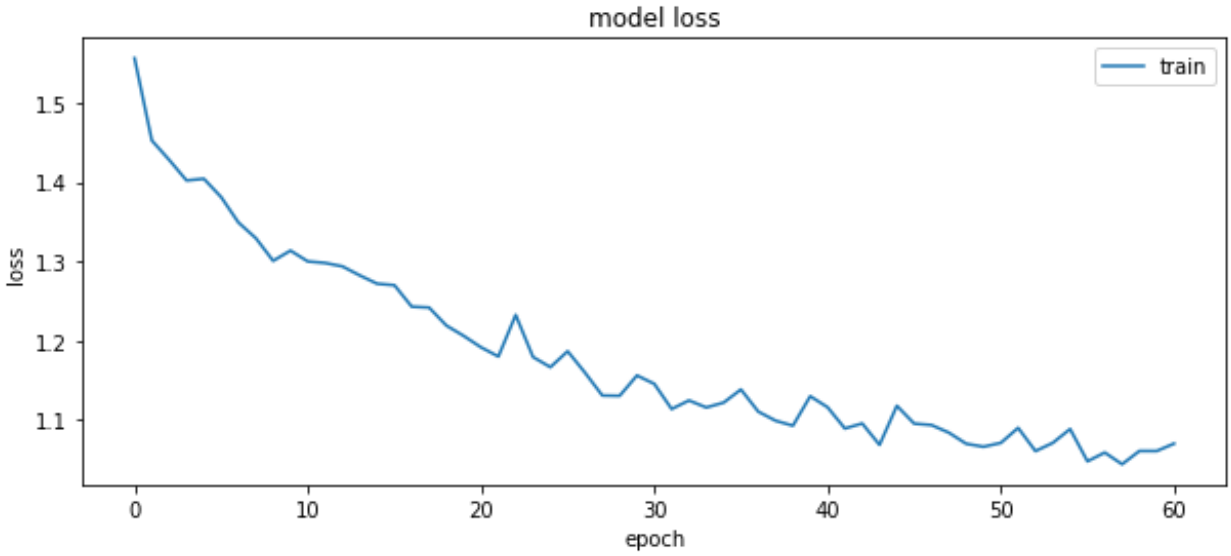


به درستی ۰/۷ روی train و ۰/۶۳ روی validation و سپس روی داده های تست به ۰/۵۶ رسیدیم که ایندفعه میبینیم رو داده های trainعملکرد بهتری بوده و باز روی داده های تست تفاوت چندانی حاصل نشده و تقریبا همان نتایج بدست آمده است.

مدل ۵:

این مدل مثل مدل قبلی (مدل ۴) است با این تفاوت که با optimizer sgd پیاده سازی شده است

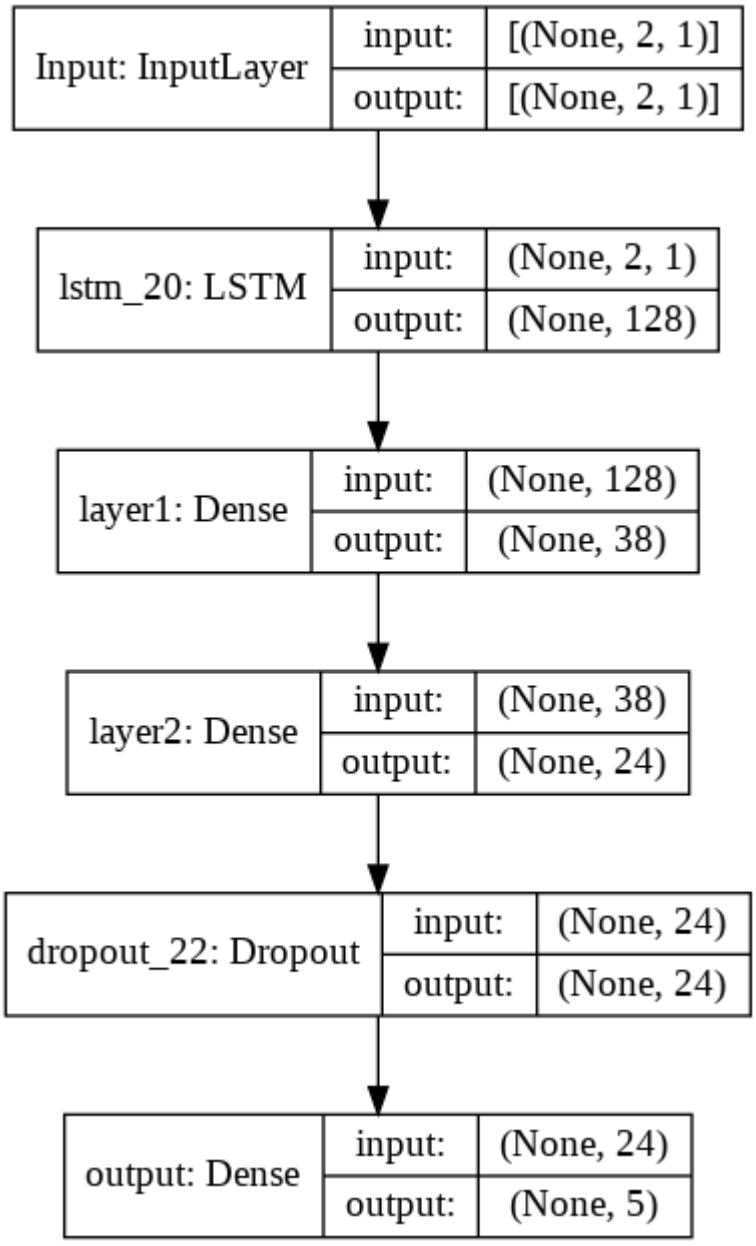




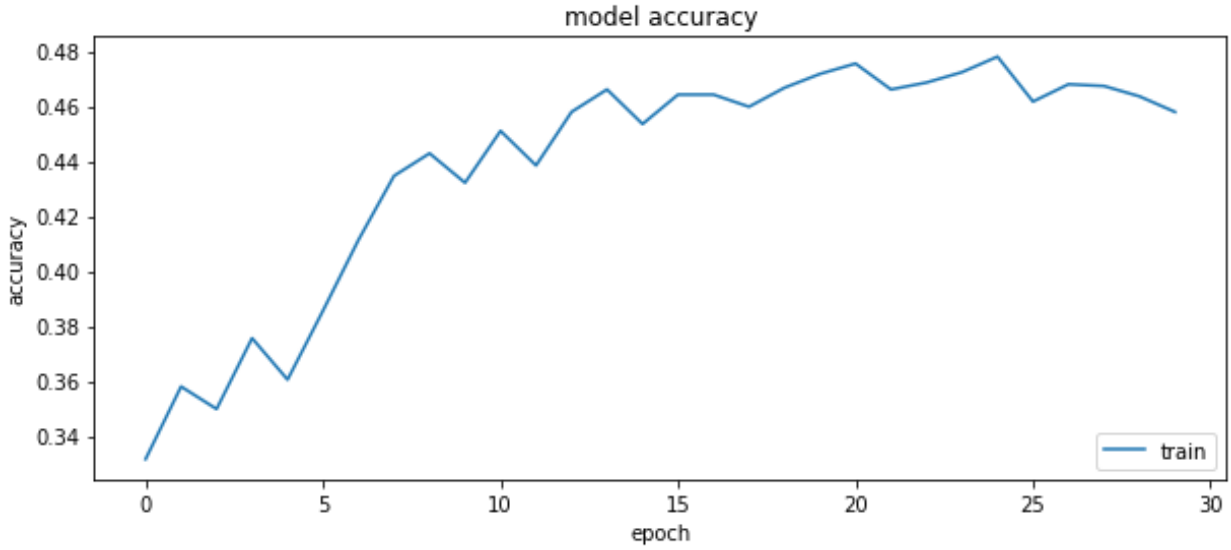
طبق تصویر بالا خطای ۱/۰۴ داشتیم و به درستی حدودا ۰/۶۱ روی train و حدودا ۰/۵۸ روی validation و در آخر روی داده های تست به درستی ۰/۵ رسیدیم که میبینیم نسبت به مدل قبلی عملکرد ضعیف تری ثبت کرده است

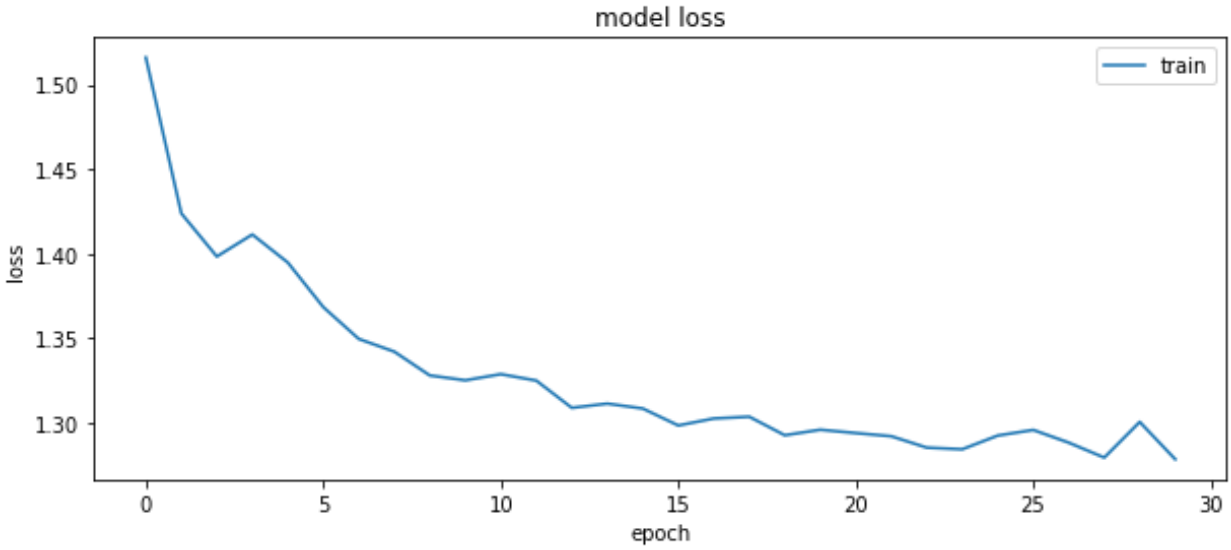
مدل ۶ :

این دفعه هم یک تغییر بنیادی نسبت به قبل داریم یعنی ایندفعه از استخراج کننده های mfcc و cens استفاده نمیکنیم بلکه از poly extraction استفاده میکنیم که به ما از هر صدا ۲ ویژگی میدهد که این یعنی باید به شبکه مان یک بردار ۲ تایی بدهیم طبق شکل زیر



و باز هم از optimizer adam که عملکرد بهتری داشته استفاده میکنیم



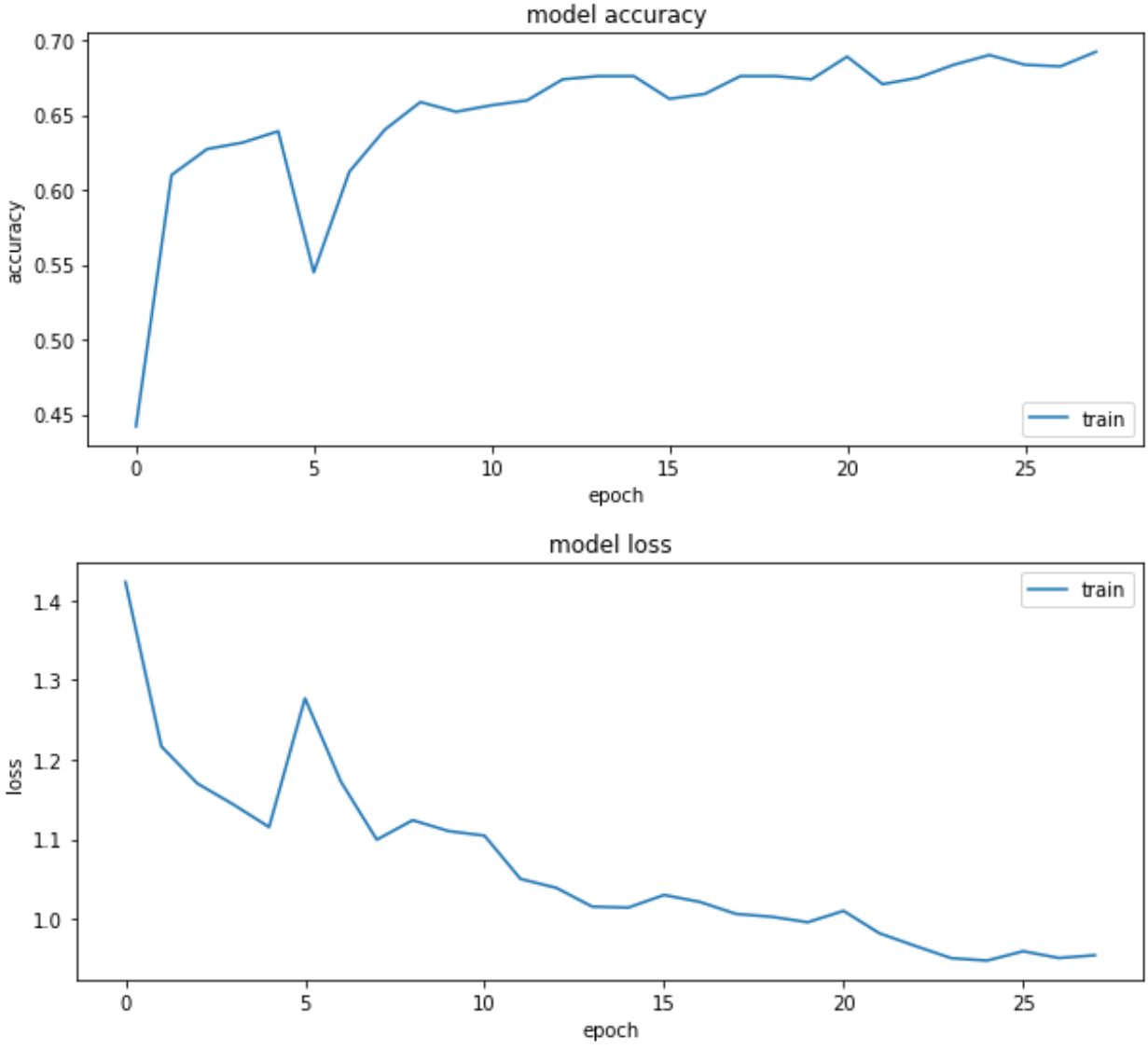


طبق تصاویر به خطای ۱/۳ و درستی ۰/۴۸ روی train و ۰/۴۶ روی validation و سپس ۰/۴۲ روی داده های تست داشتیم که بین همه مدل های تا به اینجا عملکرد ضعیف تری داشته و نشان میدهد عملکرد mfcc خیلی بهتر بوده.

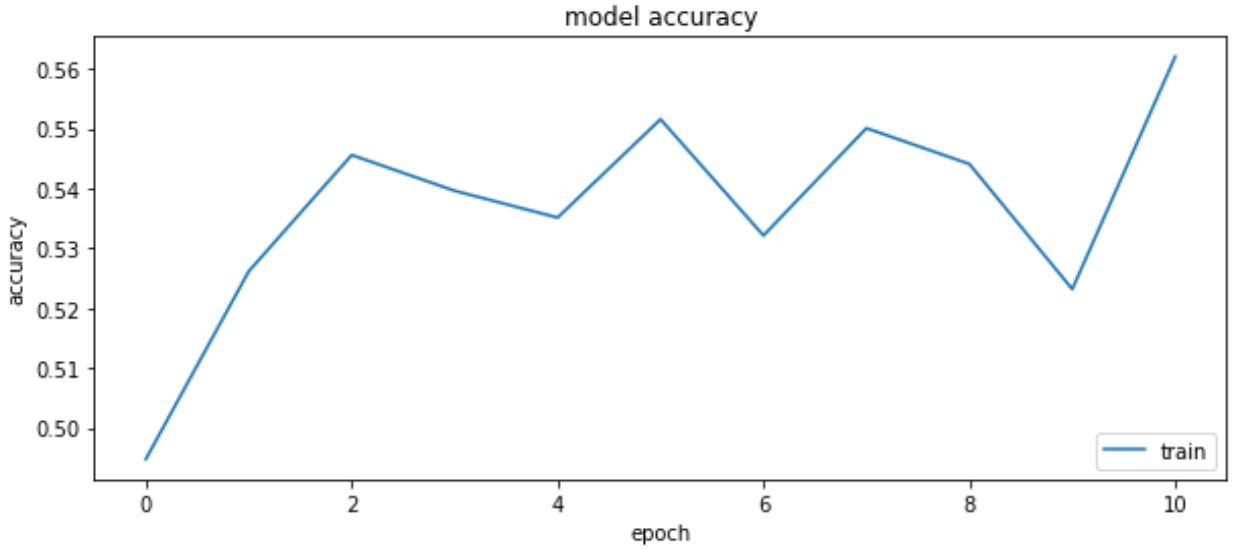
مدل ۷:

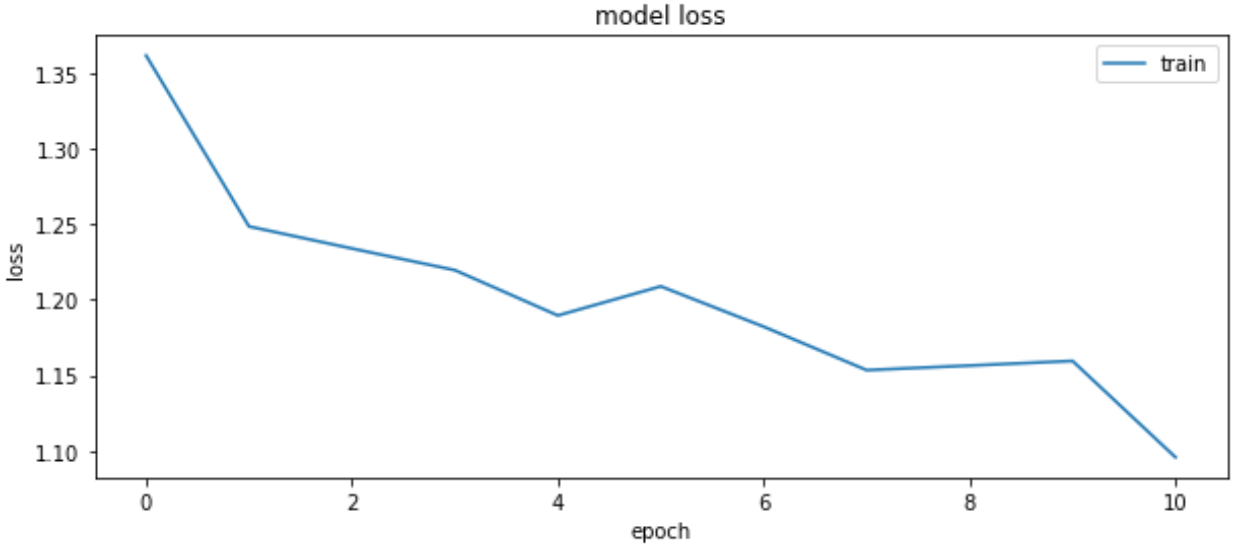
اما در این مدل ما آمدیم جنسیت صدا ها را جدا کردیم تا ببنیم تاثیر جنسیت میتواند مفید باشد یه برعکس

آمدیم همه ی داده ها را هم داده های train هم داده های تست با توجه به اسم فایل که جنسیت توش لحاظ شده جدا کردیم و سپس مدلمان را روی داده های مرد train آموزش داده و سپس روی داده های مرد تست اجرا کردیم و سپس دوباره روی داده های زن trainاموزش داده و روی داده های زن تست اجرا کردیم و نتایج به شکل زیر بوده است



روی داده های مرد خطای ۰/۹۲ حدودا داشتیم و درستی ۰/۷ روی train و ۰/۶۸ روی validation و در نهایت روی داده های مرد تست درستی ۰/۶ حدودا داشتیم

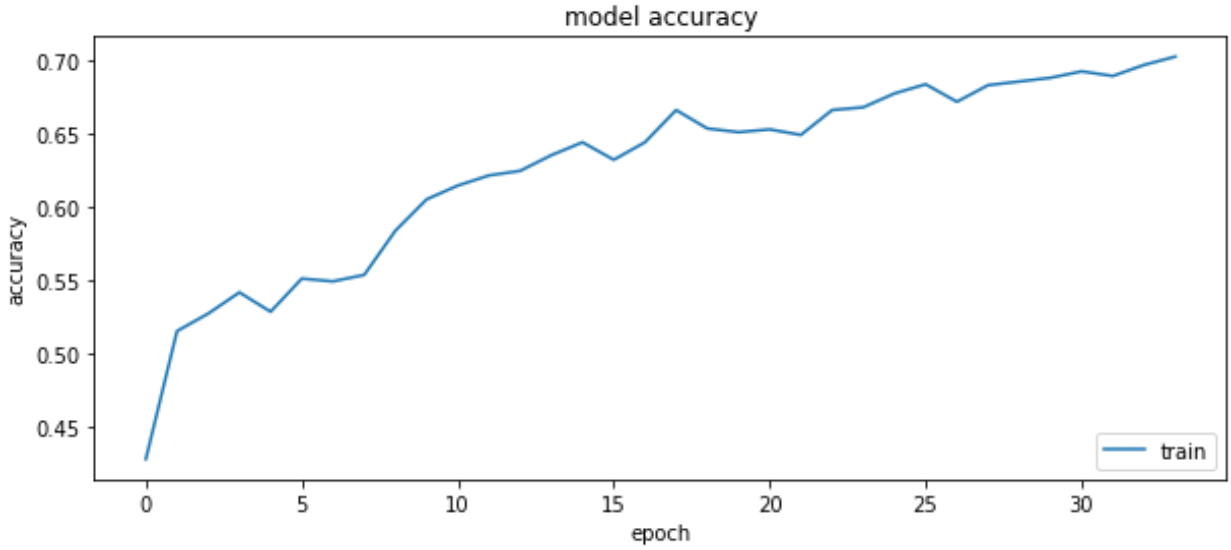


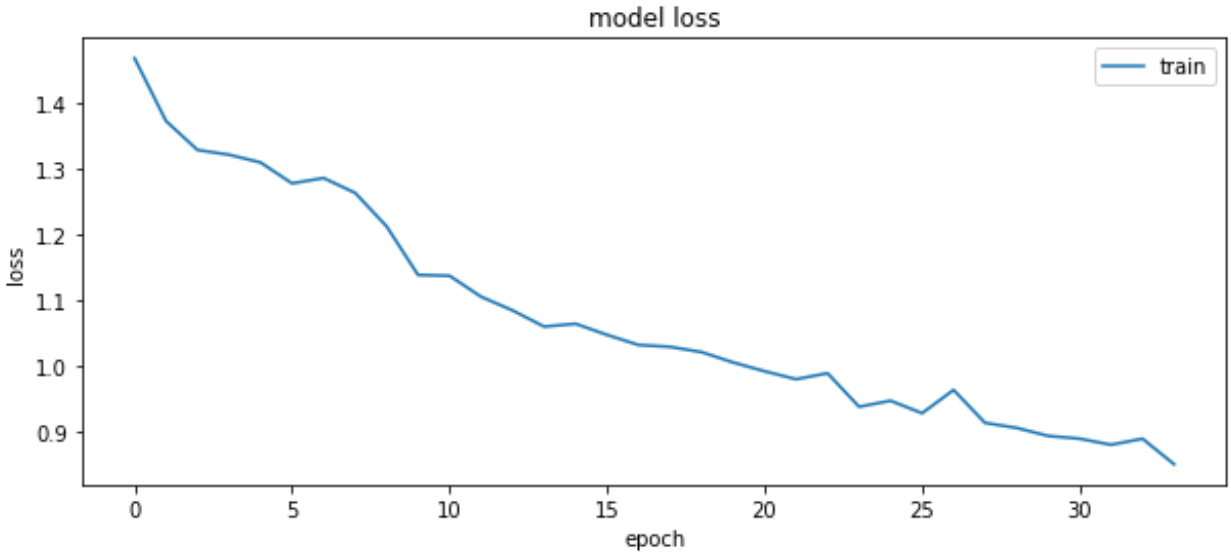


سپس رو داده های زن خطای ۱/۱ و درستی حدودا ۰/۵۶ روی train و ۰/۵ روی validation و در اخر روی داده های زن تست ۰/۵۵ داشتیم که روی داده های مرد تقریبا عملکرد بهتری نسبت به قبل وجود داشت ولی روی داده های زن خیلی بهبود خاصی نداشتیم

مدل ۸ :

در این مدل بجای استفاده از شبکه LSTM از شبکه GRU استفاده کردیم

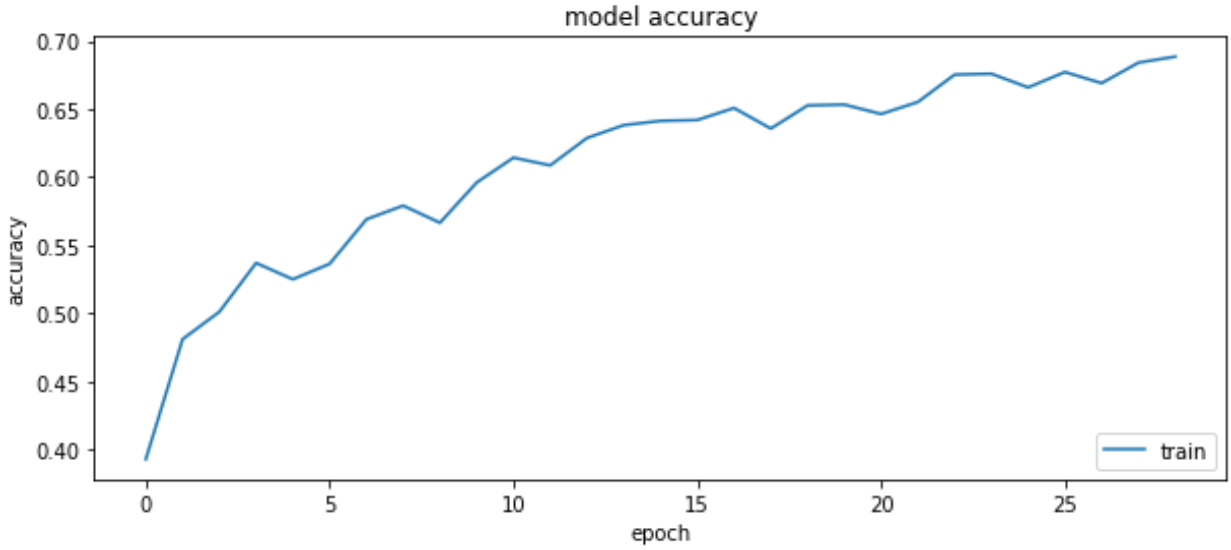


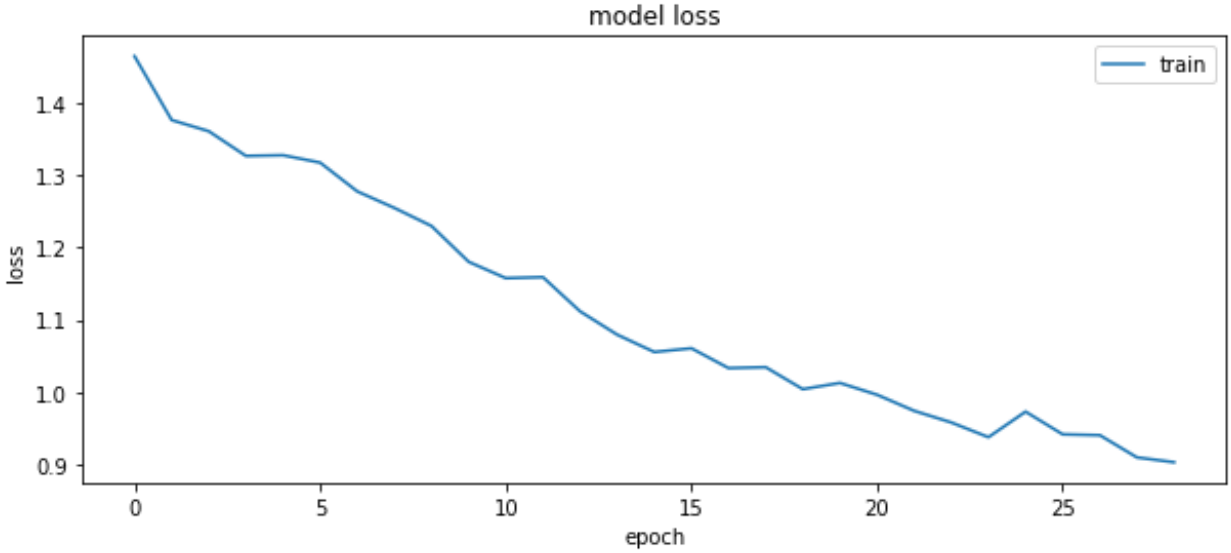


به خطای ۰/۹ رسیدیم و درستی ۰/۷ روی train و ۰/۶۱ روی validation و در نهایت روی تست درستی ۰/۴۹ داشتیم که روی داده های تست از ضعیف ترین عملکرد ها بود

مدل ۹ :

ایندفعه باز از شبکه LSTM استفاده شده با این تفاوت که درلایه denseاخر از تابع sigmoid بجای tanh استفاده کردیم





در این مدل به خطای ۰/۹ رسیدیم و به درستی ۰/۷ روی train و ۰/۶۳ روی validation رسیدیم و درستی ۰/۵۶ روی داده های تست داشتیم.

نتیجه گیری:

با توجه به برسی مدل های مختلف روی داده های تست مدل های ۱ و ۳ و ۴ تقریبا عملکرد های بهتری نسبت به بقیه داشتن یعنی مدل با شبکه LSTM و optimzer های adam و rmsprop در حالی که

optimizer sgd به خوبی دو تای قبلی نبود و استفاده از استخراج کننده های mfcc و cens درحالی که استفاده از poly اصلا خوب نبود.

همینطور جدا کردن داده ها بر حسب جنسیت هم در کل عملکرد خوبی داشت مخصوصا روی جنسیت مرد ها

استفاده از GRU در حل مسئله ما خیلی مناسب نبود.