«*In The Name Of GOD*»

دانشگاه صنعتی امیرکبیر

(پلی تکنیک تهران)

[HW-06-Report]

[NEURAL COMPUTING AND DEEP LEARNING]

Hasan Masroor | [403131030] | June 10, 2025

"فهرست مطالب تمرين 06"

Question 1 …… ... 2

1) ……..... 2

2) ……..... 2

3) ……..... 4

4) ……..... 17

5) ……..... 19

6) …….. 21

7) ……..... 25

Question 2 ……..... 29

1) ……..... 29

2) ……..... 29

3) ……..... 29

4) ……..... 31

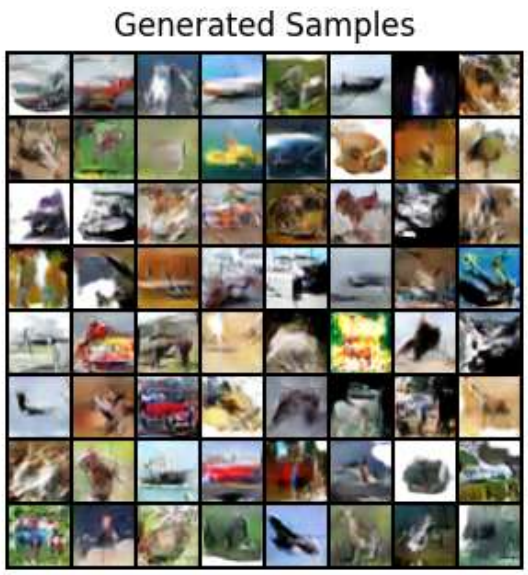
Problem 6: Generative Adversarial Networks (GAN)

Question 1:

1&2.

در این تمرین، هدف ما بررسی و ارزیابی مدل‌های مختلف شبکه‌های مولد تقابلی (GAN) در تولید تصاویر واقعی است. ابتدا ما مدل پایه DCGAN را پیاده‌سازی کرده و سپس با اعمال تغییرات مختلف در ساختار و تکنیک‌ها، سعی کردیم عملکرد آن را بهبود دهیم. در طول تمرین، آزمایش‌های مختلفی انجام دادیم که شامل تحلیل تأثیر پارامترهایی مانند اندازه بردار نویز ورودی، استفاده از Batch Normalization و انتخاب توابع فعال‌سازی متفاوت بود. بعد از پیاده‌سازی مدل‌های پایه، به سراغ مدل‌های Conditional GAN و Wasserstein GAN رفتیم تا قابلیت‌های هر یک را در تولید تصاویر با کیفیت و پایدار بررسی کنیم. سپس برای مقایسه کیفیت تصاویر تولیدی با تصاویر واقعی، از معیارهای مختلفی مانند Inception Score و FID استفاده کردیم. در نهایت، توزیع تصاویر واقعی و تولیدی را در فضای ویژگی با استفاده از تکنیک‌هایی مثل t-SNE یا UMAP مورد بررسی قرار دادیم تا میزان شباهت و کیفیت تصاویر تولیدی را ارزیابی کنیم.

در این قسمت، ما ابتدا مدل پایه DCGAN را برای تولید تصاویر از مجموعه داده CIFAR-10 پیاده‌سازی کردیم. ابتدا با استفاده از شبکه‌های کانولوشن ترانسپوز(ConvTranspose2d) ، مولد (Generator) و متمایزکننده (Discriminator) را ساختیم. مولد تصاویر تصادفی را از بردار نویز به تصاویر 32x32 تبدیل می‌کند، در حالی که متمایزکننده تلاش می‌کند تا تصاویر واقعی و تولیدی را از هم تشخیص دهد، سپس با استفاده از تابع ضرر BCELoss، مدل آموزش دید و بهینه‌سازی با الگوریتم Adam انجام شد. در طول آموزش، به صورت دوره‌ای تصاویر تولیدشده ذخیره و مقایسه شدند و با استفاده از معیارهای Inception Score (IS) و Frechet Inception Distance (FID) کیفیت تصاویر تولیدی را ارزیابی کردیم و در نهایت برای تجزیه و تحلیل بهتر تصاویر واقعی و تولیدشده را در خروجی زیر نمایش دادیم:

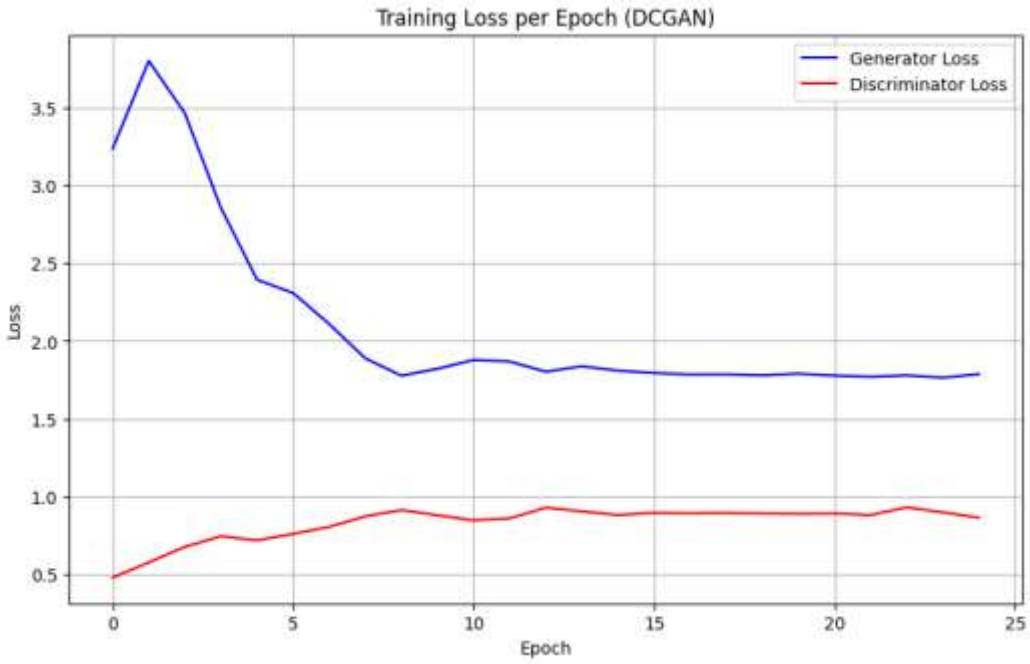


همانطور که در تصاویر بالا می‌بینیم نمونه‌های تولیدشده هرچند ساختارهای کلی را بازتولید می‌کنند، اما به دلیل پدیده‌Mode Averaging ، با محوشدگی لبه‌ها و فقدان جزئیات مواجه می‌شوند که در مقایسه با تصاویر واضح و متنوع مجموعه داده CIFAR-10 قابل‌توجه است. این امر نشان‌دهنده چالش مدل در یادگیری کامل توزیع داده‌ها و دستیابی به همگرایی مطلوب است و می‌تواند به دلیل تعداد اپک ها هم باشند و با افزایش آنها بهبود بهتری حاصل شود.

FID 54.68 IS 3.65 ± 0.04

با توجه به نتایج به‌دست‌آمده از معیارهای ارزیابی، Inception Score (IS) برابر با 3.65 نشان‌دهنده کیفیت نسبی خوب تصاویر تولیدی است اما هنوز فاصله قابل توجهی با مقادیر ایده‌آل دارد. همچنین، مقدار Frechet Inception Distance (FID) برابر با 54.68 نشان‌دهنده تفاوت قابل توجه میان توزیع ویژگی‌های تصاویر واقعی و تولیدی است و نیاز به بهبود بیشتر در کیفیت تصاویر تولیدی و هم‌راستایی بهتر با تصاویر واقعی است.

در ادامه نیز نمودارهای Loss مربوط به مولد و متمایزکننده را برای تحلیل بهتر نشان دادیم و نمودار Loss نشان‌دهنده کاهش اولیه قابل‌توجه Loss مولد (Generator) تا حدود اپک 5 است که با تثبیت در سطح 1.5 تا 2.0 همراه می‌شود، در حالی که Loss متمایزکننده (Discriminator) به‌تدریج افزایش یافته و در محدوده 0.5 تا 1.0 پایدار می‌ماند. نمودار Loss را در پایین نمایش دادیم:



3.

برای بررسی تأثیر اندازه بردار نویز ورودی بر عملکرد مدل DCGAN، آزمایش‌هایی با دو مقدار متفاوت برای ابعاد بردار نویز، یعنی 50 و 200 انجام دادیم. هدف از این آزمایش‌ها، ارزیابی تأثیر اندازه فضای نهان بر کیفیت و تنوع تصاویر تولید شده بود. در این مرحله تمامی تنظیمات دیگر ثابت نگه داشته شدند تا بتوانیم به‌صورت کنترل‌شده فقط نقش بردار نویز ورودی را تحلیل کنیم.

ساختار مدل در هر دو حالت ثابت بود و شامل یک Generator و یک Discriminator با معماری کلاسیک DCGAN می‌شد. در Generator بردار نویز از اندازه مشخص‌شده به‌صورت تصادفی تولید و پس از عبور از چند لایه ConvTranspose2d و BatchNorm وReLU، تصویر با ابعاد 32x32 ساخته می‌شد.
Discriminator نیز وظیفه تفکیک تصاویر واقعی از جعلی را بر عهده داشت و شامل چند لایه Conv2d همراه با LeakyReLU و در نهایت تابع خروجی Sigmoid بود.

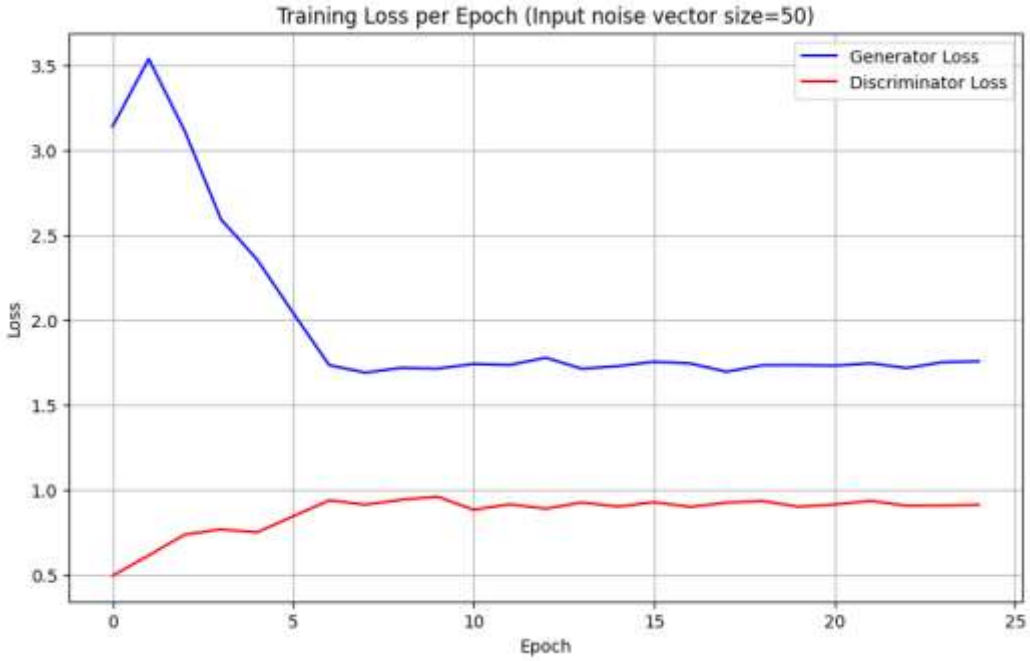
در هر دو آزمایش، از تابع زیان Binary Cross-Entropy و بهینه‌ساز Adam برای به‌روزرسانی شبکه‌ها استفاده کردیم. برای ارزیابی کیفی خروجی‌ها، پس از پایان آموزش در هر حالت،ده هزار تصویر تولید و با استفاده از معیارهای IS و FID تحلیل شدند. علاوه بر آن، نمودار تغییرات لاس Generator و Discriminator در طول دوره آموزش نیز مثل بخش قبل رسم شد تا روند یادگیری مدل در مواجهه با بردارهای نویز متفاوت مورد بررسی قرار گیرد. این مقایسه به ما کمک می‌کند تا درک بهتری از نقش اندازه فضای نهان در کیفیت و تنوع تصاویر تولیدی داشته باشیم.

-Input noise vector size:

- Size=50

نمودار Loss برای مدل با اندازه بردار نویز 50 نشان‌دهنده کاهش قابل‌توجه اولیه Loss Generator از 3.5 به حدود 1.8 در طی 5 اپک اول است که با تثبیت نسبی در محدوده 1.5 تا 2.0 تا پایان 25 اپک همراه می‌شود، در حالی که Loss Discriminator پس از افزایش اولیه، به‌تدریج در محدوده 0.5 تا 1.0 پایدار می‌ماند. این الگو حاکی از تطابق مناسب Generator با بازخورد Discriminator و تلاش برای یادگیری توزیع داده‌ها است که با مقادیر ارزیابی IS 3.77 و FID 57.65 تأیید می‌شود. این مقادیر نشان‌دهنده کیفیت قابل‌قبول تصاویر تولیدشده است، هرچند هنوز با تصاویر واقعی فاصله دارد دارند. ثبات نسبی Lossها در فازهای پایانی بیانگر دستیابی به یک تعادل نسبی در آموزش است.

nz50 IS 3.77 ± 0.10 | FID 57.65

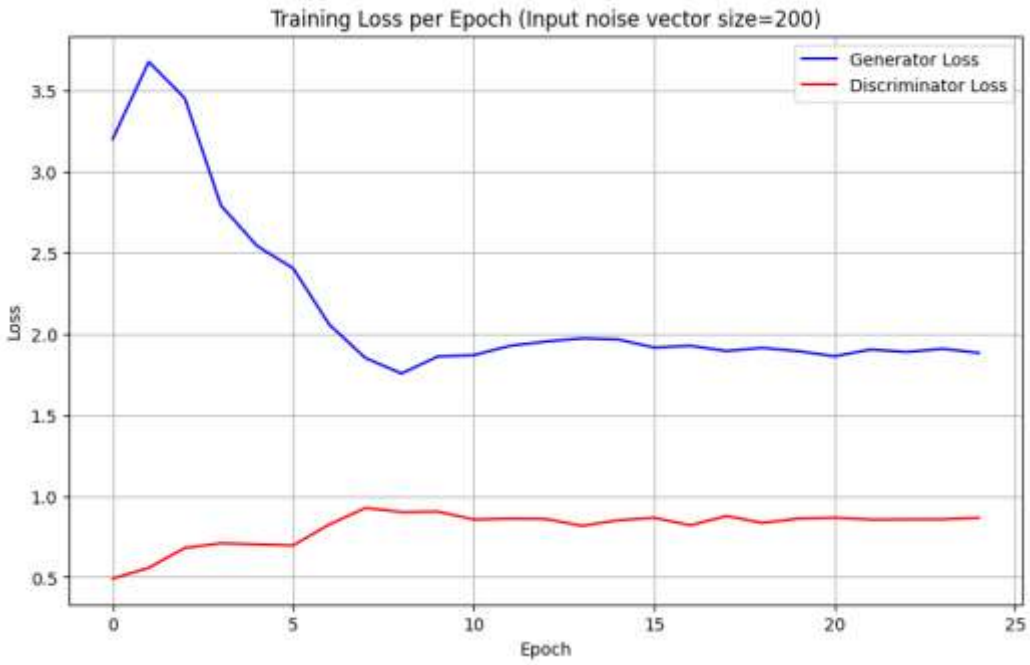


- Size=200

نمودار Loss برای مدل با اندازه بردار نویزرا 200 مثل بخش قبل می‌توانیم تحلیل کنیمو این نمودار نشان‌دهنده کاهش قابل‌توجه اولیه Loss Generator از 3.5 به حدود 1.8 در طی 5 اپک اول است که با تثبیت نسبی در این محدوده تا پایان 25 اپک همراه می‌شود، در حالی که Loss Discriminator پس از افزایش اولیه در محدوده

0.5 تا 1.0 به‌صورت پایدار باقی می‌ماند. این رفتار منعکس‌کننده توانایی Generator در تطبیق با فضای Latent بزرگ‌تر و تولید تنوع بیشتر است، که با مقادیر ارزیابی IS 3.60 و FID 59.14 پشتیبانی می‌شود. این مقادیر نشان‌دهنده کیفیت تصاویر تولیدشده در سطحی مشابه با اندازه 50، اما با کمی کاهش IS و افزایش FID است. ثبات نسبیLoss ها در فازهای پایانی حاکی از تعادل مناسب بین Generator و Discriminator است و با این حال افزایش اندازه بردار نویز ممکن است پیچیدگی یادگیری را بالا برده باشد.

nz200 IS 3.60 ± 0.10 | FID 59.14



-Batch Normalization:

- **Only Generator**

در این بخش، به‌منظور بررسی تأثیر Batch Normalization بر کیفیت و پایداری آموزش مدلDCGAN، سه سناریوی مختلف طراحی و اجرا شده‌اند. در این آزمایش‌ها، لایه‌های Batch Normalization به‌صورت جداگانه و ترکیبی روی دو بخش اصلی مدل، یعنی Generator و Discriminator اعمال شدند. حالت اول شامل استفاده از Batch Normalization فقط درGenerator، حالت دوم تنها درDiscriminator و حالت سوم در هر دو شبکه بود. هدف اصلی این بررسی، ارزیابی تأثیر نرمال‌سازی داخلی لایه‌ها بر روند همگرایی مدل،

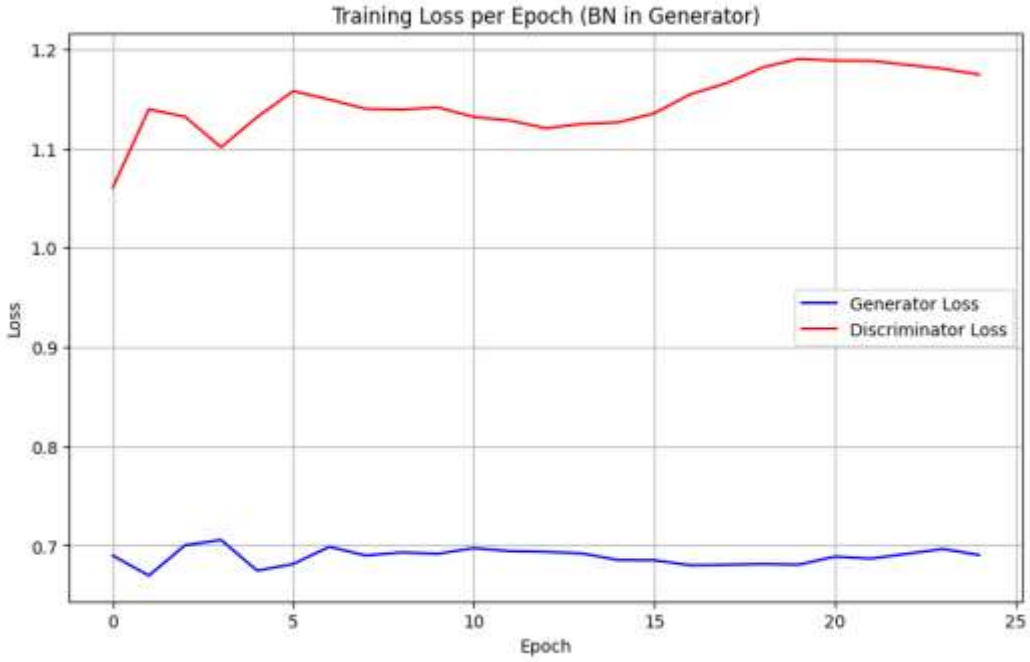
پایداری lossها و کیفیت نهایی تصاویر تولیدی است. در تمامی آزمایش‌ها، معماری کلی هر دو شبکه ثابت باقی ماند و تنها تفاوت در فعال یا غیرفعال بودن لایه‌های BatchNorm2d بود. پیاده‌سازی این کنترل از طریق پارامترهای bn_G و bn_D انجام شد. درGenerator ، سه لایه BatchNorm2d پس از لایه‌های ConvTranspose2d و پیش از ReLU قرار دارند. در Discriminator نیز دو لایه BatchNorm2d در مسیر پردازش تصاویر پس از لایه‌های Convolution درج می‌شوند. طبق روال فرآیند آموزش در هر سناریو به مدت ۲۵ اپک انجام شده است. در هر دوره، ابتدا Discriminator با تصاویر واقعی و تولیدی آموزش می‌بیند، سپس Generator به‌گونه‌ای به‌روزرسانی می‌شود که بتواند Discriminator را فریب دهد. همچنین از تابع BCELoss به عنوان تابع هزینه و بهینه‌ساز Adam استفاده شد

در ادامه خروجی های حالت اول یعنی فقط روی مولد را اعمال و نمایش دادیم:

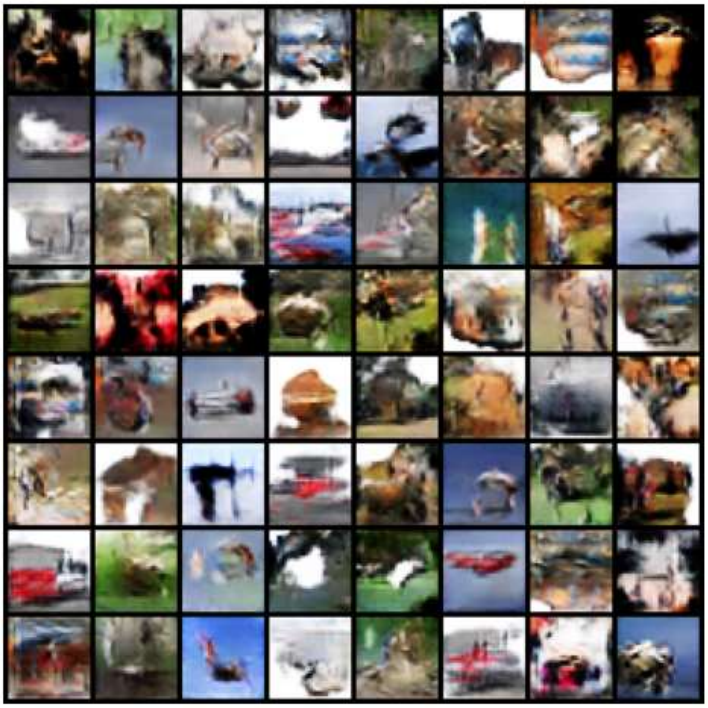
IS 2.42 ± 0.03 FID 135.17

نمودار Loss برای مدل با اعمال Batch Normalization (BN) تنها در Generator نشان‌دهنده کاهش اولیه Loss Generator به حدود 0.7 و تثبیت آن در محدوده 0.7 تا 0.8 در طول 25 اپک است، در حالی که Loss Discriminator پس از افزایش اولیه به حدود 1.1، با نوسانات جزئی در محدوده 1.0 تا 1.2 باقی می‌ماند. این الگو بیانگر تلاش Generator برای تطبیق با داده‌ها با استفاده از BN است اما مقادیر ارزیابی IS 2.42 و FID 135.17 نشان‌دهنده کیفیت پایین تصاویر تولیدشده و فاصله قابل‌توجه از توزیع واقعی هستند که با مشاهده

کم کیفیت و کم‌وضوح بودن تصاویر نمونه‌ها هم‌راستاست. ثبات نسبی Loss Generator ممکن است به دلیل تأثیر مثبت BN در کاهش نویز داخلی باشد.

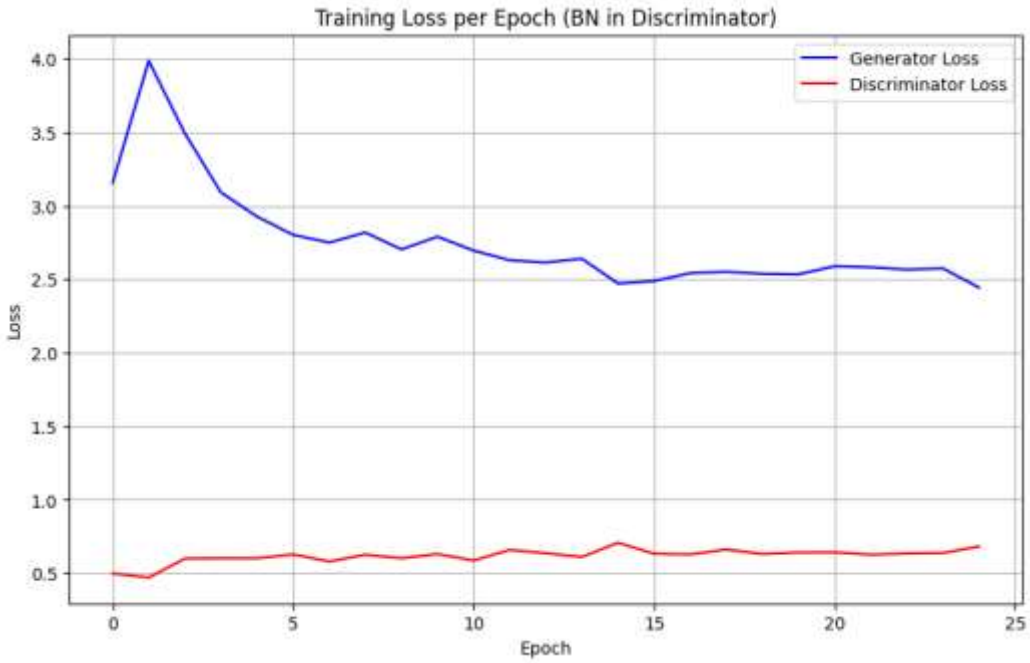


- **Only Discriminator**



نمودار Loss برای مدل با اعمال Batch Normalization (BN) تنها در Discriminator نشان‌دهنده کاهش قابل‌توجه اولیه Loss Generator از 4.0 به حدود 2.5 در طی 5 اپک اول است که با تثبیت نسبی در این محدوده تا پایان 25 اپک همراه می‌شود، در حالی که Loss Discriminator پس از افزایش اولیه در محدوده 0.5 تا 1.0 به‌صورت پایدار باقی می‌ماند. این الگو حاکی از تلاش Generator برای تطبیق با داده‌ها با وجود تأثیر BN در Discriminator است که با مقادیر ارزیابی IS 3.48 و FID 62.29 نیز حاکی از این ماجرا است. این مقادیر نشان‌دهنده تلاش مدل برای تولید تصاویر با کیفیت قابل‌قبول هستند، هرچند تصاویر خروجی دارای مقداری نویز و محوشدگی هستند که می‌تواند به دلیل کم بودن تعداد اپک ها و آموزش ناکافی مدل باشد.

IS 3.48 ± 0.07 FID 62.29

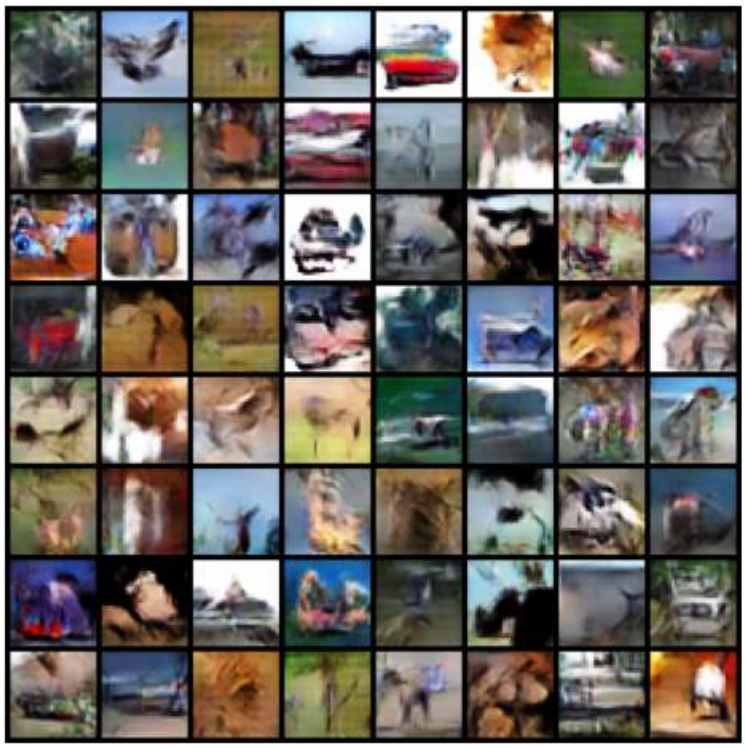


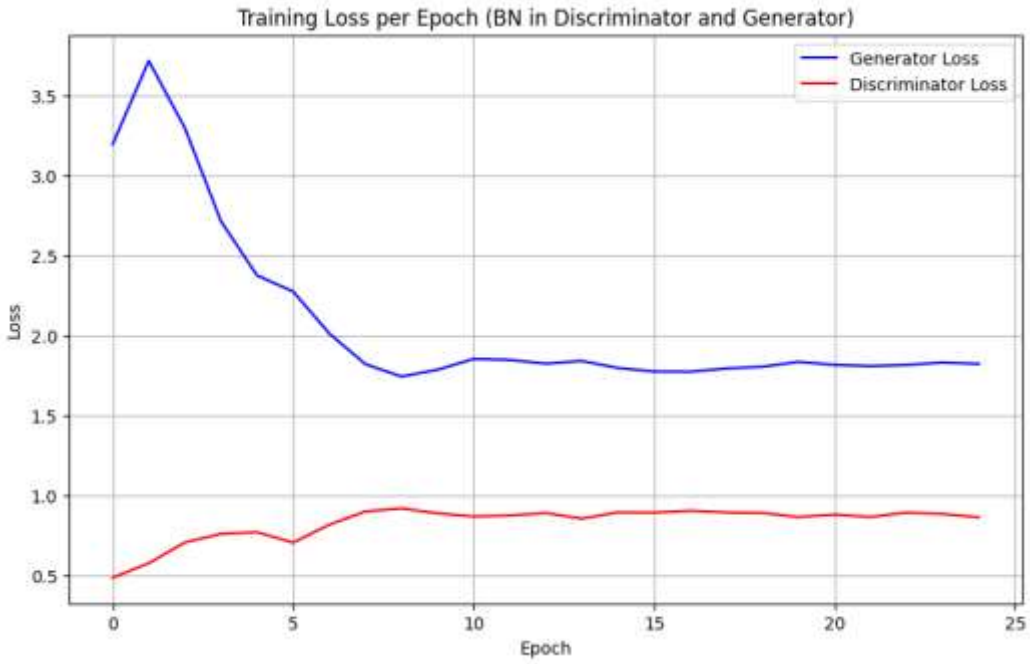
- **Generator & Discriminator**

نمودار Loss برای مدل با اعمال Batch Normalization (BN) هم در Generator و هم در Discriminator نشان‌دهنده کاهش قابل‌توجه اولیه Loss Generator از 3.5 به حدود 2.0 تا 2.5 در طی 5 اپک اول است که با تثبیت نسبی در این محدوده تا پایان 25 اپک همراه می‌شود در حالی که Loss Discriminator پس از افزایش اولیه، در محدوده 0.5 تا 1.0 به‌صورت پایدار باقی می‌ماند. این الگو حاکی از

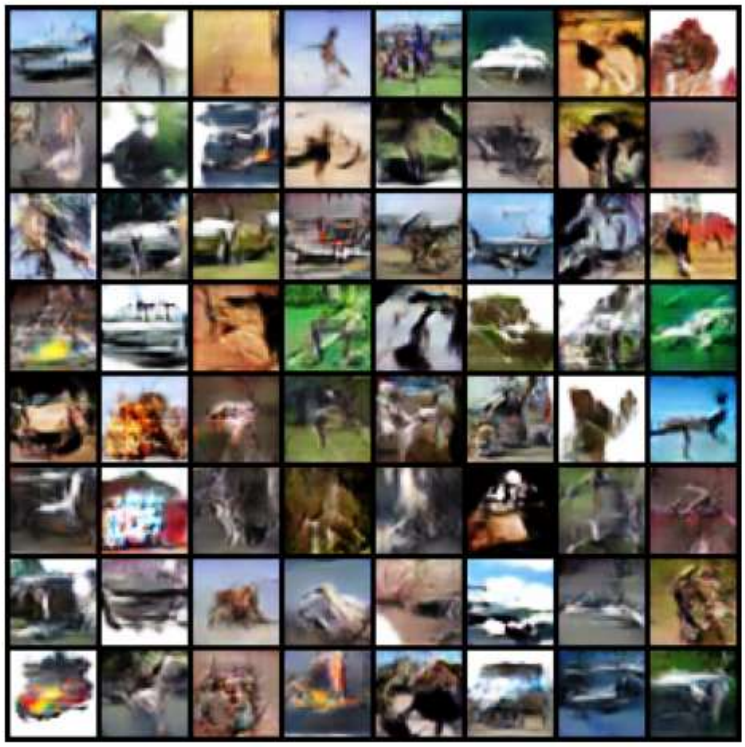
هماهنگی مناسب بین Generator و Discriminator در مقابل دو حالت قبلی یعنی فقط مولد یا فقط متمایزکننده است که مقادیر ارزیابی IS 3.54 و FID 57.07 نیز عملکرد بهتر این بخش نسبت دو حالت گذشته را نشان می دهد. این مقادیر نشان‌دهنده کیفیت قابل‌قبول تصاویر تولیدشده هستند، هرچند تصاویر نمونه‌ها دارای مقداری نویز و وضوح محدود هستند. ثبات نسبی Lossها در فازهای پایانی نیز بیانگر پیشرفت در فرآیند آموزش و تعادل مناسب بین اجزای مدل است.

IS 3.54 ± 0.09 FID 57.07



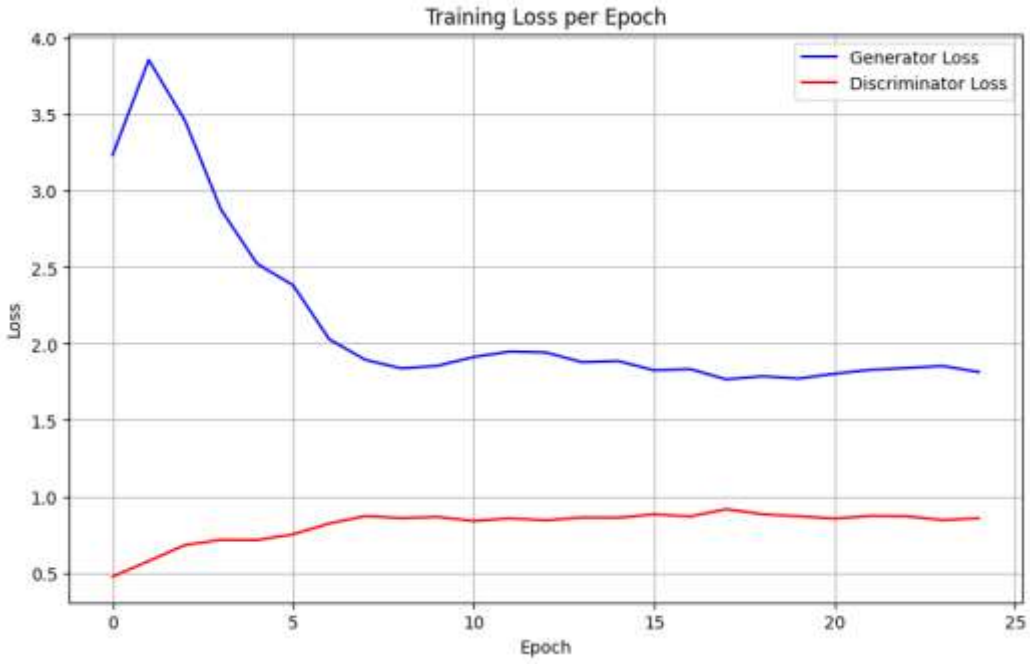


- **Label Smoothing:**



در این بخش به‌منظور بررسی تأثیر Label Smoothing بر پایداری فرآیند آموزش و کیفیت تصاویر تولیدشده، تغییری در مقدار برچسب واقعی در تابع هزینه Discriminator اعمال شده است. به‌جای استفاده از مقدار 1 برای برچسب تصاویر واقعی، مقدار نرم‌شده‌ی 0.9 انتخاب شده است. این تکنیک که با نام label smoothing شناخته می‌شود و با هدف جلوگیری از بیش‌اعتماد شدن Discriminator طراحی شده و از غلبه‌ی بیش‌ازحد آن بر Generator جلوگیری می‌کند. برای پیاده‌سازی این تکنیک، مقدار برچسب واقعی از طریق پارامتر real_label در پیکربندی تنظیم شده و به‌صورت شرطی از طریق کلید ls کنترل می‌شود. در صورتی که ls=True باشد، برچسب‌های واقعی در طول آموزش مقدار 0.9 دریافت می‌کنند و در غیر این صورت برابر 1 باقی می‌مانند. این برچسب‌ها در قالب تنسور real_lbl برای استفاده در BCELoss آماده می‌شوند و در محاسبه‌ی loss مربوط به Discriminator و Generator به کار می‌روند. در سایر بخش‌های مدل، ساختار شبکه‌های Generator و Discriminator مشابه تنظیمات پایه DCGAN حفظ شده و تنها تغییر اعمال‌شده مربوط به جایگزینی مقدار برچسب واقعی است. در ادامه خروجی‌های این بخش را مشاهده می‌کنیم:

IS 3.68 ± 0.07 FID 56.72



نمودار Loss برای مدل با اعمال Label Smoothing نشان‌دهنده کاهش قابل‌توجه اولیه Loss Generator از 4.0 به حدود 2.0 تا 2.5 در طی epoch 5 اول است، که با تثبیت نسبی در این محدوده تا پایان epoch 25 همراه می‌شود، در حالی که Loss Discriminator پس از افزایش اولیه، در محدوده 0.5 تا 1.0 به‌صورت پایدار باقی می‌ماند. این الگو حاکی از تأثیر مثبت Label Smoothing در بهبود تطابق Generator با داده‌ها و تعادل مناسب با Discriminator است، که با مقادیر ارزیابی IS 3.68 ± 0.07 و FID 56.72 پشتیبانی می‌شود؛ این مقادیر نشان‌دهنده کیفیت بهتری در تصاویر تولیدشده نسبت به تنظیمات قبلی هستند و بیانگر کاهش فاصله از توزیع واقعی است.

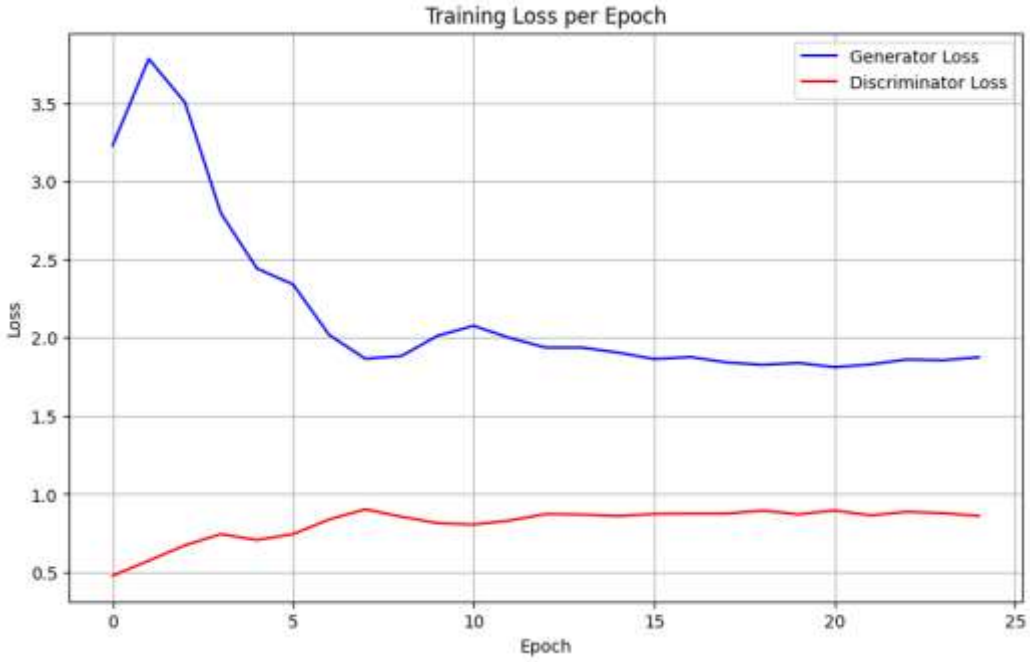
-Activation Function:

- **Generator = ReLU, Discriminator = LeakyReLU**

در این بخش، برای بررسی تأثیر تابع فعال‌سازی بر عملکرد DCGAN، سه ترکیب متفاوت از توابع فعال‌سازی در ساختار Generator و Discriminator مورد آزمایش قرار گرفت. در حالت اول، از ترکیب رایج ReLU در Generator و LeakyReLU در Discriminator استفاده کردیم. این ترکیب باعث می‌شود Generator بتواند گرادیان‌های واضح‌تری برای تولید تصاویر یاد بگیرد و در عین حال Discriminator با استفاده از LeakyReLU دچار مشکل vanishing gradient نشود. در حالت دوم ترکیب ReLU در Generator و Tanh در Discriminator به کار گرفته شد. با اینکه Tanh در لایه خروجی Generator متداول است، استفاده از آن در Discriminator به دلیل اشباع گرادیان در بازه‌های بزرگ‌تر چندان توصیه نمی‌شود. نتایج این آزمایش نیز نشان داد که این ترکیب باعث ناپایداری در آموزش و کاهش کیفیت خروجی‌ها می‌شود. در حالت سوم، هر دو شبکه از LeakyReLU استفاده کردند. این انتخاب به Generator امکان عبور بهتر گرادیان را می‌دهد و از قفل شدن نورون‌ها جلوگیری می‌کند. در برخی تکرارها، این ترکیب باعث همگرایی پایدارتر مدل و کاهش نوسانات Loss شد، گرچه گاهی کیفیت جزئیات تصویر کاهش یافت. خروجی های حالت اول به صورت زیر حاصل شده است:

FID 58.07 IS 3.48 ± 0.05

نمودار Loss برای مدل با استفاده از ReLU در Generator و LeakyReLU در Discriminator نشان‌دهنده کاهش قابل‌توجه اولیه Loss Generator از 3.5 به حدود 2.0 تا 2.5 در طی 5 اپک اول است در حالی که Loss Discriminator پس از افزایش اولیه، در محدوده 0.5 تا 1.0 به‌صورت پایدار باقی می‌ماند. مقادیر ارزیابی IS 3.48 و FID 58.07 نشان‌دهنده کیفیت متوسط تصاویر تولیدشده با تنوع قابل‌قبول و فاصله نسبی از توزیع واقعی هستند. همچنین این مقادیر نشان‌دهنده کیفیت قابل‌قبول تصاویر تولیدشده هستند و بیانگر تلاش مدل برای تطابق با توزیع داده‌هاست. ثباتLoss ها در فازهای پایانی، منعکس‌کننده تعادل خوب بین Generator و Discriminator است. نمودار لاس این بخش را در ادامه نمایش دادیم:

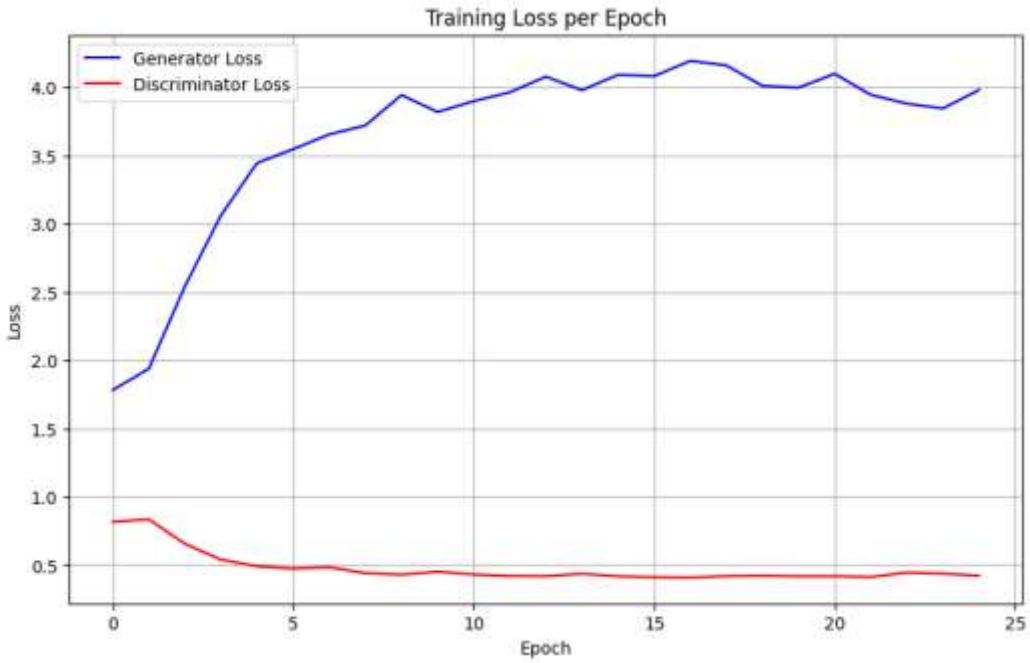


- **Generator = ReLU, Discriminator = Tanh**

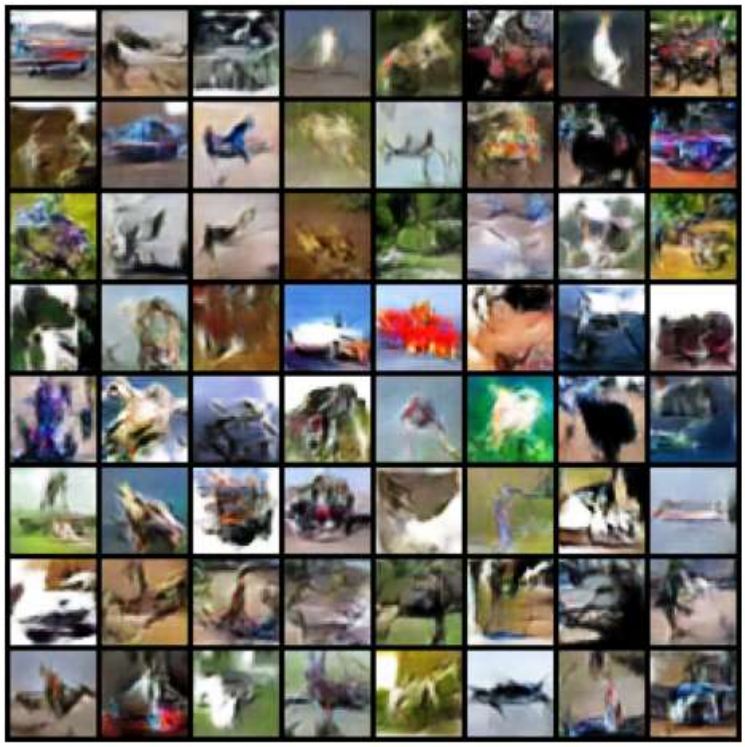
نمودار Loss برای مدل با استفاده از ReLU در Generator و Tanh در Discriminator نشان‌دهنده افزایش قابل‌توجه اولیه Loss Generator از 2.0 به حدود 4.0 در طی 5 اپک اول است، در حالی که Loss Discriminatorپس از کاهش اولیه، در محدوده 0.5 تا 1.0 به‌صورت پایدار باقی می‌ماند. مقادیر ارزیابی IS 2.38 و FID 150.21 نشان‌دهنده چالش‌هایی در کیفیت تصاویر تولیدشده با تنوع محدود و فاصله قابل‌توجه از توزیع واقعی هستند و بدترین حالت بین تمامی بخش های این تمرین می باشد و به دلیل اشباع گرادیان در بازه‌های بزرگ‌تر چندان توصیه نمی‌شود. نتایج این آزمایش نیز نشان داد که این ترکیب باعث ناپایداری در آموزش و کاهش چشمگیر کیفیت خروجی‌ها می‌شود.

در ادامه نیز خروجی های این قسمت را برای درک بصری آوردیم:

FID 150.21 IS 2.38 ± 0.02

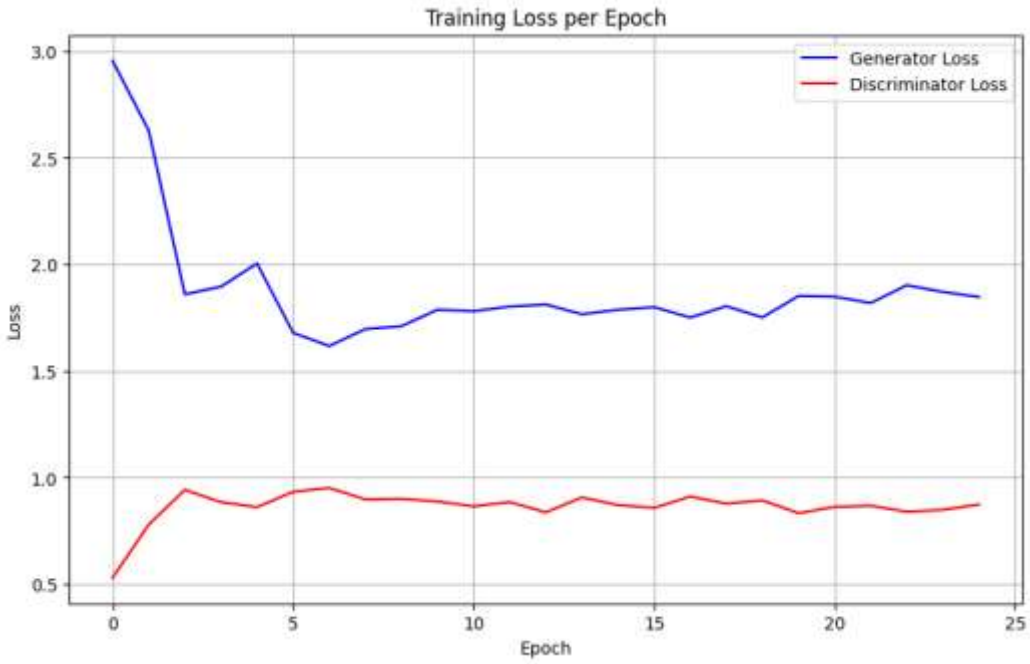


- Generator = LeakyReLU, Discriminator = LeakyReLU



نمودار Loss برای مدل با استفاده از LeakyReLU هم در Generator و هم در Discriminator نشان‌دهنده کاهش قابل‌توجه اولیه Loss Generator از 3.0 به حدود 2.0 در طی 5 اپک اول است، در حالی که Loss Discriminator پس از افزایش اولیه، در محدوده 0.5 تا 1 به‌صورت پایدار باقی می‌ماند. مقادیر ارزیابی IS 3.65 و FID 58.77 نشان‌دهنده کیفیت متوسط تصاویر تولیدشده با تنوع قابل‌قبول و فاصله نسبی از توزیع واقعی هستند. همچنین این مقادیر نشان‌دهنده کیفیت قابل‌قبول تصاویر تولیدشده هستند و بیانگر تلاش مدل برای تطابق با توزیع داده‌هاست. ثباتLoss ها در فازهای پایانی، منعکس‌کننده تعادل خوب بین Generator و Discriminator است.

FID 58.77 IS 3.65 ± 0.06



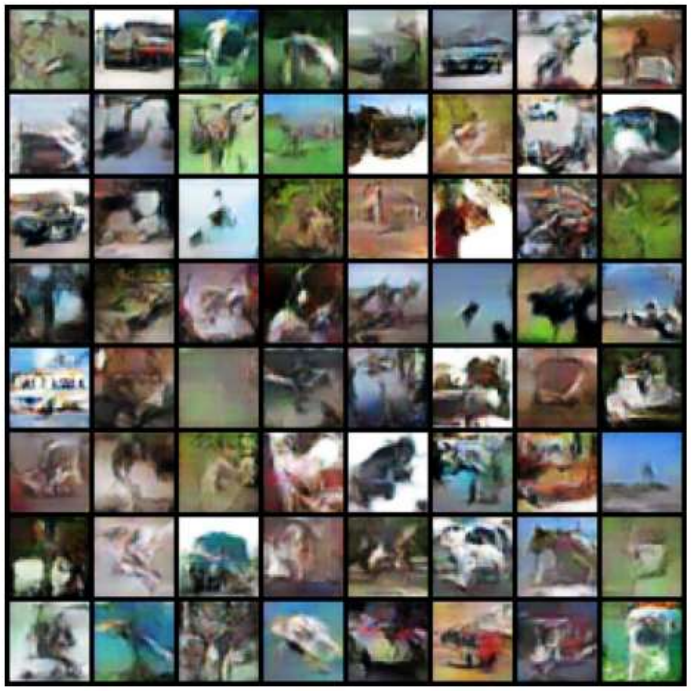
.4

برای پیاده‌سازی مدل Conditional GAN (cGAN) که توانایی کنترل کلاس تصویر خروجی با استفاده از برچسب‌ها را داشته باشد، ابتدا یک معماری کلاسیک cGAN طراحی کردیم. در این مدل Generator با استفاده از ورودی تصادفی و برچسب کلاس، تصویر تولید می‌کند. این برچسب‌ها به‌وسیله یک لایه Embedding به ویژگی‌های ورودی Generator اضافه می‌شوند تا مدل قادر به تولید تصاویر متنوع از کلاس‌های مختلف باشد. از طرف دیگر، Discriminator به‌عنوان یک شبکه تفکیک‌کننده، وظیفه تشخیص تصاویر واقعی و مصنوعی را دارد اما این‌بار علاوه بر تصویر، برچسب کلاس نیز به ورودی‌های آن افزوده می‌شود تا مدل بتواند بر اساس کلاس‌های مختلف، واقعی بودن یا نبودن تصویر را شبیه‌سازی کند.

در بخش آموزش، شبکه‌های Generator و Discriminator به‌طور متناوب با استفاده از تابع زیان Binary Cross-Entropy به‌روز می‌شوند. این به‌روزرسانی‌ها باعث می‌شود که Generator بتواند تصاویر با کیفیت و واقع‌گرایانه‌تری تولید کند و Discriminator نیز به‌طور مؤثرتری تصاویر واقعی و مصنوعی را از هم تشخیص دهد.

در نهایت نیز نمونه ای از تصاویر تولیدشده به همراه نمونه های 5 کلاس اول را در خروجی نمایش دادیم:

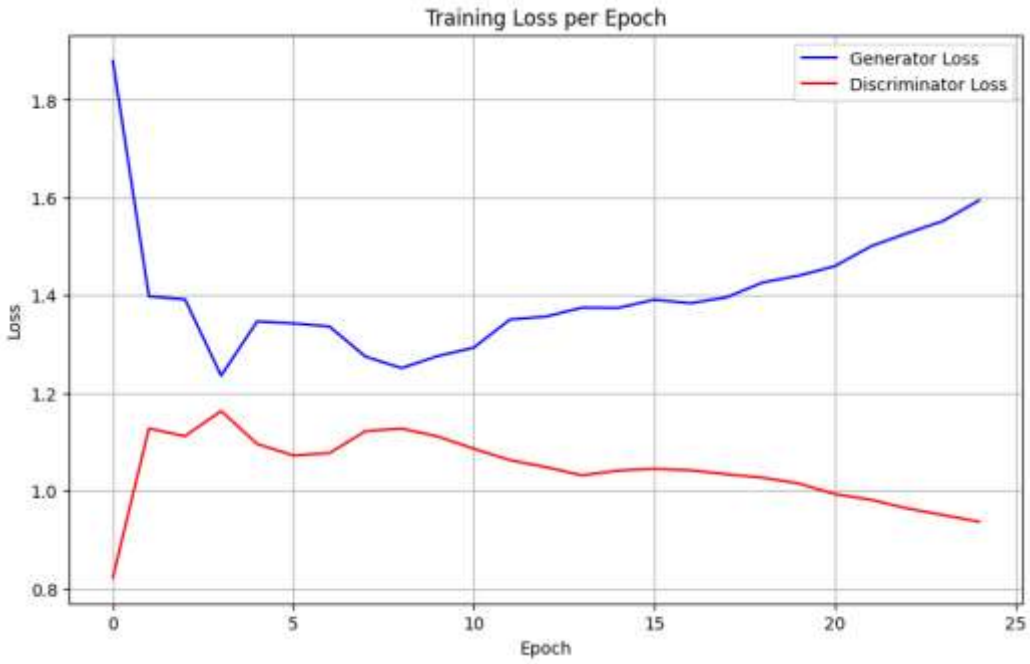
تصاویر تولیدشده توسط مدلcGAN، با وجود تلاش برای بازتولید کلاس‌های مشخص (مانند airplane، automobile، bird، cat، deer) از نظر وضوح و جزئیات کمی ضعیف بوده و تحت تأثیر نویز و محوشدگی قرار دارند حاکی از آن است که مدل هنوز به همگرایی مطلوب نرسیده و ممکن است با افزایش تعداد اپک بهینه تر ود و عملکرد بهتری از خود نشان دهد.



با توجه به مقادیر بدست آمده از ارزیابی مدل، می‌توان گفت که کیفیت تصاویر تولیدی به‌طور کلی قابل قبول است، اما هنوز جای پیشرفت وجود دارد. با توجه به مقدار IS که برابر 3.23 است، مدل توانسته است تصاویر نسبتاً متنوع و با کیفیتی تولید کند، هرچند که این مقدار به اندازه‌ای بالا نیست که نشان‌دهنده تولید تصاویر کاملاً طبیعی باشد. همچنین، مقدار FID برابر با 72.94 است که نشان‌دهنده فاصله بین توزیع ویژگی‌های تصاویر واقعی و تولید شده دارد. این مقادیر به ما می‌گویند که مدل هنوز به صورت کامل قادر به تولید تصاویری با کیفیت مشابه تصاویر واقعی نیست و نیاز به بهبود دارد.

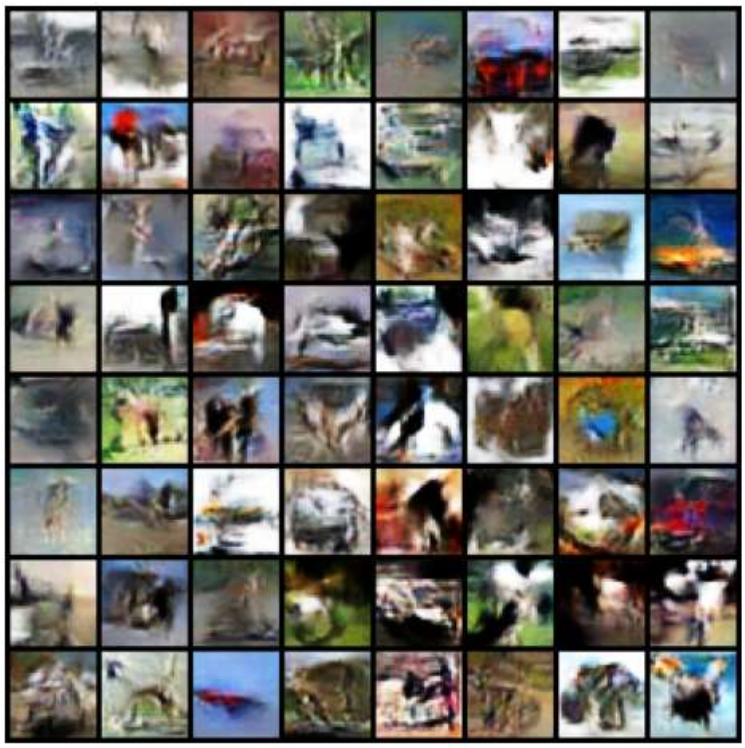
FID 72.94 IS 3.23 ± 0.08

نمودار Loss نشان‌دهنده کاهش اولیه Loss مربوط به Generator تا حدود 5 اپک و سپس افزایش تدریجی آن تا سطح 1.6 در پایان دوره‌های آموزشی است، در حالی که Loss مربوط بهDiscriminator پس از افزایش اولیه، به‌تدریج کاهش یافته و در محدوده 0.8 تا 1.0 به تثبیت نسبی نزدیک می‌شود؛ این روند بیانگر کاهش پایداری این مدل و احتمال ضعف مولد در فریب متمایزکننده است که می‌تواند ناشی از تعداد اپک کمتر باشد.



.5

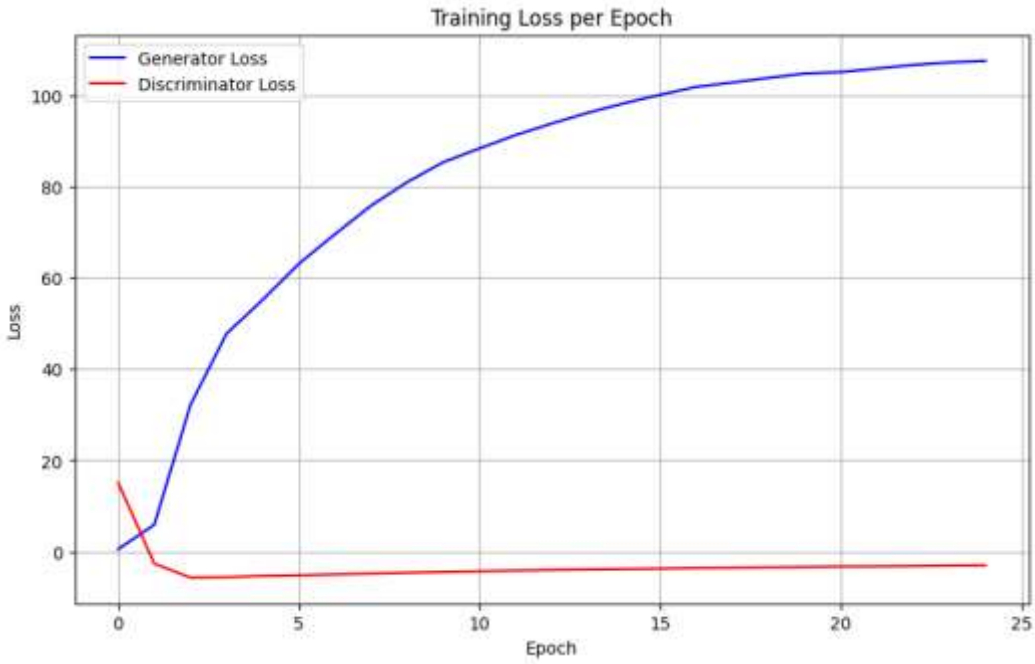
برای پیاده‌سازی مدل WGAN-GP (Wasserstein GAN with Gradient Penalty) ، ابتدا معماری اصلی این مدل را طراحی کردیم که از یک Generator و Discriminator استاندارد تشکیل شده است. در این مدل، Generator با ورودی تصادفی، تصاویری تولید می‌کند که باید از لحاظ توزیع مشابه با داده‌های واقعی باشند. Discriminator یا همانCritic ، این بار نه به‌صورت کلاسیک، بلکه به‌عنوان یک تابع فاصله Wasserstein عمل می‌کند و مقادیر امتیازی برای هر تصویر تولید شده ارائه می‌دهد. برای بهبود عملکرد مدل و جلوگیری از ناپایداری‌های معمول در آموزش GAN ها، از تکنیک Gradient Penalty استفاده می‌کنیم که با اعمال یک جریمه به گرادیان‌ها موجب بهبود همگرایی مدل می‌شود. در بخش آموزش، Generator و Discriminator به‌طور متناوب با استفاده از تابع زیان Wasserstein و جریمه گرادیان به‌روز می‌شوند. به‌طور خاص، Discriminator باید مقادیر مشابه‌تری برای تصاویر واقعی و مصنوعی ایجاد کند و Generator باید تصاویری تولید کند که Discriminator قادر به تمایز آن‌ها از تصاویر واقعی نباشد. این به‌روزرسانی‌ها باعث می‌شود که Generator بتواند تصاویر واقع‌گرایانه‌تری تولید کند و Discriminator به‌طور مؤثرتری این تصاویر را ارزیابی کند. در ادامه خروجی تصاویر این بخش را نیز نمایش می‌دهیم:



برای ارزیابی مدل WGAN-GP نیز از دو معیار IS و FID استفاده کردیم. مقدار IS برابر با 3.29 به‌دست آمد که نشان‌دهنده توانایی مدل در تولید تصاویر متنوع و با ویژگی‌های پیچیده است؛ همچنین، مقدار FID برابر با 72.94 به‌دست آمد که از فاصله قابل توجهی میان توزیع تصاویر واقعی و تولید شده حکایت دارد. با این حال، این مقدار هنوز در محدوده متوسط قرار دارد و بهبودهایی در تولید تصاویر واقعی‌تر می‌تواند نتایج بهتری ارائه دهد.

IS 3.29 ± 0.07 FID 72.94

نمودار Loss برای مدل WGAN-GP نشان‌دهنده رفتار متمایز در فرآیند آموزشی است، با افزایش تدریجی Loss Generator از مقادیر اولیه به بیش از 100 در طول 25 اپک، در حالی که Loss Discriminator پس از کاهش اولیه به سطح نزدیک به صفر در یک دامنه پایین و پایدار باقی می‌ماند. این الگو حاکی از تلاش Generator برای اکتشاف گسترده‌تر فضای Latent و تولید تنوع بیشتر در تصاویر است، در حالی که Discriminator با ثبات در عملکرد خود بازخورد مؤثری برای بهبود کیفیت ارائه می‌دهد. برای بهینه‌سازی بیشتر، افزایش تعداد اپک‌ها می‌تواند به دستیابی به همگرایی بهینه و کیفیت بالاتر تصاویر منجر شود.



.6

برای تحلیل و بررسی بهتر ویژگی‌های یادگرفته‌شده توسط مدل‌های GAN، در این بخش از روش کاهش ابعاد داده‌ها با استفاده از t-SNE بهره بردیم. این تکنیک به ما امکان می‌دهد که نمونه‌های تولیدشده به عنوان مثال توسط دو مدل DCGAN و cGAN را در فضای دو بعدی به تصویر بکشیم و ساختار و توزیع داده‌های تولید شده را به‌صورت بصری مورد ارزیابی قرار دهیم. با مقایسه نتایج t-SNE برای هر دو مدل می‌توانیم تفاوت‌ها و شباهت‌های کیفیت و تنوع نمونه‌های تولید شده را بهتر درک کنیم و در نتیجه عملکرد مدل‌ها را از دیدگاه درک ویژگی‌های پنهان بررسی کنیم.

- **DCGAN:**

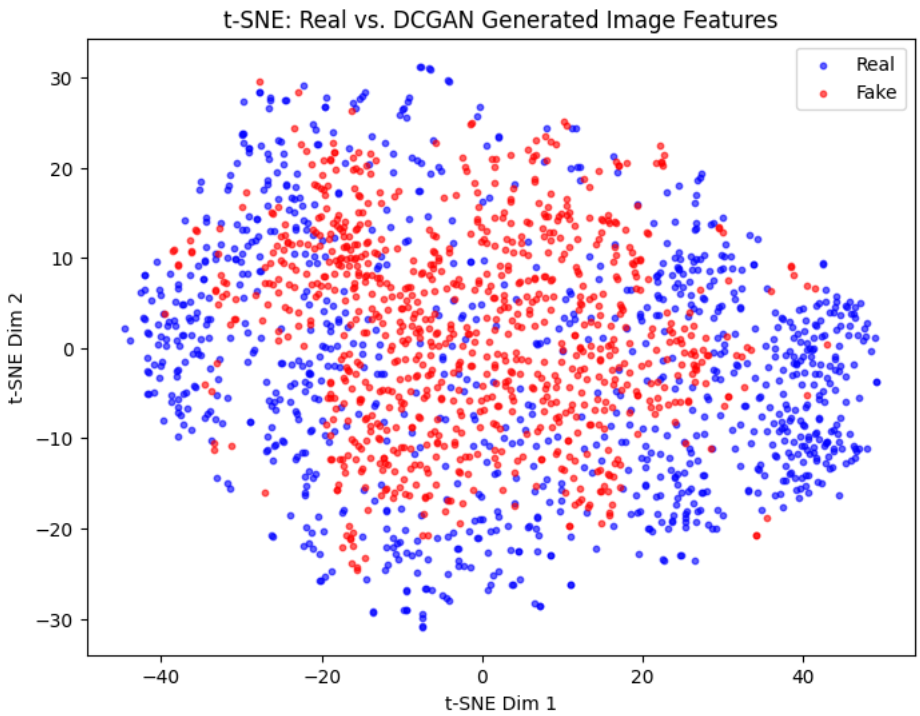
در این بخش از همان مدل پارت 1 استفاده کردیم و برای این منظور ابتدا همان مدل DCGAN آموزش‌دیده را به‌کار گرفتیم و 1000 تصویر تولید کردیم. سپس به منظور بررسی ویژگی‌های استخراج‌شده از این تصاویر در مقایسه با نمونه‌های واقعی، از مدل ResNet-18 از پیش‌آموزش‌دیده استفاده کردیم تا بردارهای ویژگی هر تصویر (هم واقعی و هم تولیدشده) را استخراج کنیم. این مرحله کمک می‌کند تا با نگاه دقیق‌تر و در فضای ویژگی، کیفیت و تنوع تصاویر تولیدی را ارزیابی کنیم. پس از استخراج بردارهای ویژگی، از t-SNE استفاده کردیم تا این ویژگی‌های با ابعاد بالا را به فضای دوبعدی نگاشت کنیم و امکان مشاهده‌ی بصری خوشه‌بندی و توزیع تصاویر را فراهم کنیم. این نگاشت دو بعدی به ما امکان می‌دهد تا شباهت یا اختلاف توزیع داده‌های واقعی و تولیدشده را به شکل

واضح‌تری بررسی کنیم و نقاط قوت یا ضعف مدل را بهتر درک کنیم. در نهایت، نمودارهای حاصل از t-SNE رسم شدند که در آن‌ها می‌توان مشاهده کرد که آیا تصاویر تولیدی مدل DCGAN توانسته‌اند پوشش خوبی از فضای ویژگی داده‌های واقعی داشته باشند یا خیر.

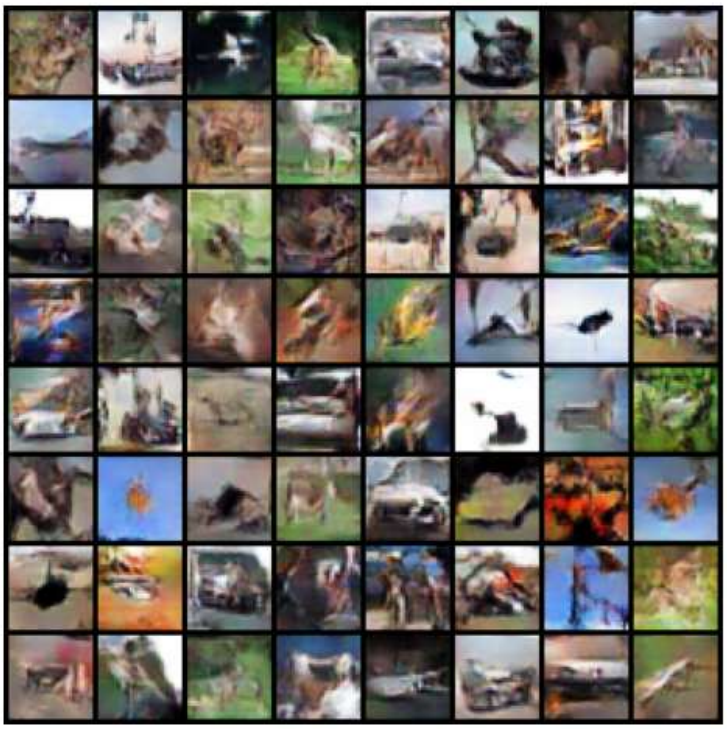
در ادامه به تحلیل خروجی‌های این قسمت که در ادامه نمایش داده شده اند می‌پردازیم:

نمودار t-SNE که ویژگی‌های تصاویر واقعی (آبی) و تولیدشده توسط DCGAN (قرمز) را در فضای دوبعدی نشان می‌دهد، بیانگر پراکندگی قابل‌توجه هر دو دسته است که با همپوشانی گسترده همراه است، اما توزیع تصاویر تولیدشده کمی گسترده‌تر و پراکنده‌تر از تصاویر واقعی به نظر می‌رسد که می‌تواند نشان‌دهنده تنوع بیشتر یا ناهمگونی در تولید تصاویر باشد. مقادیر ارزیابی IS 3.57 و FID 56.98 حاکی از کیفیت متوسط تا قابل‌قبول تصاویر تولیدشده هستند، با تنوع مناسب (IS) و فاصله نسبی از توزیع واقعی (FID) که نشان‌دهنده تلاش موفق مدل برای تقلید از داده‌های اصلی است. ثبات پراکندگی در نمودار و نزدیکی نسبی خوشه‌ها، منعکس‌کننده تعادل نسبی بین Generator و Discriminator است، هرچند پراکندگی بیشتر تصاویر تولیدشده ممکن است نیاز به تنظیمات بیشتر برای همگرایی دقیق‌تر و یا تعداد اپک بیشتر برای آموزش کافی مدل باشد.

FID 56.98 IS 3.57 ± 0.10

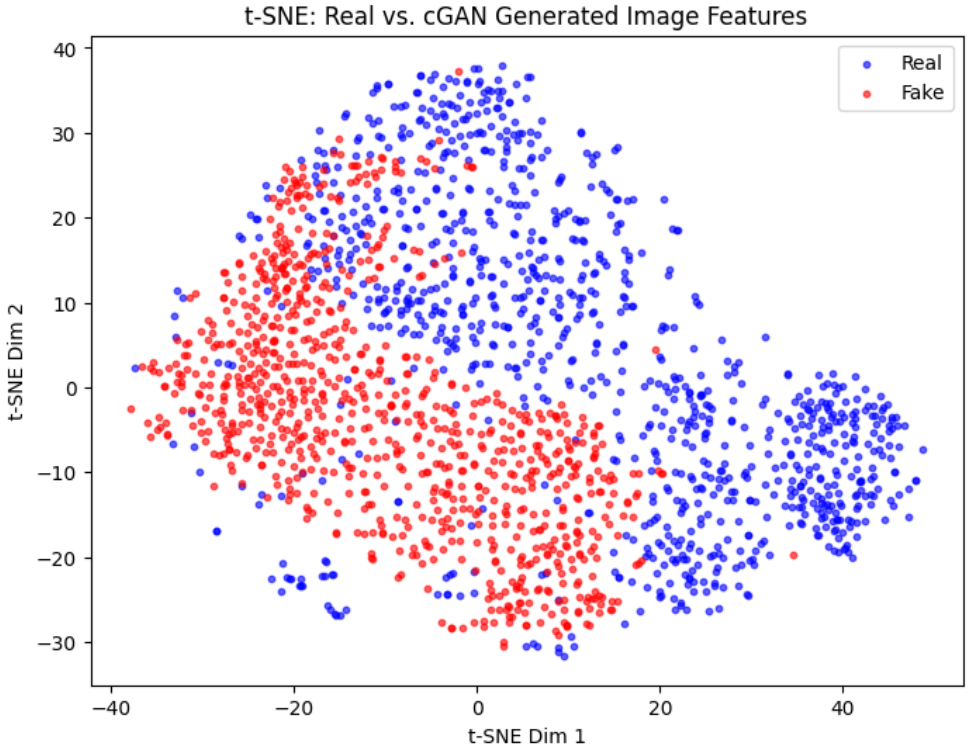

t-SNE: Real vs. DCGAN Generated Image Features
Real
Fake
t-SNE Dim 2
t-SNE Dim 1
30
20
10
0
−10
−20
−30
−40
−20
0
20
40

- **CGAN:**



نمودار t-SNE که ویژگی‌های تصاویر واقعی (آبی) و تولیدشده توسط cGAN (قرمز) را در فضای دوبعدی نشان می‌دهد، پراکندگی گسترده‌ای را برای هر دو دسته نمایش می‌دهد، با همپوشانی قابل‌توجه اما توزیع تصاویر تولیدشده که به‌طور محسوسی از تصاویر واقعی جدا شده و در بخش‌هایی پراکندگی کمتری دارد، که می‌تواند نشان‌دهنده محدودیت در تنوع یا انطباق ناقص با توزیع واقعی باشد. مقادیر ارزیابی IS 3.16 و FID 88.66 حاکی از کیفیت پایین‌تر تصاویر تولیدشده نسبت به مدل‌های بهینه هستند، با تنوع محدود (IS پایین) و فاصله قابل‌توجه از توزیع واقعی (FID بالا) که نشان‌دهنده چالش‌هایی در تطابق مدل با داده‌های اصلی است. ثبات نسبی پراکندگی در نمودار و جدایی نسبی خوشه‌ها، منعکس‌کننده تلاش مدل برای حفظ تعادل بین Generator و Discriminator است، اما پراکندگی ناهمگن و فاصله بیشتر تصاویر تولیدشده از واقعی، نشان می دهد که عملکرد بدتری نسبت به مدل cGAN که بالاتر نشان دادیم دارد.

FID 88.66 IS 3.16 ± 0.03



.7

❖ **نقاط قوت و ضعف GANها در مقایسه با سایر مدل‌های مولد مانند VAEها**

-نقاط قوت:

- **کیفیت بصری بسیار بالا** GANها: به دلیل ساختار رقابتی بین Generator و Discriminator، قادر به تولید نمونه‌هایی با کیفیت بصری بسیار بالا و جزئیات دقیق هستند. این ویژگی در کاربردهایی مثل تولید چهره‌های واقعی یا هنر دیجیتال، باعث برتری محسوس GANها نسبت به مدل‌هایی مانند VAE یا Flow ها شده است.
- **عدم نیاز به تعریف صریح تابع چگالی احتمال:** برخلاف VAE و Normalizing Flow که برای مدل‌سازی داده‌ها نیاز به تعریف صریح توزیع احتمال دارند، GANها تنها به یک تابع افتراق برای تشخیص واقعی یا جعلی بودن نمونه نیاز دارند و نیازی به مدل‌سازی دقیق احتمال داده‌ها ندارند.
- **توانایی مدل‌سازی توزیع‌های پیچیده:** معماری‌های GAN به دلیل انعطاف بالای خود قادرند توزیع‌های پیچیده و پرجزئیاتی را از داده‌های واقعی بیاموزند، در حالی‌که بسیاری از مدل‌های مولد دیگر مانند VAE معمولاً در مدل‌سازی چنین توزیع‌هایی دچار ساده‌سازی بیش از حد می‌شوند.
- **سرعت تولید بالا در زمان نمونه‌گیری:** برخلاف مدل‌های مولدی مثل PixelCNN که نمونه‌سازی را به‌صورت ترتیبی و کند انجام می‌دهند، درGANها می‌توان با یک بار عبور دادن بردار نهفته از Generator، یک تصویر کامل را فوراً تولید کرد.

-نقاط ضعف:

- **عدم دسترسی به احتمال صریح برای داده‌ها** GAN :یک مدل مولد implicit است؛ به این معنا که نمی‌توان احتمال تولید یک نمونه خاص (likelihood) را مستقیماً محاسبه کرد. این موضوع باعث محدودیت در کاربردهای آماری یا فشرده‌سازی می‌شود، برخلاف VAE وFlow ها که به‌راحتی likelihood را ارائه می‌دهند.
- **چالش در کنترل فضای نهفته** :در مدل‌هایی مانند VAE یا Flow، فضای نهفته دارای ساختار مشخصی است که امکان انجام عملیات‌هایی مثل interpolating یا manipulating ویژگی‌های خاص فراهم است. اما درGAN ها، مگر در نسخه‌هایی خاص مثل StyleGAN یا InfoGAN، این فضا کنترل‌ناپذیرتر و پیچیده‌تر است.
- **پایداری پایین در آموزش** :برخلاف مدل‌هایی مانند VAE یا PixelCNN که معمولاً روند آموزش نسبتاً پایدار دارند، آموزش GAN ممکن است با نوسان‌های زیاد، فروپاشی حالت، یا واگرایی مواجه شود. این چالش در بخش بعدی به‌تفصیل بررسی شده است.
- **نیاز به تنظیمات حساس و دقیق** :عملکرد GAN بسیار به طراحی دقیق شبکه، انتخاب بهینه‌ساز، نرخ یادگیری، استفاده از تکنیک‌هایی مانند label smoothing یا spectral normalization وابسته است. در حالی‌که مدل‌هایی مثل VAE نسبت به این تنظیمات حساسیت کمتری دارند.

در مجموع GANها با وجود چالش‌های آموزشی، به دلیل کیفیت بصری بالا و انعطاف‌پذیری در مدل‌سازی توزیع‌های پیچیده، برتری قابل‌توجهی نسبت بهVAE ها دارند اما برای بهره‌مندی کامل از پتانسیل آن‌ها، نیاز به تنظیمات دقیق و تکنیک‌های پیشرفته است.

❖ چالش‌های آموزش GAN :

آموزش GAN به دلیل ساختار رقابتی و غیرمستقیم آن (به‌صورت یک بازی مین-مکس بین دو شبکه) با چالش‌های مفهومی و عددی متعددی مواجه است که در ادامه به مهم‌ترین آن‌ها می‌پردازیم:

- **ناپایداری در آموزش (Training Instability):** تعادل بین Generator و Discriminator بسیار شکننده است. اگر Discriminator خیلی سریع و قوی یاد بگیرد، Gradientsتولیدشده برای Generator به‌شدت ضعیف یا صفر می‌شوند (vanishing gradients)، در نتیجه Generator دیگر قادر به یادگیری مؤثر نیست. از طرف دیگر اگر Generator خیلی زود پیشرفت کند، Discriminator دیگر نمی‌تواند تفاوت نمونه‌های واقعی و جعلی را تشخیص دهد و یادگیری متوقف می‌شود. این بی‌تعادلی منجر به نوسان در عملکرد و افت کیفیت خروجی‌ها می‌شود.
- **فروپاشی حالت (Mode Collapse):** در این حالت، Generatorبه‌جای یادگیری توزیع کامل داده‌ها، تنها روی تولید چند نمونه محدود (یا حتی یک نمونه تکراری) تمرکز می‌کند. این پدیده زمانی اتفاق می‌افتد که تولید نمونه‌های خاص باعث فریب آسان Discriminator می‌شود، و Generator برای کاهش زیان خود، به تولید آن‌ها بسنده می‌کند. نتیجه، تنوع کم در خروجی‌ها و کاهش پوشش توزیع داده‌های واقعی است.
- **همگرایی دشوار (Convergence Difficulty):** برخلاف مدل‌های مولد کلاسیک که معمولاً بهینه‌سازی یک تابع زیان مشخص و محدب انجام می‌شود، آموزش GAN معادل حل یک بازی دوسویه مین‌مکس است که پاسخ تعادلی آن ممکن است وجود نداشته باشد یا به‌سختی قابل دستیابی باشد. در عمل، این منجر به نوسانات دائمی در lossها، عدم بهبود کیفیت خروجی، یا حتی افت آن در اواخر آموزش می‌شود.
- **حساسیت شدید به تنظیمات و ابرپارامترها:** یکی از چالش‌های اساسی در آموزش GAN، حساسیت بالای آن به تنظیمات اولیه و انتخاب دقیق ابرپارامترهاست. نرخ یادگیری Generator و Discriminator باید با دقت تنظیم شود، چون اختلاف زیاد بین آن‌ها ممکن است باعث غلبه یکی بر دیگری و ناپایداری مدل شود. انتخاب بهینه‌ساز مناسب مانند Adam و استفاده از تکنیک‌هایی مثل Batch Normalization (در Generator) می‌تواند به پایداری آموزش کمک کند. همچنین روش‌هایی مانند Label Smoothing، Spectral Normalization و Gradient Penalty اغلب برای جلوگیری از نوسانات شدید و فروپاشی حالت به کار می‌روند. در نهایت، تعادل در تعداد به‌روزرسانی‌های هر شبکه نیز اهمیت دارد و گاهی لازم است Discriminator بیشتر از Generator آموزش داده شود تا رقابت مؤثر بین آن‌ها حفظ شود..

- **مشکل تشخیص معیار توقف مناسب:** از آن‌جا که lossهای G و D مستقیماً بیانگر کیفیت نمونه‌ها نیستند، یافتن معیاری دقیق برای تشخیص پایان مناسب آموزش دشوار است. مدل ممکن است ظاهراً lossمناسبی داشته باشد اما خروجی‌هایی بی‌کیفیت تولید کند، یا بالعکس.

در مجموع، آموزش GANها به دلیل ساختار رقابتی و پیچیدگی مین‌مکس با چالش‌هایی نظیر ناپایداری، فروپاشی حالت، و همگرایی دشوار مواجه است، که حساسیت به تنظیمات و عدم وجود معیار توقف مناسب آن را پیچیده‌تر می‌کند. با این حال، استفاده از تکنیک‌های پیشرفته می‌تواند به تثبیت و بهبود فرآیند یادگیری کمک کند.

❖ راهکارهای پیشنهادی برای بهبود فرایند یادگیری و تثبیت آموزش

با توجه به چالش‌های جدی در آموزش GANها مانند ناپایداری، فروپاشی حالت و حساسیت به تنظیمات، مجموعه‌ای از تکنیک‌ها و اصلاحات را برای بهبود فرایند آموزش موجود هستند که در ادامه به مهم‌ترین آن‌ها اشاره می‌کنیم:

- **استفاده از Soft Labels:** در این روش به‌جای برچسب‌های دقیق 1 یا 0، از مقادیری مانند 0.9 برای تصاویر واقعی استفاده می‌شود. این تکنیک باعث می‌شود Discriminator با اطمینان کمتری عمل کند و فضای یادگیری بهتری برای Generator فراهم شود. ما نیز در تمرین از این روش بهره گرفتیم.

- **معماری‌های بهبود یافته مانند WGAN و WGAN-GP:** مدل Wasserstein GAN با جایگزینی تابع هزینه کلاسیک با فاصله Wasserstein و حذف تابع Sigmoid در خروجی Discriminator (که در این مدل به‌عنوان "critic" شناخته می‌شود)، پایداری بیشتری را فراهم کرده است. همچنین افزوده شدن تکنیک Gradient Penalty در WGAN-GP به کاهش نوسانات و جلوگیری از انفجار گرادیان کمک کرده است.

- **Mini-batch Discrimination:** این تکنیک با تحلیل روابط بین نمونه‌های داخل هر batch به Discriminator کمک می‌کند تا به شباهت بیش‌ازحد تصاویر تولیدی حساس‌تر شود، که می‌تواند خطر فروپاشی حالت (Mode Collapse) را کاهش دهد.

- **نرخ یادگیری تفکیک‌شده برای Generator و Discriminator:** تنظیم نرخ یادگیری متفاوت برای دو شبکه می‌تواند به تعادل بهتر در رقابت کمک کند. به‌طور مثال، در برخی مدل‌ها نرخ یادگیری کمتر برای Discriminator باعث جلوگیری از تسلط سریع آن بر Generator می‌شود.

- **نرمال‌سازی مناسب:** استفاده از Batch Normalization در لایه‌های Generator باعث همگرایی پایدارتر و سریع‌تر می‌شود. همچنین Spectral Normalization در Discriminator برای کنترل مقادیر گرادیان و بهبود پایداری شبکه پیشنهاد شده است.

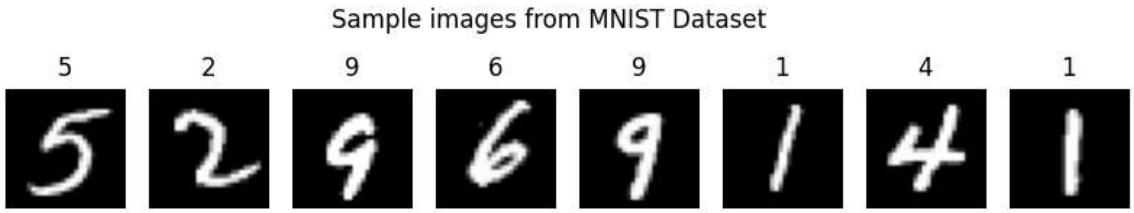
- **استفاده از معیارهای ارزیابی مانند IS و FID:** برای بررسی کیفیت و تنوع خروجی‌ها، استفاده از شاخص‌هایی نظیر Inception Score (IS) و Frechet Inception Distance (FID) ضروری است. این معیارها کمک می‌کنند نه‌تنها کیفیت بصری بلکه توان مدل در تولید تصاویر متنوع نیز پایش شود.
- **افزایش ظرفیت مدل یا استفاده از معماری‌های مدرن‌تر:** در مواردی استفاده از معماری‌هایی مانند StyleGAN یا BigGAN با طراحی دقیق‌تر و بلوک‌های تخصصی‌تر، موجب بهبود چشم‌گیر در کیفیت خروجی و پایداری آموزش شده‌اند.

مجموع این راهکارها اگر به‌درستی و متناسب با مسئله به‌کار گرفته شوند، می‌توانند به طرز قابل‌توجهی روند یادگیریGAN ها را تثبیت کرده و نتایجی با کیفیت بالا و تنوع مناسب ارائه دهند.

Question 2:

1.

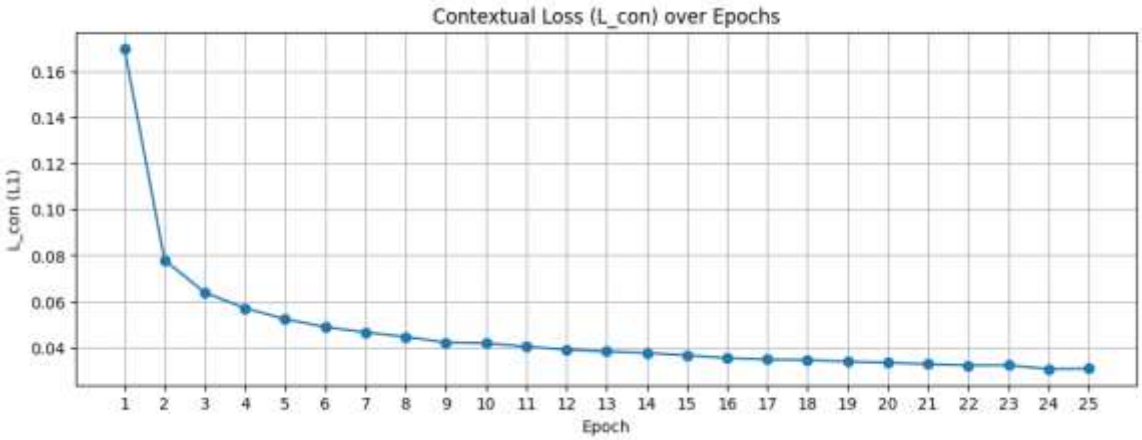
در این پارت از مجموعه داده‌ی استاندارد MNIST که شامل تصاویر عددهای دست‌نویس 0 تا 9 استفاده کردیم. داده‌ها با استفاده از کتابخانه‌ی torchvision لود و به قالب تنسور تبدیل شدند. در نهایت نیز چند نمونه از نمونه های این دیتاست را در ادامه نمایش دادیم:



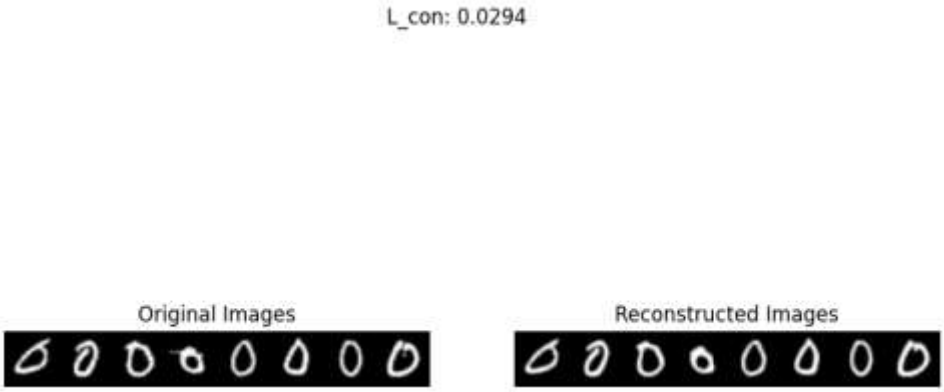
2&3.

در پارت دوم و سوم برای بازسازی تصاویر MNIST، یک خودکدگذار قرینه طراحی کردیم که از یک Encoder و یک Decoder تشکیل شده است. ساختار Encoder شامل چهار لایه‌ی کانولوشنی با فیلترهای 64، 128، 256 و در نهایت 100 است که بازنمایی فشرده تصویر را در قالب یک تنسور 100×1×1 تولید می‌کند. Decoder نیز به‌صورت قرینه عمل کرده و با استفاده از لایه‌های ConvTranspose، تصویر اولیه را بازسازی می‌کند. در تمامی لایه‌ها از Batch Normalization و توابع فعال‌سازی LeakyReLU یا ReLU استفاده شده است.

برای ارزیابی کیفیت بازسازی از تابع Contextual Loss (L_con) استفاده کردیم که در اینجا با معیار L1 (MAE) پیاده‌سازی شده و میزان اختلاف پیکسل‌به‌پیکسل بین تصویر اصلی و بازسازی‌شده را محاسبه می‌کند. برای آموزش با توجه به مقاله ذکرشده تنها تصاویر مربوط به یک کلاس (مثلا کلاس 0) از دیتاست MNIST فیلتر شده و مورد استفاده قرار گرفتند تا شرایط یادگیری فقط از داده‌های نرمال برقرار باشد. شبکه با استفاده از بهینه‌ساز Adam به مدت 25 اپک آموزش داده شد. در هر اپک مقدار L_con ثبت و نمودار آن رسم گردید و در ادامه نمودار مربوطه و همچنین نمونه‌هایی از تصاویر واقعی و بازسازی‌شده را برای بررسی عملکرد مدل نیز نمایش داده‌ایم:



نمودار L_con بالا که کاهش Contextual Loss را طی 25 اپک آموزشی نشان می‌دهد، بیانگر بهبود مداوم کیفیت بازسازی تصاویر MNIST است، به‌طوری که مقدار اولیه حدود 0.16 به تدریج به زیر 0.04 در دوره‌های پایانی رسیده است. این کاهش تند در مراحل اولیه (تا حدود اپک 5) و سپس روند ملایم‌تر تا پایان حاکی از تطابق سریع اولیه مدل با داده‌ها و همگرایی پایدار در فازهای بعدی است. ثبات کم‌نوسان L_con در اواخر آموزش، منعکس‌کننده تعادل مناسب در فرآیند یادگیری خودکدگذار و توانایی آن در بازسازی دقیق‌تر تصاویر کلاس 0 است که با مشاهده بصری تصاویر بازسازی‌شده نیز تأیید می‌شود.



تصاویر بازسازی‌شده از مجموعه داده MNIST کلاس 0 که با مقدار L_con =0.0294 ارزیابی شده‌اند، نشان‌دهنده توانایی خوب خودکدگذار در بازسازی شکل کلی اعداد صفر هستند، هرچند برخی جزئیات ظریف (مانند لبه‌های ناهموار) در مقایسه با تصاویر اصلی کمی گم شده‌اند. این مقدار کم L_con تأییدکننده کاهش خطا طی آموزش و تطابق مناسب مدل با داده‌های نرمال است، اما تفاوت‌های بصری جزئی بین تصاویر اصلی و بازسازی‌شده حاکی از پتانسیل بهبود بیشتر در دقت بازسازی با تنظیمات اضافی یا افزایش دوره‌های آموزشی است.

.4

در این پارت از یک Encoder مشترک در سه بخش مختلف استفاده شده است: فشرده‌سازی تصویر ورودی، فشرده‌سازی تصویر بازسازی‌شده ($\hat{x}$) و ورودی شبکه Discriminator. ساختار Encoder شامل چند لایه کانولوشن و BatchNorm است که بردار z را استخراج می‌کند. Decoder نیز با لایه‌های Transposed Convolution تصویر $\hat{x}$ را بازسازی می‌کند.Generator از ترکیب Encoder و Decoder ساخته شده و AuxiliaryEncoder همان Encoder را روی $\hat{x}$ اعمال می‌کند تا $\hat{z}$ به‌دست آید. شبکه Discriminator نیز از لایه‌های Fully Connected برای تشخیص واقعی یا جعلی بودن بردار z استفاده می‌کند. در پایان یک batch نمونه از مدل عبور داده می‌شود و خروجی هر بخش چاپ می‌گردد. این طراحی به‌خوبی هدف اشتراک‌گذاری Encoder را در سه مرحله مدل محقق کرده است.

```
Input image shape:      torch.Size([64, 1, 32, 32])
Reconstructed x̂ shape: torch.Size([64, 1, 32, 32])
z shape:                torch.Size([64, 100, 1, 1])
ẑ shape:               torch.Size([64, 100, 1, 1])
D(z) shape:             torch.Size([64, 1])
```

«. . . خردادماه 1404 . . .»