

# گزارش تمرین هفتم درس رایانش عصبی و یادگیری عمیق

تمرین هفتم:

آشنایی با شبکه‌های GAN و CGAN

نام دانشجو: مجید ادیبیان

شماره دانشجویی: ۴۰۰۱۳۱۰۷۸

نام استاد درس: دکتر صفابخش

زمستان ۱۴۰۱

## بخش اول (سوالات تئوری):

**الف)** در شبکه‌های مولد تقابلی در واقع سعی می‌شود که توزیع داده‌ها یادگرفته شود. برای این منظور یک توزیع اولیه ساده ورودی در نظر می‌گیریم و شبکه عصبی سعی می‌کند این توزیع را به توزیع داده‌های اصلی تبدیل کند. برای این منظور از یک توزیع ساده مانند گوسی هر بار به صورت تصادفی نمونه برداری می‌شود و شبکه عصبی این نمونه را به توزیع داده‌های عصبی تبدیل کرده و داده نهایی (مانند تصویر نهایی) ساخته می‌شود که این نمونه تصادفی از توزیع اولیه همان نویز ورودی به شبکه مولد تقابلی است.

با توجه به مطالب گفته شده مشخص است که تغییر نوع توزیع در نویز ورودی باعث می‌شود شبکه وزن‌های متفاوتی برای تبدیل آن به توزیع داده‌های اصلی یادبگیرد و بسته به نوع داده‌های نهایی که می‌خواهیم تولید کنیم نوع توزیع اولیه می‌تواند کار شبکه را سخت‌تر یا ساده‌تر کند و در نتیجه زمان آموزش شبکه بیشتر یا کمتر شود.

**ب)** توضیح موارد گفته شده به شرح زیر است:

**Modal Collapse:** شبکه مولد تقابلی از این نظر دارای محدودیت هستند که تنها بر روی داده‌های مربوط به یک دسته یا گروه (single modal) می‌توانند آموزش ببینند و در اثر آموزش بر روی داده‌های مربوط به چند دسته، آموزش آن‌ها دچار مشکل می‌شوند و در این حالت، مدل بدون توجه به ورودی تنها چند نمونه مشخص را تولید می‌کند و با این کار متمایزگر را فریب می‌دهد.

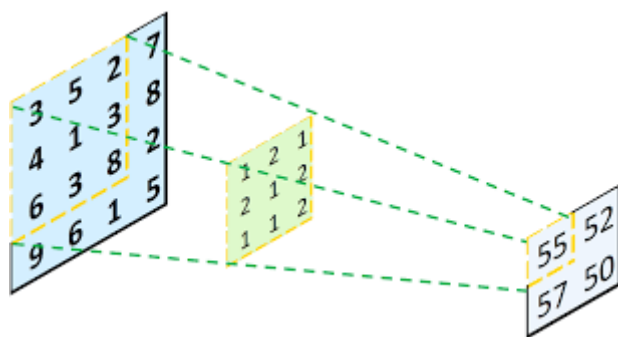
راه حل: اولین راه کاری که برای حل این مشکل به نظر می‌رسد استفاده از گروه‌بندی داده‌ها است که باعث می‌شود که متمایزگر با در نظر گرفتن گروه داده‌ها تشخیص را در زیر گروه‌ها انجام دهد. همچنین می‌توان برای هر گروه از داده‌ها یک شبکه مولد تقابلی مجزا آموزش داد که البته زمان و هزینه محاسباتی بیشتری دارد و نتایج به اندازه کافی مطلوب نخواهد بود. راه کار دیگری که می‌توان برای حل این مشکل انجام داد، آموزش متمایزگر بر روی داده‌های مصنوعی و قدیمی است که توسط مدل مولد ساخته شده است.

**diminished gradient:** این مشکل به معنای آن است که در طی آموزش شبکه مولد تقابلی و به خصوص در قدم‌های اولیه‌ی آن متمایزگر به سادگی می‌تواند نمونه‌های واقعی را از نمونه‌های غیر واقعی تشخیص دهد و در نتیجه احتمالی که برای نمونه‌های ساختگی در نظر می‌گیرد نزدیک به صفر است که با توجه به رابطه خطا در این شبکه‌ها، باعث می‌شود تغییرات گرادیان در مدل مولد نزدیک به صفر باشد و مدل به خوبی آموزش نبیند.

راه حل: برای حل این مشکل می توان رابطه محاسبه خطا در قسمت مولد را به رابطه زیر تغییر داد که در آن خطا به گونه ای محاسبه می شود که سعی می شود مقدار آن به جای مینیمم ماکسیمم شود و در نتیجه در محاسبه گرادیان به جای حرکت در جهت نزول گرادیان در جهت افزایش گرادیان انجام می شود.

$$-\nabla_{\theta g} \log \left( 1 - D \left( G(z^i) \right) \right) \rightarrow \nabla_{\theta g} \log \left( D \left( G(z^i) \right) \right)$$

**ج) شبکه DC-GAN** هم در قسمت مولد و هم متمایزگر از شبکه های عمیق کانولوشنی استفاده می کند. قسمت متمایزگر شامل لایه های کانولوشنی strid, batch normalization و تابع فعال سازی LeakyRelu است به طوری که در طی لایه ها به تدریج تعداد کانال ها افزایش می یابد و ابعاد هر کانال کاهش دارد تا در نهایت یک دسته بندی انجام می شود تا تشخیص داده واقعی و مصنوعی انجام شود. قسمت مولد نیز شامل لایه های کانولوشنی ترانهاده، batch normalization و تابع فعال سازی LeakyRelu است به طوری که از یک بردار نویز ورودی به مرور در طی لایه ها تعداد کانال ها کاهش و ابعاد هر کانال افزایش می یابد تا در نهایت به ابعاد یک تصویر برسیم. کانولوشن ترانهاده: در این نوع کانولوشن به جای آن که یک کرنل بر روی ماتریس لغزنده شود و درایه به درایه در هم ضرب و جمع شوند و یک مقدار از آن به دست آید، به نوعی عکس این عمل انجام می شود. در این حالت هر درایه از ماتریس در تک تک درایه های کرنل ضرب می شود و هر بار یک مقدار تولید می شود که در نتیجه این عمل، هر درایه از ماتریس به چند درایه به اندازه کرنل تبدیل می شود و عملاً افزایش بعد صورت می گیرد. تصویر این فرایند در شکل زیر دیده می شود که در آن ماتریس دو در دوی سمت راست به ماتریس چهار در چهار سمت چپ تبدیل شده است:



**تولید تصویر از متن با DC-GAN:** برای تولید تصویر از متن ورودی با استفاده از شبکه ی DC-GAN ابتدا متن ورودی کد می شود و برداری از آن ساخته می شود. سپس این بردار با بردار نویز تولیدی از توزیع کنار هم قرار

می گیرند و بردار بزرگتری را می سازند. در ادامه این بردار به چند لایه کانولوشن ترانهاده داده می شود تا در نهایت از بردار ورودی تصویر متناسب با آن ساخته شود.

**د)** در آموزش شبکه های مولد و تمایزگر در هر قدم از آموزش شبکه ابتدا مدل تمایزگر  $n$  بار بر روی یک batch از داده ها آموزش می بیند و سپس مدل مولد یک بار آموزش می بیند و این فرایند در هر قدم تکرار می شود. در هر بار آموزش قسمت تمایزگر ابتدا به تعداد یک batch داده مصنوعی که به کمک مدل مولد ساخته می شود (بدون تغییر وزن های مدل مولد) به همراه یک batch از داده های واقعی به تمایزگر داده شده و  $loss$  حساب می شود و وزن هایش به روز می شوند. در آموزش قسمت مولد هم به کمک یک batch داده نویزی یک batch داده مصنوعی تولید می شود و سپس این داده ها به متمایزگر داده می شود و خطای تشخیص مصنوعی بودن آن ها به دست می آید (بدون تغییر وزن مدل متمایزگر) و این خطا به عنوان خطای مدل مولد استفاده شده و وزن های مدل به روز می شود.

## بخش دوم (پیاده سازی):

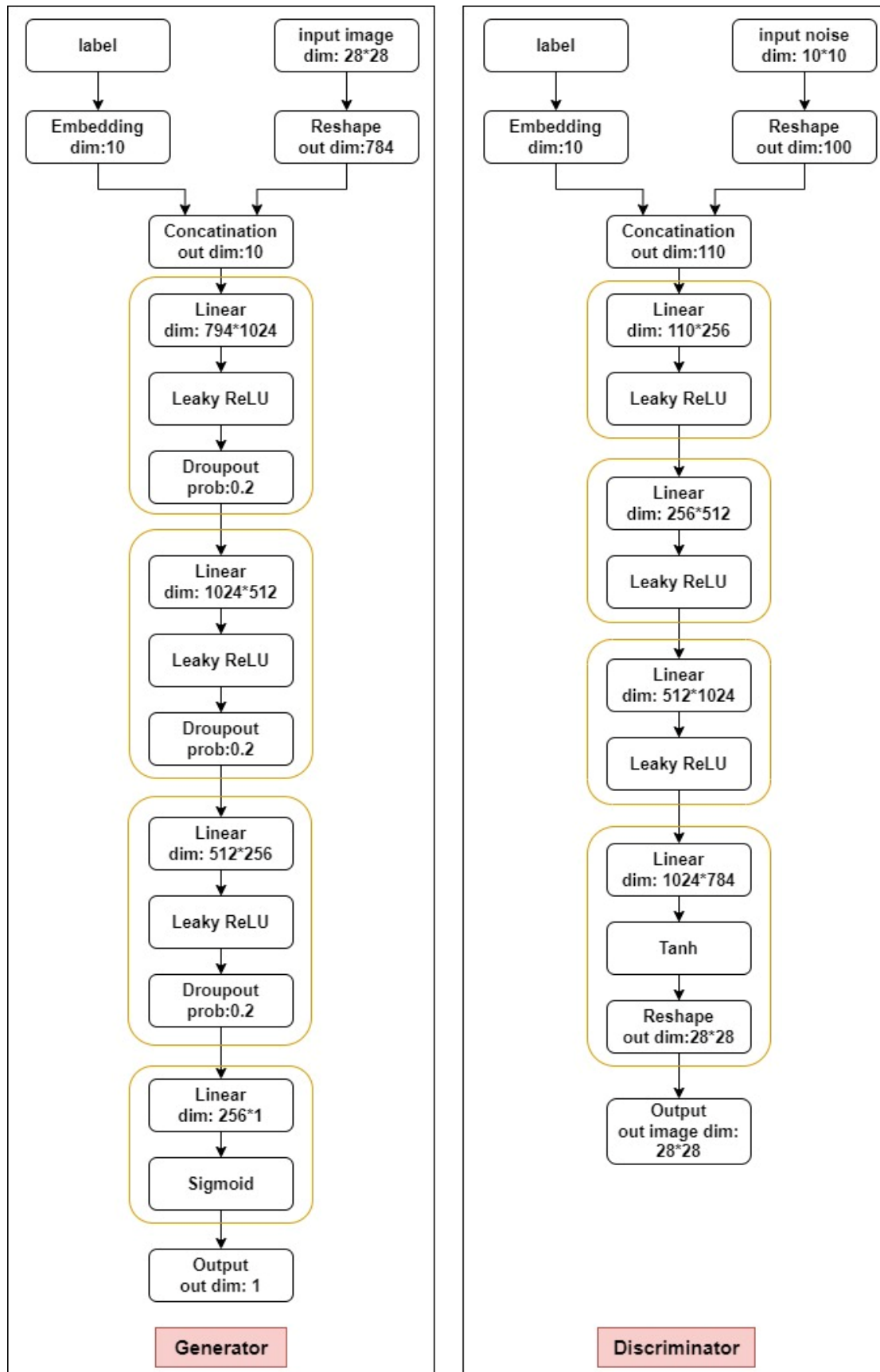
۱) فرض کنیم داده ورودی یک تصویر است که بسته به رنگی بودن یا نبودنش دارای یک یا سه کانال می باشد. همچنین برچسب این داده یک عدد است که می تواند به بردار one-hot تبدیل شود. این به این معنی است که اگر مثلاً  $N$  برچسب متفاوت داریم می توان برداری  $N$  تایی داشت که یکی از درایه هایش یک و بقیه درایه هایش صفر است. سپس می توان با یک لایه dense این بردار را به برداری با طول  $h*w$  تبدیل کرد که  $h$  طول تصاویر و  $w$  عرض تصاویر است. در انتها بردار حاصل به یک ماتریس دو بعدی reshape می شود و یک ماتریس  $h$  در  $w$  ساخته می شود. حال باید این ماتریس را با تصویر حاصل ترکیب کرد تا هر یک از ماژول های generator و discriminator برچسب داده را هم در نظر بگیرند که برای این کار می توان این ماتریس را به عنوان یک کانال دیگر به تصویر ساخته شده یا واقعی اضافه کرد. همچنین در صورتی که هر تصویر به یک بردار تبدیل می شود، می توان برچسب آن تصویر پس از تعبیه سازی به آن concat شود و سپس فرایند گفته شده طی خواهد شد.

۲) در قسمت تمایزگر به صورت صریح برچسب داده در نظر گرفته نمی شود بلکه مانند قبل سعی می کند حقیقی یا ساختگی بودن داده ورودی خود که این بار ترکیب تصویر و برچسب است را تشخیص دهد. این موضوع باعث می شود به صورت ضمنی برچسب داده هم در تشخیص لحاظ شود. از طرفی چون برچسب داده های real با خود داده ها مطابقت دارد، قسمت مولد سعی می کند داده ای تولید کند که به برچسبش مطابقت داشته باشد تا متمایزگر را به اشتباه بیندازد.

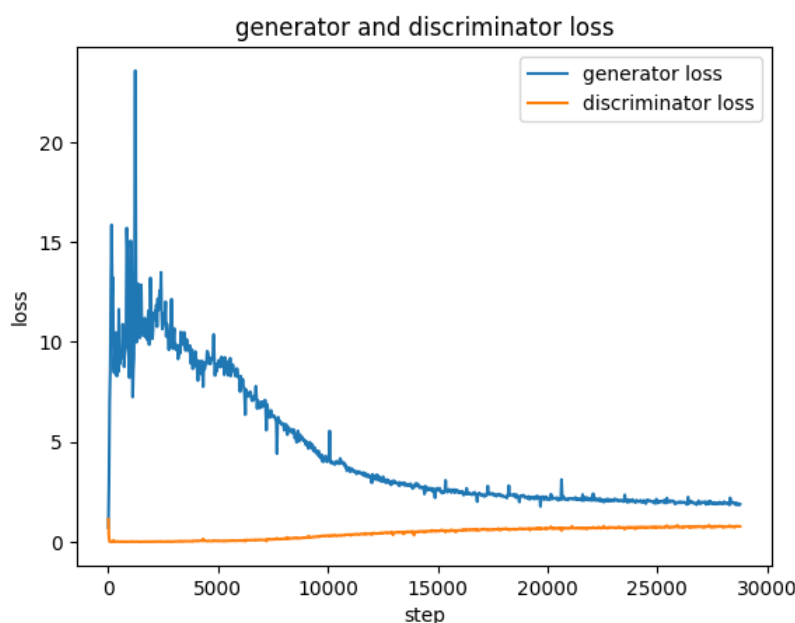
۳) توضیح پیاده‌سازی مدل: در پیاده‌سازی مورد نظر از `pytorch` استفاده شده است. ابتدا داده‌های MNIST از کتابخانه `torchvision.datasets` اضافه شده است و مقادیر موجود در این داده‌ها `transform` شده‌اند تا میانگین و انحراف معیار ۰.۵ داشته باشند. سپس `dataloader` با `batch size` برابر ۱۲۸ ساخته شده است.

در قدم بعد مدل‌های متمایزگر و مولد ساخته شده‌اند. در پیاده‌سازی مدل متمایزگر ابتدا از چند لایه کانولوشن استفاده شده که نتایج مطلوبی به دست نیامد و سپس تنها از لایه‌های خطی استفاده شده است. گراف این دو شبکه به صورت زیر هستند. همان‌طور که دیده می‌شود در قسمت متمایزگر ابتدا تصویر ورودی به بردار تبدیل شده است و برچسب آن نیز تعبیه شده است و سپس این دو بردار با هم `concat` شده‌اند. در قسمت مولد یک نویز ورودی گرفته شده که ۱۰ در ۱۰ است و به صورت بردار ۱۰۰ تایی دریافت می‌شود و همچنین برچسب مورد نظر که مانند قسمت متمایزگر ابتدا تعبیه می‌شود تا به بردار تبدیل شود و سپس با بردار نویز ورودی `concat` می‌شود. همچنین دیده می‌شود که این دو شبکه تنها از لایه‌های `linear` با توابع فعال‌سازی `leaky Relu` ساخته شده‌اند و معماری پیچیده‌ای ندارند. در قسمت متمایزگر از `dropout` نیز استفاده شده است که تعمیم‌پذیری مدل بالاتر رود.

برای `loss` در دو مدل از `BCE` و برای تابع بهینه‌ساز از `Adam` با نرخ یادگیری ۰.۰۰۰۱ استفاده شده است. در ادامه برای آموزش هر یک از دو مدل توابعی نوشته شده که در هر قدم خروجی مربوط به مدل دیگر را می‌گیرد و `loss` را محاسبه کرده و وزن‌ها را به‌روز می‌کند. در آموزش همزمان دو مدل نیز فرایند طراحی شده به این صورت است که در هر قدم از آموزش ابتدا مدل متمایزگر تا ۵ قدم آموزش می‌بیند و سپس مدل مولد یک قدم آموزش می‌بیند.

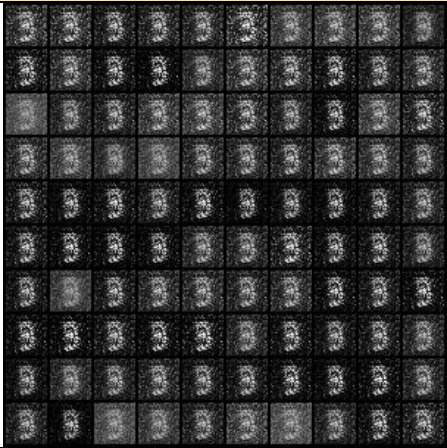
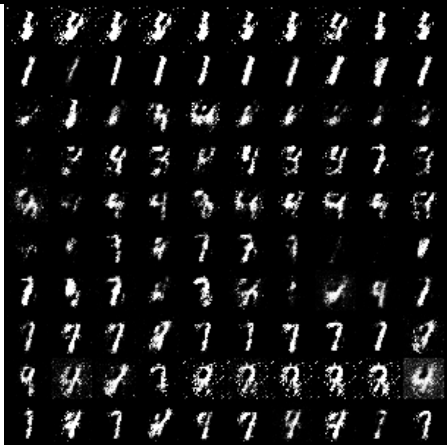



پس از آموزش مدل‌ها تا ۶۰ اپاک با batch size برابر ۱۲۸ نمودار تغییرات خطا برای تابع مولد و متمایزگر به صورت زیر است:



**تحلیل تغییرات خطای دو مدل:** همان‌طور که در نمودار تغییرات خطای دو مدل دیده می‌شود ابتدا خطای مدل مولد بالاست و خطای مدل متمایزگر کم است که دلیل این امر آن است که چون در ابتدای آموزش مولد تصاویر کاملاً تصادفی و نویزی تولید می‌کند تشخیص مصنوعی بودن تصویر برای مدل متمایزگر بسیار ساده است و در نتیجه خطای مدل متمایزگر پایین و مولد بالا خواهد بود. در ادامه با بهبود عملکرد مدل مولد خطای این مدل کاهش می‌یابد و به دلیل شبیه شدن نتایج به نتایج واقعی تشخیص آن برای مدل متمایزگر سخت می‌شود و در نتیجه خطای مدل متمایزگر مقداری افزایش یافته است.

**نمایش خروجی‌ها:** همچنین پس از هر اپاک از آموزش خروجی مدل مولد را برای تولید اعداد بررسی کرده‌ایم و خروجی آن ذخیره شد است. در جدول زیر این خروجی‌ها پس از هر ۱۰ اپاک آموزش نشان داده شده است:

خروجی مدل مولد	شماره ایپاک
	۱
	۱۰
	۲۰



	0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 5 5 5 5 5 5 5 5 5 5 6 6 6 6 6 6 6 6 6 6 7 7 7 7 7 7 7 7 7 7 8 8 8 8 8 8 8 8 8 8 9 9 9 9 9 9 9 9 9 9		۳.
	0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 5 5 5 5 5 5 5 5 5 5 6 6 6 6 6 6 6 6 6 6 7 7 7 7 7 7 7 7 7 7 8 8 8 8 8 8 8 8 8 8 9 9 9 9 9 9 9 9 9 9		۴.
	0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 5 5 5 5 5 5 5 5 5 5 6 6 6 6 6 6 6 6 6 6 7 7 7 7 7 7 7 7 7 7 8 8 8 8 8 8 8 8 8 8 9 9 9 9 9 9 9 9 9 9		۵.
	0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 5 5 5 5 5 5 5 5 5 5 6 6 6 6 6 6 6 6 6 6 7 7 7 7 7 7 7 7 7 7 8 8 8 8 8 8 8 8 8 8 9 9 9 9 9 9 9 9 9 9		۶.

## امتیازی:

مجموعه ۶۰ تصویر تولید شده پس از هر ایپاک آموزشی با استفاده از کتابخانه imageio به یک گیف تبدیل شده است که نتیجه آن در کنار فایل گزارش قرار گرفته است.