

ادیب رضائی - امیرمحمد خسروی	نام و نام خانوادگی
810198386 - 810198401	شماره دانشجویی
۱۴۰۱.۰۳.۱۴	تاریخ ارسال گزارش



به نام خدا
 دانشگاه تهران
 دانشکده مهندسی
 برق و کامپیوتر



درس شبکه‌های عصبی و یادگیری عمیق
 تمرین امتیازی

فهرست

1

پاسخ ۱. در آپلود دوستمان خواهد بود

1

پاسخ ۲ - تشخیص خشونت در فیلم

شكل‌ها

.1.1

جدول‌ها

پاسخ ۱. در آپلود دوستمان خواهد بود

پاسخ ۲ - تشخیص خشونت در فیلم

۱-۲- دریافت و پیش پردازش دادگان

ابتدا داده‌ها را به روش گفته شده دانلود و unzip می‌کنیم. بعد از آن آرایه‌ای از آدرس ویدیوها ایجاد می‌کنیم تا در آینده از آن برای خواندن فایل‌ها استفاده کنیم همچنین با توجه به اسم فایل‌ها یک آرایه متناظر ایجاد می‌کنیم که label هر ویدیو را در خود نگه می‌دارد. برای فایل‌های با پیشوند fi مقدار ۱ و برای فایل‌های دیگر مقدار ۰ در نظر می‌گیریم.

پس از خواندن فایل‌های ویدیویی داده‌ها را مورد بررسی قرار می‌دهیم. نتایج زیر حاصل بررسی‌ها می‌باشد:

ماکزیمم و مینیمم تعداد فریم‌ها در ویدیوها (اکثر ویدیوها تعداد فریم برابر با 41 دارند):

Maximum number of frames : 49 Image : ./HockeyFights\fi190_xvid.avi

Minumum number of frames : 40 Image : ./HockeyFights\fi157_xvid.avi

ماکزیمم و مینیمم طول و عرض فریم‌های ویدیوها (اغلب ویدیوها $360 * 288$ هستند):

Maximum videos widths : 360 Image : ./HockeyFights\fi100_xvid.avi

Minimum videos widths : 360 Image : ./HockeyFights\fi100_xvid.avi

Maximum videos heights : 360 Image : ./HockeyFights\fi100_xvid.avi

Minumum videos heights : 288 Image : ./HockeyFights\fi100_xvid.avi

با توجه به اطلاعات بالا تعداد فریم‌هایی که میخواهیم از هر ویدیو استخراج کنیم را 20 و اندازه فریم‌ها را نیز $360 * 288$ در نظر می‌گیریم. سپس با استفاده از آرایه حاوی آدرس ویدیوها، آنها را خوانده و در یک آرایه numpy ذخیره می‌کنیم. آرایه مذکور به صورت زیر خواهد بود:

$(1000, 20, 288, 360, 3)$

به این معنا که 1000 داده داریم که هر کدام شامل 20 فریم و هر فریم $360 * 288$ میباشد و دارای 3 کanal نیز هستند.

5 فریم متوالی از یک ویدیو رندوم را نمایش میدهیم. نتیجه به صورت زیر است:



شکل 1. پنج فریم متوالی

حال مراحل Augmentation ذکر شده در مقاله را انجام می‌دهیم: یک تابع remove_dark_edges تعریف میکنیم که حاشیه‌های سیاه تصاویر را حذف می‌کند. نمونه تصاویر ورودی و خروجی این تابع و همچنین اندازه این تصاویر در شکل زیر آمده است:

Original image shape : $(288, 360, 3)$

Preprocessed image shape : $(199, 331, 3)$



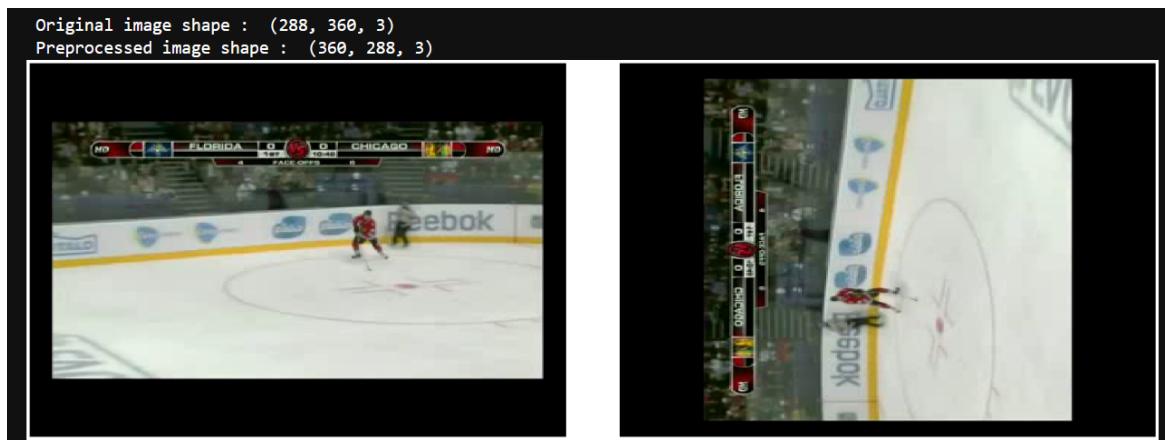
شکل 2. ورودی و خروجی remove_dark_edges

سپس تابعی تعریف می‌کنیم به نام random_crop که به صورت رندوم یکی از 5 مدل مختلف crop را رو تصویر اعمال می‌کند. یک نمونه ورودی و خروجی این تابع به صورت زیر خواهد بود:



شکل 3. ورودی و خروجی random_crop

در نهایت نیز تابع transpose_image را تعریف می‌کنیم که تصویر را transpose می‌کند. نمونه ورودی و خروجی اندازه تصاویر برای این تابع به صورت زیر است:



شکل 4. ورودی و خروجی transpose_image

حال برای هر دو فریم متوالی هر ویدیو (به ازای هر ویدیو 20 فریم داریم) با استفاده از تابع absdiff اختلاف دو تصویر را محاسبه می‌کنیم. در نتیجه به ازای هر ویدیو 10 فریم اختلاف خواهیم داشت. یک نمونه از دو فریم و اختلاف آن ها در شکل زیر نشان داده شده است:



شکل 5. ورودی و خروجی absdiff

در مرحله بعد فریم ها و لیبل ها را شافل میکنیم و روی برخی از آنها random_crop، روی برخی از آنها transpose و روی بعضی دیگر نیز remove_dark_edge اجرا میکنیم. سپس روی تمام زوج فریم های متواالی absdiff اجرا میکنیم تا در نهایت آرایه داده های ما دارای ای به صورت زیر شود:

$$(1000, 10, 200, 330, 3)$$

حجمی که داده ها به خود اختصاص می دهند به صورت زیر است:

Memory space taken by the numpy array: 1980000000 bytes

Memory space taken by the numpy array: 1933593.75 kilobytes

Memory space taken by the numpy array: 1888.275146484375 megabytes

در نهایت داده ها برای ورودی دادن به مدل آماده میشوند. 5 نمونه تصویر مربوط به یک ویدیو در شکل زیر نشان داده شده اند (در تمرین گفته شده است 10 تصویر نمایش داده شوند اما به دلیل کوچک شدن تصاویر و به هم ریختن ترتیب آن ها به 5 تصویر بسنده شده است):



شکل 6. پنج تصویر پیش پردازش شده مربوط به یک ویدیو

2-2- پیاده سازی مدل و آموزش

حال داده ها را که shape آن ها به صورت زیر است را به مدل ResNet50 که از قبل آموزش داده شده است میدهیم. برای این کار در یک حلقه 1000 تایی هر بار 10 فریم مربوط به یک ویدیو را به مدل میدهیم و predict میکنیم:

$$(1000, 10, 200, 330, 3)$$

ورودی مدل ResNet50 به صورت $(3, 200, 330)$ است و خروجی آن نیز به صورت numpy $(10, 7, 11, 2048)$ است که ما 1000 عدد از این نوع خروجی را در یک آرایه قرار میدهیم تا به عنوان ورودی مدل بعد از آن استفاده کنیم. shape نهایی داده به صورت زیر خواهد بود (حدود 6 گیگابایت فضای اشغال میکند):

$$(1000, 10, 7, 11, 2048)$$

سپس مدل ConvLSTM2D را میسازیم. ورودی این مدل به صورت $(10, 7, 11, 2048)$ است. تعداد فیلترهای آن 3 در 3 و دارای $kernel_size = 256$ است. پس از لایه BatchNormalization یک لایه ConvLSTM2D میگیریم.

در این مرحله داده ها را باید با استفاده از flatten به یک بعد کاهش دهیم تا برای ورودی دادن به شبکه fully connected آماده شود. سپس خروجی این لایه را به یک لایه با 1000 نورون میدهیم. لایه بعدی 256 نورون و لایه بعد از آن 10 نورون خواهد داشت. در نهایت نیز یک لایه با یک نورون داریم که لیبل را مشخص خواهد کرد. این لایه softmax activation function است. مدل را با

ذکر شده در صورت تمرین و مقاله یعنی RMSprop و همچنین تابع $\text{loss} = \text{optimizer}$ کامپایل میکنیم. مدل به صورت زیر است:

(None, 7, 11, 256)		
Model: "model"		
Layer (type)	Output Shape	Param #
=====	=====	=====
input_2 (InputLayer)	[(None, 10, 7, 11, 2048) 0]	
conv_lstm2d (ConvLSTM2D)	(None, 7, 11, 256)	21234688
batch_normalization (BatchN ormalization)	(None, 7, 11, 256)	1024
flatten (Flatten)	(None, 19712)	0
dense (Dense)	(None, 1000)	19713000
dense_1 (Dense)	(None, 256)	256256
dense_2 (Dense)	(None, 10)	2570
dense_3 (Dense)	(None, 1)	11
=====	=====	=====
Total params: 41,207,549		
Trainable params: 41,207,037		
Non-trainable params: 512		

ConvLSTM2D model summary .7 شکل

حال داده ها و لیبل ها را به سه دسته validation، train، test تقسیم میکنیم. طبق مقاله تعداد هر کدام از این دسته ها به صورت زیر خواهد بود:

Training set size: 800

Testing set size: 160

Validation set size: 40

مقادیر داده ها نیز به صورت زیر خواهد بود:

800, 10, 7, 11, 2048)

(160, 10, 7, 11, 2048)

(40, 10, 7, 11, 2048)

در این مرحله با epoch 10 learning rate = 0.001 و batch size = 2 همچنین در مدل را train میکنیم.

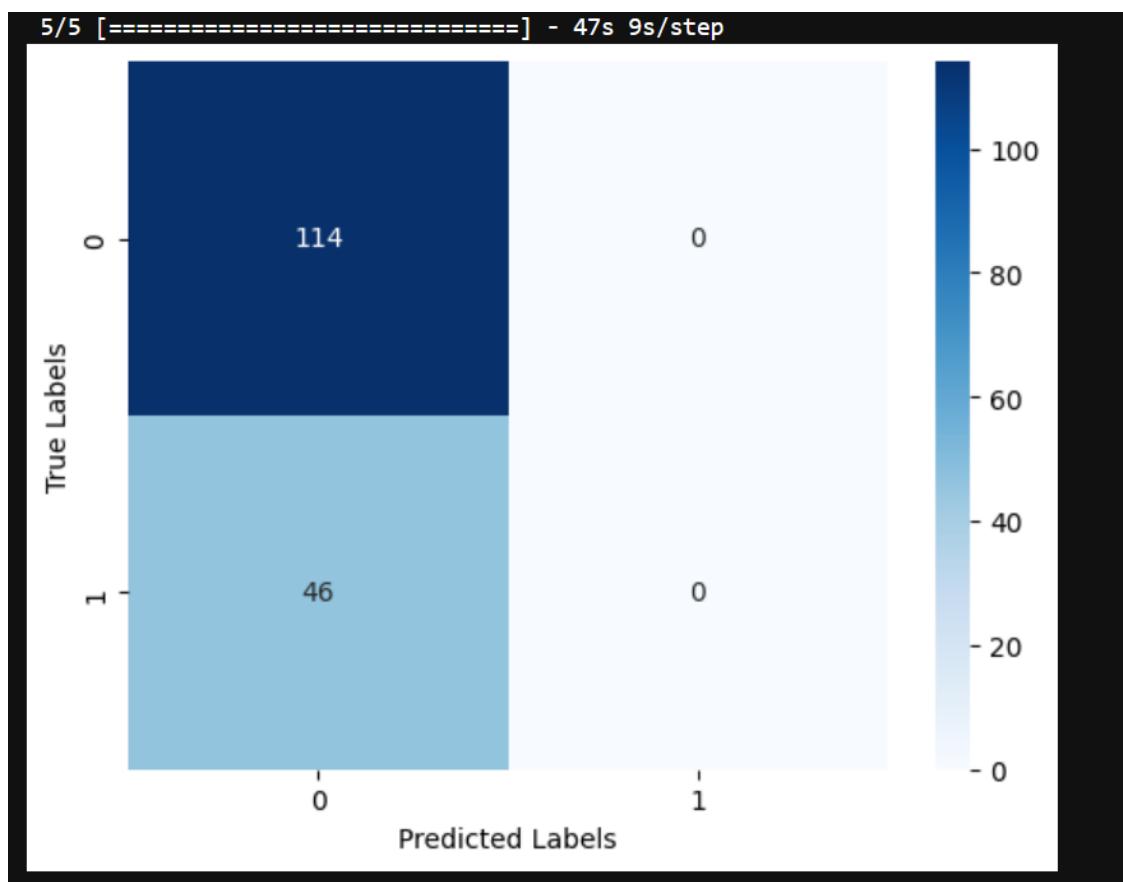
```
Epoch 1/10
400/400 [=====] - 1048s 3s/step - loss: 0.9124 - val_loss: 0.6714
Epoch 2/10
400/400 [=====] - 1014s 3s/step - loss: 0.6524 - val_loss: 0.6577
Epoch 3/10
400/400 [=====] - 1063s 3s/step - loss: 0.6360 - val_loss: 0.6508
Epoch 4/10
400/400 [=====] - 1072s 3s/step - loss: 0.6264 - val_loss: 0.6481
Epoch 5/10
171/400 [=====>.....] - ETA: 10:00 - loss: 0.6104
```

3- نتایج

پس از train کردن مدل آن را evaluate میکنیم و نتایج به صورت زیر است:
مقدار loss روی داده های تست:

```
5/5 [=====] - 48s 10s/step - loss: 0.6014
0.6013756990432739
```

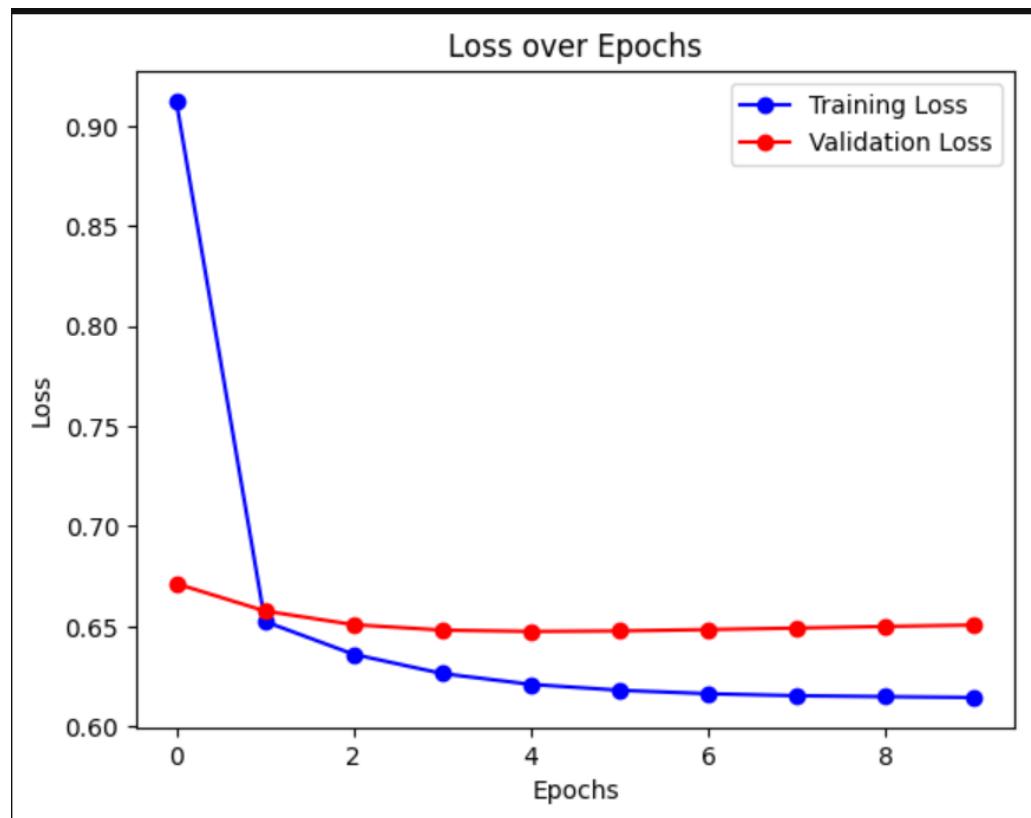
ماتریس در هم ریختگی:



مقدار precision، recall، F1-score:

	precision	recall	f1-score	support
0	0.71	1.00	0.83	114
1	0.00	0.00	0.00	46
accuracy			0.71	160
macro avg	0.36	0.50	0.42	160
weighted avg	0.51	0.71	0.59	160

نمودار خط:



شنبه