به نام خدا

دانشگاه شهید بهشتی

دانشکده علوم ریاضی

گزارش تمرین چهار شبکه عصبی

زهرا دهقانی تفتی (۹۶۲۲۲۰۳۷)

دی ۹۹

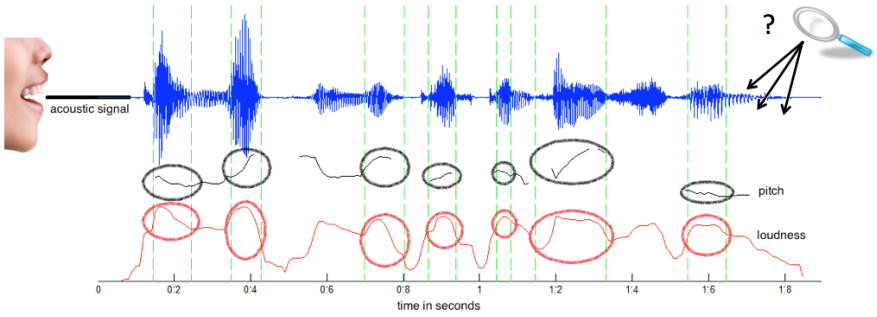
فهرست مطالب

راه حل و ایده‌های کلی

تشخیص احساسات

در این پروژه ما با تعدادی صدای ضبط شده‌ی افراد مختلف (زن‌ها و مردهای مختلف) با احساسات مختلف سر و کار داریم. هدف ما این است که با گرفتن این صداها بتوانیم پیش بینی کنیم که این صدا مربوط به کدام یک از کلاس احساسات می‌باشد.در این پروژه هدف استفاده از شبکه های بازگشتی مانند LSTM است, زیرا این نوع شبکه‌ها برای دریافت و کار با داده‌های توالی مثل صدا که ترتیب توالی مهم است استفاده می‌شوند. در این بخش از پروژه ما صدای زن و مرد را جدا نمی گیریم و همین ممکن است خطای زیادی را در یادگیری مدل ما ایجاد کند زیرا ویژگی صدای زن و مرد فرق دارد و این دقیق نیست که برای مثال صدای زن خوشحال و مرد خوشحال را در یک کلاس قرار دهیم. در این بخش فقط احساسات را جدا می‌کنیم. در این پروژه ۵ کلاس احساسات داریم که هر احساسات را با حرف اول آن احساس نشان می دهیم و این احساسات عبارتند از A مربوط به کلاس خشم، H مربوط به کلاس شادی، S مربوط به کلاس غم، W مربوط به کلاس شگفتی و N مربوط به کلاس بی تفاوت است.

در این پروژه چون با فایل‌های صدا سر و کار داریم از کتابخانه librosa برای خواندن فایل‌ها و استخراج ویژگی از آن ها استفاده می‌کنیم. در این تمرین ورودی مدل ما به صورت یک آرایه‌ای ۲۵ بعدی است که این آرایه ویژگی هر صدا را نشان می‌دهد یعنی برای هر صدا ۲۵ تایم استپ در نظر گرفته‌ایم و در هر تایم استپ ویژگی‌ای را استخراج کردیم که آن را با یک عدد نشان می دهیم. باید با دادن این آرایه ها به مدل به همراه لیبل کلاس آن ها که در یکی از این ۵ کلاس است مدل را آموزش دهیم تا مدل این الگوها را یاد بگیرد. در نهایت بعد از یادگیری مدل، باید آرایه‌ای از ویژگی داده های تست به مدل داده شود و خروجی مدل، برچسب کلاسی است که برای هر صدا پیش بینی کرده است. کار ما به صورت شکل زیر است که با دادن صدا باید بتوانیم کلاس آن را تشخیص دهیم.



در ابتدا با استفاده از کتابخانه‌ی os نام فایل‌های مربوط به کلاس آموزشی این تمرین را در یک لیست به نام file_name ذخیره می‌کنیم. همانطور که می‌بینیم نام فایل ها ترکیبی از اعداد و حروف هستند. هر فایل متشکل از عدد ٤ رقمی است که در ابتدای نام فایل آمده که شماره فایل را نشان می‌دهد و برای هر فایل یکتا است و بعد از عدد چهار رقمی ۲ حرف دارد. حرف اول نشان دهنده‌ی جنسیت تولید کننده ی صدا است و حرف دوم احساس آن صدا را نشان می‌دهد. در واقع متا دیتای ما در

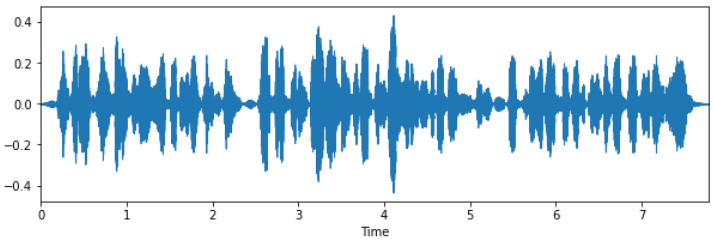
نام فایل است و ما باید آن ها را از نام فایل استخراج کنیم. برای مثال `1345FN.wav` به این معنی است که صدا مربوط به فایل ۱۳۴۵ است، تولید کننده صدا زن بوده و احساس این صدا از نوع بی تفاوت است.

بعد از اینکار برای آشنایی با داده‌های آموزشی از هر نوع احساس فارغ از اینکه صدا مربوط به زن یا مرد است، یک صدا را به نمایش می گذاریم که برای اینکار باید از کتابخانه IPython استفاده کنیم و برای آن صدا نمودار waveform مربوط به آن را رسم می‌کنیم که برای اینکار نیز به کتابخانه librosa نیاز داریم. برای مثال اول نمودار spectrogram را نیز رسم می‌کنیم.

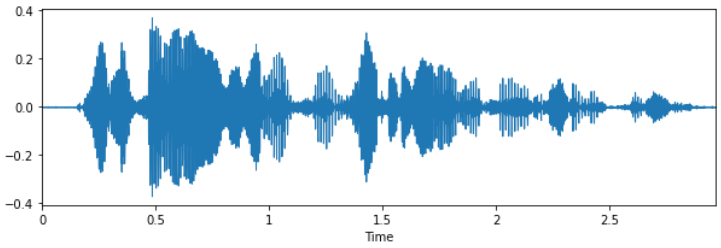
نمودار waveform یا همان نمودار شکل موج، چگونگی جا به جایی مولکول های هوا با گذشت زمان را نشان می دهد. دامنه‌ی موج، قدرت اثر آن موج است و هرچه دامنه بیشتر باشد مولکول‌های هوا بیشتر جابجا می شوند.

نمودار spectrogram یک راه دیگر برای نمایش صدا است که سطح انرژی مختلف صدا را با رنگ های مختلف و سایه ها نشان می دهد.

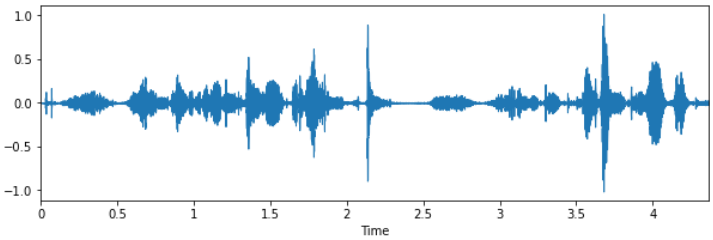
برای آشنایی با شکل موج صدا از هر نوع کلاس تعدادی از آن ها را چاپ کنیم. برای مثال شکل موج زیر مربوط به کلاس خشم است که در نوت بوک نمودار spectrogram آن نیز رسم شده است.



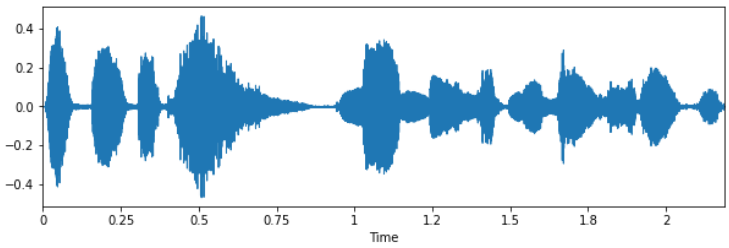
شکل زیر مثال مربوط به کلاس بی تفاوت است.



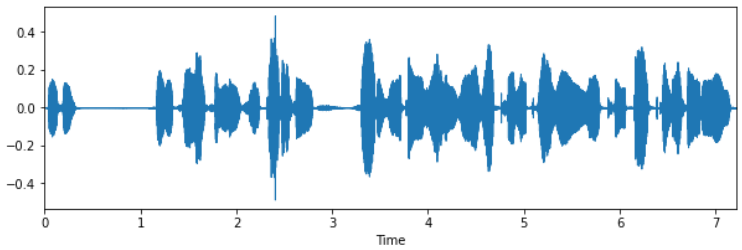
شکل زیر مثال مربوط به کلاس غم است.



شکل زیر مربوط به کلاس شادی است.



شکل زیر مربوط به کلاس شگفتی است.



همانطور که در این شکل موج ها می بینیم شکل موج کلاس شادی شبیه شکل موج کلاس‌های بی تفاوت و غمگین است. در داده‌های آموزشی اولیه که داریم تعداد داده‌ها از هر نوع به شکل زیر است:

```
number of N:  690
number of W:  149
number of S:  302
number of A:  723
number of H:  130
```

همانطور که می بینیم تعداد داده‌های کلاس بی تفاوت و خشم از بقیه‌ی داده ها بیشتر است و انتظار می رود مدل این الگوها را بهتر یاد بگیرد و همچنین مدل به سمت این دو کلاس بایاس داشته باشد زیرا تعداد این داده ها بیشتر است و از بقیه کلاس ها تعداد کافی نمونه نداریم بنابراین یادگیری مدل روی این دو داده بیشتر است. این یکی از دلایلی است که مدلی که در انتها می سازیم داده های کلاس شاد را درست تشخیص نمی دهد و آن ها را به عنوان کلاس خشم یا شادی تشخیص می دهد که در انتهای گزارش بیشتر به آن می پردازیم.

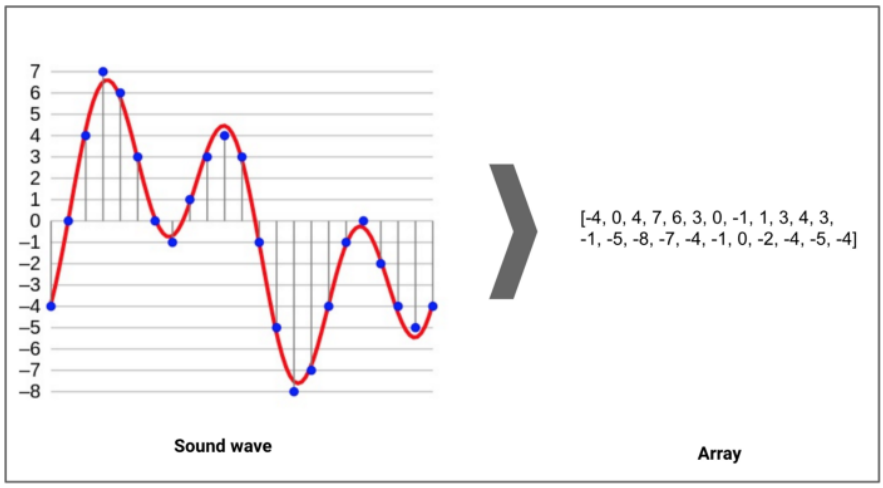
بعد از اینکار باید اطلاعاتی را از نام فایل استخراج کنیم که در اینجا برای برچسب هر فایل در داده‌ی آموزشی، حرف ٦ ام فایل را که نشان دهنده ی احساس آن صدا است را در فایل feeling_label ذخیره می کنیم که در واقع خروجی ما است.

چون این خروجی ها به شکل حرف هستند واز نوع دسته بندی می باشند برای اینکه به شکل قابل فهم برای شبکه عصبی در بیاید باید از label encoding استفاده کنیم که به هر حرف عدد نسبت می دهد. چون ما در اینجا ٥ نوع احساس داریم اعداد ٠ تا ٤ را به احساسات نسبت می دهد. حال برای اینکه در یادگیری مدل اشکال به وجود نیاید و این عددها را وزن دار نبیند

(برای مثال اگر به شادی عدد ٤ را نسبت می دهیم این برداشت برای مدل اینجاد نشود که عدد ٤ وزن بیشتری نسبت به ۰ دارد.)

از دستور to categorical استفاده می‌کنیم تا به تعداد دسته ها که در اینجا ۵ تا است ستون درست کند و داده متعلق به هر احساسی که بود در آن ستون یک بگذارد.

بعد از این کار باید ویژگی‌های مربوط به هر صدا را استخراج کنیم . ما اینکار را با استفاده از متد mfcc که در کتابخانه librosa قرار دارد و برای استخراج ویژگی از موج های صوتی استفاده می‌شود ، می‌کنیم و برای راحتی کار با آن یک فانکشن به نام extract_mfcc می سازیم که ورودی آن نام فایلی است که در کلاس مربوط به داده های آموزشی است و خروجی آن ویژگی ها است. برای بدست آورن ویژگی از یک صدا، اول موج آن را به تعدادی تایم استپ تقسیم می‌کند که ما در اینجا به طور دلخواه ۲۵ تایم استپ در نظر گرفته ایم و در هر کدام از این تایم استپ ها ویژگی‌ای بدست می آید در نهایت ویژگی ما ۲۵ بعد دارد. این روند مانند شکل زیر است اما در شکل زیر ۲۳ تایم استپ داریم.



بعد از استخراج ویژگی برای ۱۹۹٤ صدایی که برای آموزش داریم روی آن با استفاده از reshape تغییراتی انجام می‌دهیم تا این آرایه از ویژگی‌ها به صورت یک آرایه ی ۲۵ بعدی در بیاید. تا اینجای کار ورودی داده‌های آموزشی ما در آرایه‌ی sound_features قرار دارد که ۱۹۹٤ عنصر دارد و هر عنصر ۲۵ بعد دارد و خروجی آن‌ها در آرایه‌ی feeling_label قرار دارد که آن هم ۱۹۹٤ عنصر دارد و هر عنصر ۵ بعدی است.

حالا نوبت نرمال سازی داده‌ها است. ما داده‌ها رو طوری نرمال می‌کنیم که میانگین آن‌ها برابر با ۰ باشد و واریانس آن‌ها برابر با ۱ باشد در واقع آن‌ها را به شکل توزیع نرمال استاندارد در می آوریم. این کار را هم برای داده‌های آموزشی و هم تست انجام می دهیم.

سپس داده های آموزشی را به دو بخش x_train و x_val یعنی داده های آموزشی و اعتبار سنجی تقسیم می‌کنیم. ۲۰ درصد داده‌های آموزشی اولیه را به عنوان داده‌ی اعتبارسنجی و بقیه را به عنوان داده‌های آموزشی در نظر می‌گیریم.

سپس فایل result.csv که در مسئله به ما داده شده و شامل نام فایل‌های تست و احساسات آن‌ها است را می‌خوانیم تا بتوانیم عملکرد مدل روی داده‌های تست را با کلاس واقعی‌ای که در آن هستند برای مدل‌ها مقایسه کنیم. نام فایل‌های تست را در لیست filename_test ذخیره می‌کنیم و احساسات مربوط به آن‌ها نیز که در ستون فایل csv است را در فایل y_actual_test ذخیره می‌کنیم. لیست filename_test فقط شامل نام فایل تست بدون پسوند wav است. این پسوند را نیز به هر عنصری که در این لیست قرار دارد اضافه می‌کنیم تا طبق همین لیست بتوانیم وِیژگی مربوط به داده‌های تست را از مکانی که صوتشان در آن هستند استخراج کنیم.

تمام کارهایی که برای داده‌های آموزشی انجام شد مثل label encoding، استخراج ویژگی و نرمال سازی را برای داده‌های تست نیز انجام می‌دهیم. ورودی داده‌های تست در فایل test_sound_features و خروجی آن در فایل y_actual_test قرار دارد.

بعد از آماده سازی داده‌ها به شکل قابل فهم برای شبکه عصبی نوبت آموزش مدل و امتحان کردن مدل‌های مختلف است.

ارزیابی نتایج

در این بخش ما به بررسی و تحلیل ٦ مدلی که ساخته‌ایم می‌پردازیم. در تمام مدل‌های ما از lstm استفاده شده است اما در مدل آخر به جای lstm از GRU استفاده شده است. لایه‌ی ورودی در تمام این مدل ها باید به صورتی باشد که قادر به دریافت ویژگی های ما که ۲۵ بعدی هستند باشد در خروجی نیز چون ما ۵ کلاس داریم پس ۵ نورون برای لایه‌ی خروجی قرار می‌دهیم و تابع فعالیت softmax استفاده می کنیم تا مانند تابع احتمال، احتمال بودن خروجی در هر یک از این کلاس ها مشخص شود. در نهایت برای اینکه ببینیم ورودی ما متعلق به چه کلاسی است از تابع argmax روی پیش بینی مدل استفاده می‌شود تا شماره کلاسی که بیشترین احتمال دارد چاپ شود. بعد از اینکار برای اینکه خروجی ما همان برچسب حرفی باشد و احساسات را نشان دهد باید کاری برعکس کاری برای label encoding و to_categorical کردیم را انجام دهیم.

به اینصورت که ابتدا آرایه که از پیش بینی مدل بدست آمده و روی آن argmax زده ایم را با استفاده از flatten به آرایه یک بعدی تبدیل می‌کنیم سپس این شماره ها را با استفاده از دستور inverse.transform به برچسب حرفی آن ها تناظر می‌دهیم.

آزمایش ۱:

در این مدل ساختاری مانند شکل زیر داریم. در این مدل از یک لایه LSTM استفاده شده است. همچنین از تابع هزینه cross entropy و بهینه گر Adam استفاده شده است.

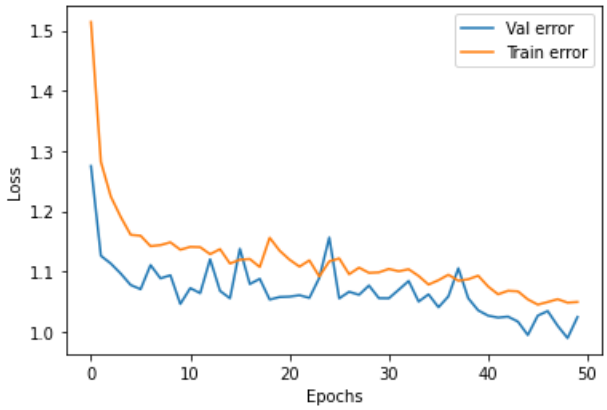
```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input (InputLayer)           [(None, 25, 1)]           0
_________________________________________________________________
lstm (LSTM)                  (None, 128)               66560
_________________________________________________________________
layer1 (Dense)               (None, 64)                8256
_________________________________________________________________
layer2 (Dense)               (None, 32)                2080
_________________________________________________________________
dropout (Dropout)            (None, 32)                0
_________________________________________________________________
layer3 (Dense)               (None, 16)                528
_________________________________________________________________
output (Dense)               (None, 5)                 85
=================================================================
Total params: 77,509
Trainable params: 77,509
Non-trainable params: 0
```

مدل را با داده های آموزشی آموزش می‌دهیم و طی ۵۰ ایپاک و با سایز دسته ٦٤ به نتیجه زیر روی داده های آموزشی و اعتبارسنجی می‌رسیم.

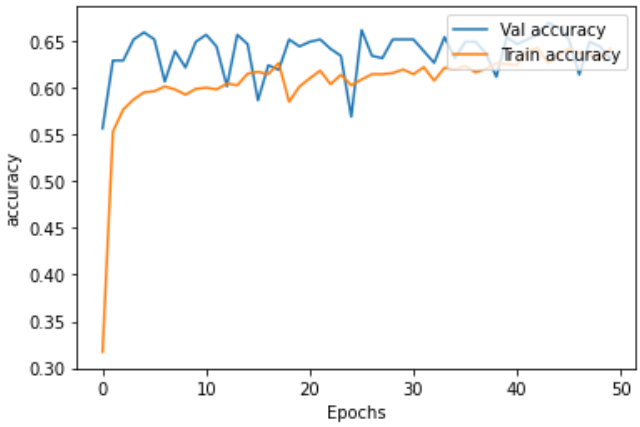
```
Epoch 50/50
```

```
25/25 [===] - 0s 13ms/step - loss: 1.0376 - accuracy: 0.6465 - val_loss: 1.0245 -
val_accuracy: 0.6316
```

طبق شکل زیر در این مدل خطا روی دادههای آموزشی و اعتبارسنجی کاهش پیدا میکند. اما در این روند نوسان زیادی داریم که علت های مختلفی می تواند داشته باشد . یکی از حدس های ممکن این است که سایز بچ را کم گرفته ایم زیرا هرچه سایز دسته بندی بیشتر باشد مدل حساسیت کمتری به نویز دارد. در مدل بعد که با استفاده از همین معماری ساخته شده سایز دسته را افزایش می دهیم تا تاثیر آن را ببینیم.



در شکل زیر نیز مشاهده می‌شود در طی زمان دقت روی داده‌های آموزشی و اعتبارسنجی بیشتر می‌شود زیرا داده ها را به تعداد ۵۰ بار از اول می‌بینند و به خوبی آن‌ها را یاد می‌گیرند. ولی این روند هم با نوسان زیادی همراه است و این نشان می‌دهد مدل ما به اندازه کافی مناسب نیست.



آزمایش۲:

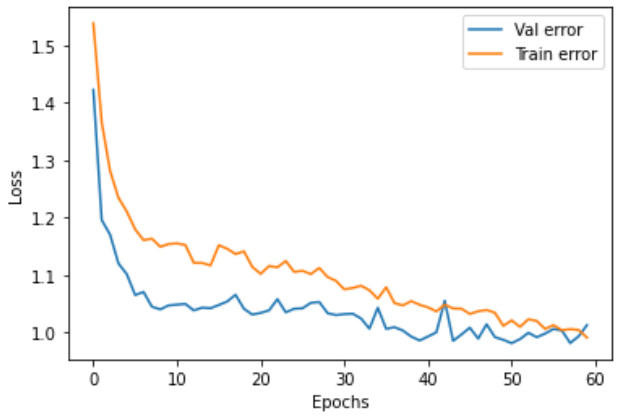
معماری این مدل دقیقا مانند معماری مدل قبل است و تنها تفاوتی که دارد این است که سایز دسته بندی را در این مدل دوبرابر مدل قبل و به تعداد ۱۲۸ تا گرفته ایم. همچنین در اینجا تعداد ایپاک ها را به ۶۰ افزایش دادیم تا مدل تعداد مرتبه بیشتری داده های آموزشی را از اول مشاهده کند. بعد از ۶۰ ایپاک روی این مدل به نتیجه زیر می‌رسیم.

```
Epoch 60/60
```

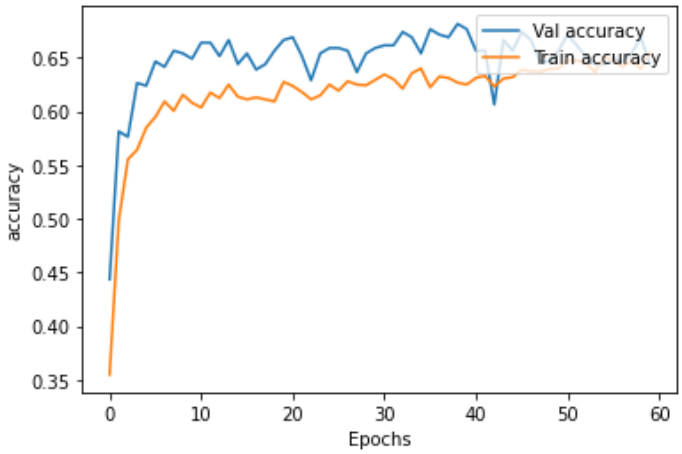
```
13/13 [=====] - 0s 22ms/step - loss: 0.9337 - accuracy: 0.6681 - val_loss: 1.0133 -
val_accuracy: 0.6441
```

همانطور که می‌بینیم در این مدل نسبت به مدل قبل پیشرفت خوبی داشته ایم. در این مدل خطا روی داده‌های آموزشی و اعتبارسنجی نسبت به مدل قبل کمتر شده همچنین دقت روی این دو داده بیشتر شده است که این نتیجه‌ی افزایش تعداد ایپاک است که به یادگیری بهتر و بیشتر مدل کمک کرده است زیرا توانسته داده ها را بیشتر از مدل قبل مشاهده کند.

طبق شکل زیر خطا روی داده‌های آموزشی و اعتبارسنجی با سرعت خوبی کم می‌شود و این روند نوسان کم تری نسبت به مدل قبل دارد و دلیل آن افزایش سایز دسته بندی است. چون هرچقدر سایز دسته ها بیشتر شود حساسیت مدل نسبت به نویز کم‌تر می‌شود زیرا در هر دسته با برآیند داده‌های آن دسته سروکار داریم و این اثر داده های نویزی و outlayer را خنثی می‌کند.



در شکل زیر مشاهده می‌شود دقت روی داده‌های آموزشی و اعتبارسنجی افزایش پیدا می‌کند.و نوسان این نمودار نیز نسبت به مدل قبل کمتر شده است.



تا اینجای کار این مدل بهترین مدل ما است. میخواهیم ببینیم این مدل برای پیش بینی داده های تست چقدر خوب عمل می‌کند.و به نتیجه زیر می‌رسیم.

```
4/4 [==============================] - 0s 4ms/step -
Test loss: 1.0451319217681885
Test accuracy: 0.6475771069526672
```

کار آزمایش کردن را ادامه می‌دهیم.

آزمایش۳:

این مدل ساختار زیر را دارد. تفاوت این مدل با مدل ۲ این است که ما در اینجا تعداد یونیت‌های LSTM را دو برابر کرده‌ایم و تعدادی نورون و لایه نسبت به مدل دو بیشتر داریم.

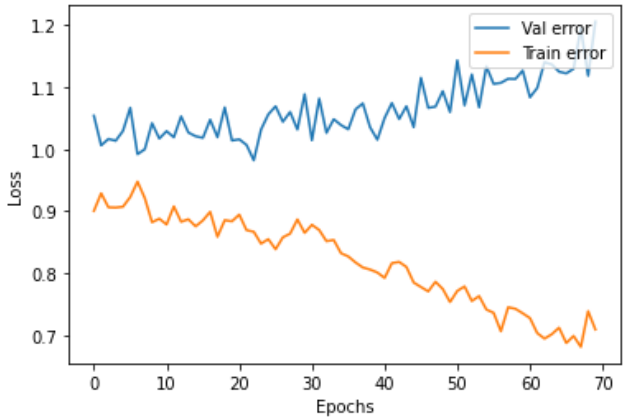
```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input (InputLayer)           [(None, 25, 1)]           0
_________________________________________________________________
lstm_4 (LSTM)                (None, 256)               264192
_________________________________________________________________
layer1 (Dense)               (None, 128)               32896
_________________________________________________________________
layer2 (Dense)               (None, 64)                8256
_________________________________________________________________
dropout_6 (Dropout)          (None, 64)                0
_________________________________________________________________
layer3 (Dense)               (None, 32)                2080
_________________________________________________________________
layer4 (Dense)               (None, 16)                528
_________________________________________________________________
dropout_7 (Dropout)          (None, 16)                0
_________________________________________________________________
output (Dense)               (None, 5)                 85
=================================================================
Total params: 308,037
Trainable params: 308,037
Non-trainable params: 0
```

همچنین در این مدل سایز دسته ها را نسبت به مدل ۲ دو برابر کرده‌ایم و ۲۵٦ در نظر گرفتیم و تعداد ایپاک ها را به ۷۰ رسانده ایم. به نتیجه زیر می‌رسیم.

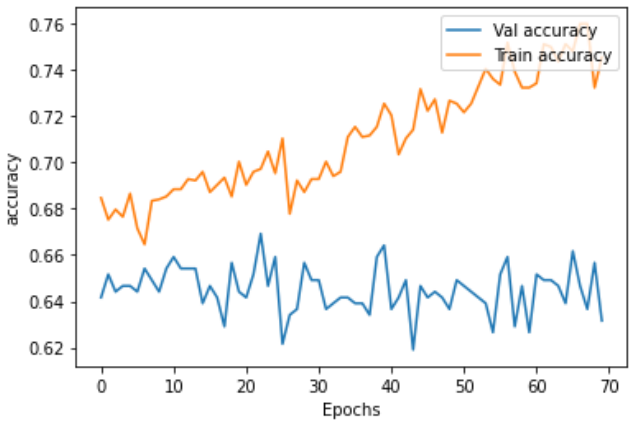
```
Epoch 70/70
7/7 [===] - 0s 13ms/step - loss: 0.7096 - accuracy: 0.7455 - val_loss: 1.2055 -
                                                        val_accuracy: 0.6316
```

همانطور که می‌بینیم دقت روی داده‌های آموزشی نسبت به مدل دو بیشتر شده همچنین خطا روی داده های آموزشی کمتر شده است اما خطا روی داده های اعتبارسنجی بیشتر شده و دقت نیز کم تر شده است. بنظر می‌رسد این مدل قدرت تعمیم کمی دارد و برای داده هایی که در آموزش آن استفاده نشده اند نمی‌تواند خوب عمل کند. حال برای روشن تر شدن قضیه به شکل زیر نگاه می‌کنیم. در شکل زیر مشاهده می شود که مدل overfit می شود زیرا خطا روی داده های آموزشی روند کاهشی دارد ولی خطا روی داده‌های اعتبارسنجی روند افزایشی دارد و این به آن معنی است که مدل در حال حفظ داده‌های آموزشی است. این مدل خوب نیست و باید به دنبال مدل بهتری باشیم. تعداد پارامترهای زیادی که در این مدل داریم و پیچیده شدن این مدل به دلیل اضافه کردن یونیت‌های LSTM و افزایش لایه‌ها و تعداد نورون‌ها از دلایل overfit شدن مدل ما است. در این مدل حدود ٤ برابر مدل ۲ پارامتر داریم که این ظرفیت یادگیری مدل را بیشتر می‌کند و احتمال مدل برای overfit شدن بیشتر می

شود. از دیگر دلایل overfit شدن مدل این است که تعداد اپاک ها را افزایش دادیم و همانطور که می‌دانیم آموزش مدل در طولانی مدت و افزایش دقت روی داده‌های آموزشی منجر به حفظ داده‌های آموزشی و در نهایت overfit شدن مدل می‌شود.



در شکل زیر روند دقت مدل روی داده‌های آموزشی و اعتبارسنجی را نشان می‌دهد. همانطور که می‌بینیم دقت روی داده‌های آموزشی در حال افزایش است و در واقع در حال حفظ کردن داده‌ها است نه یادگیری الگوی آن‌ها در حالی که دقت روی داده‌های اعتبار سنجی روند افزایشی ندارد و در هر دو نمودار نوسان خیلی زیادی داریم.



مدل ۳ مدل خوبی نیست زیرا پیچیدگی آن خیلی بیشتر از پیچیدگی مسئله است و ظرفیت یادگیری خیلی زیادی دارد به طوری که علاوه بر حفظ کردن داده‌ها نویز آن‌ها را نیز حفظ می‌کند و این دلیل نوسان زیاد در نمودارها است. در مدل‌های بعدی باید پیچیدگی مدل نسبت به مدل ۳ کمتر باشد.

با این که این مدل خوب نیست اما برای آزمایش می‌خواهیم روی داده‌های آموزشی که در آموزش مدل استفاده شده است و مدل دقت و خطای کمتری روی این داده‌ها نسبت به داده‌های مشاهده نشده دارد، پیش بینی انجام دهیم.

مقدار واقعی برچسب داده‌های آموزشی را داریم، مقدار پیش بینی شده توسط مدل ۳ برای این داده‌ها را نیز بدست می‌آوریم و در یک جدول قرار می‌دهیم. تعدادی از این داده‌ها را چاپ می‌کنیم. برای نمونه دو مورد از مواردی که اشتباه گرفته اند را چاپ می‌کنیم. اولین موردی که در نوت بوک چاپ کردیم مربوط به کلاس شادی بوده اما به عنوان خشم تشخیص داده

است. بنظر می رسد دلیل آن این است که تن صدای شخص خیلی بلند است و به اشتباه عنوان حالت خشم در نظر گرفته شده است و اینکه تعداد داده‌های مربوط به کلاس شادی کم است و تعداد داده‌های مربوط به خشم زیاد است نیز بی تاثیر نیست و باعث بایاس مدل می‌شود.

مورد دومی که صدای آن را چاپ کردیم مربوط به کلاس شادی است اما باز به اشتباه به عنوان کلاس بی تفاوت در نظر گرفته شده. باز هم دلیل این اشتباه می‌تواند این باشد که تعداد داده های مربوط به کلاس بی تفاوت در داده های آموزشی ما خیلی زیاد است و باعث بایاس مدل شده است. همچنین این که ما جنسیت‌ها را در مورد صدا جدا در نظر نگرفته‌ایم هم یکی از دلایل خطای مدل ما است.

آزمایش ٤:

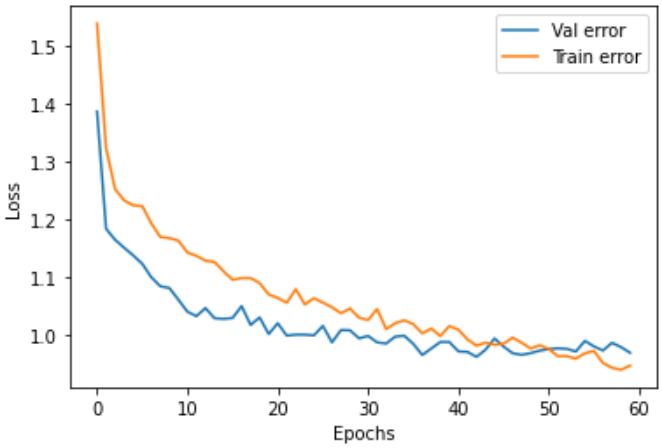
این مدل دقیقا مشابه مدل ۲ است با این تفاوت که در اینجا تعداد یونیت های LSTM را در دو لایه‌ی LSTM قرار دادیم. یعنی تعداد یونیتی برابری با مدل ۲ دارد اما تعداد لایه LSTM دو برابر شده است. بخاطر این کار تعداد پارامترهای ما نسبت به مدل ۲ حدود ۲۰۰۰۰ تا کمتر شده است و طبعا پیچیدگی مدل نیز کم می‌شود.

در این مدل سایز دسته بندی را برابر با ۱۲۸ و تعداد ایپاک‌ها را ٦۰ گرفته‌ایم. طبق این مدل به نتیجه‌ی زیر می‌رسیم.

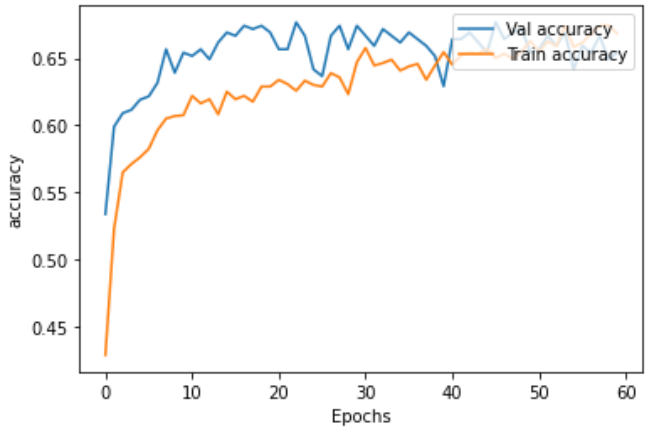
```
Epoch 60/60
13/13 [===] - 0s 10ms/step - loss: 0.9254 - accuracy: 0.6681 - val_loss: 0.9690 -
                                                        val_accuracy: 0.6541
```

همانطور که میبینیم این مدل نسبت به مدل ۲ خطای کمتری هم روی داده های آموزشی هم روی داد های اعتبارسنجی دارد. و دقت نیز روی این داده ها افزایش می یابد.

طبق شکل زیر خطا روی هر دو نمونه داده با سرعت مناسب و نوسان معقولی کم می شود. ولی در ایپاک ۵۰ مدل در آستانه overfit شدن قرار می گیرد. با توجه به اینکه تعداد پارامترهای این مدل نسبت به مدل ۲ کمتر است انتظار می رود قدرت تعمیم آن بیشتر باشد، تنها دلیل می تواند این باشد که اضافه شدن لایه های LSTM احتمال بروز overfit را افزایش می دهند و ظرفیت یادگیری مدل زیاد می شود در نتیجه تمایل مدل به حفظ کردن داده ها نسبت به یادگیری بیشتر می شود. زیرا لایه‌های LSTM محاسبات خیلی زیادی در درون خود دارند و همچنین این لایه ها حافظه دارند که همان cell state آن‌ها است، پس گرچه تعداد پارامترهای این مدل نسبت به مدل ۲ کمتر است اما چون یک لایه‌ی LSTM نسبت به آن بیشتر دارد باعث می‌شود حافظه بیشتری داشته باشد و ظرفیت یادگیری آن بیشتر شود و باعث overfit شدن مدل شود.



شکل زیر مربوط به دقت مدل روی داده‌های آموزشی و داده‌های اعتبارسنجی است.



میخواهیم ببینیم این مدل برای پیش بینی داده های تست چقدر خوب عمل می‌کند.و به نتیجه زیر می‌رسیم.

```
4/4 [==============================] -
Test loss: 1.0536582469940186
Test accuracy: 0.6387665271759033
```

دقت این مدل روی داده های تست کمتر از مدل ۲ است و همچنین خطای آن روی داده های تست بیشتر است نسبت به آن و نشان می دهد این مدل قدرت تعمیم کمتری نسبت به مدل ۲ دارد .بنابراین تا اینجا مدل ۲ بهترین مدل ما است.

آزمایش۵:

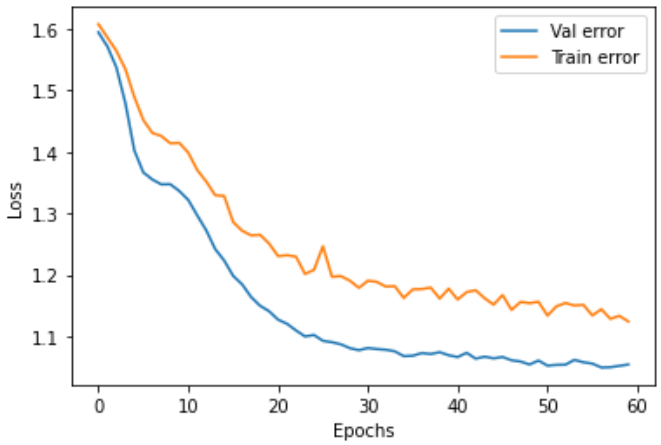
در این قسمت معماری ما مشابه مدل دوم است ولی تعداد لایه ها و تعداد نورون ها را کم تر می کنیم همچنین تعداد یونتی های LSTM را بین تعداد ویژگی های ورودی(۲۵) و تعداد خروجی(۵) می‌گیریم تا ببینیم عملکرد مدل چگونه است. به دلیل این نوع معماری، در این مدل تعداد پارامترها نسبت به بقیه مدل ها خیلی کمتر است. تعداد ایپاک را ٦٠ و سایز دسته بندی را ۱۲۸ در نظر می‌گیریم. و به نتیجه زیر می‌رسیم.

```
Epoch 60/60
```

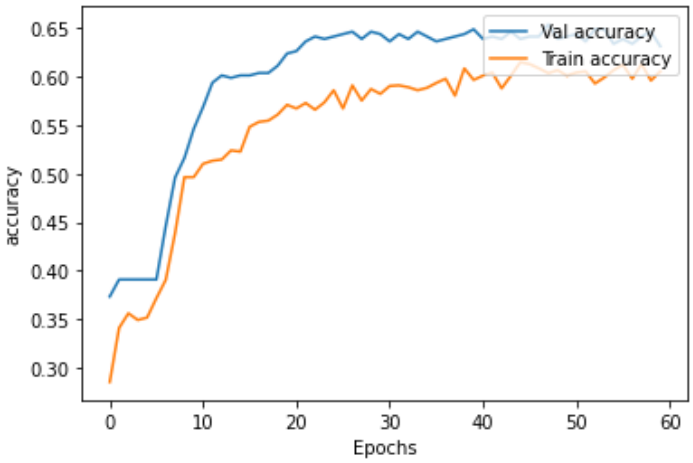
```
13/13 [===] - 0s 6ms/step - loss: 1.1166 - accuracy: 0.6245 - val_loss: 1.0545 -
val_accuracy: 0.6316
```

دقت این مدل نسبت به مدل ۲ کمتر شده همچنین خطا روی داده های اعتبارسنجی و آموزشی نسبت به مدل دو افزایش داشته است و این بخاطر این است که مدل ما نسبت به مدل ۲ خیلی ساده تر است و پیچیدگی آن کمتر است و این دلیل بر این هم هست که مدل ما overfit نمی شود و ظرفیت بالایی برای حفظ کردن ندارد.

شکل زیر خطای مدل را روی داده‌های آموزشی و اعتبارسنجی نشان می‌دهد.



شکل زیر دقت مدل را روی داده‌های آموزشی و اعتبارسنجی نشان می‌دهد.



میخواهیم ببینیم این مدل برای پیش بینی داده های تست چقدر خوب عمل می‌کند.و به نتیجه زیر می‌رسیم.

```
4/4 [==============================]
Test loss: 1.1445835828781128
Test accuracy: 0.5859031081199646
```

نسبت به دقت مدل و خطای مدل روی داده های اعتبارسنجی و آموزشی این مدل تعمیم خوبی روی داده های تست دارد.ولی همچنان مدل ۲ بهترین مدل ما است.

آزمایش۶:

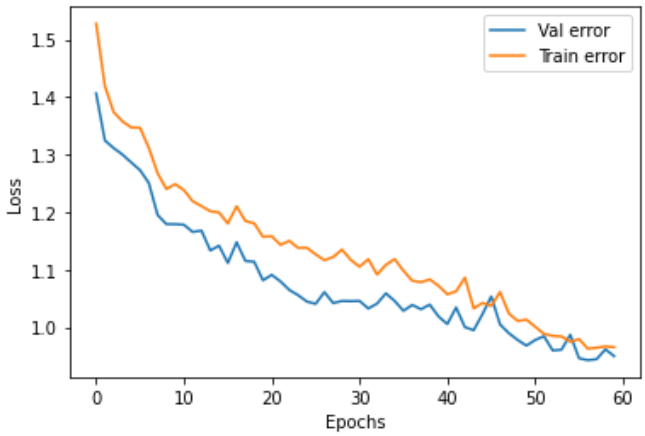
حال چون مدل ۲ بهترین مدل ما بوده می‌خواهیم روی آن تغییراتی انجام دهیم تا اگر امکان دارد بهتر شود. GRU معمولا عملکرد بهتری نسبت به LSTM دارند برای همین در این مدل که دقیقا مشابه مدل ۲ است به جای لایه LSTM لایه GRU قرار می دهیم.

همانطور که می‌بینیم در این مدل نسبت به مدل ۲ تعداد پارامترهای کمتری داریم و این به دلیل استفاده از لایه GRU است زیرا در GRU که مشابه LSTM است محاسبات و پارامترهای کمتری نسبت به LSTM داریم و این به دلیل آن است که در GRU یکسری از گیت‌ها نسبت به LSTM با هم ادغام شده اند.

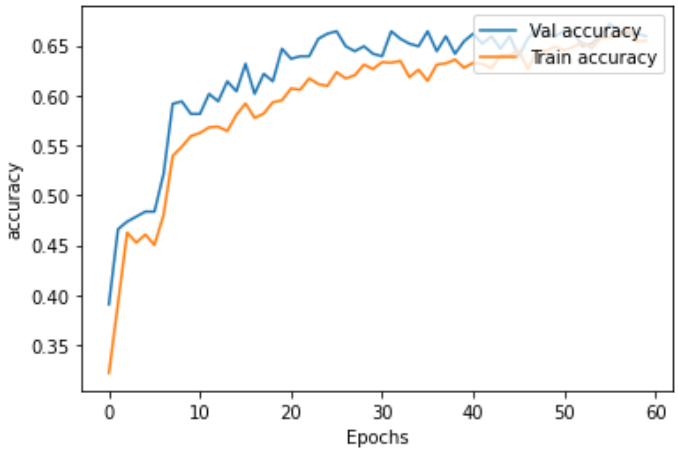
طی ۶۰ ایپاک و سایز دسته بندی ۱۲۸ به نتیجه زیر می‌رسیم.

```
Epoch 60/60
13/13 [===] - 0s 6ms/step - loss: 0.9501 - accuracy: 0.6531 - val_loss: 0.9501 -
                                                        val_accuracy: 0.6591
```

در شکل زیر می بینیم که خطا روند کاهشی را روی داده های اعتبارسنجی و آموزشی دارد اما این سرعت کاهش نسبت به مدل ۲ کمتر است. این مدل نسبت به مدل ۲ خطا روی داده های آموزشی بیشتر شده و دقت نیز روی داده های آموزشی کمی کم تر شده اما خطا روی داده های اعتبارسنجی کم تر شده و دقت برای آن افزایش پیدا کرده است. و همچنین طبق قطعه کد بعدی پیش بینی آن برای داده های تست، خطا روی داده های تست نسبت به مدل ۲ کمتر شده است و دقت نیز کمی کمتر شده. بنظر می رسد این مدل قدرت تعمیم بیشتری نسبت به مدل ۲ دارد زیرا دقت آن روی داده های اعتبارسنجی که در آموزش استفاده نشده اند افزایش یافته همچنین خطا روی داده های تست و اعتبارسنجی نسبت به مدل ۲ کمتر شده است . پس از همین مدل برای پیش بینی استفاده می شود.



شکل زیر دقت را روی داده‌های آموزشی و اعتبارسنجی نشان می‌دهد.



این مدل برای داده‌های تست به صورت زیر عمل می‌کند.

```
4/4 [==============================]
Test loss: 1.025498390197754
Test accuracy: 0.6431717872619629
```

طبق توضیحات بالا از این مدل برای پیش بینی داده‌های تست استفاده می‌کنیم.

پیش بینی داده‌های تست با مدل۶:

در این قسمت مدل ٦ را لود می‌کنیم سپس برای پیش بینی داده‌های تست از آن استفاده می‌کنیم. مقادیر پیش بینی شده و برچسب‌های واقعی را در دو ستون قرار می دهیم و به همراه نام فایل داده‌های تست در یک فایل به نام result_emotion.csv ذخیره می‌کنیم که همراه با گزارش ضمیمه می‌شود.

سپس برای ارزیابی مقادیر پیش بینی شده میخواهیم ببینیم که از هر کلاس چند نمونه در مقادیر پیش بینی شده است که به نتیجه زیر می‌رسیم.

predictedvalues	Id	actualvalues
A	87	87
N	117	117
S	22	22
W	1	1

طبق قطعه کد زیر می بینیم که مدل ما هیچ صدایی را به عنوان خوشحال تشخیص نداده است در صورتی که ما در داده‌های تست ۲۲ صدا مربوط به کلاس خوشحال داریم. این مدل بیشتر صداها را به عنوان کلاس بی تفاوت تشخیص داده، یکی از علت ها این است که که مدل ما که انقدر دقیق نیست و دلیل دیگر این است که جنسیت ها جدا نشده اند. زیرا صداهای زن و مرد در هر احساسات فرق دارد.

سپس در نوت بوک دو نمونه از صداهای تست را که به اشتباه تشخیص داده شدند را چاپ می‌کنیم.

بعد از این کارها برای بررسی مدل و اینکه روی چه کلاس‌هایی بهتر عمل می‌کند از ماتریس confusion استفاده می‌کنیم.

برای نوشتن این ماتریس از کد موجود در کگل استفاده شده است(https://www.kaggle.com/ritzing/speech-emotion-recognition-with-cnn).

برای نوشتن این ماتریس یک فانکشن نوشته شده است.در ابتدا باید نام کلاس‌های موجود در داده هایمان را در classes ذخیره کنیم. سپس نام این کلاس ها بر حسب الفبای مرتب شده است. بعد از اینکار از confusion_matrix که در کتابخانه sklearn.metrics است استفاده می کنیم به این صورت که برچسب واقعی داده های تست و برچسب پیش بینی شده برای آن ها را به آن می‌دهیم و نتیجه را در متغیری به اسم c نگه میداریم. سپس این متغیر را به همراه نام کلاس‌ها به فانکشنی که برای رسم ماتریس نوشته ایم می دهیم. این متغیر c در واقع این اعداد صحیح که در ماتریس نوشته شده است را نشان می‌دهد.

همچنین با استفاده از accuracy_score که آن هم در کتابخانه sklearn است می توان با دادن برچسب واقعی و پیش بینی شده دقت را به دست آورد.

طبق ماتریس رسم شده میبینیم می بینیم که اکثر داده های مربوط به کلاس بی تفاوت درست تشخیص داده شده اند که دلیل آن این است که تعداد داده های مربوط به کلاس بی تفاوت در داده های آموزشی ما زیاد است و حدود یک سوم داده ها از این نوع کلاس اند پس مدل بیشتر آن ها را یاد می گیرد و بایاس بیشتری به سمت این کلاس دارد پس طبیعی است که بعضی از داده ها را به اشتباه به عنوان کلاس بی تفاوت تشخیص دهد..

طبق این ماتریس هیچ داده ای به عنوان داده ی خوشحال تشخیص داده نشده در صورتی که در داده های تستمان ما ۲۲ نمونه مربوط به این کلاس داریم.. ۱۰ تا از داده های کلاس خوشحال به عنوان خشم تشخیص داده شده اند که یکی از دلایل آن می تواند این باشد که الگوی صدایی این دو کلاس تقریبا مشابه یکدیگر است و اینکه تعداد داده‌های مربوط به کلاس خشم نیز در داده‌های آموزشی نسبت به بقیه کلاس ها بیشتر است و مدل به سمت این کلاس بایاس دارد. و ۱۰ تای دیگر از داده هایی که متعلق به کلاس شاد بودند را به عنوان کلاس بی تفات شناسایی کرده است که دلیل آن بالاتر گفته شد.

داده های مربوط به کلاس خشم نیز تا حد زیادی درست تشخیص داده شده اند که دلیل آن این است که در داده های آموزشی تعداد ۷۲۰ تا داده مربوط به این کلاس داشتیم و این کلاس نیز به خوبی یاد گرفته شده و بایاس به سمت این کلاس نیز هست.

پس این مدل روی داده های مربوط به کلاس بی تفاوت و خشم ، مخصوصا بی تفاوت خیلی خوب عمل می کند و دلیل آن این است که تعداد خوبی از این نوع داده ها را در داده های آموزشی دیده ایم و یادگرفته ایم و همچنین به دلیل تعداد زیادشان مدل به سمت این کلاس ها بایاس پیدا می‌کند و بعضی از داده ها را به اشتباه به یکی از این دو کلاس نسبت می دهد.

جمع بندی و نتیجه‌گیری

طبق آزمایش‌های انجام شده به این نتیجه می‌رسیم که لایه‌های LSTM پتانسیل overfit شدن را دارند زیرا محاسبات آن ها پیچیده است و حافظه ای که دارند باعث افزایش ظرفیت یادگیری مدل می‌شود بنابراین باید دقت داشته باشیم که تعداد یونیتی LSTM و تعداد لایه‌های ان را متناسب با پیچیدگی مسئله انتخاب کنیم.

لایه‌های GRU معمولا عملکرد بهتری نسبت به LSTM دارند زیرا به دلیل ادغام شدن یکسری از گیت ها نسبت به LSTM باعث می‌شود محاسبات کم تری داشته باشند و همچنین باعث می‌شود تعداد پارامترهای مدل نیز کمتر شود بنابراین با کم شدن تعداد پارامترها پیچیدگی مدل کم می‌شود و احتمال overfit شدن کاهش می‌یابد.

همچنین طبق پیش بینی‌هایی که انجام شد مدل در کلاس‌هایی بهتر عمل می‌کند که تعداد آن ها در داده های آموزشی بیشتر بوده است. در اینجا تعداد داده‌های مربوط به کلاس خشم و بی تفاوت در داده‌های آموزشی نسبت به بقیه داده ها خیلی بیشتر است و این عدم توازن سبب بایاس مدل یه سمت این دو کلاس می‌شود.

ما در این پروژه تاثیر جنسیت را بررسی نکرده ایم و همین باعث شده دقت مدل کم تر شود زیرا جنسیت تولید کننده ی صدا نیز مهم است و برای مثال؛ بین صدای تولید شده توسط زنی که خوشحال است و مردی که خوشحال است تفاوت وجود دارد ولی در مدل ما این دو را یکسان می‌گیریم و این باعث گیج شدن مدل می‌شود.

راه های زیادی برای افزایش دقت مدل وجود دارد مثلا یک راه ممکن این است که تعداد تایم استپ ها را افزایش دهیم این کار شاید باعث افزایش دقت مدل و استخراج تعداد بیشتری ویژگی شود و یا اینکه از روش‌های استخراج ویژگی دیگری به جز mfcc استفاده شود که به دلیل کمبود وقت و طول کشیدن این قسمت از این کار صرف نظر کردیم.

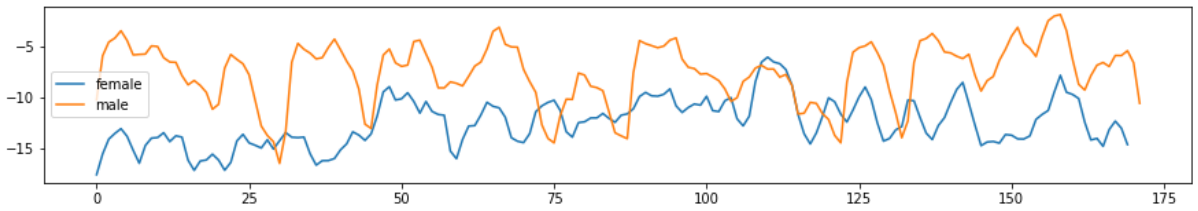
قسمت امتیازی پروژه (در نظر گرفتن جنسیت)

این قسمت از پروژه برای جلوگیری از شلوغ شدن، در یک نوت بوک دیگر به نام Proj4-2_emtiazi.ipynb زده شده است.

قسمت های ابتدایی این نوت بوک دقیقا مانند کاری است که در نوت بوک قبلی انجام دادیم و شامل خواندن فایل های صدا، استخراج ویژگی از آن ها و نرمال سازی است. فقط یک کاری که در این نوت بوک انجام شده و در نوت بوک قبلی (قسمت هایی که در قبل توضیح داده شد) در نظر گرفته نشده این است که در label ها یا همان خروجی ها علاوه بر کلاس احساسات، جنسیت نیز در نظر گرفته شده است و در واقع ما ۱۰ کلاس داریم. زیرا ۲ نوع جنسیت و ۵ نوع احساس داریم که در مجموع ۱۰ نوع کلاس داریم و برای مثال زن شاد و مرد شاد جدا در نظر گرفته شده. و همچنین برای آشنایی با نوع صداها و شکل موج آن ها از انواع احساسات بی تفاوت، شاد و خشم و از هر دو جنسیت صداها و شکل موج آن ها را چاپ کردیم و سپس

اطلاعات استخراج شده توسطmfcc برای هر دو از جنسیت ها که در یک کلاس هستند را در یک نمودار با هم کشیدیم تا ویژگی آن ها را باهم مقایسه کنیم.

برای مثال در نمودار زیر که مربوط به ویژگی های موج صدای زن و مرد برای احساس شادی می باشد می بینیم که عدد ویژگی های مربوط به صدای مرد نسبت به زن معمولا بیشتر است و در واقع نشان دهنده ی این است که در این صداهای انتخاب شده طول موج مربوط به صدای مرد بیشتر از زن بود است.



در ایتدای پروژه تعداد داده های مربوط به هرنوع از کلاس ها را در داده های آموزشی بدست آوردیم که به صورت زیر است.

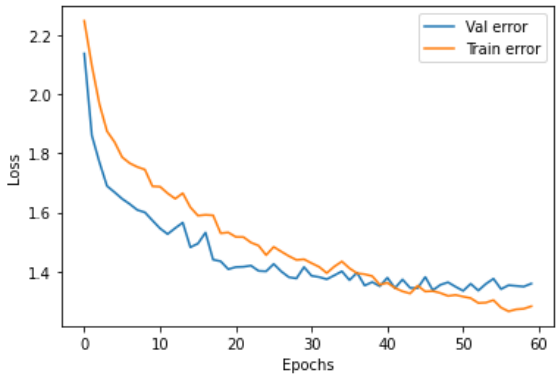
```
number of FS:  182
number of MA:  415
number of FW:  81
number of MH:  59
number of FH:  71
number of FA:  308
number of MW:  68
number of FN:  197
number of MN:  493
number of MS:  120
```

طبق نتیجه بالا می بینیم که تعداد داده ها از نوع مرد بی تفاوت و مرد خشمگین نسبت به بقیه داده ها بیشتر است و احتمال می رود به همین دلیل و به دلیل عدم توازن در تعداد داده ها در کلاس‌ها ، مدل ما به سمت این دو کلاس بایاس کند.

حرف اول هر کلاس نشان دهنده جنسیت و حرف دوم نشان دهنده احساس است.حال به سراغ آموزش مدل می رویم.

آزمایش ۱:

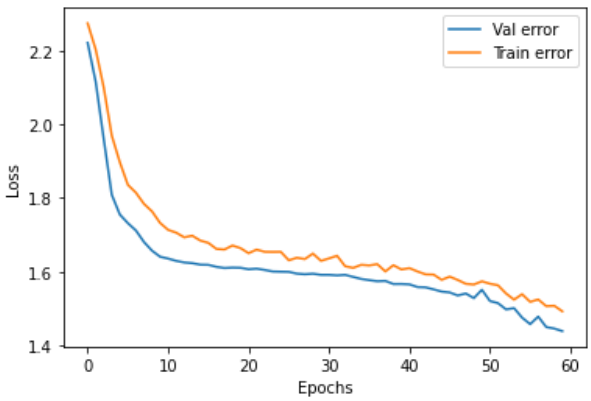
بعد از آموزش این مدل به شکل زیر می رسیم که نشان دهنده‌ی خطا روی داده های آموزشی و اعتبارسنجی است. همانطور که در شکل زیر میبینیم خطا روی داده های آموزشی و اعتبارسنجی کم می شود اما مدل از ایپاک ۴۰ در آستانه overfit شدن است و این به معنی آن است که مدل در حال حفظ داده ها است و قدرت تعمیم مدل کاهش پیدا می کند. این به این معنی است که مدل پیچیده تر از مسئله ما است و به جای یادیگری الگوها در حال حفظ آن هاست. پس باید به دنبال مدل بهتری باشیم که پیچیدگی کمتری از این مدل داشته باشد.



آزمایش۲:

در اینجا مدل را نسبت به مدل قبل ساده تر می کنیم به اینصورت که تعداد یونیت های LSTM را ۲۵ تا در نظر می گیریم حدود یک چهارم تعداد در مدل قبل تا جلوی اورفیت شدن را بگیریم. همانطور که می بینیم در این مدل تعداد پارامترها نسبت به مدل قبل به شدت کاهش یافته است که به دلیل کم شدن یونیت ها است.

همانطور که در شکل زیر میبینیم خطا روی داده های آموزشی و اعتبارسنجی کاهش می یابد اما این کاهش سرعت کمی دارد و طبق نتایج بدست آمده از آموزش مدل طی ٦٠ ایپاک، خطا و دقت روی داده های آموزشی و اعتبارسنجی مدل ۱ ما نسبت به این مدل نتیجه بهتری دارد. این به دلیل این است که مدل ما خیلی ساده تر شده است و همین باعث می شود داده های آموزشی را به خوبی یاد نگیرد (البته underfit نشده ولی سرعت کمی دارد و دقت آن نسبت به مدل قبل کم تر است. و همچنین خطای آن نیز نسبت به مدل ۱ حتی در حالت قبل از اورفیت شدن مدل یک کمتر است.)



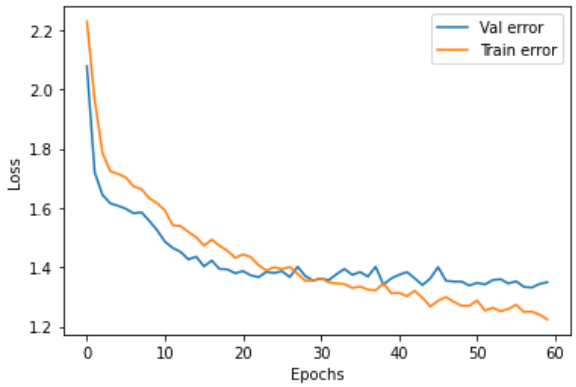
آزمایش کردن را ادامه می دهیم.

آزمایش۳:

در این مدل تعداد یونیت های LSTM را بین تعداد یونیت های این لایه در مدل۱ و ۲ می گیریم یعنی تعداد یونیت ها را عددی بین ۲۵ و ۱۰۰ میگیریم که ما در اینجا عدد ٦٤ را انتخاب کردیم تا درجه پیچیدگی مدل بین این دو باشد.

همانطور که در شکل می بینیم خطا روی داده های آموزشی و اعتبارسنجی کاهش پیدا می کند اما از ایپاک حدود ۳۰ ام مدل در آستانه اورفیت شدن قرار می گیرد و طبق مقایسه ی خطا مدل روی داده های آموزشی و ولید می بینیم که مدل ۱ بهتر از

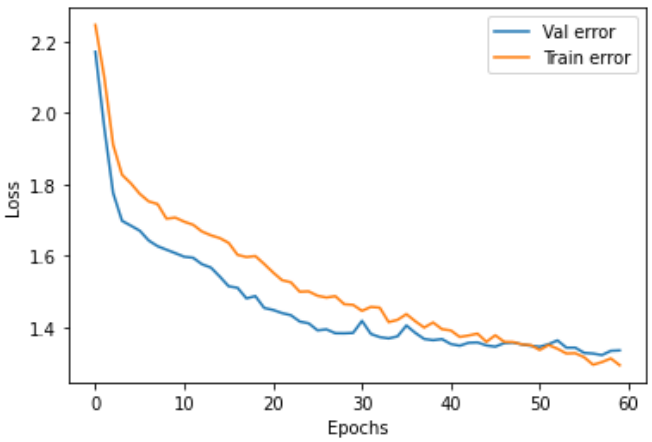
این مدل است. و این نشان می دهد که هنوز هم پیچیدگی مدل نسبت به مسئله زیاد است. یک راه دیگر برای جلوگیری از اورفیت شدن کم تر کردن تعداد ایپاک هاست تا مدت طولانی آموزش نبیند؛ اما ما می خواهیم مدل تعداد مرتبه بیشتری داده ها را ببینید بنابرین سعی می کنیم معماری بهتری پیدا کنیم.



آزمایش ٤:

ما در اینجا تعداد یونیت ها در LSTM را بین تعداد آن در مدل ۲ و مدل ۳ گرفته ایم تا به مدلی بین این دو برسیم که نه خیلی ساده باشد که داده ها را یاد نگیرد و نه خیلی پیچیده باشد که داده رو حفظ کند و این کار باعث می شود تعداد پارامترهای مدل کم شود و مدل زیاد پیچیده نشود

با توجه به نمودار می بینیم که خطا روی داده های اموزشی و ولید در حال کاهش است. و نسبت به مدل یک اوروفیت کم تری داریم و خطا هم روی داده های آموزشی و هم روی داده های ولید نسبت به مدل یک کاهش داشته است. و دقت نیز بیشتر شده. این به دلیل این است که پیچیدگی مدل با مسئله تناسب دارد. پس تا اینجا این بهترین مدل ماست.



دقت و خطای این مدل را روی داده های تست بدست می‌آوریم و به نتیجه زیر می رسیم.

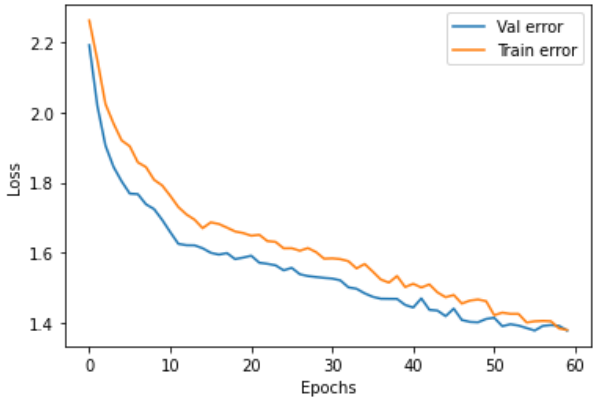
```
Test loss: 1.3341903686523438
Test accuracy: 0.5770925283432007
```

حال می‌خواهیم اگر امکان دارد این مدل را با استفاده از لایه GRU بهبود دهیم.

آزمایش۵:

همانطور که در شکل زیر مشاهده می شود خطای این مدل روی داده های آموزشی و ولید در حال کاهش است. این مدل نسبت به مدل ٤ تعداد پارامترهای کم تری دارد زیر در آن از GRU استفاده کرده ایم و این لایه معماری ساده تری نسبت به LSTM دارد. همچنین این لایه باعث سریع تر شدن مدل هم می شود. اما طبق نتایج نهایی بدست آمده مدل ٤ نسبت به این مدل خطای کمتری روی داده های ولید و آموزشی دارد همچنین دقت بالاتری روی این داده ها دارد.

این می تواند به این دلیل باشد که چون معماری GRU ساده تر است و محاسبات کمتری دارد و به طبع پارامترهای کم تری دارد نسبت به مسئله ساده است و باعث شده مسئله را به خوبی مدل ٤ یاد نگیرد.



پیش بینی داده‌های تست با مدل٤:

حالا نوبت آن است که داده های تست را با بهترین مدلی که بدست آوردیم پیش بینی کنیم.

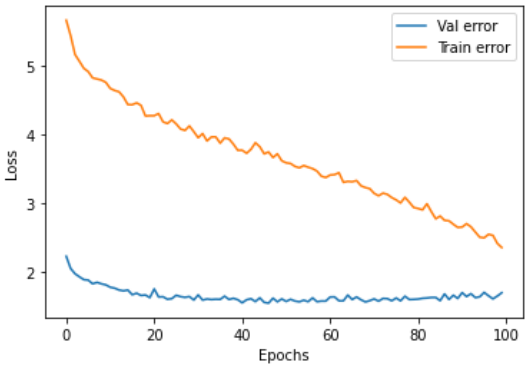
طبق پیش بینی که در نوت بوک داشتیم هیچ داده ای به عنوان شاد تشخیص داده نشده و دلیل آن می تواند این باشد که مدل داده های مربوط به کلاس شاد را خیلی کم دیده چون تعداد آن ها در داده های اموزشی کم بوده و به دلیل عدم توازن مدل این اتفاق افتاده است.

سپس ماتریس مربوط به این مدل را رسم میکنیم. همانطور که میبینیم مدل روی کلاس مرد بی تفاوت و مرد خشمگین نسبت به بقیه بهتر عمل می کند و دلیل آن زیاد بودن داده ها در کلاس آموزشی از این نوع کلاس ها است که باعث شده مدل این نوع کلاس ها را بهتر یاد بگیرد و به سمت آن ها بایاس داشته باشد.

حال برای بهبود عملکرد مدل و جلوگیری از بایاس از روشی استفاده می کنیم که به هر کلاس وزن نسبت دهد و در هنگام fit کردن مدل از آن استفاده می کنیم.

مدل٦ با در نظر گرفتن وزن برای هر کلاس

طبق مدل های مختلفی که با عوض کردن پارامترها، تعداد لایه ها و نورن ها ساختیم این مدل برای کلاس ها با در نظر گفتن وزن، بهترین بود. خطا روی داده های آموزشی و ولید در حال کاهش است اما این کاهش سرعت مناسبی ندارد.

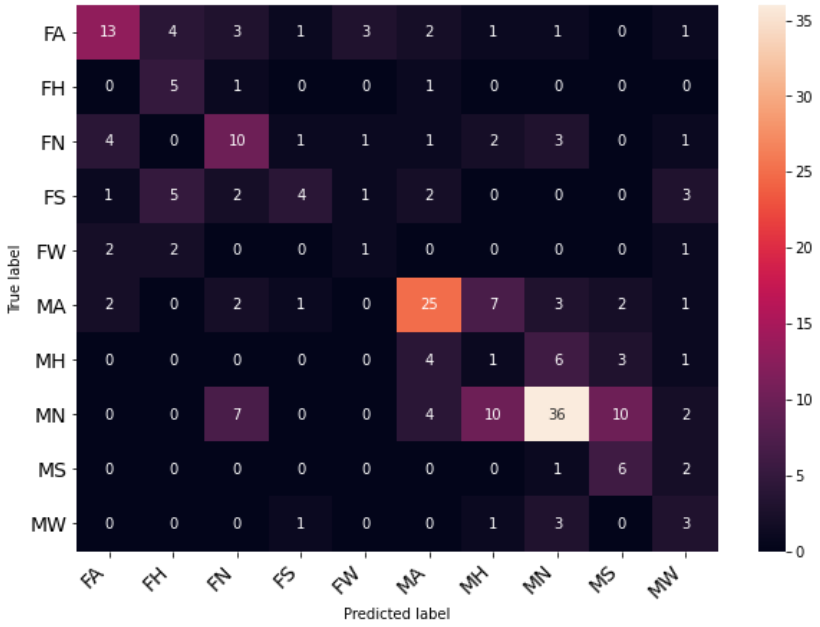


این مدل قبلا روی همین داده ها بدون در نظر گرفتن وزن خوب عمل می کرد و حتی اورفیت شده بود ولی اینجا بدتر عمل می کند. و بنظر می رسد یادگیری وزن ها مسئله را پیچیده تر میکند و نیاز به مدل پیچیده تر هم هست ولی ما به دلیل کمبود وقت از آزمایش بیشتر صرف نظر می کنیم و با همین مدل روی داده های تست پیش بینی انجام می دهیم.

دقت و خطای این مدل روی داده های تست به صورت زیر است و همانطور که میبینیم دقت و خطای خوبی ندارد.

```
Test loss: 1.6297072172164917
Test accuracy: 0.458149790763855
```

ماتریس مربوط به این مدل را رسم می کنیم.



همانطور که می بینیم این مدل در پیش بینی اش بر خلاف مدل های قبلی هر ۱۰ کلاس را تشخیص داده است. در مدلهای قبلی به دلیل بایاس تعدادی از کلاس ها مثل کلاس شادی اصلا تشخیص داده نمی شدند.

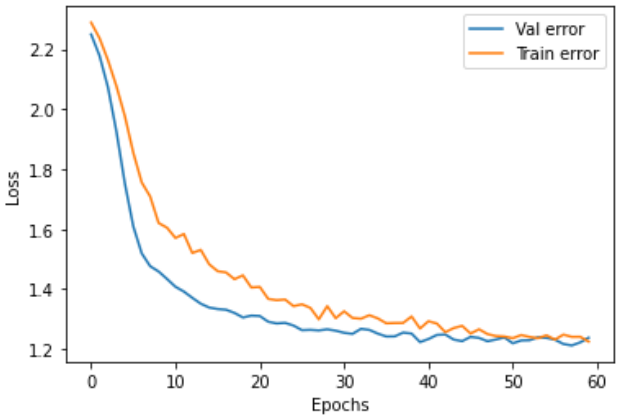
همانطور که در ماتریس می بینیم این مدل کمتر به سمت دو کلاس مرد بی تفاوت و مرد خشمگین بایاس دارد و برای بقیه ی کلاس ها هم تا حدودی خوب عمل می کند. برای مثال در مدل قبل در ماتریس دیدیم که هیچ داده ای به عنوان زن شاد تشخیص داده نشده بود در صورتی که در اینجا می بینیم ۱۶ داده به عنوان زن شاد تشخیص داده شده که از این بین ۵ تای آنها درست بود است. تشخیص های غلط دیگر به دلیل دقت کم و خطای مدل است. ولی همچنان مدل روی دو کلاس مرد بی تفاوت و مرد خشمگین بهتر عمل می کنند.زیرا از این دو کلاس بیشتر نمونه داریم.

مدل ترکیبی:

حال می خواهیم برای آزمایش یک مدل به روش ترکیبی بزنیم یعنی یک مدل بسازیم که فقط داده های مربوط به جنسیت زن را دریاقت کند و مدل دیگر فقط داده های مربوط به کلاس مرد را دریاافت کند و درنهایت این دو مدل را با هم ترکیب کنیم.

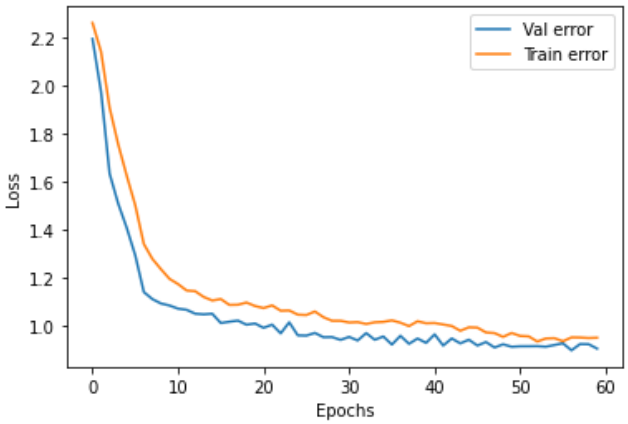
ابتدا باید ویژگی های مرد و زن و برچسب های آن ها را جدا کنیم. در داده های ما ۸۳۹ صدا مربوط به زن و ۱۱۵۵ صدا مربوط به مرد است.

حال دو نمونه مدل با معماری مشابه معماری مدل ٤ (مدل مربوط به قسمت امتیازی) که عملکرد خوبی روی این داده ها داشت می سازیم. یک مدل رافقط روی داده های مربوط به جنسیت زن اموزش می دهیم و نمودار خطا روی داده های آموزشی و ولید برای آن به صورت زیر است.



همانطور که می بینیم خطا روی داده های آموزشی و تست کم می شود و مدل خوبی است.

حال مدل بعدی را برای آموزش فقط روی داده های مربوط به جنسیت مرد می سازیم و به نمودار زیر می رسیم.



همانطور که در این نمودار نیز دیده می‌شود خطا روی داده های آموزشی و ولید کم می شود و این مدل نیز خوب است.

حالا داده های تست را به هر دو مدل می دهیم و خروجی آن ها را ترکیب می کنیم به این صورت که میانگین پیش بینی مدل مربوط به مرد و پیش بینی مدل مربوط به زن را به عنوان پیش بینی مدل ترکیبی در نظر می گیریم.

طبق نتیجه که از ترکیب این دو مدل با میانگین گیری بدست می آید، می بینیم که نتیجه ی مناسبی ندارد و فقط صداها را در ٤ کلاس دسته بندی می کند. در واقع این روش میانگین گیری مناسب نیست و احتمالا با روش میانگین وزن دار بهتر کار می کند.

همانطور که می بینیم الان دقت مدل ترکیبی ما روی داده های تست ۰,۰۳ درصد است و این خیلی کم است و باعث شده اشتباهات زیادی داشته باشد مثلا تعداد خیلی زیادی از صدای مردان بی تفاوت را به عنوان زن با احساس شگفتی شناسایی کرده!!.

ولی معمولا مدل های ترکیبی عملکرد خوبی دارند به شرط آن که مدل هایی که با هم ترکیب می شوند هم دقت و خطای قابل قبولی داشته باشند در نتیجه ترکیب آن ها مدل خوبی را می سازد. یکی از دلایل ممکن که این مدل بد عمل می کند این است که مدل مربوط به داده های زن و مدل مربوط به داده های مرد عملکرد خوبی نداشته اند. زیرا در ساخت این مدل ها وزن برای کلاس ها در نظر گرفته نشده و همچنان بایاس به سمت دو کلاسی که تعداد بیشتری نمونه در داده های اموزشی ما دارند وجود دارد.

نتیجه گیری از قسمت امتیازی

جدا کردن جنسیت به بهبود یادگیری مدل کمک می‌کند زیرا ویژگی هایی که از صداهای زن و مردی که در یک کلاس احساسات هستند بدست می آید طبق نمودارهایی که در ابتدا رسم شد متفاوت هستند ودر نظر نگرفتن جنسیت در صدا باعث می شود مدل یادگیری خوبی نداشته باشد.

همچنین برای جلوگیری از بایاس به سمت کلاسی خاص به دلیل عدم توازن داده ها در کلاس ها، می توان به هر کلاسی با توجه به تعدادش در داده ها وزن نسبت داد. با این روش بایاس مدل کم تر می شود ولی چون مدل باید این وزن ها را نیز یاد بگیرد نیاز به مدل پیچیده تر است نسبت به زمانی که وزن را در نظر نگرفته بودیم.

مدل های ترکیبی نیز عملکرد خوبی می توانند داشته باشند ولی در این پروژه عملکرد خوبی نداشت و این به دلیل میانگین گیری (روش ترکیب دو مدل) پیش بینی های دو مدل بود.