به نام خدا

دانشگاه صنعتی امیرکبیر (پلی تکنیک تهران)

دانشکده مهندسی کامپیوتر

شبکه‌های عصبی

تکلیف پنجم

استاد درس: دکتر صفابخش

امیرحسین کاشانی

۴۰۰۱۳۱۰۷۱

amkkashani@gmail.com

نیم سال اول ۱۴۰۱-۱۴۰۲

فهرست

بخش یک

الف)

Data windowing

هنگـامی کـه از مـدل‌هایی اسـتفاده مـی‌کنیم کـه نیازمنـد ورودی گـرفتن چنـد ورودی بـه صـورت همزمـان هسـتند (بـه عبـارت دیگـر چنـد سـطر از اطلاعـات را همزمـان می‌خواهنـد) از Data windowing اسـتفاده مـی‌کنیم. بـه ایـن صـورت کـه N گام مورد نظر کـه در مسـاله مـا N سـطر قبلـی داده مـا هسـتند را بـه عنـوان یـک نمونـه داده جدیـد فـرض مـی‌کنیم و بـه مدل خودمان به عنوان داده آموزش یا تست وارد می‌کنیم.

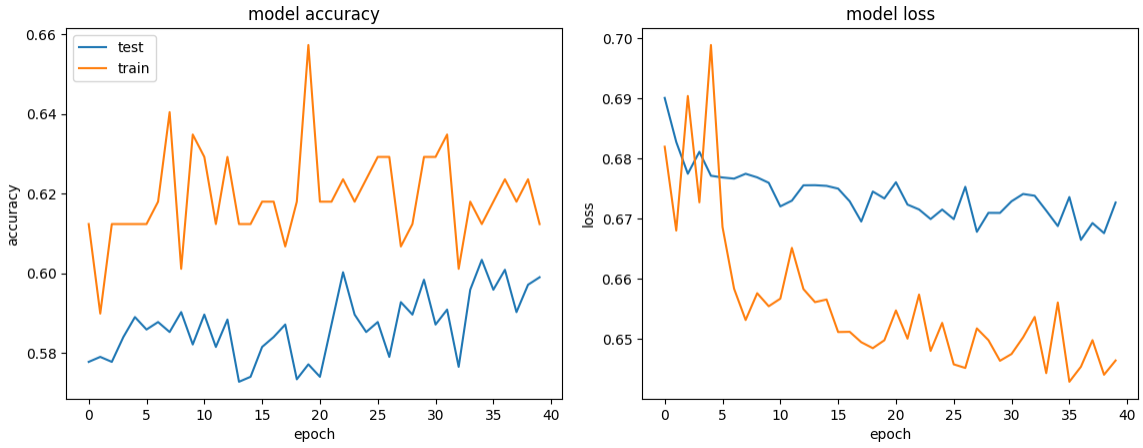
پنجره‌هـای اسـتفاده شـده در ایـن مسـاله overlap دار فـرض شـده انـد. در صـورتی کـه داده بـه مقـدار کـافی وجـود داشـته باشـد می‌تـوان میـزان ایـن overlap را کـاهش یـا بـه صـفر رسـاند امـا باتوجـه بـه اینکـه در ایـن مسـاله تعـداد داده بیشـتر اهمیـت بـالاتری دارد دنبالـه‌های مـا بیشـترین overlap ممکـن را بایـک دیگـر دارنـد بـرای مثـال بـا سـایز پنجـره ۲۰ ، دنبالـه اول ۱ تا ۲۰ ، دنباله دوم ۲ تا ۲۱ و ... می‌باشد.

ب)

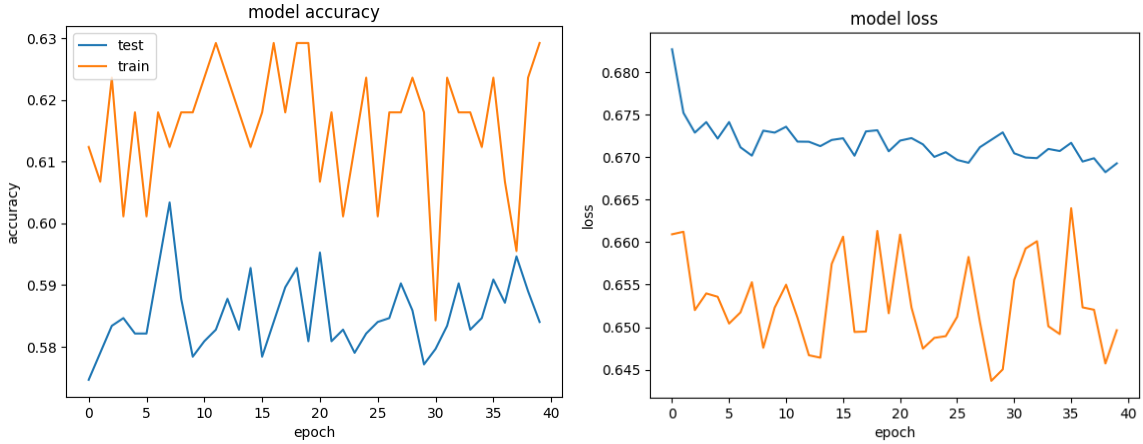
در ایـن بخـش دو مـدل آمـوزش داده شـده اسـت مـدل ابتـدایی بـا Simple RNN بـوده و مـدل دوم بـا اسـتفاده از LSTM بوده است نتایج بدست آمده از هردو مدل تقریبا مشابه یک دیگر بوده وعملکردی نزدیک به ۶۰ درصد داشته است.

** در پیاده سازی لیبل‌ها ۰ و ۱ در نظر گرفته شده است.

Simple RNN

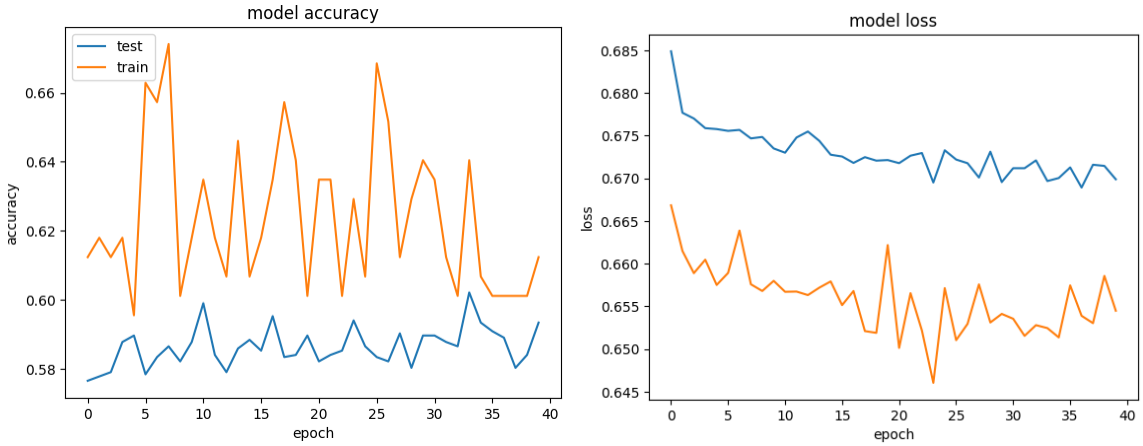


LSTM



ج)

عملکرد شبکه کانولوشنی نیز مشابه بخش قبل بوده اما در داده train توانسته دقت بالاتری بدست آورد که این باتوجه به وجود حدود ۴ هزار متغیر برای یادگیر که دوبرابر مدل‌های قبلی بوده است و فقط روی داده‌های آموزشی می‌باشد چندان ارزشمند نیست و می‌توان گفت که عملکرد مدل‌های در این مساله تقریبا مشابه یکدیگر بوده اند.



بخش دوم

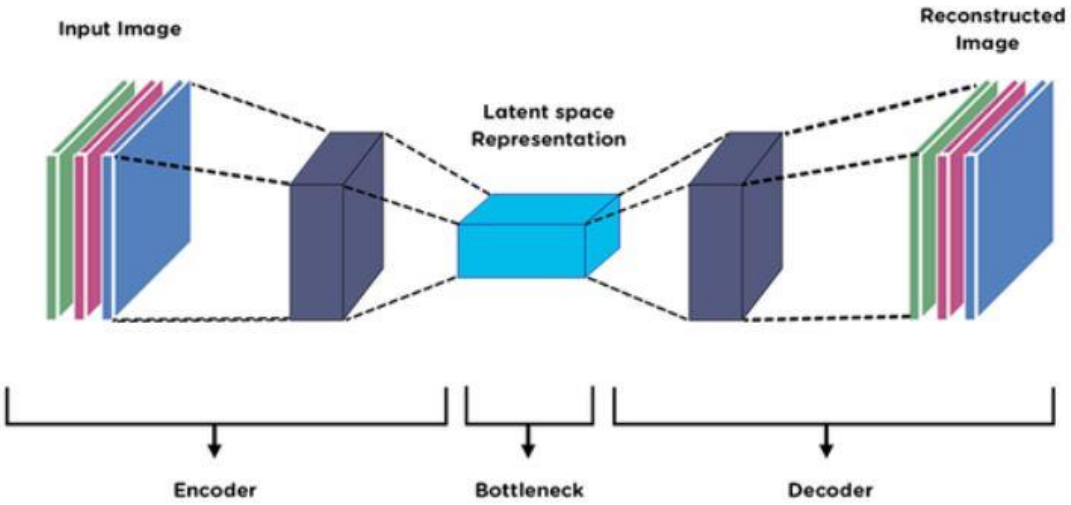
د)

آنومالی به داده یا دنباله ای از داده‌ها یا پترن‌هایی گفته می‌شود که نمی‌توان آن‌ها را با استفاده رفتار معمول سیستم توجیه نمود. بعضا در مواردی به آنومالی داده پرت نیز گفته می‌شود اما رویکرد ما با آنومالی متفاوت است و در خیلی از وظایف هدف تشخیص آن است برخلاف داده پرت که هدف حذف آن است.

آنومالی‌ها را به سه دسته تقسیم می‌کنند که عبارت اند از Additive outlier ,Temporal changes , Level Shift

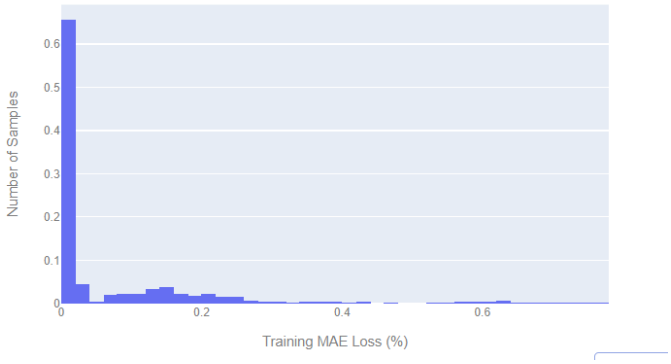
- Additive outlier : رشد یا کاهش ناگهانی در داده‌ها
- Temporal changes: بوجود آمدن ناهنجاری در بازه ای از داده‌ها به دلیلی خاص
- Level shift : تاثیر پذیری از یک عامل بیرونی موجب تغییرات کلی و نسبتا بلند مدت در داده‌های ما بشود

در این مقاله برای تشخیص آنومالی از Auto Encoder/decoder استفاده شده است. در این سیستم مانند تعریف encoder به دنبال ساخت یک بازنمایی (representation) از داده‌های ورودی است که عموما این بازنمایی در ابعاد کوچک تری از داده ورودی می‌باشد(مانند شکل زیر). از طرف دیگر decoder به دنبال ساخت داده اولیه از روی بازنمایی ساخته شده توسط encoder هست. در نهایت این مجموعه در کنار یک دیگر آموزش می‌بینند به طوری که این سیستم سعی دارد تا با ورودی گرفتن X، همان X را در خروجی ایجاد کند و خطای مدل نیز از اختلاف این دو بدست می‌آید.



در این تسک با استفاده از بازه‌های زمانی قصد داریم تا داده‌های آنومالی را تشخیص دهیم و به این صورت عمل می‌شود که پنجره‌های ۳۰ تایی پس از نرمال سازی از داده‌ها تشکیل می‌شود و به عنوان ورودی به مدل وارد می‌گردد تا مدل باید بتواند همان ورودی را در خروجی ایجاد کند. به عنوان معیار شباهت از MAE استفاده شده است.

شکل زیر توزیع به دست آمده از خطای بازنمایی داده‌های آموزش را بیان می‌کند. برای تشخیص داده‌های آنومالی هر دنباله ای که خطای reconstruction آن یا همان فاصله خروجی آن تا خودش، بیشتر از ماکسیمم فاصل موجود در داده‌های آموزش باشد را به عنوان آنومالی اعلام می‌کند. به بیان دیگر ما یک مدل ایجاد کردیم که می‌تواند داده‌های که عادی هستند (آنومالی نیستند) را به خوبی encode و decode کند پس هر جا که به خوبی کار نکند متوجه می‌شویم که داده ی ورودی آنومالی است.



تحلیل مدل

مدل به کار رفته برای encode , decode به شکل زیر است که به راحتی می‌توان حالت تقارن بین encoder , decoder را در آن دید.

```
21    model = keras.Sequential(
22        [
23         layers.Input(shape=(num_steps, num_features)),
24         layers.Conv1D(filters=32, kernel_size = 15, padding = 'same', data_format=
      'channels_last',
                       dilation_rate = 1, activation = 'linear'),
26         layers.LSTM(units = 25, activation = 'tanh', name = 'LSTM_layer_1',return_sequences=
      False),
27         layers.RepeatVector(num_steps),
28         layers.LSTM(units = 25, activation = 'tanh', name = 'LSTM_layer_2', return_sequences=
      True),
29         layers.Conv1D(filters = 32, kernel_size = 15, padding = 'same', data_format =
      'channels_last',
30                       dilation_rate = 1, activation = 'linear'),
31         layers.TimeDistributed(layers.Dense(1, activation = 'linear'))
32        ]
33    )
```

ه)

در این مقاله با اســتفاده از یک روش self_supervise با بهره گیری ازauto encoder ها قصـد دارد تا بخش‌های mask شـده یا پوشـانده شـده از تصـویر را بازیابی نمایید. برای این کار از یک معماری نامتقارن اسـتفاده شـده اسـت که در آن بخش encoder پیچیده تر و بخش decoder سبک تر فرض گردیده است. عکس ورودی به بخش‌هایی با سایز مساوری تقسیم می‌شود و بصورت رندوم (در بخش آموزش) بخش‌هایی از آن ماسـک می‌شـود. در بخش آموزش ۷۵ درصـد بخش‌ها را ماسـک کرده اسـت،encoder فقط بخش‌هایی که دارای داده هستند و ماسک نشده‌اند را به عنوان ورودی می‌گیرد ، این کار باعث کم شدن بار محاسباتی و بهبود سه برابری سرعت یادگیری می‌شود. بعد از اینکه encoder برداری با طول تعداد patchها ماسک نشده ساخت مقادیربدست آمده در مکان مربوطه در تصویر جایگذاری شده (در زمانی که داده ها به encoder وارد می شود یک shuffle صورت می گیرد و وقتی از encoder خارج می گردد unshuffle صورت می گیرد) و با مابقی نقاط ماسک شده به عنوان ورودی به decoder داده می‌شوند و decoder با توجه به مفاهیم اسـتخراج شـده از encoder و اطلاعات مکانی بدسـت آمده از جایگذاری آن‌ها باید تصـویر اصـلی را بازنمایی کند تا به این صـورت بتواند تصـویر اصـلی را بدسـت آورد. در این مدل بخش decoder مسـتقل از مدل بوده و بعد از فاز پیش آموزش از مدل جدا خواهد گردید. این موضــوع که بعد از جداشــدن decoder چه می کنیم در مقاله به شــکل کامل حل و بحث نشده است و صرفا بیان شده وبیشتر تمرکز بر روی بخش پیش آموزش بوده‌است.