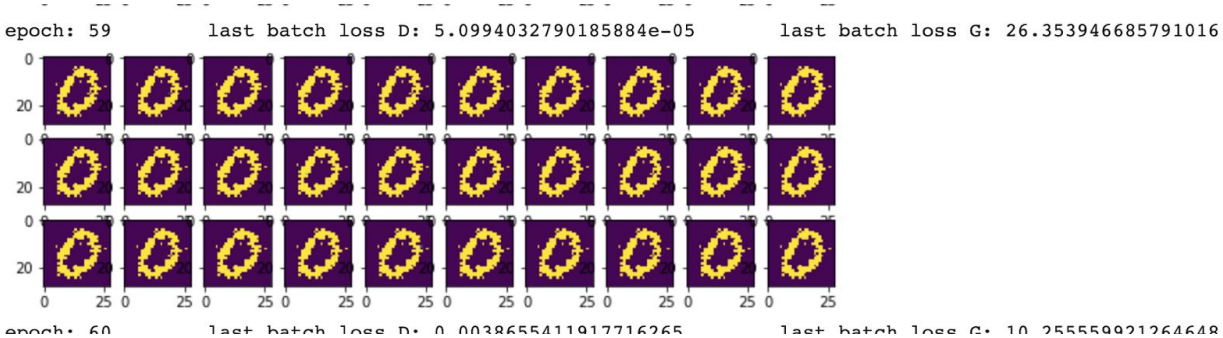
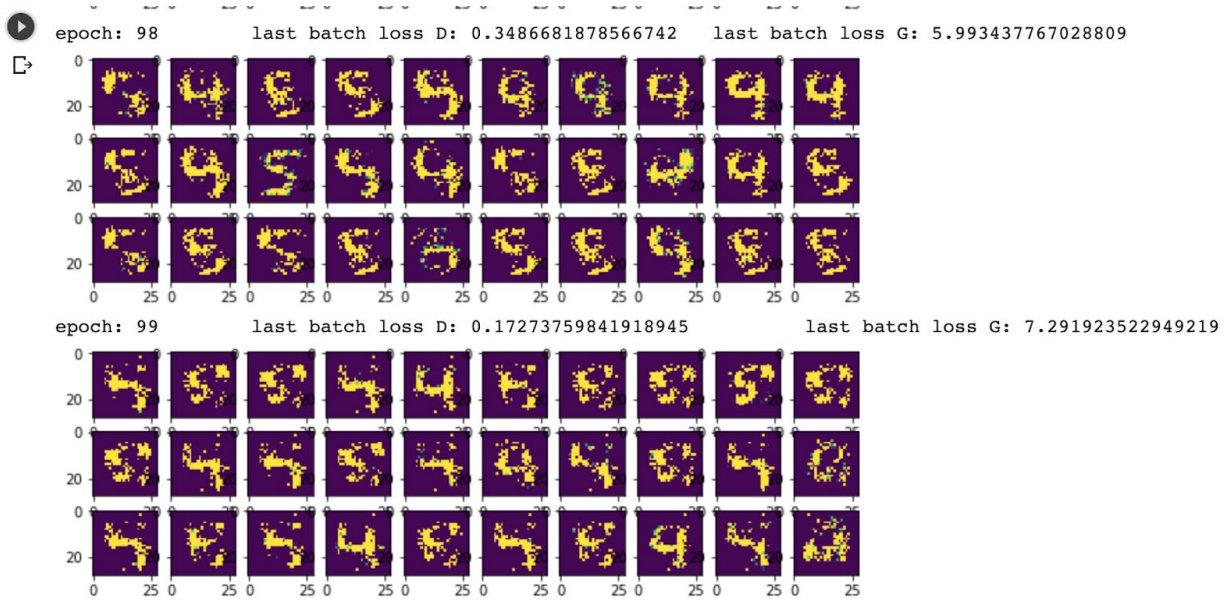
دانیال ملک محمد - 94100092 - یادگیری عمیق - تمرین پنجم - سوال هفتم

بخش الف)

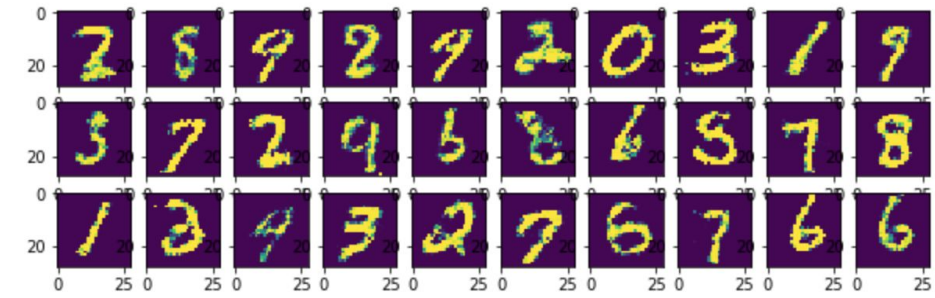
توزیع یونیفرم:





توزیع نرمال:

```
epoch: 99        last batch loss D: 1.3221588134765625   last batch loss G: 0.856677770614624
```



همانطور که می بینید یادگیری با توزیع نرمال موفقیت آمیز بود اما با یونیفرم در ابتدا کلا یک شکل صفر مانند بود و در نهایت تصاویر اسپارس مانندی از اعداد هستند.
دلیل ضعیف بودن این روش این است که توزیع نرمال سعی می کند که تمرکز دستریبیوشن نویز در فضایش فشرده تر باشد، همانطور که در VAE توزیع پرایور Q اینگونه است. و واقعا منیفولدی که بیانگر اطلاعات باشد در یک فضای فشرده قابل توصیف است و توزیع یونیفرم باعث بخش شدن و سخت شدن یادگیری می شود.

توجیه دیگر شاید این باشد که وقتی توزیع یونیفرم می گیریم، طبق چیزی که در سوال ۴ بخش آخر دیدیم، ساپورت توزیع مدل فضای خیلی کوچکی را تشکیل می دهد و در ابتدا با توزیع داده ی اصلی هیچ اشتراکی ندارد و گرادیان JS را صفر می کند اما توزیع نرمال این مشکل را ندارد و ساپورتش تقریبا کل فضا است.

بخش ب)

برای این بخش بنده به این صورت عمل کردم:

درکل لایه ی اول در تمیزدهنده و مولد، با پدینگ عکس را به اندازه ی 32*32 تبدیل می کند تا برای ادامه ی کار عکس اندازه اش رند باشد همچنین مقادیر گری اسکیل هر پیکسل را طوری نرمالایز کردیم که مقادیرشان بین ۱ و ۱- باشد تا با خروجی tanh ای که در لایه نهایی مولد استفاده می شود، هماهنگ شوند.
همچنین تمام کانولوشن هایی که انجام می شوند، اندازه ی فیلتر ۴در۴ با استراید ۲ و پدینگ ۱ است تا با هربار انجام طول و عرض عکس دقیقا نصف شود.

بخش تمیزدهنده:

ابتدا یک لایه ی دراپ آوت با p= 0.3 وجود دارد تا تعدادی ازپیکسل های عکس را سیاه کند و کار تمیزدهنده را دشوار کند، چون قبل از اعمال این کار مشکلی که همواره وجود داشت این بود که تمیزدهنده خیلی قوی می شد و لاس خیلی کمتری داشت و یکباره مدل از حالت تقریبا مطلوب، به حالتی کاملا بی معنی جهش می کرد و ثابت می ماند.

سپس عکس ها با کانولوشن به سایز ۱۶ در ۱۶ و عمق ۱۶ می روند.

سپس سه بار این ماژول تکرار می شود:

کانولوشن،batch normalization، تابع فعالسازی Leaky Relu 0.2

و هربار طول و عرض عکس تقسیم بر۲ می شوند اما عمق عکس دوبرابر می شود.

بعد از این سه ماژول عکس ۱ در ۱ است و ۱۲۸ عمق دارد، روی این یک کانولوشن با اندازه فیلتر خروجی ۱ می زنیم، درواقع یک MLP است و روی آن در نهایت یک فعالسازی سیگموید زده می شود و بدین ترتیب یک عدد اسکالر بین صفر و یک خروجی داده می شود.

بخش مولد:

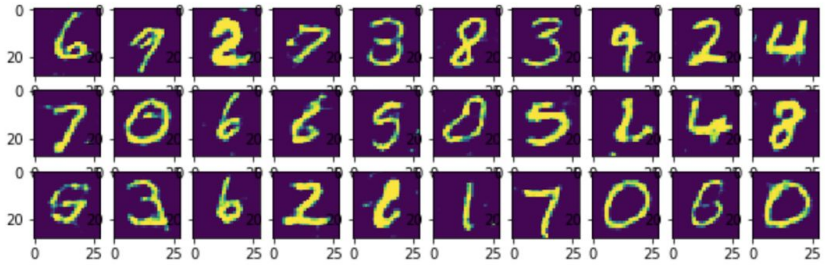
دقیقا معماریش معکوس معماری بخش تمیز دهنده است.
به عنوان ورودی یک نویز ۱۲۸ بعدی می گیرد و آن را با کانولوشن ترنسپوز خطی به تصویر ۴*۴ با عمق ۶۴ می برد.(فیلتر ۴* ۴ با استراید ۱)

سپس دوبار همان ماژول مذکور در قسمت تمیزدهنده را (اما با کانولوشن ترنسپوز) روی داده اعمال می شود تا به عکس ۱۶ در ۱۶ به عمق ۱۶ برسد.
در نهایت یک کانولوشن ترنسپوز نهایی با تابع فعالسازی tanh و چنل خروجی ۱، تصویر نهایی را ارایه می دهد.

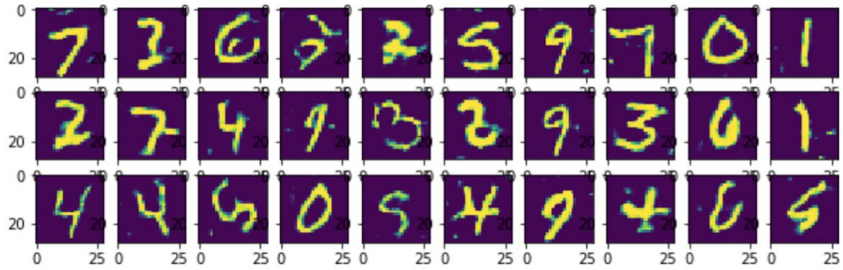
این معماری مزیتش این است که انگار مولد و تمیزدهند دو ماژول معکوس هم هستند که مولد یک داده ی ۱۲۸ بعدی را به عکس تبدیل می کند و تمیزدهنده عکس را به داده ی ۱۲۸ بعدی تبدیل بازمی گرداند سپس از روی آن تشخیص می دهد که واقعی یا تقلبی است. انگار داده ها در کل دو نوع نمایش دارند، یکی به صورت عکس و دومی نمایشی در منیفولد ۱۲۸ بعدی مذکور.
مزیت دیگر این روش این است که به دلیل عمق کم کانولوشن ها و ترتیب افزایش و کاهش عمق، پارامترها خیلی کمتر شده اند و روند آموزش به نسبت سریع صورت می گیرد به طوری که هر اپیاک (((((((((طول می کشد.(توجه کنید چون عکس ها گری اسکیل هستند و فقط یک عدد از ۱۰ عدد ممکن را نشان می دهند، نیازی به عمق زیاد مثل معماری های کانولوشن مربوط به عکس های عادی ندارند)

در پایین عکس های از مولد می بینید:

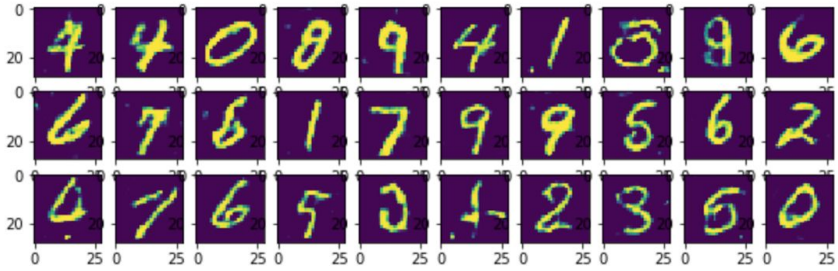
```
epoch: 86        last batch loss D: 0.32387077808380127          last batch loss G: 6.83490514755249
```



```
epoch: 87        last batch loss D: 0.6074679493904114   last batch loss G: 8.19876480102539
```



```
epoch: 67         last batch loss D: 0.060048554092645645         last batch loss G: 4.947624206542969
```



```
epoch: 99         last batch loss D: 0.2948535978794098    last batch loss G: 3.215184211730957
```