دانشگاه صنعتی امیرکبیر

دانشکده مهندسی پزشکی

گزارش پروژه DSP

پردازش صدای قلب

دانشجو

سیدابوالفضل مرتضوی ۹۸۳۳۰۶۳

تیر ۱۴۰۲

فهرست مطالب

توضیح کلی

با توجه به اینکه تعداد داده‌های ارسالی برای آموزش کم است مدل‌های مورد استفاده نمی‌توانند به خوبی با داده‌ها منطبق شوند. برای نتیجه بهتر به سایت کگل مراجعه شد و دیتای مورد نظر به طور کامل از آن سایت لود شد و این بار از شبکه‌های RNN برای بررسی استفاده شد. در فایل ارسالی دو نوت‌بوک پایتون موجود است.

مدل برروی داده‌های ارسالی

** این بخش اعمال مدل صرفا برروی داده‌های ارسالی شمااست. فایل مربوط به توضیحات این بخش **DSP_Project** نام دارد.

۲-۱ توضیحات

در انجام این پروژه از زبان برنامه نویسی پایتون و همچنین کتابخانه scikit learn استفاده شد. تعداد ۲۱۰ داده برای آموزش و ۹۰ داده برای تست استفاده شدند. در انجام این پروژه چند چالش وجود داشت که به ترتیب در زیر بیان می‌شوند:

- داده‌ها به یک شکل واحد نام گذاری نشده بودند وهمین امر باعث می‌شد تا وارد کردن دیتا به مشکل بخورد. برای حل این مشکل از کتابخانه OS وقطعه کد زیر استفاده شد تا اسامی داده‌ها به یک شکل واحد تبدیل شوند.

```
1. def main():
2.     i = 0
3.     path='E:/validation-physionet/'
4.     for filename in os.listdir(path):
5.         my_dest ="audio" + str(i) + ".wav"
6.         my_source =path + filename
7.         my_dest =path + my_dest
8.         # rename() function will
9.         # rename all the files
10.        os.rename(my_source, my_dest)
11.        i += 1
12.# Driver Code
13.if __name__ == '__main__':
```

```
14.     # Calling main() function
15.     main()
```

در این بخش ابتدا اسامی فایل‌ها دریافت و سپس به فرمت audio i تغییر پیدا کرد. اندیس i صرفا شماره این داده است. در تغییر اسامی فایل‌ها توجه شد که ترتیب داده‌ها به هم نریزد.

- مشکل دیگر طول داده‌ها بود که یکسان نبودند و باعث می‌شد تا نتوان ماتریسی واحد برای کل داده‌ها در نظر گرفت. برای رفع این مشکل ابتدا طول بلندترین داده پیدا شد و سپس طول بقیه داده‌ها با زیرو پدینگ به طول مورد نظر رسید.
- برچسب‌های داده شده در فایل CSV فاقد هدینگ بودند و همین امر باعث می‌شد که نتوان با استفاده از کتابخانه‌های موجود برچسب‌ها را اضافه کرد. برای حل این مشکل در فایل CSV به برچسب‌ها هدینگY تعلق گرفت تا اضافه کردن آن‌ها به آسانی صورت گیرد.

۲-۲ مراحل انجام

** در نوت بوک ارسالی تمامی مراحل اسامی خاص خود را دارند و از هم مجزا هستند.

۲-۲-۱ Import Data

در این مرحله ماتریس X_Data با ابعاد ۲۴۴۰۰۰*۳۰۰ که حامل تمامی داده‌ها است تولید شده و سپس داده‌ها بر این ماتریس افزوده می‌شوند. از آن‌جا که مدل‌های موجود برروی آرایه‌های نامپای عمل می‌کنند فایل‌ها با فرمت نامپای فراخوانی شدند.(۲۴۴۰۰۰ طول بلندترین داده موجود است.). برای وارد کردن داده‌ها از کتابخانه Scipy و تابع wav استفاده شد.

۲-۲-۲ Import Labels

در این بخش برچسب‌های داده‌ها وارد شده و در آرایه Y_Data ذخیره شدند. ابعاد این آرایه ۳۰۰*۱ است.

۲-۲-۳ Choosing test and train data

در این بخش ۲۱۰ داده برای آموزش و ۹۰ داده برای آزمودن صحت آموزش انتخاب شدند.

۲-۲-۴ Using models

در این بخش ۴ مدل از مدل‌های موجود برای آموزش استفاده شدند.

۲-۲-۴-۱ SVC(Sigmoid)

در این بخش از متد SVC با کرنل سیگموئید استفاده شد. کرنل سیگموئید یکی از کارآمد ترین کرنل‌ها برای تفکیک‌های دوکلاسه است. صحت به دست آمده با این متد برابر ۷۵٪ می‌باشد.

۲-۲-۴-۲ SVC(rbf)

کرنل دیگری که برای جداسازی داده‌ها کارآمد است کرنل rbf است. در این بخش اما صحت به دست آمده قابل قبول نبوده وبرابر با۴۶٪ است. علت این امر است است که این نوع کرنل بیشتر برای جداسازی داده‌ها در سه بعد و به صورت کروی مناسب است.

۲-۲-۴-۳ SVC(linear)

این نوع کرنل برای داده‌هایی که به صورت خطی جدایی پذیر هستند مناسب است. با اعمال این کرنل به داده‌ها صحت ۶۴٪ به دست آمد که قابل قبول است. این امر نشان می‌دهد که داده‌ها به صورت خطی قابل تفکیک هستند.

۲-۲-۴-۴ SVC(poly)

در این بخش از کرنل poly استفاده شد. این نوع کرنل با چند جمله‌ای درجه سه اعمال شد و صحت ۴۱٪ را نشان داد. باافزایش درجه آموزش بهتر صورت می‌گیرد اما خطر overfit افزایش می‌یابد.

مدل برروی کل داده‌ها

۳-۱ توضیحات

در ادامه توضیحات برای نوت بوک **DSP_Project2** نوشته می‌شود.

در این بخش داده‌های مورد بررسی از سایت کگل برروی نوت بوک اضافه شدند زیرا حجم دیتاست کلی زیاد بود و سیستم بنده قادر به ترین کردن داده‌ها نبود. به علت طولانی بودن مراحل این بخش، توضیحات مربوط به هر بخش کد مستقیما از document توابع و کتابخانه‌های استفاده شده در فایل نوت بوک کامنت شده‌اند.

۳-۲ مراحل انجام

۳-۲-۱ Import Data

مراحل انجام همانند بخش قبل می‌باشد که ابتدا دیتا و لیبل‌های مربوط به داده‌های سالم و ناسالم اضافه شدند. سپس یک لودر برای خواندن فایل‌های صوتی با کتابخانه librosa نوشته شد. در این بخش از تابع mfcc در کتابخانه librosa استفاده شد تا فیچرهای مورد استفاده در شبکه RNN استخراج شوند.

۳-۲-۲ Preprocessing and Encoding

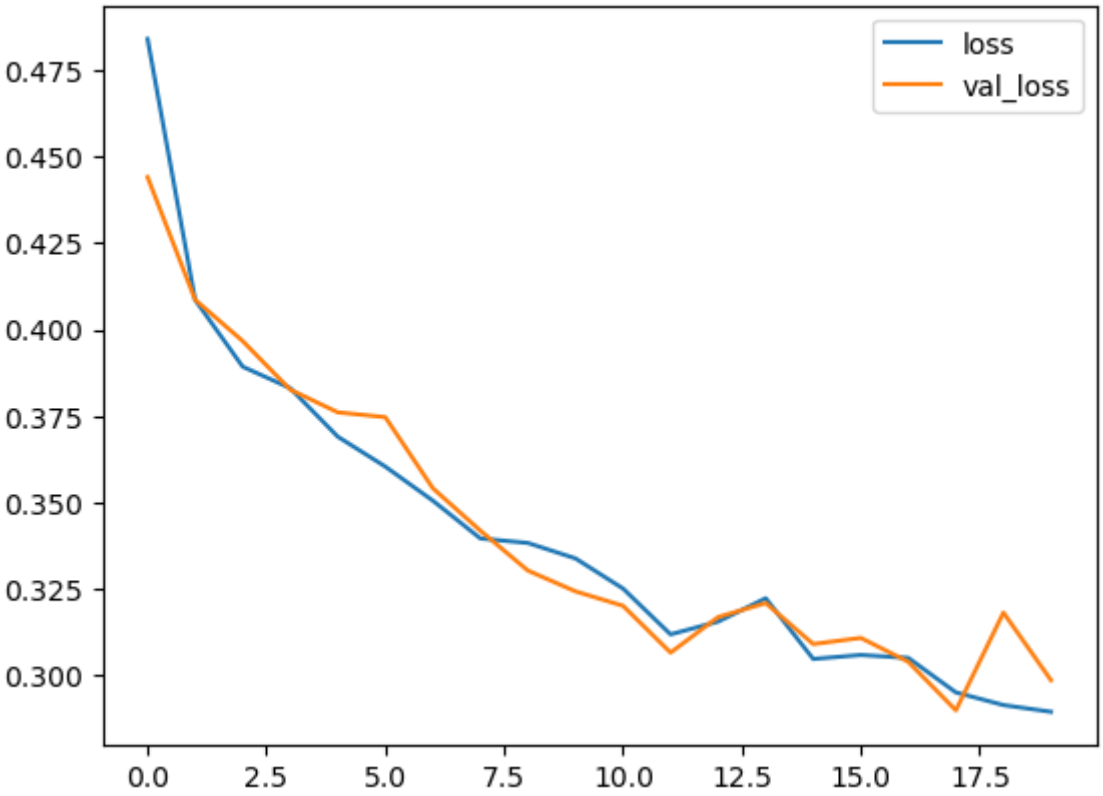
در این بخش دادها برای لودر نوشته شده در بخش قبل فرستاده می‌شوند. در حین لود شدن دیتا همزمان طول مربوط به داده‌ها با زیروپدینگ یکسان می‌شود. این عمل برای داده‌های سالم و ناسالم مربوط به بخش‌های تست و ترین انجام می‌شود.

۳-۲-۳ Concatenation

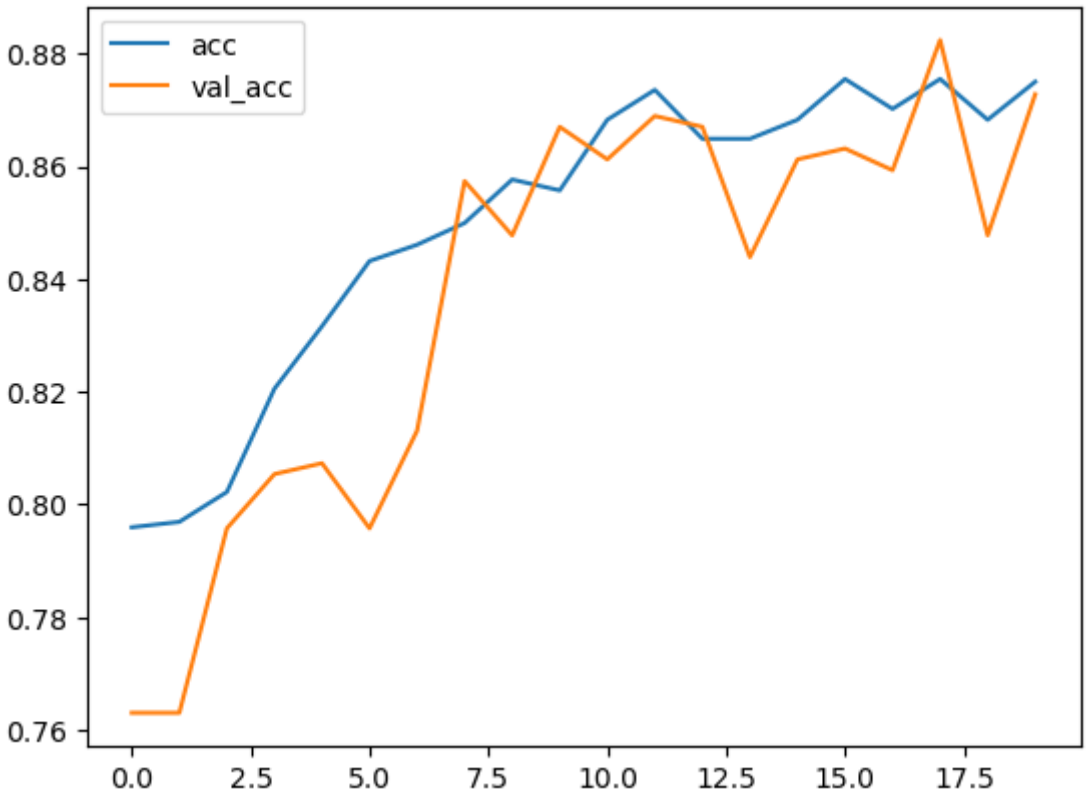
در این بخش داده‌های تست و ترین مرتب می‌شوند. به این صورت که داده‌های سالم و ناسالم هربخش(تست و ترین) در یک آرایه جمع شده و داده‌های تست و ترین را تشکیل می‌دهند.

۳-۲-۴ Model

در نهایت مدل LTSM که یک مدل از شبکه‌های RNN است برای ترین تشکیل شد. این نوع مدل با توجه به منابع ذکرشده در پایان گزارش از مدل‌های پرکاربرد برای پردازش صوت می‌باشند.



شکل ۳.۱ نمودار تغییرات خطا



شکل ۳.۲ نمودار تغییرات صحت

در آخر صحت پیش‌بینی در این مدل ۸۸٪ است که نسبت به مدل‌های قبلی بیشتر است.

```
130/130 [==============================] - 2s 8ms/step
21/21 [==============================] - 0s 12ms/step - loss: 0.2781 - acc: 0.8750
Model evaluation accuracy:  88 %
```

شکل ۳.۳ خروجی پیشبینی

منابع

1. https://towardsdatascience.com/recognizing-speech-commands-using-recurrent-neural-networks-with-attention-c2b2ba17c837
2. https://ieeexplore.ieee.org/document/8551185
3. https://keras.io/examples/vision/video_classification/
4. https://www.apriorit.com/dev-blog/609-ai-long-short-term-memory-video-classification