به نام خدا

پروژه درس مبانی بینایی کامپیوتر

دانشکده مهندسی کامپیوتر

دانشگاه علم و صنعت ایران

استاد محمدرضا محمدی

نیم سال دوم ۱۴۰۳-۱۴۰۲

مبحث :

Face Anti Spoofing

ارائه دهندگان :

آریا شهسوار

ریحانه شاهرخیان

بخش اول - جمع آوری داده:

قسمت اول- آموزش: برای آموزش مدل از داده های CASIA_faceAntisp استفاده کردیم. دلیل انتخاب این مجموعه داده به نسبت سایر گزینه ها حجم کمتر (برای دانلود سریعتر- تقریبا ۸۰۰ مگابایت در مقایسه با سایر مجموعه دادگان با حجم های بیشتر از ۲ گیگابیات) و همچنین استفاده تعداد خوبی از منابع از این دیتاست برای تسک مشابه است. این مجموعه داده شامل ۲ پوشه یکی مربوط به داده های train و دیگری مربوط به داده های test می باشد. داده های train شامل ۲۰ پوشه که هر کدام شامل ۱۲ ویدیو بودند و داده های test شامل ۳۰ پوشه (مشابه train هر کدام شامل ۱۲ ویدیو). همچنین داده های label نداشتند و نیاز بود به صورت دستی برچسب‌گذاری شود که با بررسی فایل های موجود در هر پوشه به این رسیدیم که همه آنها از الگوی ثابتی برای spoof بودن/نبودن پیروی میکنند. به گونه ای که ویدیوهای 'HR_4.avi.1', 'HR_1.avi', '2.avi', 'avi' زنده بودند و بقیه spoof بنابراین بر این اساس برچسب‌گذاری را نیز انجام دادیم و چون داده تست این دیتاست را نیاز نداشتیم برای train استفاده کردیم.

قسمت دوم- تست: برای تست هر دو حالت نیاز بود که داده هایی برای تست جمع آوری شود. برای اینکار سعی کردیم با بررسی داده های آموزشی و با توجه به خواسته سوال حالت های مختلفی برای تست مدل به کار ببریم :

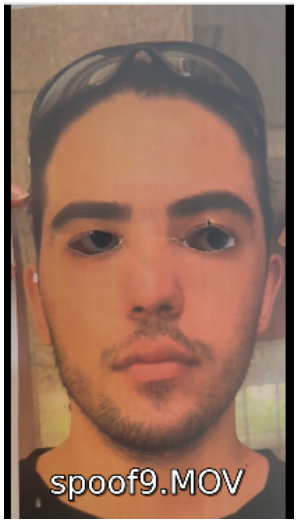
۱ - دسته ویدیو های real: از تعدادی از افراد با سنین و جنسیت ها گرفته شده که شرایط محیطی (شامل مکان - نور - زاویه و ...) در هر کدام متفاوت است.

۲ - دسته ویدیوهای fake: این دسته از تصاویر با توجه به خواسته سوال و دیتاست های آموزشی به چند طریق آماده شده است:

- گرفتن ویدیو از عکس فرد بر روی گوشی

- گرفتن ویدیو از عکس فرد بر روی کاغذ

- پوشاندن چهره فرد با عکس خود (مکان دو چشم در عکس بریده شده)



بخش دوم - پیش پردازش داده ها:

با توجه به اینکه ورودی که به ما داده میشد ویدیو یک شخص بود نیاز بود تا در ابتدا یک فریم از آن را استخراج کنیم. یکی از دید هایی که نسبت به استخراج فریم از تصویر داشتیم این بود که فریمی را انتخاب کنیم که نسبت به فریم پیشین خود بیشترین تغییر را داشته باشد (به این صورت احتمالا در عکس واقعی این فریم مربوط به حالتی خواهد بود که فرد پلک میزند) که در حالت فیک اینطور نیست. با این حال با توجه به اینکه در بخش دسته ویدیو های فیک حالت هایی که وجود داشت که پلک زدن نیز اتفاق می افتاد تصمیم نهایی بر این شد که یک فریم (فریم وسط) را به عنوان فریم اصلی تصویر استخراج کنیم.

در مرحله بعدی نیاز بود تا چهره فرد را از فریم اصلی تصویر بدست بیاوریم که این کار نیاز بوسیله ماژول face_recognition به سادگی قابل حل بود. همچنین این مورد نیز باید در نظر گرفته شود که نیاز بود تا تمامی داده ها دارای یک سایز یکسانی باشند پس عملیات resize نیز انجام میشود.

بخش سوم - استخراج ویژگی:

برای استخراج ویژگی ها از lbp_histogram و کانال s در فرمت hsv استفاده کردیم. دلیل استفاده از کانال s، تفاوت بسیار خوب آن در دو حالت زنده و spoof است. نمونه‌ی خروجی کانال s از دو تصویر spoof و زنده‌ی یک شخص:

Spoof

Non-spoof:

سپس دو ویژگی در آمده را باهم concate کردیم و به عنوان feature ورودی به svm می‌دهیم تا کار classification را برای ما انجام دهد. همچین بخشی از داده‌ها را به عنوان داده‌ی validation انتخاب کردیم که دقت آن به صورت زیر است:

```
Feature extraction method accuracy: 0.9083333333333333
```

روی داده‌های تست جمع آوری شده نیز این مقادیر را خروجی گرفتیم:

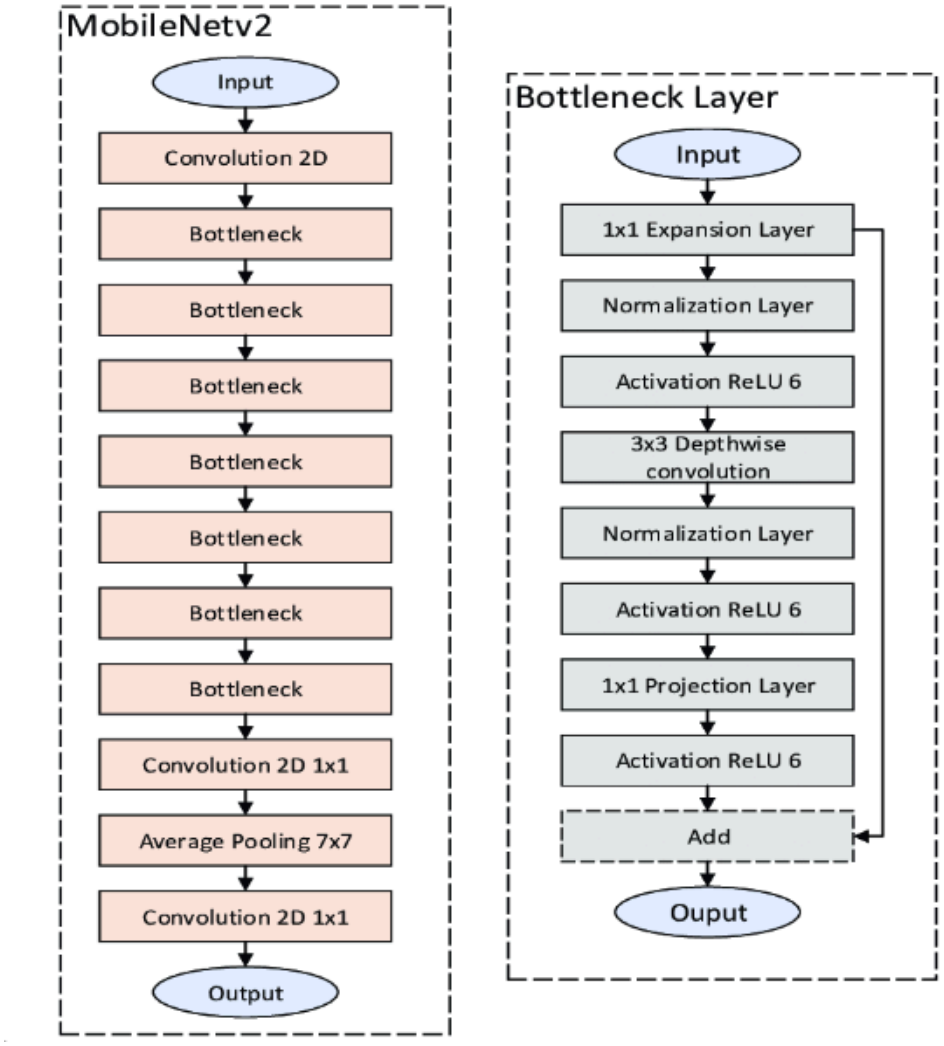
filename	liveness_score	liveness_score_crop	liveness_score_frequency
anti-spoof1	1	1	1
anti-spoof2	0	1	1
anti-spoof3	0	0	1
anti-spoof4	0	0	1
anti-spoof5	0	0	1
anti-spoof6	1	1	1
anti-spoof7	1	1	1
anti-spoof8	1	0	1
anti-spoof9	1	1	1
spoof1	0	1	1
spoof2	0	0	1
spoof3	0	0	1
spoof4	0	1	1
spoof5	1	1	1
spoof6	0	0	1
spoof7	0	1	1
spoof8	0	0	1
spoof9	1	1	1
spoof10	1	0	1
spoof11	0	0	1

که در ستون سوم همه‌ی موارد را زنده تشخیص داده: (ولی درباره‌ي ستون اول دقت 65 درصد و برای فریم‌های crop شده دقت 55 درصد داریم که مطلوب نیستند.)

البته ویژگی‌های دیگری مانند fourier - HoG و ... نیز تست شد ولی خروجی مطلوبی گرفته نشد.

بخش چهارم - مدل یادگیری عمیق:

در این قسمت مدل‌های آماده‌ی زیادی جهت استفاده به عنوان baseline وجود داشت که ما MobileNetV2 را انتخاب کردیم:

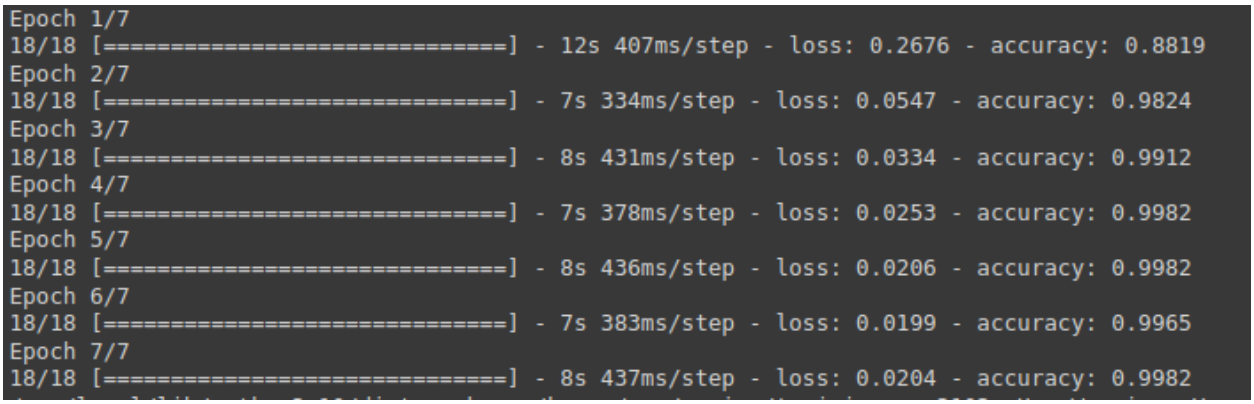


سپس یک لایه GAP و بعد آن Dense ران کردیم و در نهایت یک Dense با یک نورون برای تشخیص زنده بودن یا نبودن داریم.

در این بخش هم ابتدا یک قسمتی از داده‌ها را برای validation انتخاب کردیم تا بفهمیم تا چه epoch ای مشکل overfit نداریم و خروجی نهایی رو تست به این شکل شد:

filename	liveness_score	liveness_score_crop	liveness_score_frequency
anti-spoof1	0.9860291	0.886060833930969	0.49011075
anti-spoof2	0.9940369	0.869118928909302	0.37556037
anti-spoof3	0.9995659	0.93857604265213	0.36781752
anti-spoof4	0.995777	0.995321929454803	0.7736416
anti-spoof5	0.8473715	0.358389258384705	0.29056457
anti-spoof6	0.9925096	0.838878989219666	0.44120994
anti-spoof7	0.9947425	0.968680799007416	0.47626716
anti-spoof8	0.9996941	0.962214946746826	0.3116397
anti-spoof9	0.9905934	0.991879165172577	0.3829256
spoof1	0.9893231	0.984255850315094	0.35621613
spoof2	0.08956659	0.0378418117761612	0.6054576
spoof3	0.3818883	0.0272018481045961	0.6291491
spoof4	0.71064746	0.628495931625366	0.2133216
spoof5	0.08808319	0.267544776201248	0.34722653
spoof6	0.087301336	0.86078268289566	0.17002606
spoof7	0.0015884337	0.653576374053955	0.886807
spoof8	0.027212229	0.621666014194489	0.62063575
spoof9	0.05941801	0.152895227074623	0.558425
spoof10	0.096296914	0	0.506841
spoof11	0.07956236	0.770594120025635	0.03254518

و بار دیگر با توجه به اینکه تعداد epoch لازم را میدانستیم تمام دیتا را به مدل دادیم تا train شود(حدس زدیم با افزایش تعداد ورودی ها عملکرد بهتر میشود):



filename	liveness_score	liveness_score_crop	liveness_score_frequency
anti-spoof1	0.9104929	0.357545375823975	0.0013139222
anti-spoof2	0.9924036	0.0968254283070564	0.00053528004
anti-spoof3	0.99951804	0.722456216812134	0.0006474526
anti-spoof4	0.97103477	0.741030395030975	0.019697806
anti-spoof5	0.79574317	0.0724463313817978	0.00052970136
anti-spoof6	0.9603984	0.525684416294098	0.0013201268
anti-spoof7	0.87766355	0.61590439081192	0.00065331877
anti-spoof8	0.99957246	0.565047562122345	0.0026581844
anti-spoof9	0.9735333	0.985405147075653	0.0005051418
spoof1	0.2863061	0.269973874092102	0.00318603
spoof2	0.01328704	0.00115578796248883	0.003948256
spoof3	0.17333582	0.000764872005674988	0.0014895209
spoof4	0.0017281512	0.00215042126365006	0.00046497057
spoof5	0.0029267415	0.00304553215391934	0.00038646423
spoof6	0.0032387653	0.0298207011073828	0.0008402442
spoof7	0.0000038684007	0.0998818278312683	0.008412397
spoof8	0.00043647335	0.456633120775223	0.001810181
spoof9	0.00018804775	0.000349557027220726	0.0042497856
spoof10	0.005578859	0	0.001531606
spoof11	0.00071664306	0.105596266686916	0.00004089431

همانطور که قابل مشاهده است میزان دقت مدل برای فریم کامل 100 درصد است و روی دیتای برش خورده از صورت 95 درصد(فقط یک خطا مشاهده می‌شود) است ولی درباره‌ی ورودی دادن فرکانس عکس، طبق پیشبینی خروجی مطلوبی نگرفتیم چون مدل طبق فرکانس آموزش ندیده است.

بخش پنجم - تست:

خروجی این بخش در توضیحات بخش‌های قبلی آورده شده است اما برای پیاده‌سازی آن از چند function استفاده شده است.

تابع calculate_liveness_score برای محاسبه‌ی میزان زنده بودن یک فریم از ویدیو است. برای بدست آوردن فریم ویدیو یک تابع extract_frame داریم که خروجی آن را به عنوان ورودی calculate_liveness_score استفاده می‌کنیم. در این تابع ابتدا سایز تصویر را به اندازه‌ی مدنظر برای ورودی مدل تبدیل می‌کنیم و سپس با صدا زدن predict میزان زنده بودن را خروجی می‌گیریم.

برای دومین قسمت یک تابع recognize_face داریم که چهره را به صورت خودکار crop کند(لوکیشن‌های اولیه‌ی کتابخانه‌ی آماده‌ی face_recognition صورت را به صورت کامل دربر نمیگیرد برای همین با اضافه کردن margin به آن مشکل را حل کردیم). حالت های مختلف margin:

Margin 30:

	filename	liveness_score	liveness_score_crop	liveness_score_frequency
0	/content/drive/My Drive/dataset/anti-spoof1.MOV	0.984466	0.237474	0.179807
1	/content/drive/My Drive/dataset/anti-spoof2.MOV	0.999389	0.226998	0.028669
2	/content/drive/My Drive/dataset/anti-spoof3.MOV	0.999944	0.370844	0.042947
3	/content/drive/My Drive/dataset/anti-spoof4.MOV	0.999586	0.991400	0.498636
4	/content/drive/My Drive/dataset/anti-spoof5.MOV	0.789392	0.973626	0.038251
5	/content/drive/My Drive/dataset/anti-spoof6.MOV	0.982925	0.016312	0.057798
6	/content/drive/My Drive/dataset/anti-spoof7.MOV	0.998245	0.975134	0.150681
7	/content/drive/My Drive/dataset/anti-spoof8.MOV	0.999932	0.200363	0.014293
8	/content/drive/My Drive/dataset/anti-spoof9.MOV	0.997665	0.967868	0.041158
9	/content/drive/My Drive/dataset/spoof1.MOV	0.272792	0.284137	0.216340
10	/content/drive/My Drive/dataset/spoof2.MOV	0.001505	0.008561	0.055270
11	/content/drive/My Drive/dataset/spoof3.MOV	0.009341	0.001098	0.145687
12	/content/drive/My Drive/dataset/spoof4.MOV	0.026444	0.000549	0.037467
13	/content/drive/My Drive/dataset/spoof5.MOV	0.000029	0.001512	0.034880
14	/content/drive/My Drive/dataset/spoof6.MOV	0.013640	0.018912	0.304346
15	/content/drive/My Drive/dataset/spoof7.MOV	0.000096	0.040923	0.752950
16	/content/drive/My Drive/dataset/spoof8.MOV	0.000691	0.007112	0.152022
17	/content/drive/My Drive/dataset/spoof9.MOV	0.000040	0.439637	0.468359
18	/content/drive/My Drive/dataset/spoof10.MOV	0.013138	0.000000	0.300801
19	/content/drive/My Drive/dataset/spoof11.MOV	0.002390	0.860999	0.004026

Margin 80:

	filename	liveness_score	liveness_score_crop	liveness_score_frequency
0	anti-spoof1	0.984466	0.603249	0.179807
1	anti-spoof2	0.999389	0.893173	0.028669
2	anti-spoof3	0.999944	0.960184	0.042947
3	anti-spoof4	0.999586	0.998859	0.498636
4	anti-spoof5	0.789392	0.099786	0.038251
5	anti-spoof6	0.982925	0.131270	0.057798
6	anti-spoof7	0.998245	0.981425	0.150681
7	anti-spoof8	0.999932	0.989856	0.014293
8	anti-spoof9	0.997665	0.999236	0.041158
9	spoof1	0.272792	0.145026	0.216340
10	spoof2	0.001505	0.000200	0.055270
11	spoof3	0.009341	0.000165	0.145687
12	spoof4	0.026444	0.032042	0.037467
13	spoof5	0.000029	0.000063	0.034880
14	spoof6	0.013640	0.028681	0.304346
15	spoof7	0.000096	0.406396	0.752950
16	spoof8	0.000691	0.205022	0.152022
17	spoof9	0.000040	0.000236	0.468359
18	spoof10	0.013138	0.000000	0.300801
19	spoof11	0.002390	0.939898	0.004026

best margins (after multiple tests) :

```
top = max(0, top - 250)
right = min(frame.shape[1], right + 40)
bottom = min(frame.shape[0], bottom + 40)
left = max(0, left - 40)
```

filename	liveness_score	liveness_score_crop	liveness_score_frequency
anti-spoof1	0.9860291	0.886060833930969	0.49011075
anti-spoof2	0.9940369	0.869118928909302	0.37556037
anti-spoof3	0.9995659	0.93857604265213	0.36781752
anti-spoof4	0.995777	0.995321929454803	0.7736416
anti-spoof5	0.8473715	0.358389258384705	0.29056457
anti-spoof6	0.9925096	0.838878989219666	0.44120994
anti-spoof7	0.9947425	0.968680799007416	0.47626716
anti-spoof8	0.9996941	0.962214946746826	0.3116397
anti-spoof9	0.9905934	0.991879165172577	0.3829256
spoof1	0.9893231	0.984255850315094	0.35621613
spoof2	0.08956659	0.0378418117761612	0.6054576
spoof3	0.3818883	0.0272018481045961	0.6291491
spoof4	0.71064746	0.628495931625366	0.2133216
spoof5	0.08808319	0.267544776201248	0.34722653
spoof6	0.087301336	0.86078268289566	0.17002606
spoof7	0.0015884337	0.653576374053955	0.886807
spoof8	0.027212229	0.621666014194489	0.62063575
spoof9	0.05941801	0.152895227074623	0.558425
spoof10	0.096296914	0	0.506841
spoof11	0.07956236	0.770594120025635	0.03254518

که در سومین حالت بهترین نتیجه را گرفتیم.
یک تابع دیگر هم برا frequency extraction داریم تا ویژگی‌های فرکانسی تصویر را دربیاوریم و به عنوان ورودی مدل بدهیم.

منابع:

https://www.ijsr.net/archive/v11i11/SR221103150851.pdf

https://www.mdpi.com/2079-9292/12/10/2199

دیگر منابع سرچ‌های گوناگون از سایت های مختلف بودند که قبل از شروع پروژه انجام دادیم و متاسفانه اطلاع نداشتیم که باید در گزارش ذکر کنیم.