تمرین چهارم پایتون
شبکه‌های عصبی بهار ۱۴۰۲

سارا قوام‌پور ۹۸۱۲۷۶۲۷۸۱

هدف این تمرین بررسی شبکه‌هایی که است که بدون ناظر عمل می‌کنند. در سوال ۱ ابتدا عملکرد دو شبکه اتوانکدر و شبکه‌تخاصمی پیچشی بر روی دیتاست cifar10 بررسی می‌شود و سپس شبکه‌های آموزش دیده‌شده بر روی دیتا به نام real در سوال ۲ بر روی دیتا جدید به نام painting تست می‌شوند و عملکرد این شبکه‌ها در رویارویی به داده از domain جدید پس از آموزش بررسی می‌شود. داده های real و painting متعلق به دیتاست domainnet هستند.

بخش اول

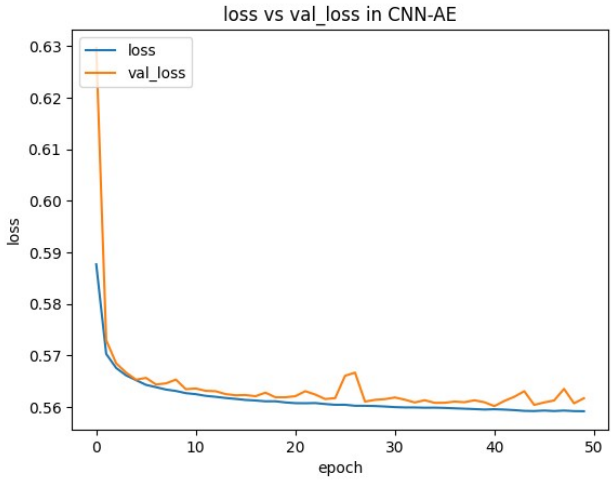
در این بخش ابتدا دو شبکه CNN-AE و DCGAN پیاده سازی می‌شوند.

الف) CNN-AE

این شبکه از یک بخش encoder تشکیل شده است که یک عکس را دریافت کرده و با استفاده از تعدادی لایه convolution ویژگی های آن را استخراج کرده و یک فضای latent می‌سازد. در ادامه برای decoder که از فضای latent سعی بر بازسازی تصویر دارد و یک تابع هزینه برای reconstruction تعریف شده است. ساختار انکدر شامل سه لایه پیچشی با فیلتر سایز ۳ و padding=same و تابع فعالیت relu می‌باشد. دولایه اول هر کدام ۳۲ فیلتر استفاده شده است و لایه سوم از ۶۴ فیلتر استفادع شده است. درنهایت این خروجی flatten میشود و این خروجی انکدر و به عبارتی فیچر های فضای نهان ورودی است.

ساختار دیکدر ابتدا ورودی که از فضای نهان تولید شده توسط انکدر دریافت می‌کند را reshape می کندبه نحوی که با استفاده از سه لایه برعکس پیچشی که هر کدام سایز را ۲ برابر میکنند در نهایت به سایز اولیه خود عکس های دیتاستcifar10 که ۳۲ است برسد. به این منظور هر لایه برعکس پیچشی دارای سایز فیلتر ۳ و stride=2 و padding=same می‌باشد و تابع فعالیت این ۳ لایه نیز relu است. بعد از هر لایه برعکس پیچشی به سبب استفاده از relu برای نگاه داشتن مقادیر در بازه ای خاص از Batch Normalization استفاده شده است. در انتها نیز یک لایه پیچشی به عنوان لایه آخر دیکدر با تابع فعالیت sigmoid استفاده می‌شود چون عکس ورودی در پیش‌پردازش به بازه ۰و۱ برده شده است پس انتخاب این تابع به سبب اینکه خروجی دیکدر باید مشابه ورودی باشد مناسب است. با آموزش اتوانکدر روی دیستاست اموزشی مقدار تابع هزینه ۰.۴۱۶۶ است و روی دیتاست validation مقدار تابع هزینه ۰.۴۲۰۴ در انتها آموزش می‌باشد. آموزش با batch_size=128 , epochs=50 انجام شد.

برای اتوانکدر از بهینه‌ساز adam و برای تابع هزینه از binary crossentropy به این دلیل که ورودی ها بین ۰و۱ هستند استفاده شده است.

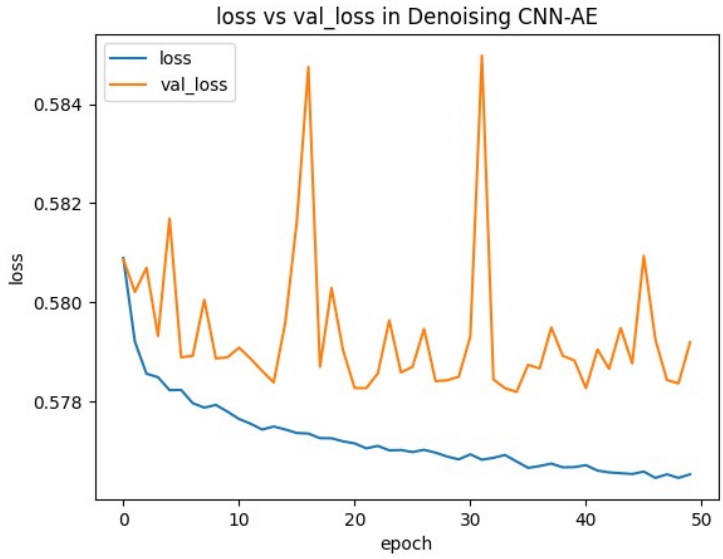


شکل ۱: مقدار تابع هزینه اتوانکدر روی داده cifar

شکل ۲: ردیف اول ۱۰ تصور واقعی از دیتاست test و ردیف دوم بازسازی شده این تصاویر توسط اتوانکدر است.

در ادامه classification نیز با استفاده از فیچر های که انکدر خروجی می‌دهد انجام‌شده‌است. از طبقه بند random forest استفاده شده است به این صورت که ابتدا این طبقه بند بر روی دیتاست آموزش fit شد. به این منظور انکدر بر روی دیستاست آموزش استفاده شده است و طبقه بند بر روی این فیچر ها فیت شده است. برای تست طبقه بند نیز ابتدا فیچر های داده های تست را با کمک انکدر دریافت کرده و بعد متد predict طبقه بند فراخوانی میشود. اکیورسی بر روی داده تست با طبقه بند random forest بر روی داده تست ۰.۴۷۶۶ می‌باشد.

در ادامه اثر بررسی وارد کردن نویز بر روی داده های ورودی شبکه اتوانکدر که همان denoising autoencoder است بررسی می‌شود. به داده های ورودی تنها مقداری نویز گوسی اعمال شده است و لیبل ها تغییری نکرده‌اند. با آموزش اتوانکدر روی دیستاست اموزشی مقدار تابع هزینه ۰.۵۷۶۲ است و روی دیتاست validation مقدار تابع هزینه ۰.۵۷۹۵ در انتها آموزش می‌باشد. آموزش با epochs=50 , batch_size=128 انجام شد.



شکل ۳

در طبقه بندی با استفاده از فیچر های تولید شده توسط انکدر آموزش دیده با داده های نویزی اکیورسی بر روی داده تست با طبقه بند random forest بر روی داده تست ۰.۴۹۸۳ می‌باشد. مشاهده می‌شود که اضافه کردن نویز منجر به یادگیری فیچر های بهتری شده‌است. که در نتیجه به اکیورسی بیشتر طبقه بند منتهی شده است.(افزایش از ۰.۴۷۶۶)

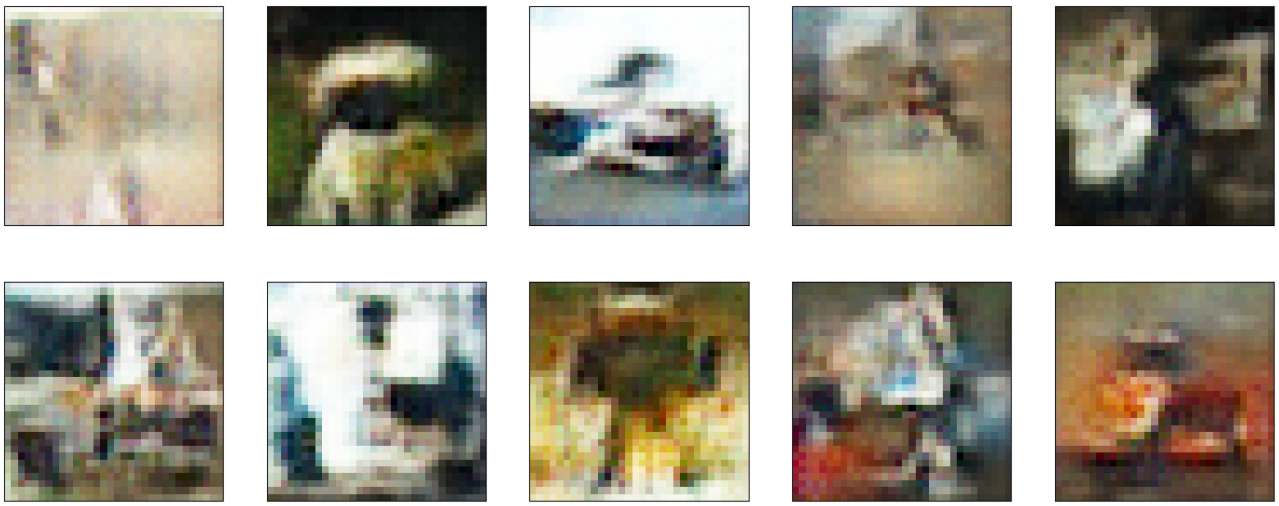
ب) DCGAN

در قسمت بعدی شبکه DCGAN را پیاده سازی می‌کنیم. برای این کار ابتدا دو شبکه generator و descriminator را ایجاد کرده که به ترتیب مسئول ساخت عکس از یک بردار رندم و تشخیص عکس واقعی از غیرواقعی هستند.

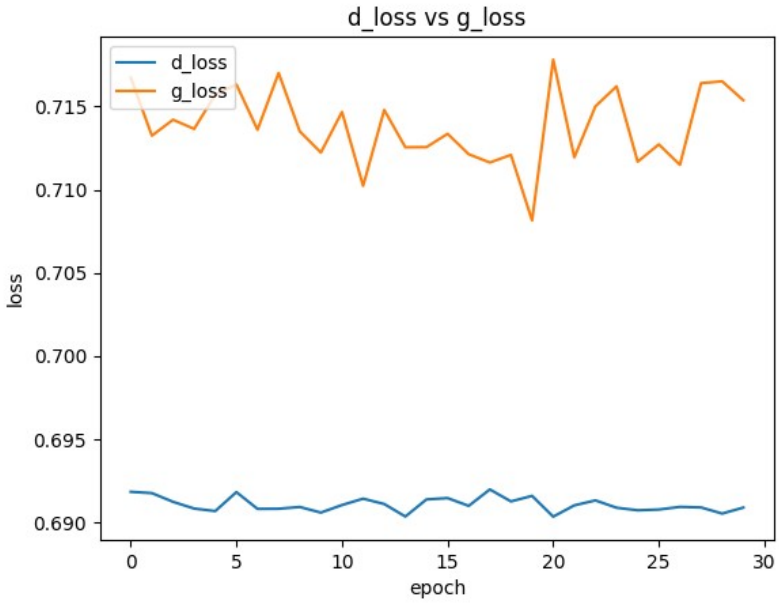
ساختار شبکه descriminator از ۴ لایه پیچشی با فیلتر سایز ۴ تشکیل شده است و تعداد فیلتر ها در هر لایه ۲ برابر می شود. برای تابع فعالسازی نیز از leaky relu استفاده شده است. سپس یک لایه flatten وجود دارد که خروجی این لایه همان فیچر های به دست‌آمده توسط این شبکه است. لایه انتهایی یک لایه dense با یک نورون و تابع سیگموید برای طبقه بندی دو تایی است.

ساختار شبکه generator نیز یک نویز با بعد ۱۰۰ دریافت می‌کند ان را به بعد ۴۰۹۶ برده و بعد ابعاد ان را به سه بعد تغییر داده و بعد از ۳ لایه معکوس پیچشی با leaky relu استفاده میکند تا عکسی با ابعاد ۳*۳۲*۲۳ تولید کند در لایه انتهایی نیز که یک لایه پیچشی است از tanh استفاده شده است چون در کد مربوط به DCGAN عکس ها به جای ۰و۱ به ۱- و ۱ مپ شده‌اند.

در آموزش هم ابتدا descriminator و بعد generator تعلیم می‌بینند. تابع هزینه مطرح شده در کلاس برای descriminator در واقع همان binary crossentropy است که لیبل ها و عکسای مربوط به عکس های واقعی و عکس های تولید شده gan در آن با ترتیب درست قرار گرفته باشند. تابع هزینه برای generator نیز به همین صورت است با لیبل تماما ۱ و خروجی descriminator بر روی عکس های توسط خودش.



شکل ۴: عکس های تولید شده توسط DCGAN



شکل ۵ :خطای دو شبکه generator و descriminator

طبق شکل۵ ، خطاهای دو شبکه generator و descriminator برخلاف هم عمل می‌کنند.

در طبقه بندی با استفاده از فیچر های استخراج شده از یک لایه قبل از لایه آخر descriminator با داده های آموزش، اکیورسی با طبقه بند random forest بر روی داده تست ۰.۴۵۳۹ می‌باشد. مشاهده می‌شود که فیچر های یاد گرفته شده توسط اتوانکدر استاندارد و اتوانکدر همراه با داده های نویزی به اکیورسی بیشتری از فیچر های یاد گرفته شده توسط DCGAN در طبقه بند رسیده‌اند. (کاهش از ۰.۴۷۶۶)

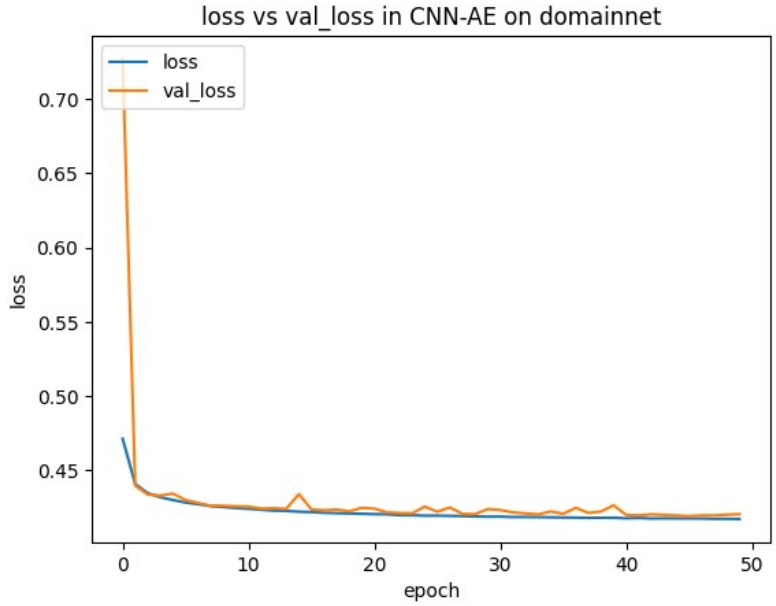
بخش دوم

در این بخش عملکرد شبکه های ساخته شده در بخش یک را بر روی دیتاستی که شامل عکس ها با domain متفاوت است بررسی می‌کنیم. دیتاست domainnet از domain های متفاوتی تشکیل شده است که در اینجا از داده های real و داده های paint آن استفاده می‌شود.

به دلیل تفاوت در سایز عکس‌های این دیتاست با ورودی شبکه اتوانکدر عکس ها در هنگام لود شدن پیش پردازش و تغییر سایز پیدا می‌کنند.

الف) CNN-AE

برای اتوانکدر داده ها بین ۰ و ۱ مپ می‌شوند و به علت کم بودن تعداد داده از ۱۰ کلاس دیتاست بای داده هارا آگمنت کرد. ساختار شبکه های انمکدر و دیکدر مشابه بخش قبل می‌باشد.



شکل ۶

شکل ۷: ردیف اول ۱۰ تصور واقعی از دیتاست painting و ردیف دوم بازسازی شده این تصاویر توسط اتوانکدر آموزش داده شده با دیتاست real است.

پس از آموزش دادن اتوانکدر بر روی داده های real ، طبقه‌بند random forest را بر روی فیچر های دیتاست real که توسط انکدر به دست آمده است، فیت کرده و سپس طبقه بند را بر روی دیتاست painting تست کرده و به اکیورسی ۰.۱۳۱۶ می‌رسیم.

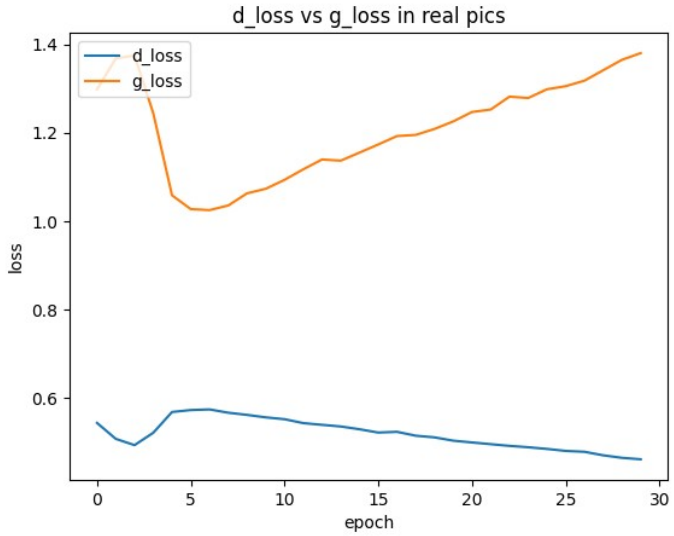
(اکیورسی روی domain جدید بدون آموزش)

دقت شود که در هنگام ساختن طبقه‌بند به آن مقدار warm_start=True داده‌شده‌است که باعث می‌شود طبق‌بند بتواند با داده‌های جدید نیز به درخت خود اضافه کند و به عبارتی دانش جدید از domain جدید را به دانش قبلی اضافه کند.

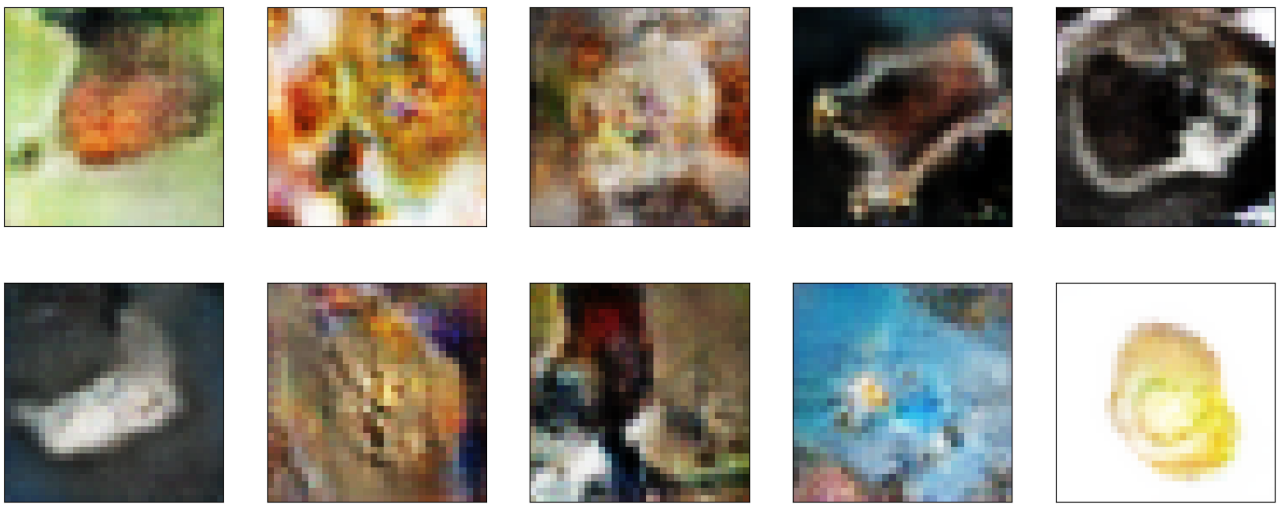
برای مقداری آموزش، ابتدا دیتاست painting به قسمت های آموزش و تست تقسیم می‌شود، سپس همان مدل قبلی طبقه بنـد روی این دیتا آموزشـی نقاشـی هم fit می‌شـود. اکیورسـی بـر روی داده تسـت نقاشـی در این حـالت ۰۱۳۷۵۵ اسـت. مقـداری افـزایش در اکیورسی تست روی domain جدید با کمی آموزش صورت گرفته است اما مقدار ناچیزی است.

ب) DCGAN

از شبکه DCGAN بخش قبلی استفاده می‌کـنیم. در ابتـدا شـبکه را بـر روی داده هـای real آگمنت شـده بین ۱- و ۱ قـرار دارنـد آموزش داده می‌شود.



شکل ۸



شکل ۹: تصاویر generate شده توسط شبکه generator آموزش داده شده بر روی داده های real.

برای classification از یک random forest classifier استفاده شده است. ابتدا فیچر های داده های real را با استفاده از مدل ویژگی که ویژگی های یادگیری شده در discriminator است را استخراج می‌کنیم و سپس random forest را با استفاده از آن آموزش می‌دهیم. پارامتر warm_start را برابر با True و n_estimators را نیز برابر با ۱ قرار می‌دهیم. این کار برای این است که بتوانیم با استفاده از fit، آموزش را با داده های paint ادامه دهیم و از ابتدا دوباره یادگیری انجام نشود.

قبل از آموزش بر روی داده های paint، دقت مدل را بررسی می‌کنیم که همانند autoencdoer در ابتدا ۱۰ درصد است که بیانگر رندم بودن است. در ادامه داده های paint را به دو قسمت train و test تقسیم می‌کنیم و با استفاده از داده های train مدل random_forest را بر روی این داده ها نیز fit می‌کنیم و در نهایت روی داده های test دقت را می‌سنجیم. مشاهده می‌شود که دقت به ۱۵ درصد افزایش پیدا می‌کند.