به نام خدا

گزارش کار پروژه کارگاه مت لب

تشخیص جنسیت بر اساس صدا

محمد جواد صاحب ناسی ۹۷۲۴۳۰۴۳ - پریسا احمدزاده راجی ۹۶۲۴۳۰۹۴

تاریخ تحویل: ۱۴۰۰/۱۰/۱۲

چکیده

در این پروژه، به طور کلی این مراحل را طی کردیم:

پیش پردازش:
1. یکسان سازی ابعاد داده های آموزشی
2. تبدیل داده های صوتی به Mel Spectrogram

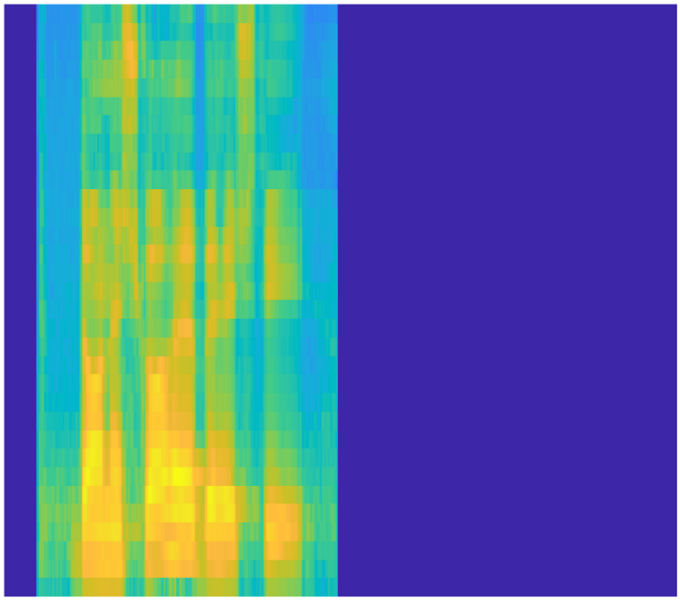
آموزش و تست:
3. انتخاب مدل (Resnet 50) و آموزش دوباره آن (رویکرد Transfer Learning)
4. ذخیره مدل برای تست

دیتاست ما شامل 1001 نمونه داده بود، که تقریبا 20% آن را برای test و validation کنار گذاشتیم (هر کدام حدود 10%). دقت (accuracy) برای validation به 94% و برای test به 90% رسید.

پیش پردازش

ما 1001 داده صوتی داریم، ابتدا داده ها را iterate می کنیم و هرکدام از فایل ها را می خوانیم و در واقع آن ها را لود می کنیم. لازم به ذکر است که داده ها تماما به فرمت wav هستند و با sample rate برابر می باشند. در ادامه با توجه به اینکه برخی داده ها مونو و برخی استریو بودند برای اینکه همه یکسان باشند با کم کردن یکی از کانال ها آن ها را تبدیل به مونو کرده ایم و در ادامه با توجه به اینکه صوت ها طول زمانی متفاوتی داشتند باید سایز آن ها را یکی می کردیم که برای اینکار از padding صفر استفاده کردیم که در واقع برای صوت هایی که طول کمتر از مقدار مشخصی داشتند مقداری صفر در ادامه به آن اضافه کردیم در واقع انگار در ادامه صوت، سکوت اضافه کرده ایم.

پس از این مرحله نوبت به بخش اصلی کار در پیش پردازش و ساخت داده های آموزش می رسد که باید صوت ها را به عکس هایی از mel spectrogram تبدیل کنیم که در ادامه مدل را در واقع برای این عکس ها آموزش می دهیم.
در واقع رویکرد ما بدین صورت بوده که داده های صوتی را به عکس هایی که ویژگی هایی از آن صوت را نشان می دهند تبدیل کردیم و مدل را بر روی آن ها آموزش دادیم.
به عنوان نمونه یک اسپکتروگرام به صورت زیر خواهد بود:



برای آموزش روی این داده ها، ابتدا در بخش پیش پردازش کل داده های مد نظر به همراه label های آن ها را ایجاد کرده ایم و سپس کل آن ها را برای آموزش و تست استفاده کرده ایم.

کد بخش پیش پردازش به صورت زیر نوشته شده است:

```
fs = 16000;%16khz
cd
D = 'C:\Users\Parisa\Desktop\AzMATLABPro\MainDataChangeToWAV';
cd C:\Users\Parisa\Desktop\AzMATLABPro\MainDataChangeToWAV
root_for_image= 'C:\Users\Parisa\Desktop\AzMATLABPro\spectrogram_images\';
name_for_image="";
filelist = dir(fullfile(D, '**\*.*'));  %get list of files and folders in any subfolder
filelist = filelist(~[filelist.isdir]);
num_of_files=length(filelist);

data=strings(3425,2);
label=strings(1001,1);
%[y,fs]=audioread("C:\Users\Parisa\Desktop\AzMATLABPro\MainDataChangeToWAV\1_female\Target\1_f_target_1.wav");
%a=size(y);
%a(2)
for i=1:1001

    temp=strcat(filelist(i).folder,"\");
    data(i,1)=strcat(temp,filelist(i).name);
    if contains(filelist(i).name,'f')==1
        data(i,2)=0;
        label(i,1)=0;
        name_for_image=strcat(string(i),"f.png");
```


همان طور که مشخص است ابتدا داده ها را که در فایل های تو در تو هستند iterate می کنیم، لازم به ذکر است که data شامل 3425 سطر به همراه دو ستون است که در واقع ستون اول آدرس فایل های صوتی و ستون دوم label آن می باشد که در کد مقدار دهی می شود.label نیز به طور جداگانه صرفا برای برچسب های معادل هر صوت تعریف شده است.
در اینجا ۱۰۰۱ داده را ساخته ایم که شامل ۱۰۰۱ سطر اول ماتریس data می باشند.
در حلقه اصلی، بررسی می کنیم که اگر در نام فایل f یا m بود آنگاه برچسب معادل را به ترتیب ۰ یا ۱ بگذارد. و همچنین name_for_image را که بعدتر برای آدرس دهی و ساخت عکس اسپکتروگرام ها مورد استفاده قرار می گیرد را اینجا می سازیم.

```
elseif contains(filelist(i).name,'m')==1
    data(i,2)=1;
    label(i,1)=1;
    name_for_image=strcat(string(i),"m.png");
end
%load from file
%data(i,1)
%class(data(i,1))
y=audioread(data(i,1));
%make all sounds mono
%check if it has two column so is stereo
col=size(y);
%col 2 will show the number of columns of y
if col(2)==2
    y=y(:,1);
end

%make audios into same size by padding zero
%177152 is the size for a audio that is around 3 seconds but we did not
%use this
%70000 ta hodood 4 second
pad=70000-col(1);
if pad>0
    y_new =[y;zeros(pad,1)];



    end
    %convert_to_melSpectrogram
    audiowrite(data(i,1),y_new,16000);
    a=audioread(data(i,1));
    melSpectrogram(a,16000);
    %temp2=strcat(root_for_image,"\")
    addr=strcat(root_for_image,name_for_image);
    axis off
    saveas(gcf,addr)

    filelist(i).name
    %filelist(i).folder;

end
save("C:\Users\Parisa\Desktop\AzMATLABPro\out.mat",'label');
%num_of_files;
%data
```

در ادامه داده را از آدرس مشخص شده می خوانیم و سایز آن را در col ذخیره می کنیم و اگر دارای دو ستون بود آنگاه یک ستون را حذف می کنیم که به مونو تبدیل شود، سپس برای اینکه سایز اسپکتروگرام ها یکسان باشد با توجه به اینکه صوت ها کمتر از ۴ ثانیه می باشند اختلاف سایز صوت را با 70000 حساب می کنیم و به اندازه آن padding صفر انجام می دهیم.
لازم به ذکر است که 177152 که کامنت شده است سایز یک صوت نمونه ۳ ثانیه ای بود اما استفاده از آن به جای 70000 صوت های ۱۱ ثانیه ای ایجاد می کرد. با توجه به نمودار هر ۱۰۰۰۰ تا حدود نیم میلی ثانیه روی نمودار اسپکتروگرام است.
بنابراین صوت ها را به صوت هایی با طول حدود ۴/۵ ثانیه تبدیل کردیم. بعد از آن صوت های جدید را به جای صوت های پیشین ذخیره کردیم و در نهایت عکس های mel spectrogram را در فایل مشخص شده ذخیره ساختیم و در نهایت هم label های ایجاد شده را در فایل out.mat ذخیره کردیم. پس از آن این داده های تولید شده را برای بخش بعدی مورد استفاده قرار دادیم.

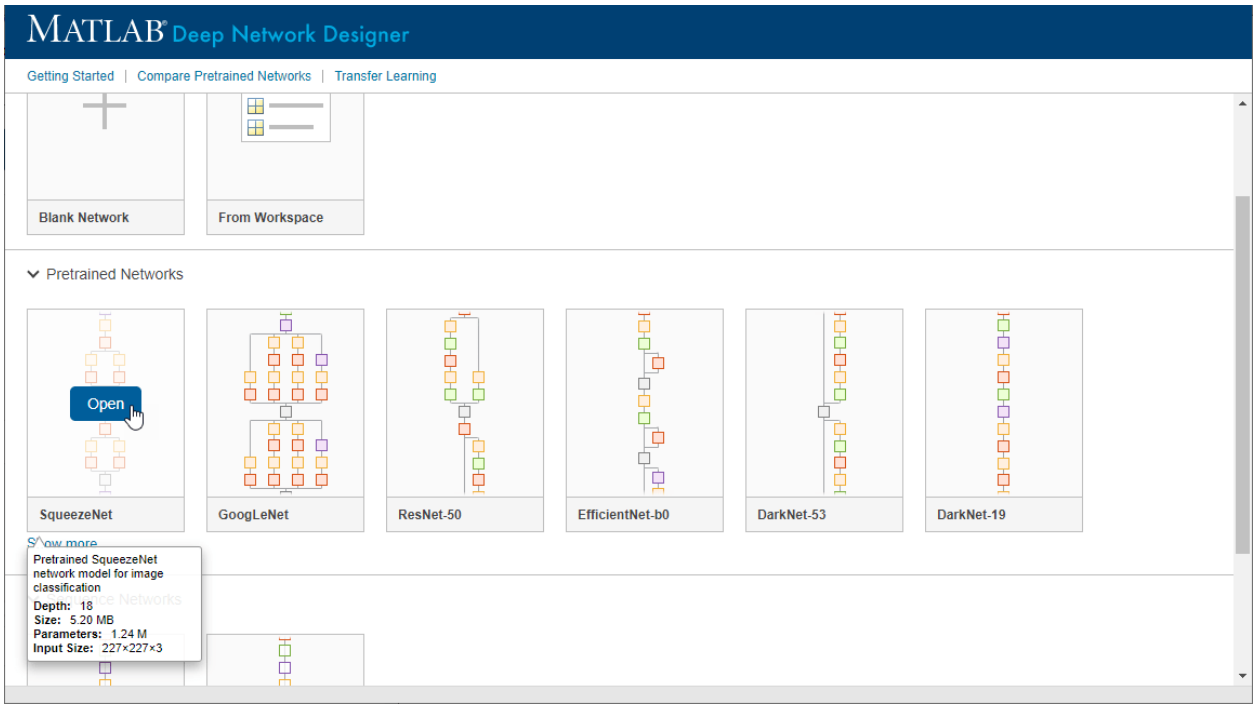
مدل و آموزش

با توجه به این که تا اینجا، داده های صوتی را به اسپکتروگرام (عملا عکس) تبدیل کرده ایم، حالا میتوانیم از روش ها و مدل های classify کردن عکس استفاده کنیم. برای مدل، Resnet 50 را انتخاب کردیم.

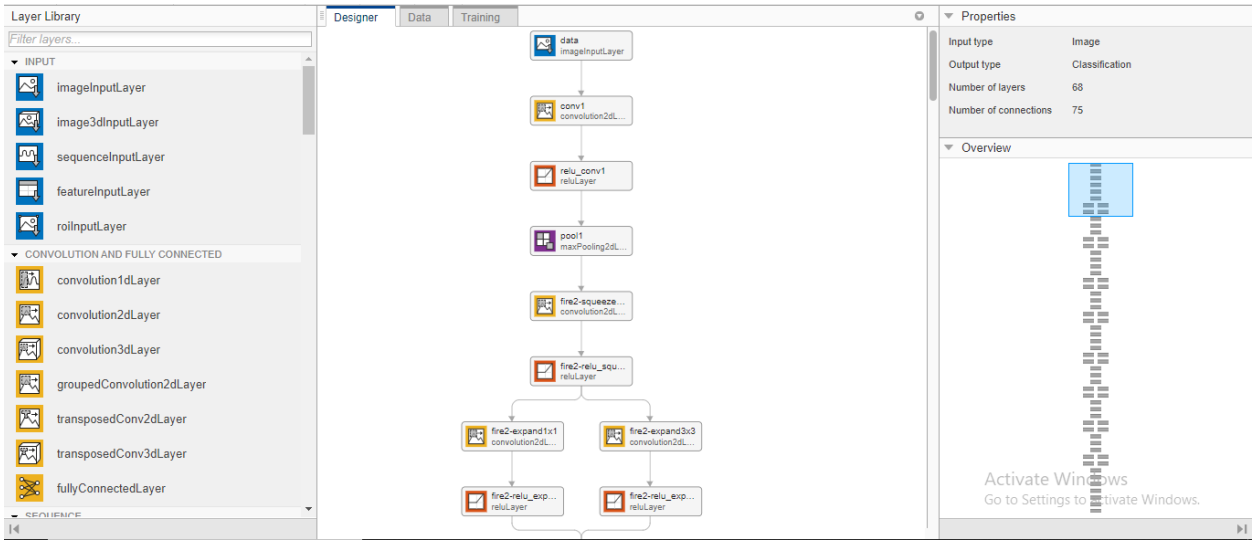
برای سادگی و با توجه به این که داده های زیادی هم نداشتیم، تصمیم گرفتیم از رویکرد transfer learning استفاده کنیم؛ به این صورت که مدل قبلا آموزش داده شده در متلب را لود کرده، تغییرات لازم را در ساختار آن اعمال کردیم (این مدل برای 1000 کلاس ساخته شده، ولی ما در این مسئله 2 کلاس داریم).

برای این منظور، به جای این که خودمان کد بنویسیم، از Deep Network Designer App در متلب استفاده کردیم، که علاوه بر ساده کردن کار ها با رابط گرافیکی، امکانات جذابی برای مانیتورینگ هم داشت.

وقتی وارد این app بشویم، میتوانیم مدل را انتخاب کنیم:

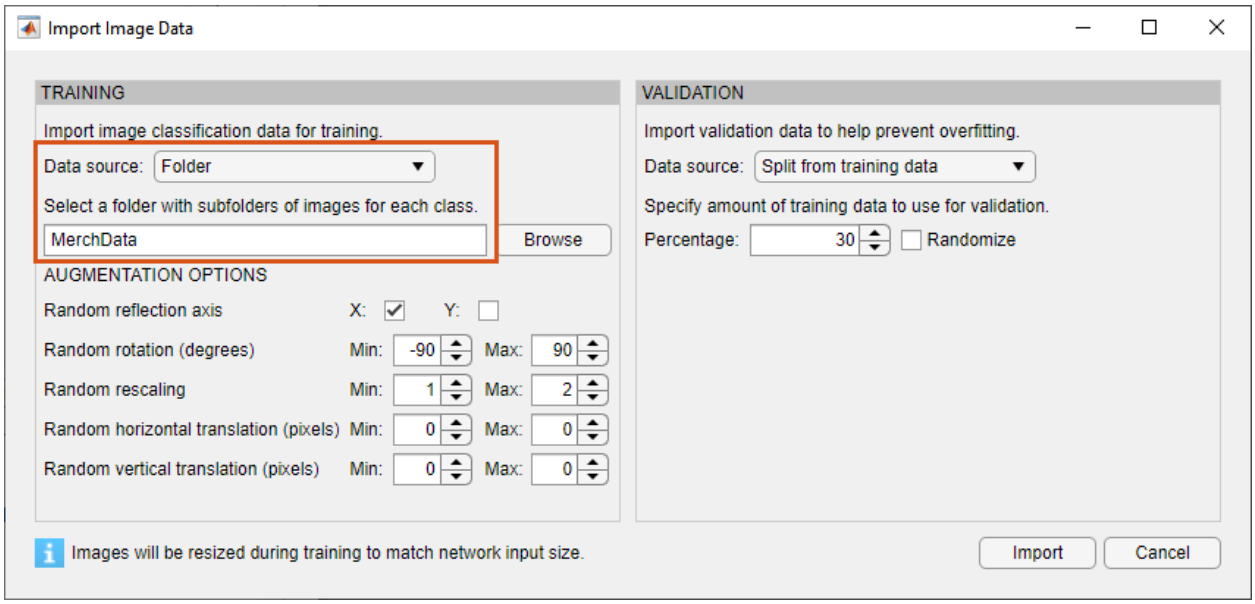


پس از انتخاب در چنین صفحه ای ساختار مدل نمایش داده میشود (تب designer):

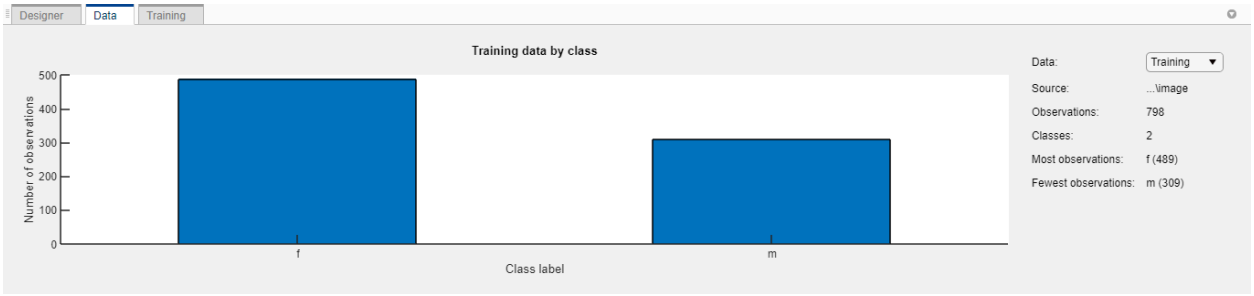


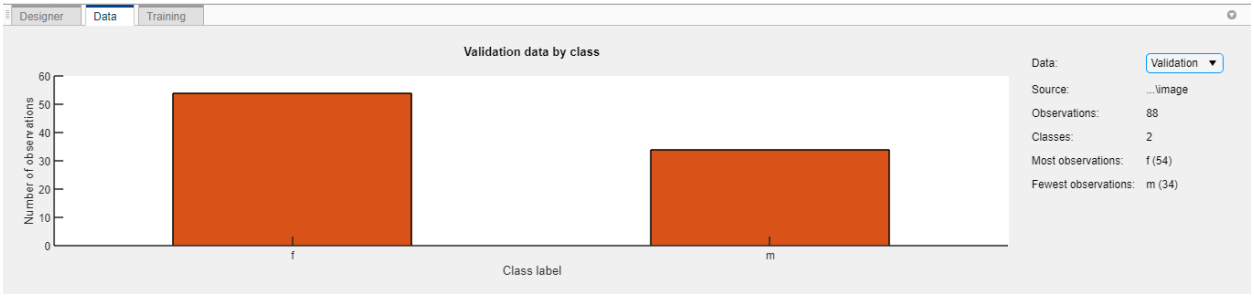
تغییرات مد نظر در ساختار شبکه را همان جا می توانیم اعمال کنیم: لایه هایی را حذف کنیم و از سمت چپ صفحه، لایه های جدیدی با مشخصات دلخواه را جایگزین آن ها کنیم.

در تب data میتوانیم دیتا ست را import کنیم (یک راه این است که آدرس پوشه شامل دیتاست را وارد کنیم. البته داخل آن باید برای هر کلاس از داده های یک پوشه جداگانه داشته باشیم). در سمت راست میتوانیم درصدی از داده ها را مشخص کنیم طوری که برای validation کنار گذاشته شوند. امکاناتی هم برای data augmentation وجود دارد. (این عکس از داکیومنتیشن برداشته شده و برای پروژه ما نیست! در این پروژه به این شکل نمی توان augmentation انجام داد.)

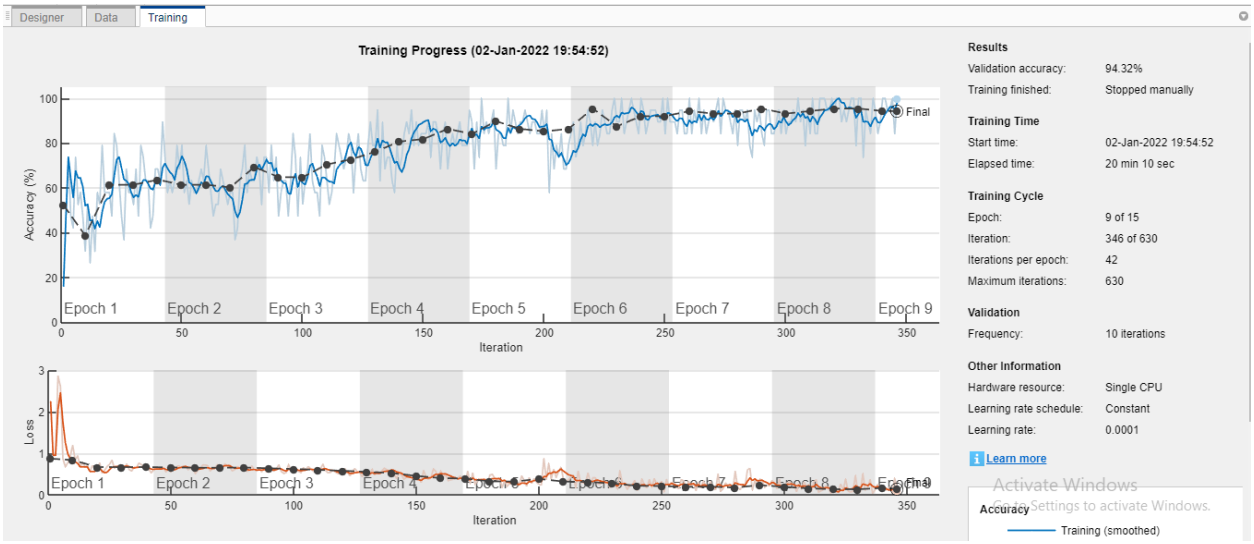


سپس یک صفحه شامل اطلاعاتی مربوط به دیتاست نمایش داده میشود:





حالا میتوانیم به تب training برویم و در قسمت training options ، تنظیمات مدنظر مان برای آموزش را اعمال کنیم. با زدن گزینه train وارد چنین صفحه ای می شویم که در آن به صورت real time نمودار های برای رصد فرایند آموزش رسم می شود (اولی برای accuracy و دومی برای loss):



بعد از اتمام فرایند (یا متوقف کردن آن) میتوانیم مدل را export کنیم تا به عنوان یک متغیر وارد working space شود. در این مرحله حتی می توان برای مدل، به صورت خودکار کد هم ساخت (با گزینه generate code for training).

این جا با دستور save، مدل را روی دیسک و در قالب فایل .mat ذخیره میکنیم تا بعدا استفاده کنیم.

تست

تا این جا یک مدل آموزش داده شده داریم که در قالب فایل .mat ذخیره شده. در این مرحله کل کاری که باید بکنیم load

کردن مدل از فایل و ورودی دادن داده های تست به آن است:

```
test_data_dir = 'test';
Files=dir(test_data_dir);

% load the saved trained network
net = coder.loadDeepLearningNetwork('trainedNetwork_1.mat');

cntr = 0;
correct = 0;
```

```
for k=1:length(Files)
   if contains(Files(k).name, '.png')
        cntr = cntr + 1;

        file_dir = strcat(test_data_dir,'/',Files(k).name);

        % load image
        img = imread(file_dir);

        % resnet accepts input in a specific size
        img = imresize(img, [227 227]);

        % prediction
        pred = classify(net,img);

        if contains(Files(k).name, string(pred))
            correct = correct+1;
        end

   end
end

disp('test accuracy: ')
disp(correct/cntr)
```

خروجی:

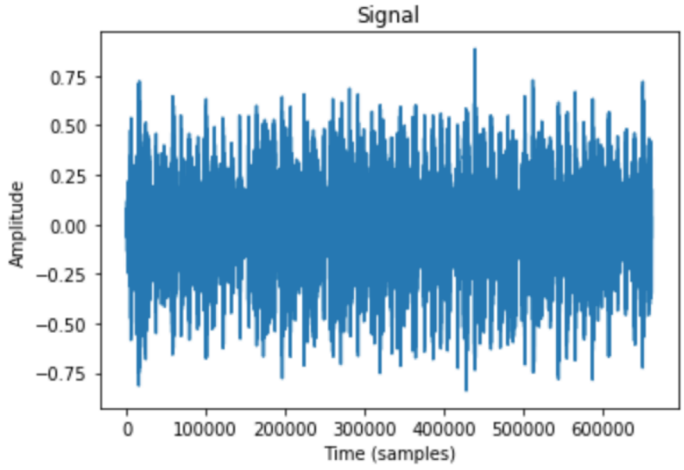
```
test accuracy:
    0.9035

>>
```

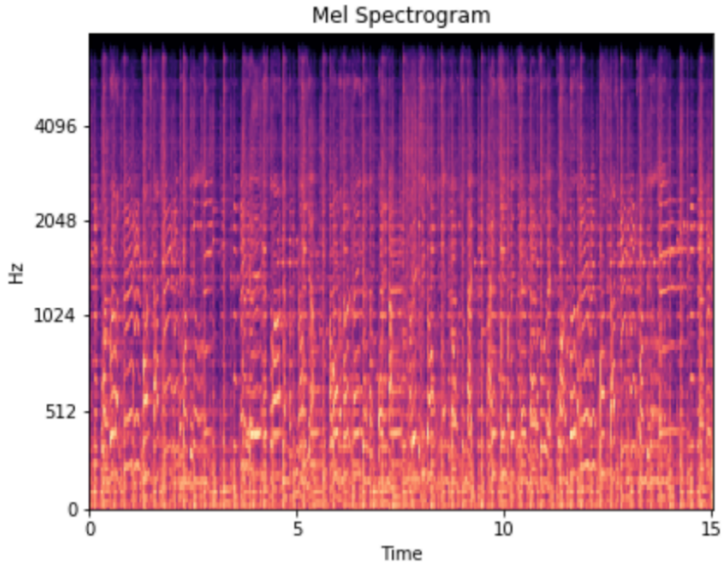
توضیحات تکمیلی:

:spectrogram

صوت های مختلف فشار هوای متفاوت ایجاد می کنند و از این طریق با هم متفاوت هستند، از این داده ها می توان نمونه برداری انجام داد و آن ها را به صورت سیگنالی و روی نمودار نمایش داد که به آن waveform می گویند.



هر سیگنال صوتی از چندین موج صدای تک فرکانسی تشکیل شده است و زمانی که نمونه برداری می کنیم صرفا بخش دامنه اعمال می شود. اما اگر از تبدیل فوریه استفاده کنیم، آنگاه سیگنال به فرکانس های مجزای آن تفکیک می شود، دراقع سیگنال را از فضای زمان به فرکانس می برد، که به آن spectrum می گویند. برای نمایش سیگنال هایی که نامتناوب هستند و محتوای فرکانسی سیگنال با زمان تغییر می کند از spectrogram استفاده می شود. Mel spectrogram بدین صورت است که محور Y در نمودار spectrogram به mel scale برده می شود که در این صورت mel spectrogram خواهیم داشت.
در واقع برخی فرکانس ها وجود دارند که انسان آن ها را به طور یکسان تشخیص می دهد، مقیاس mel با توجه به این مسئله مقیاس بندی را انجام می دهد.
شکل زیر یک نمونه از mel spectrogram می باشد.



تغییرات نسبت به چیزی که ارائه دادیم

- آموزش مدل را دوباره، با کانفیگ متفاوت و اندکی طولانی تر انجام دادیم. در نتیجه accuracy افزایش یافت.
- کدی که قبلا برای تست نوشته بودم، یک نمونه را ورودی میگرفت و پردازش می کرد. کد فعلی مجموعه داده های تست را یکجا می گیرد و accuracy محاسبه میکند.