



به نام خدا



دانشگاه تهران
دانشکده مهندسی برق و کامپیوتر
شبکه های عصبی و یادگیری عمیق

مینی پروژه ۳

سجاد پاکدامن ساوجی فاطمه حقیقی	نام و نام خانوادگی
۸۱۰۱۹۵۵۱۷ ۸۱۰۱۹۵۳۸۵	شماره دانشجویی
۱۴ مرداد ۱۳۹۹	تاریخ ارسال گزارش

فهرست گزارش سوالات

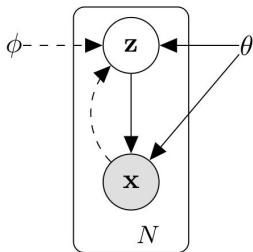
3	سوال ۱ - VAE
3	توضیح ساز و کار معماری:
4	نتایج مورد نیاز پس از اجرای شبکه:
6	پاسخ سوالات:
7	سوال ۱ - Efficient Net (امتیازی)
9	سوال ۲ - DC-GAN
9	توضیح ساز و کار معماری:
10	نتایج مورد نیاز پس از اجرای شبکه:
10	پاسخ به سوالات:
12	سوال ۳ - CGAN
12	توضیح ساز و کار معماری:
13	نتایج مورد نیاز پس از اجرای شبکه:
21	سوال ۳ - AC-GAN
21	توضیح ساز و کار معماری:
22	نتایج مورد نیاز پس از اجرای شبکه:

سوال ۱ - VAE

توضیح ساز و کار معماری:

چکیده: در این مقاله روشی برای استنباط و آموزش احتمالاتی¹ در مجموعه داده های بزرگ معرفی می شود. در نتیجه تحقیق دو مورد ارزشمند بدست آمده است، ابتدا حد پایینی برای تخمین گر بدست آمده و نشان داده می شود که با روش های معمول گرادیان بیس می توان آن ها را بهینه سازی کرد. در ادامه نشان داده می شود که تحلیل های posterior با استفاده از این تخمین گر بسیار موفق خواهد بود یا به عبارتی از فضای ثانویه می توان برای آموزش مدل های دیگر و یا استنباط استفاده کرد.

¹ stochastic variational inference and learning



شکل ۱. شمایی از رابطه متغیر z , x در مسئله

مسئله که نویسنده تلاش به حل آن دارد در شکل ۱ آورده شده است، در این شکل متغیر z متغیر پنهان است و مجموعه داده در دسترس ما به صورت $\{x^i\}_{i=1..Q} = X$ است. همچنین رابطه متغیرها و پارامتر θ از طریق خانواده چگالی احتمال $p_\theta(z|x)$ برقرار است. در این مسئله ۳ خواسته وجود داشته است، ۱. تخمین بهینه چگالی احتمال $p_\theta(z|x)$ ۲. احتمال posterior متغیر z با داشتن متغیر x . بدست آوردن چگالی مارجینال $q_\phi(z|x)$ ۳. برای این که این مسئله را حل کنیم، recognition model را به صورت $q_\phi(z|x)$ را در نظر می‌گیریم که تخمینی از چگالی $p_\theta(z|x)$ می‌باشد. در روش ارائه شده در $q_\phi(z|x)$ یا همان ϕ reconstruction parameter همراه با θ همزمان تخمین زده می‌شوند. هدف بیشینه کردن marginal log generative parameter likelihood است که بهینه کردن حد پایین زیر است.

$$\mathcal{L}(\theta, \phi; \mathbf{x}^{(i)}) = -D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})||p_\theta(\mathbf{z})) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})} [\log p_\theta(\mathbf{x}^{(i)}|\mathbf{z})]$$

مشکل تابع هزینه بالا در خوش فورم نبودن گرادیان آن نسبت به ϕ است بنابراین از تکنیک reparametrization استفاده شده و تابع هزینه به صورت زیر بدست می‌آید.

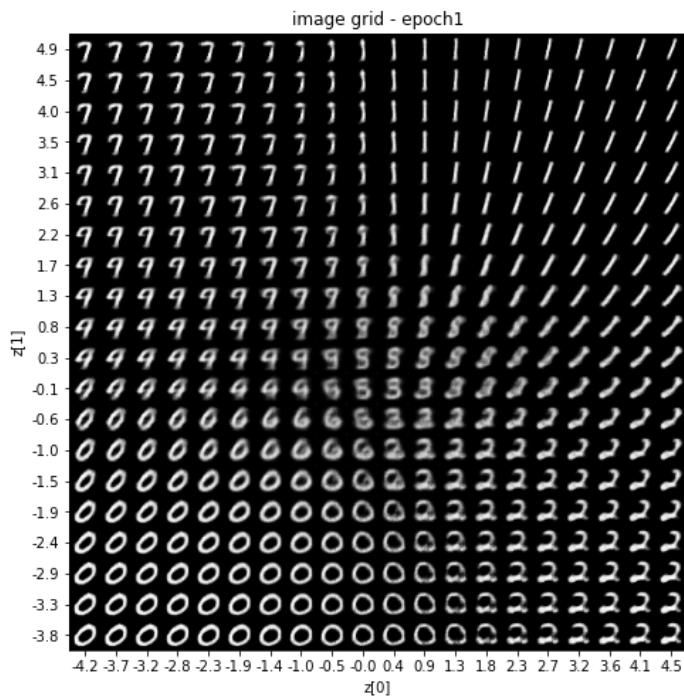
$$\tilde{\mathcal{L}}^A(\theta, \phi; \mathbf{x}^{(i)}) = \frac{1}{L} \sum_{l=1}^L \log p_\theta(\mathbf{x}^{(i)}, \mathbf{z}^{(i,l)}) - \log q_\phi(\mathbf{z}^{(i,l)}|\mathbf{x}^{(i)})$$

where $\mathbf{z}^{(i,l)} = g_\phi(\epsilon^{(i,l)}, \mathbf{x}^{(i)})$ and $\epsilon^{(l)} \sim p(\epsilon)$

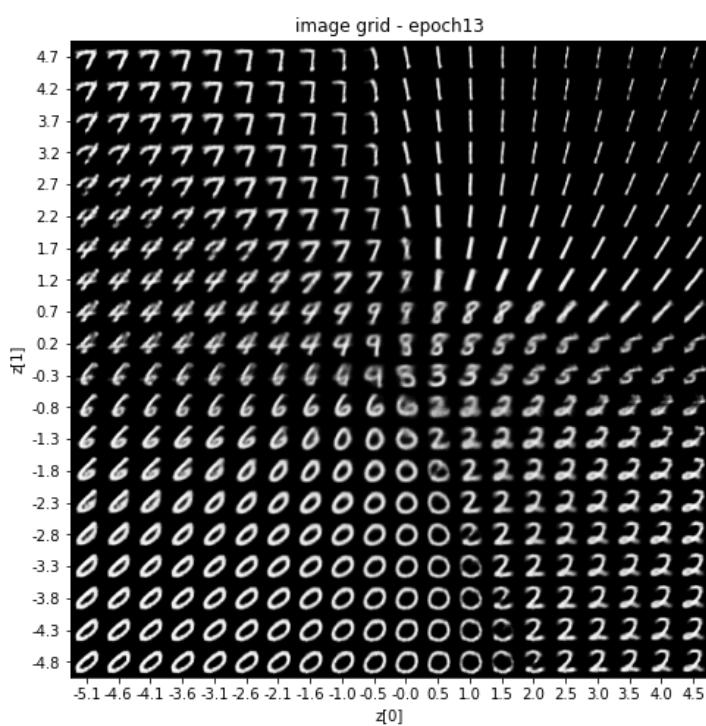
در تکنیک reparametrization با فرض پیوست بودن z توزیع احتمال $q_\phi(z|x)$ را برحسب یک تابع دترمینیستیک به صورت $g_\phi(\epsilon, x) = z$ می‌نویسیم که در آن $\epsilon \sim p(\epsilon)$ است. در شبکه نیز دقیقاً از همین روش استفاده می‌شود.

نتایج مورد نیاز پس از اجرای شبکه:

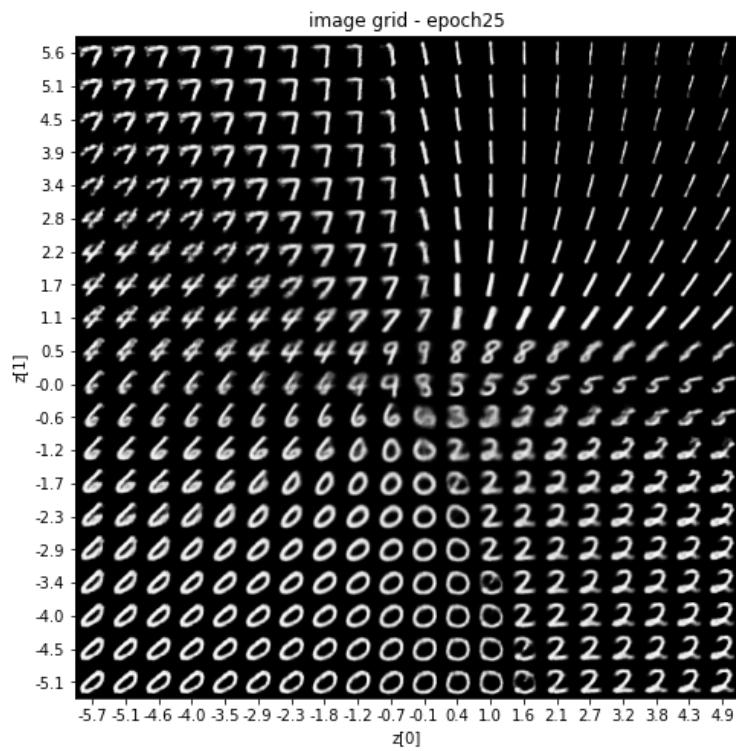
۱. نمونه تصویر تولید شده: چگونگی تغییر خروجی ها در فایل vae_grid.gif آورده شده است.



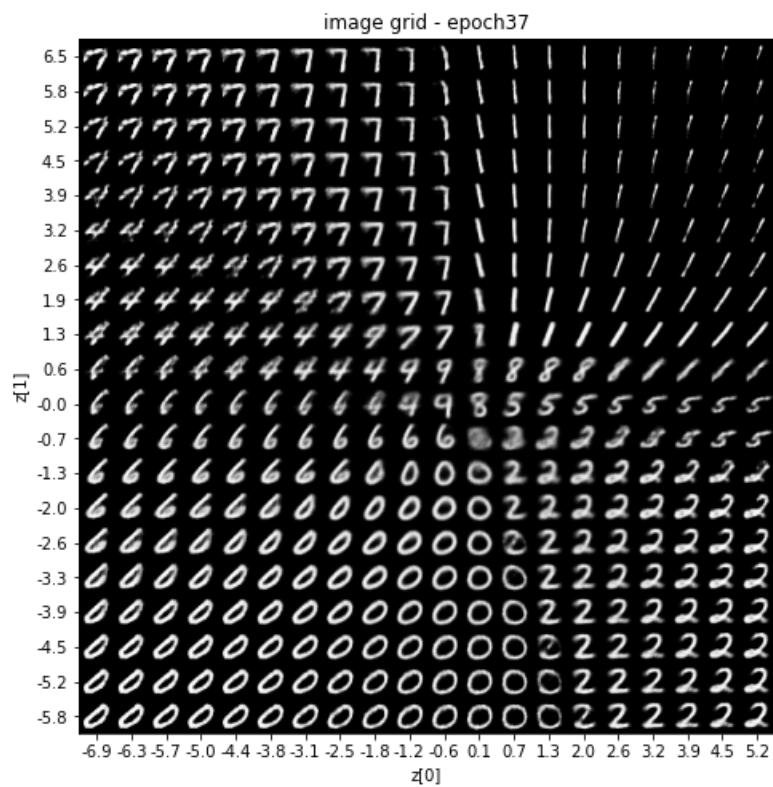
شکل ۲: نمونه تصویر تولید شده



شکل ۳: نمونه تصویر تولید شده



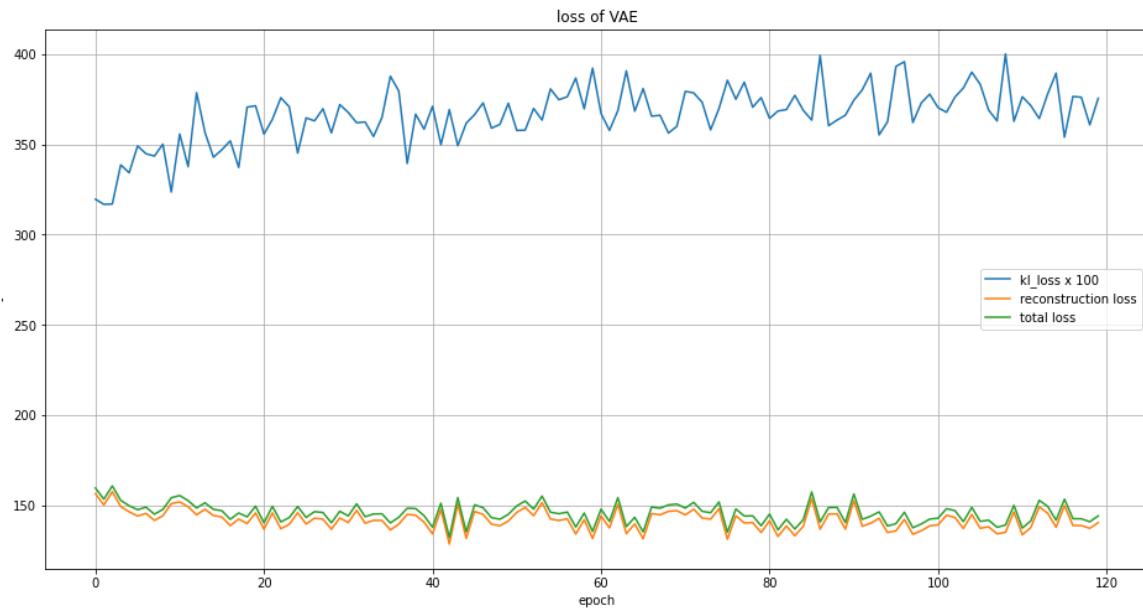
شکل ۳: نمونه تصویر تولید شده



شکل ۴: نمونه تصویر تولید شده

۲. نمودار دقت و هزینه

* دقت شود که در نمودار هزینه KL با ضریب ۱۰۰ نمایش داده شده است.



شکل ۵: نمودار هزینه در آموزش شبکه VAE

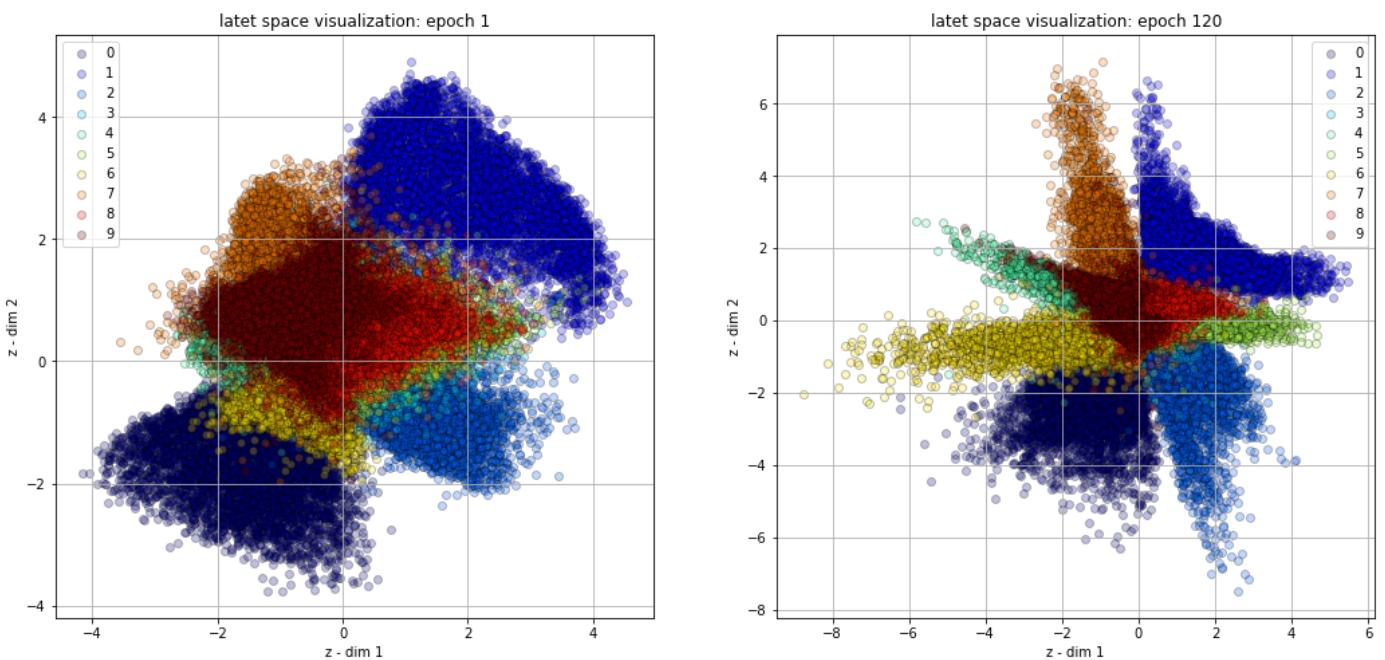
پاسخ سوالات:

(الف) ابتداً ترین تقاضاً مدل‌های VAE و GAN در نحوه کارکرد شبکه و آموزش آن است. شبکه‌های VAE مدل‌های هندسی هستند که در آن‌ها تلاش می‌شود حد پایین variational log likelihood کاهش پیدا کند. در عوض شبکه‌های GAN در مکانیزم رقابتی بهبود پیدا می‌کنند. خاصی بسیار مهم شبکه‌های VAE در همگن و پیوسته بودن فضای ثانویه است، این ویژگی به علت وجود لایه نمونه برداری ایجاد می‌شود. در فضای ثانویه VAE نقاط با فاصله هندسی کم از هم به خروجی‌های مشابهی تبدیل می‌شوند در صورتی که در GAN رابطه خاصی میان فضای ثانویه و فضای خروجی نخواهد بود. (دقت شود که فضای ثانویه در GAN‌ها همان نویز است و در شبکه‌هایی همانند infoGAN‌ها، ورودی‌های شرطی را به عنوان فضای ثانویه در نظر نمی‌گیریم. برتری GAN‌ها در ایجاد خروجی‌های واقعی تر و با کیفیت‌تر است).

(ب) در قسمت قبل نتایج آورده شده است

(ج) برای آموزش شبکه VAE از توابع هزینه KL Divergence و cross entropy استفاده کردیم، اولین تابع هزینه برای ایجاد تقسیم بین کلاس‌های مختلف است و دومین تابع هزینه برای برای نزدیک کردن کلاستر‌های داده‌ها به یک دیگر است.

(د) شما فضای ثانویه در شکل زیر آمده است، همچنین تغییرات فضای ثانویه در فرایند آموزش در فایل vae_latent.gif آورده شده است.

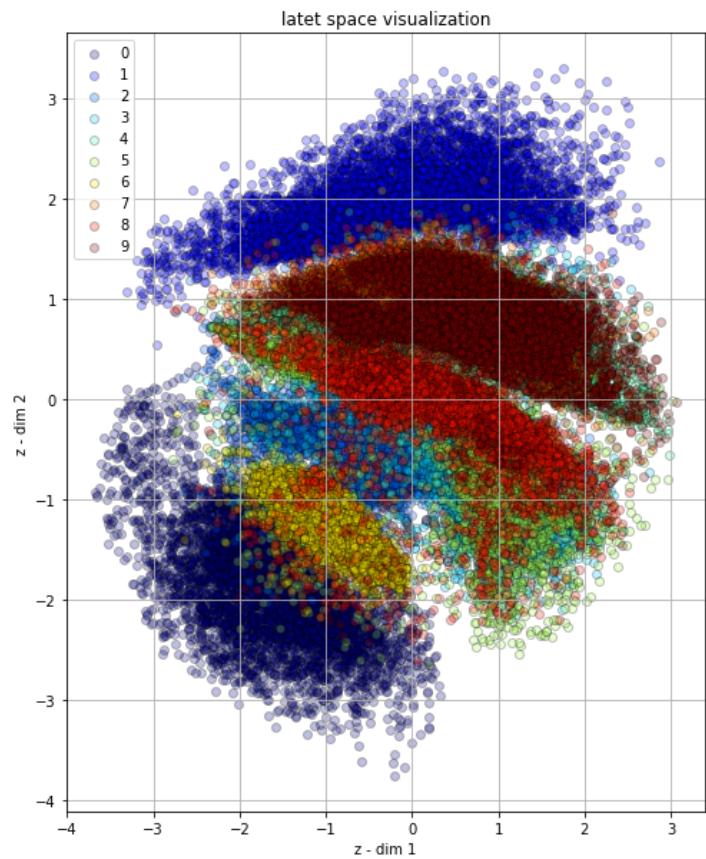


شکل ۶ : نمایش فضای ثانویه در شبکه VAE

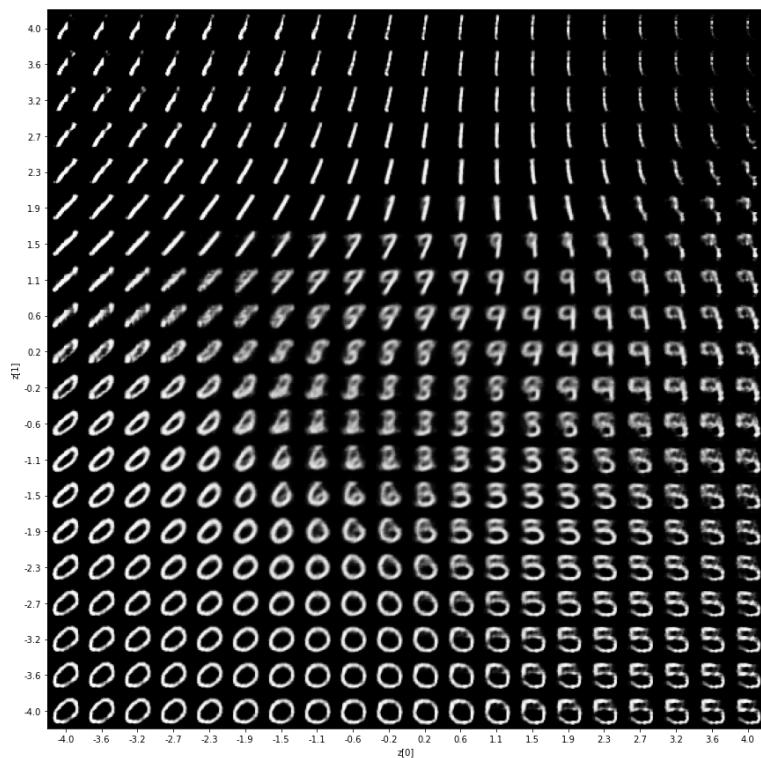
سوال ۱ - Efficient Net (امتیازی)

در این سوال با استفاده از efficient net شبکه VAE مجدداً پیاده سازی شد، از شبکه encoder استفاده می شود چرا که نسخه از پیش آموزش داده شده شبکه توانایی استخراج ویژگی خوبی دارد.

نمونه‌ای از خروجی شبکه حاصل:



شكل 7: latent space visualization



شكل 8: نمونه تصویر تولید شده

الف) توضیح ساز و کار معماری:

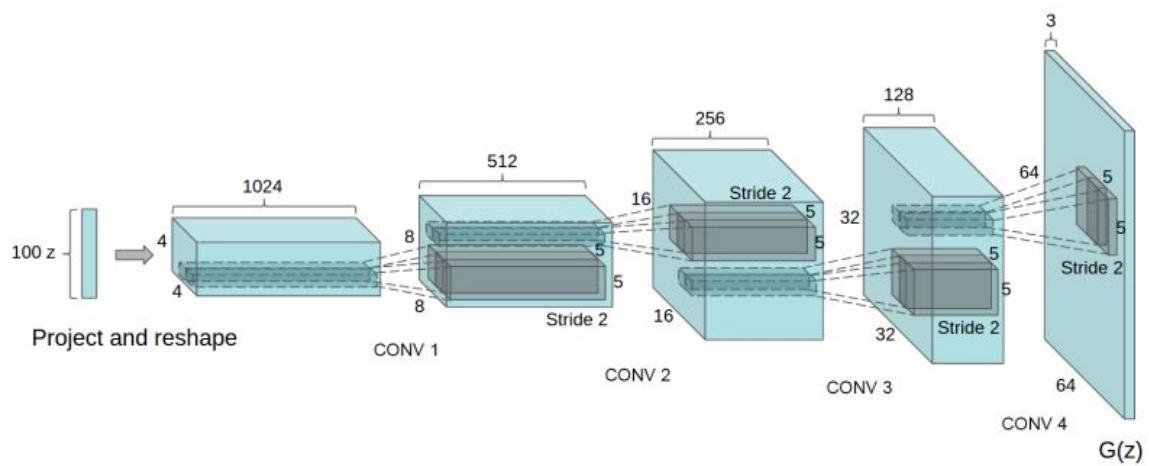
چکیده: شبکه های عصبی کانولوشنی در گذشته عملکرد چشمگیری بر مسئله های با سرپرستی داشته اند، نویسنده امیدوار است با این تحقیق توانایی عملکرد شبکه های عمیق کانولوشنی در مسئله های بی سرپرست را نیز نشان دهد. شبکه معرفی شده Deep Convolutional GAN نام دارد. نویسنده بیان می کند که شواهد کافی در نتایج وجود دارد که نشان می دهد هردو شبکه مولد و تشخیص گر به صورت سلسله مراتبی ویژگی های تصاویر را استخراج می کنند و آن را تولید می کنند.

در گذشته پیدا کردن فضایی ثانویه برای بازنمایی تصاویر حوزه فعالی در تحقیقات بوده است. پیدا کردن چنین فضایی که هم بتوان تصاویر را در آن فشرده کرد و هم بتوان تصاویر فشرده شده را بازیابی کرد از اهمیت ویژه ای برخوردار است. در این مقاله ادعا شده است که از شبکه generator و شبکه discriminator می توان برای یافتن چنین بازنمایی کمک گرفت.

در تحقیقات پیشین، عموماً مولد های GAN بسیار ناپایدار بوده اند و بعضاً تصاویر نا متعارفی تولید می کرده اند، همچنین در گذشته به هیچ شیوه ای نشان داده نشده است که Generator واقعاً چه چیزی را یاد می کیرد. این مقاله ۳ دست آورده اصلی دارد:

۱. معماری جدید به نام DCGAN معرفی می شود که نتایج نسبتاً پایداری را تولید می کند.
۲. از قسمت discriminator برای مسئله طبقه بندی تصاویر استفاده می کنیم و نشان می دهیم که نتایج با شبکه های کانولوشنی قوی رقابت می کند.
۳. فیلتر های شبکه نشان داده شده اند که بعضاً مشابه اجسام خاصی هستند.
۴. نشان داده می شود که فضای خروجی مولد خواص منحصر به فرد برداری دارد که می توان تغییرات خاصی را با استفاده از آنها ایجاد کرد.

در معماری DCGAN چند قانون وجود دارد که شبکه را پایدار می کند، اول این که شبکه تماماً کانولوشنی است و از لایه های pooling استفاده نمی شود. به جای آن از Stride خاص استفاده می شود. ثانیه استفاده از لایه های batch normalization در تمامی لایه ها بجز لایه های ورودی-خرجی regenerator و discriminator است. همچنین تمامی توابع فعال سازی در شبکه generator تابع relu است و تمامی توابع فعال ساز در discriminator تابع LeakyRelu هستند. معماری تمام کانولوشنی در شکل زیر آمده است.



شکل 9 : ساختار تمام کاتولوشنی در شبکه مولد

قسمت ب)

در این پروژه در قسمت generator شبکه به وسیله‌ی تابع initializers.RandomNormal از شبکه به وسیله‌ی نویزهای اولیه را به صورت رندم تولید می‌کنیم.

قسمت (ج) بیانگر این است که در هر مرحله از convolution چند step حرکت می‌کنیم . دلیل استفاده از آن این است که سایز output ما کمتر از input شود.

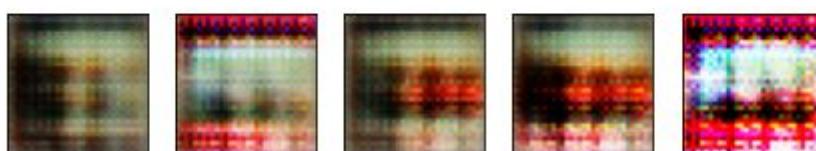
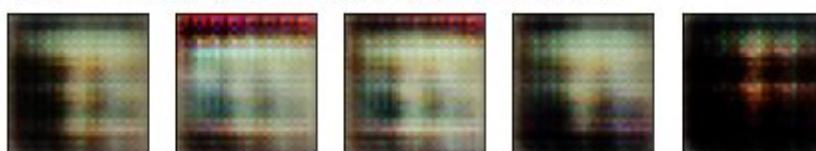
با استفاده از fractional stride convolution ، ما هسته را در منطقه‌ای از ورودی کوچکتر از خود هسته استفاده می‌کنیم

قسمت د)

نتایج مورد نیاز پس از اجرای شبکه:

۱. نمونه تصویر تولید شده پس از ایپاک اول:

epoch = 1/60, d_loss=0.479, g_loss=2.511



شکل ۱۰: نمونه تصویر تولید شده پس از ایپاک اول

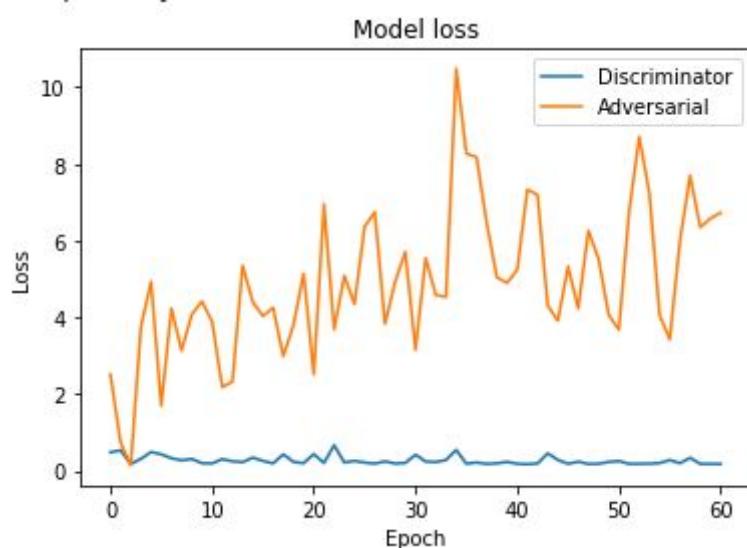
نمونه تصاویر تولید شده پس از ایپاک آخر:

epoch = 61/60, d_loss=0.175, g_loss=6.724



شکل ۱۱: نمونه تصویر تولید شده پس از ایپاک آخر

۲. نمودار دقت و هزینه‌ی این مدل به صورت زیر می‌باشد:



شکل ۱۲: تابع هزینه‌ی مدل

قسمت ۵

تابع هزینه‌ی Minimax Loss: Minimax به یک استراتژی بهینه سازی در بازی‌های مبتنی بر دو نفره برای به حداقل رساندن ضرر و هزینه برای بدترین حالت بازیکن دیگر اشاره دارد.

تابع هزینه‌ی Non-Saturating GAN Loss: به وسیله‌ی این تابع با ایجاد یک تغییر در خطای مربوط به generator از مشکل اشباح شده جلوگیری می‌کنیم.

تابع هزینه‌ی Least Squares GAN Loss: رویکرد مبتنی بر مشاهده محدودیت‌های استقاده از از دست دادن آنتروپی ضربدری باینری است که تصاویر تولید شده با تصاویر واقعی بسیار مقاومت هستند، که می‌تواند منجر به شبیه‌های بسیار کوچک یا ناپذید شده شود، و به نوبه خود، بروزرسانی کمی یا اصولی برای مدل ایجاد نمی‌کند.

تابع هزینه‌ی Wasserstein GAN Loss: با مشاهده اینکه GAN سنتی انگیزه دارد تا فاصله بین توزیع احتمال واقعی و پیش‌بینی شده را برای تصاویر واقعی و تولید شده، به اصطلاح واگرایی Kullback-Leibler یا واگرایی جنسن شانون به حداقل برساند.

سوال ۳ - CGAN

توضیح ساز و کار معماري:

چکیده: شبکه‌های GAN به تازگی عملکرد بسیار مناسبی برای مدل‌های generative ارائه داده‌اند. در این مقاله روشی برای تولید با فرض condition کردن خروجی ارائه می‌شود. برای انجام این کار متغیر z را به ورودی generator و discriminator می‌دهیم. در ادامه مقاله نشان داده می‌شود که مدل پیشنهادی می‌تواند اعداد MNIST را مشروط به لیبل آن‌ها تولید کند.

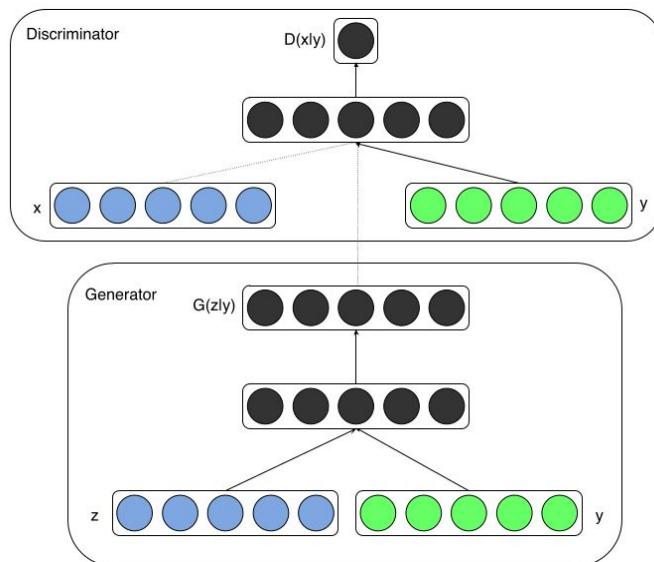
در شبکه‌های GAN که متغیر شرط وجود ندارد، هیچ کنترلی بر روی داده تولید شده وجود ندارد (برای من شهود از این موضوع به این صورت است که فرض کنید که به یک نقاش بگویی یک تصویر بکش ولی هیچ اطلاعات اضافه دیگری ندهید! تصویر کشیده شده هر چیزی می‌تواند باشد) با شبکه معرفی شده می‌تواند خروجی generator را تا حدی کنترل کرد.

در حالی که شبکه‌های کانولوشنی تسلیک طبقه بندی را به خوبی انجام می‌دهند اما مشکل اصلی آن‌ها این است که یک نگاشت یکبهیک میان ورودی و لیبل ایجاد می‌کنند، این در صورتی است که تسلیک های نگاشت یکبهیک لیبل اهمیت بیشتری دارد، همان‌طور که انسان‌های مختلف توصیفات مختلفی از یک تصویر انجام می‌دهند. برای این تسلیک بنظر می‌رسد که اگر word embedding مناسبی وجود داشته باشد، و بتوان تصویر را در آن فضا لیبل گذاری کرد، خطاهای پیش‌آمده معنی دار ترمی‌شوند.

شبکه‌های GAN از دو قسمت اصلی Generator و Discriminator ساخته شده‌اند، G برای این که توزیع p_g را بر روی داده‌های x یا بگیرد، یک نگاشت بر روی نویز z می‌سازد که نویز توزیع $(z; p_z)$ را دارد. این تابع را به صورت $G(z; \theta_g)$ نمایش می‌دهیم. و تابع $D(x; \theta_d)$ با گرفتن یک داده احتمال واقعی بودن / تولیدی بودن داده را می‌دهد. این دو شبکه همراه با یکدیگر آموخته شوند و تابع زیر را بهینه می‌کنند.

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{\text{data}}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))]$$

برای آموزش شرطی شبکه همانطور که گفته شد باید متغیر شرطی z را هم به Generator بدهیم و هم به Discriminator، بنابراین توابع به صورت $D(x; \theta_d)$ و $G(z; \theta_g)$ خواهد بود. در شکل زیر نمودار کلی شبکه CGAN آمده است.

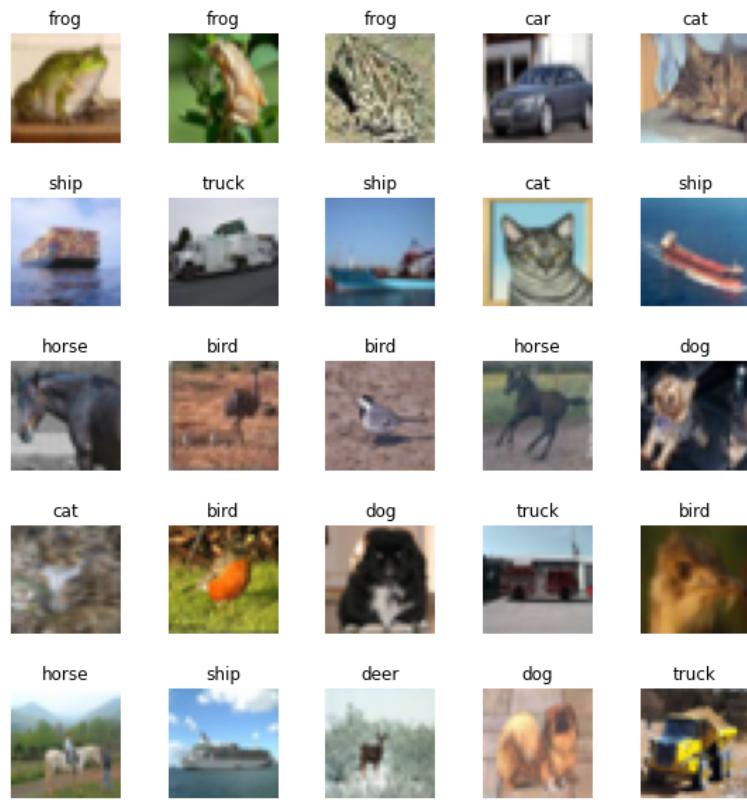


شکل ۱۳: نمودار کلی شبکه Conditional GAN

نتایج مورد نیاز پس از اجرای شبکه:

۱. نمونه تصویر تولید شده

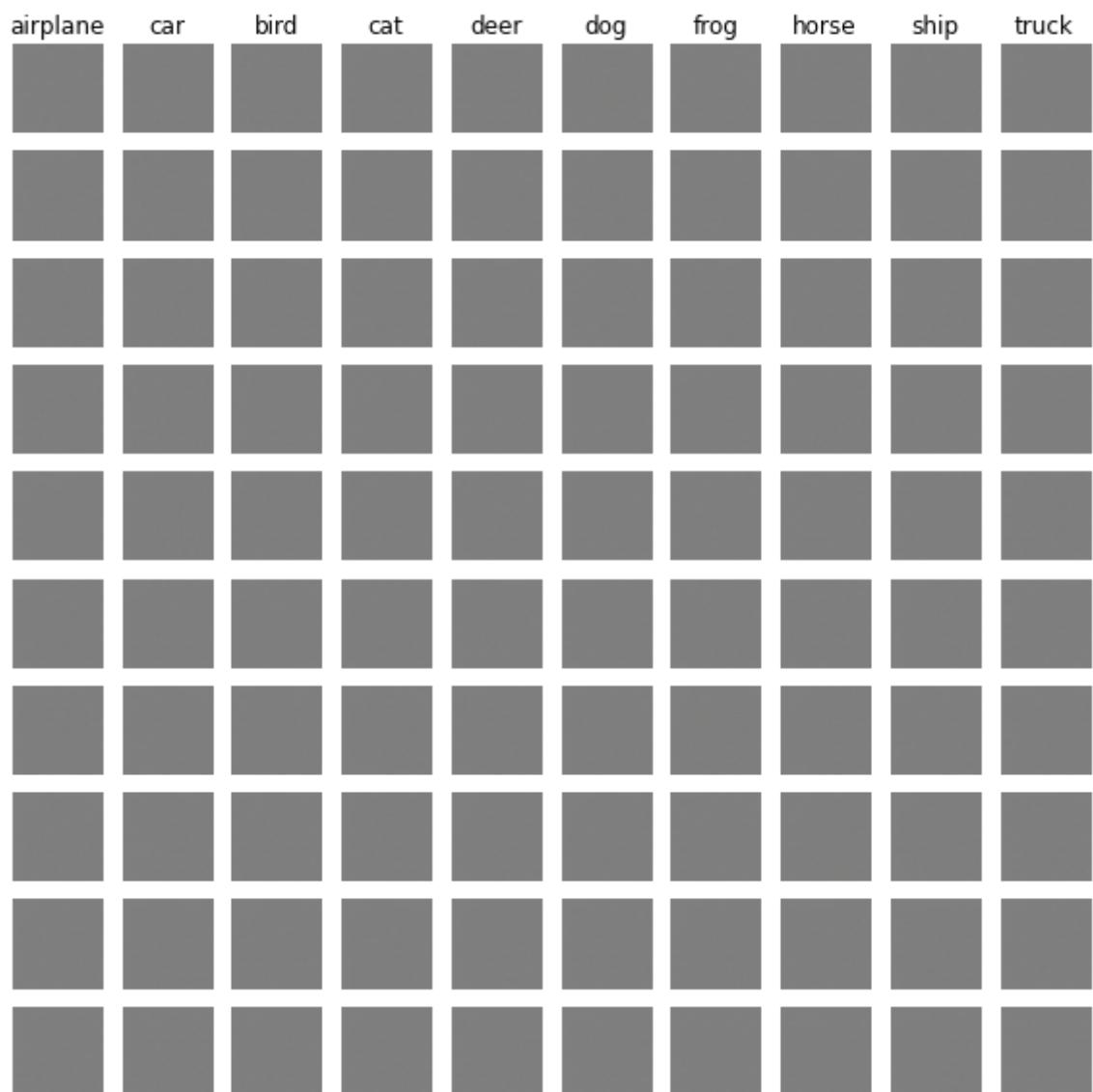
در ابتدا نمونه تصاویر مجموعه داده cifar10 در شکل زیر آورده شده است. این مجموعه داده دارای ۱۰ کلاس است.



شکل ۱۴: نمونه ای از تصاویر مجموعه داده cifar10

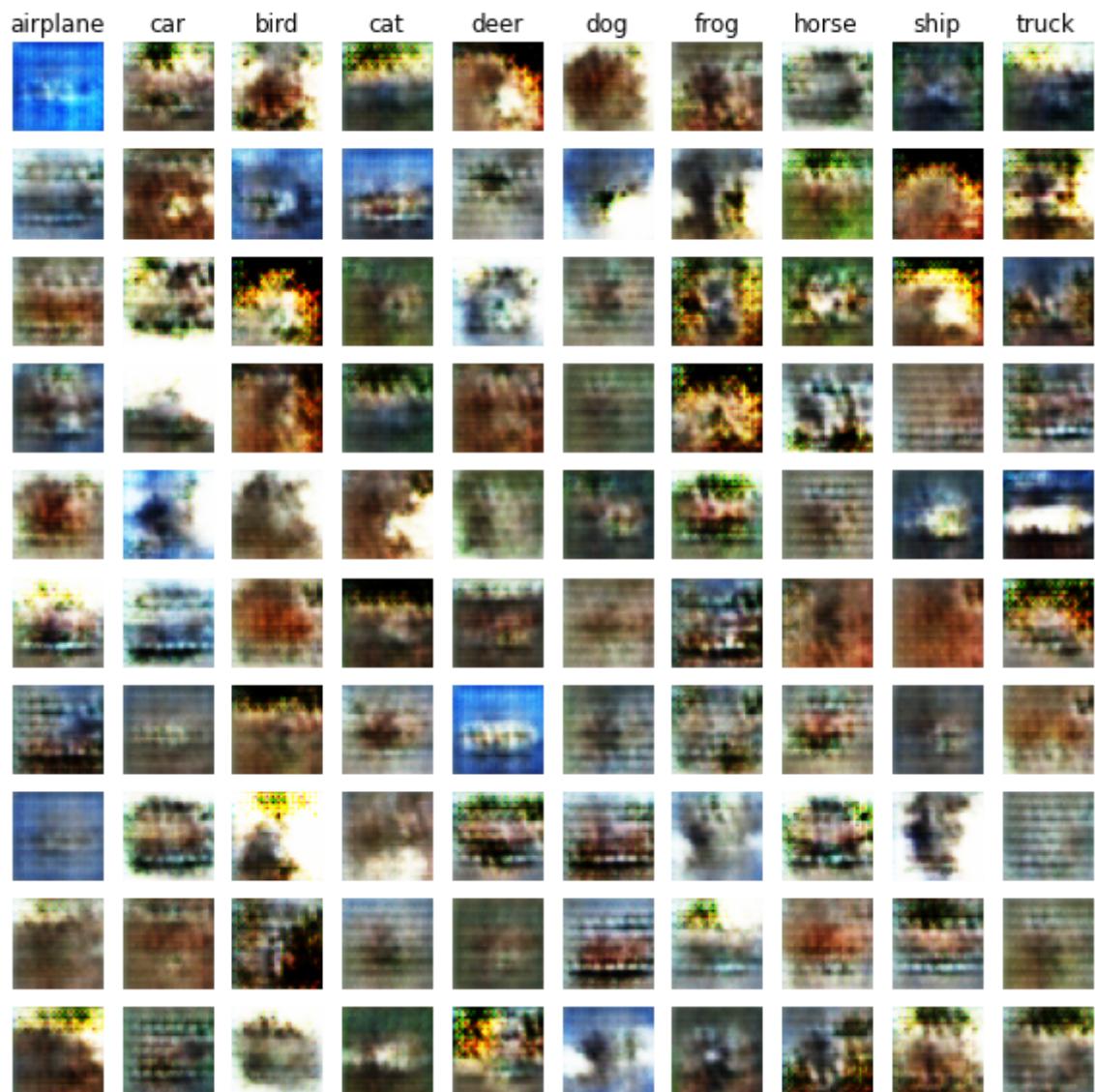
در ادامه تصاویر تولید شده در شبکه آورده شده است. همچنین تغییرات دقیق تر خروجی شبکه در فایل `cgan.gif` آورده شده است.

epoch: 0

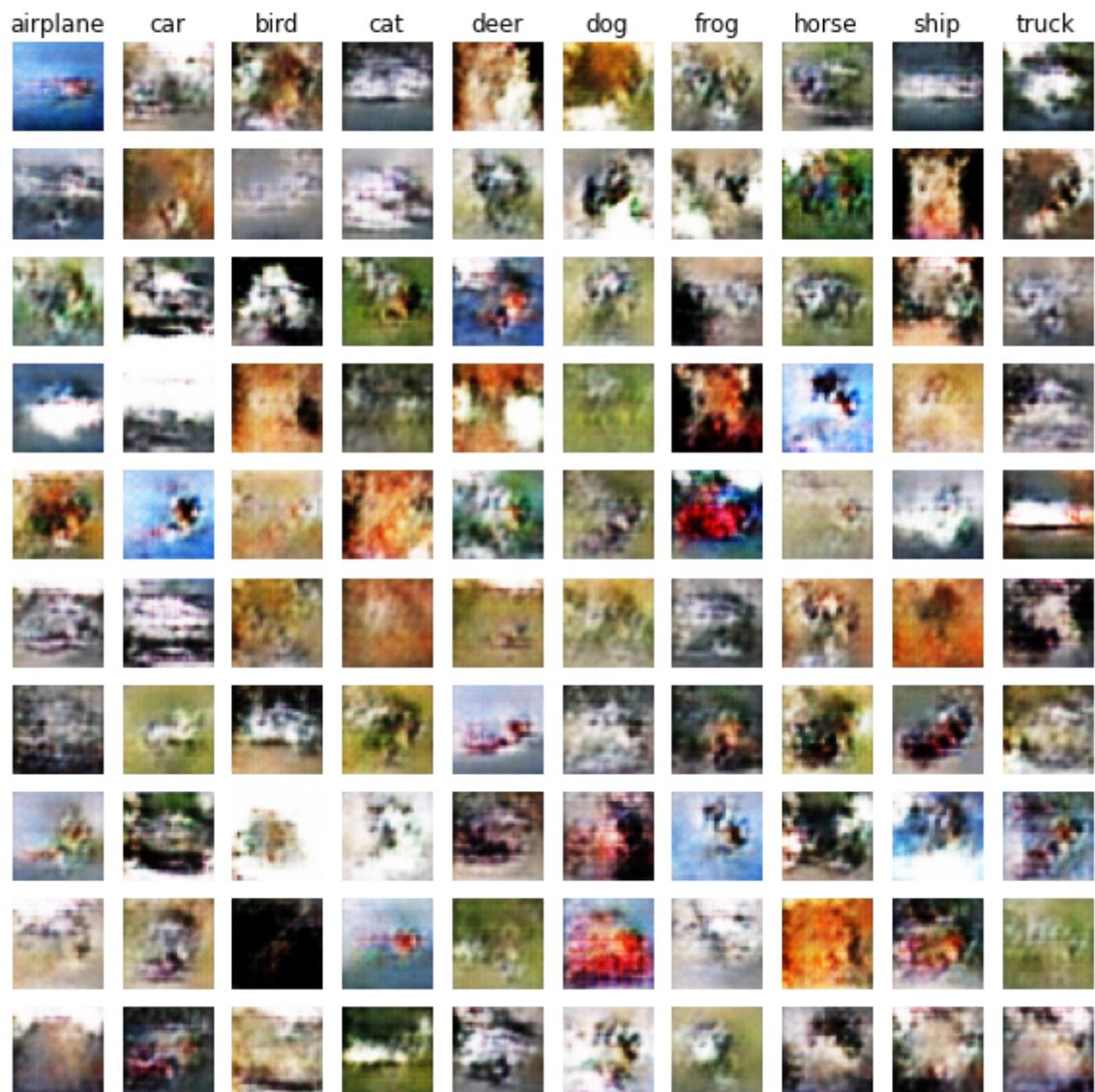


شکل ۱۵: نمونه تصاویر ایجاد شده توسط شبکه CGAN در مرحله ۰

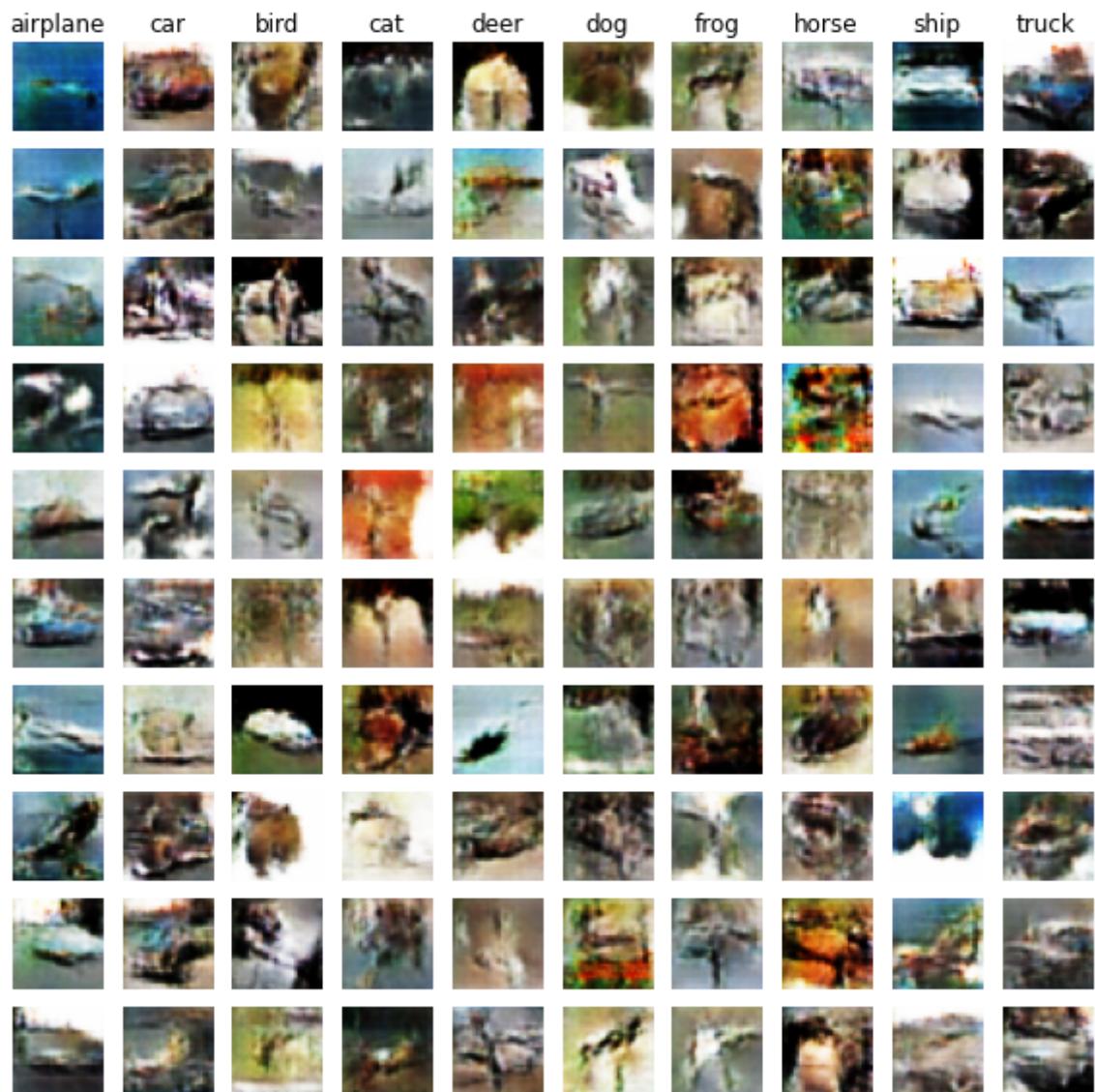
epoch: 3



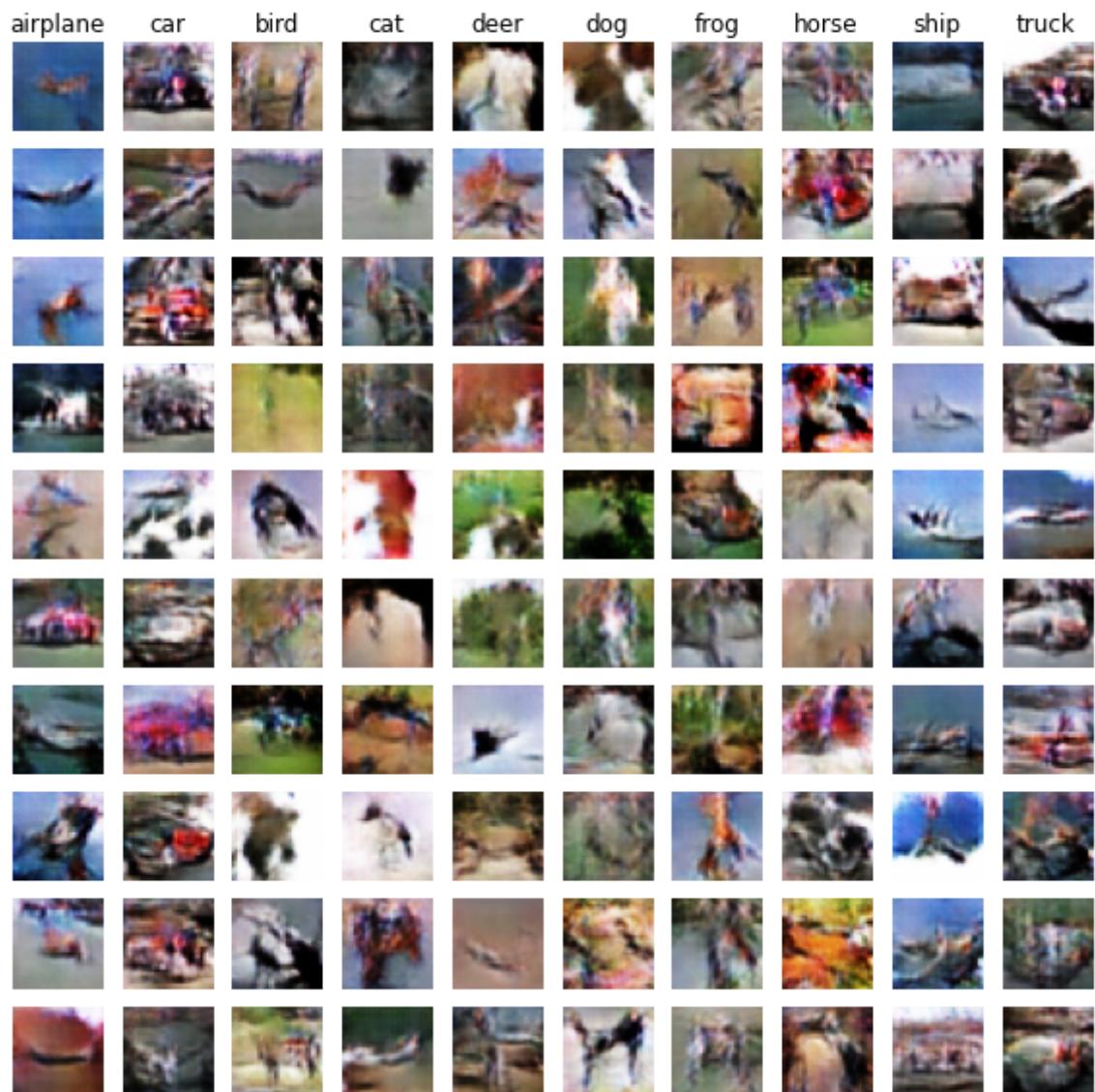
epoch: 5



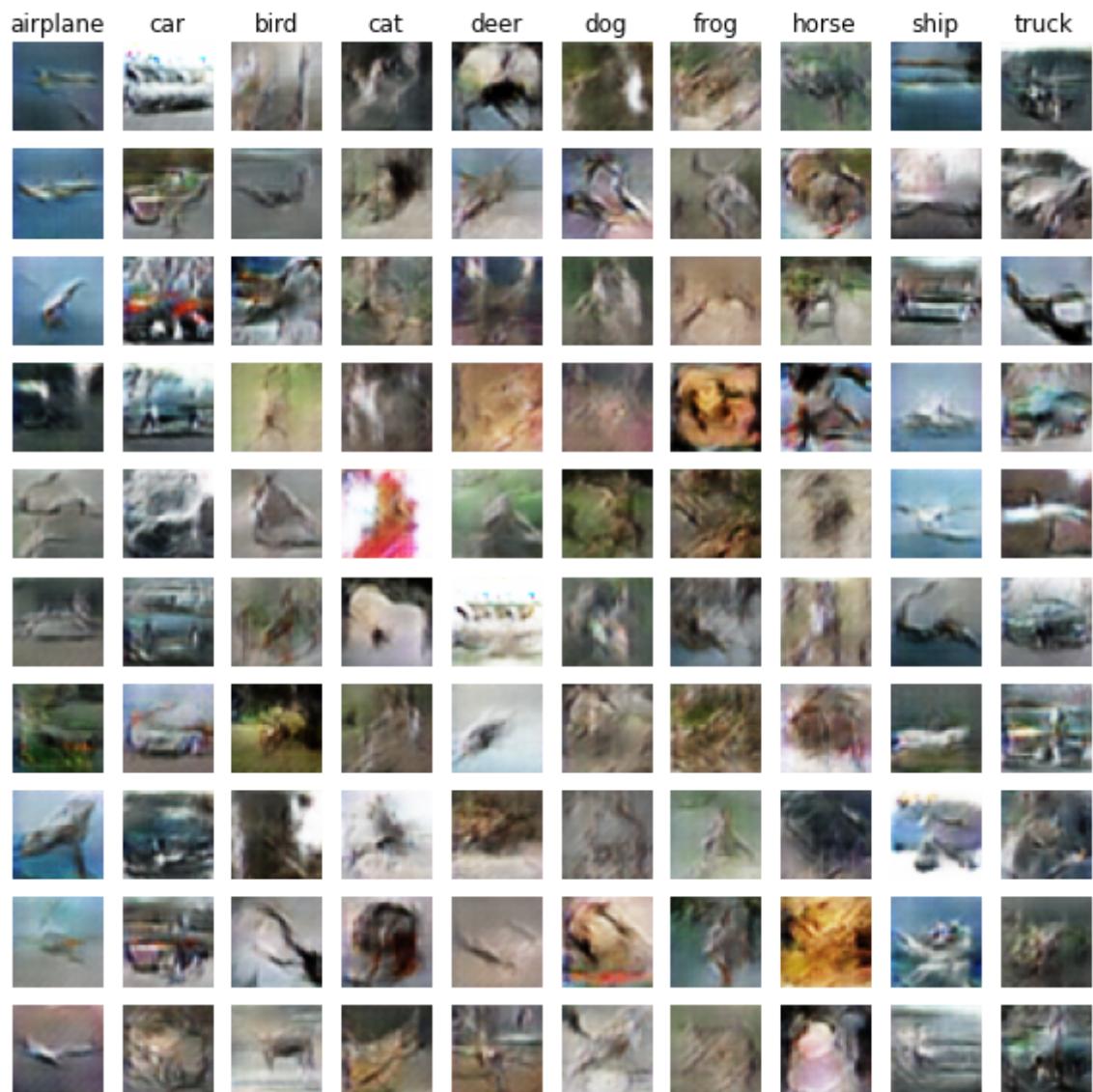
epoch: 9



epoch: 11



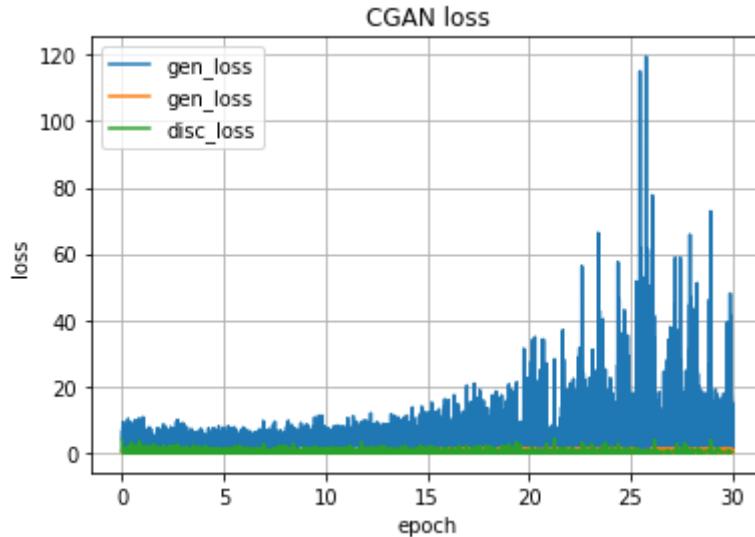
epoch: 14



شکل ۲۰ نمونه تصاویر ایجاد شده توسط شبکه CGAN در مرحله ۱۴

۲. نمودار دقیق و هزینه

در شکل زیر نمودار هزینه برای شبکه در زمان آموزش آورده شده است. اشاره به این نکته خارج از لطف نیست که دقیق مولد و تشخیص‌گر مرتبه کاهش و افزایش می‌یابندکه ناشی از ذات رقابتی این دو شبکه است.



شکل ۲۱: نمودار هزینه شبکه CGAN در حال آموزش

سوال -۳ AC-GAN

از شبکه های پیشنهاد شده در سوال ، شبکه AC-GAN انتخاب شد.

توضیح ساز و کار معماری:

چکیده: در این مقاله روش جدید برای تولید تصاویر با کیفیت بالا ارائه شده است، این روش جزو روش های class conditional information است به این معنی که در ورودی مولد^۲ علاوه بر نویز، لیبل کلاس نیز داده می‌شود. در این مقاله ادعا می‌شود که با واردن کردن class conditional information بهتر بدهست آورد. در این مقاله کاست فانکشن جدید برای خروجی discriminator می‌توان تصاویری با وضوح بهتر بدست آورد. در این مقاله شبکه مولد سعی می‌کند L_s – L_c را بیشینه کند در حالی که شبکه تقسیم‌گر^۳ تلاش می‌کند $L_c + L_s$ را بیشینه کند.

$$L_S = E[\log P(S = \text{real} \mid X_{\text{real}})] + \\ E[\log P(S = \text{fake} \mid X_{\text{fake}})]$$

$$L_C = E[\log P(C = c \mid X_{\text{real}})] + \\ E[\log P(C = c \mid X_{\text{fake}})]$$

در این سوال شبکه مطابق با پیشنهاد مقاله برای داده های cifar10 به صورت زیر ساخته شده است.

² generator

³ discriminator

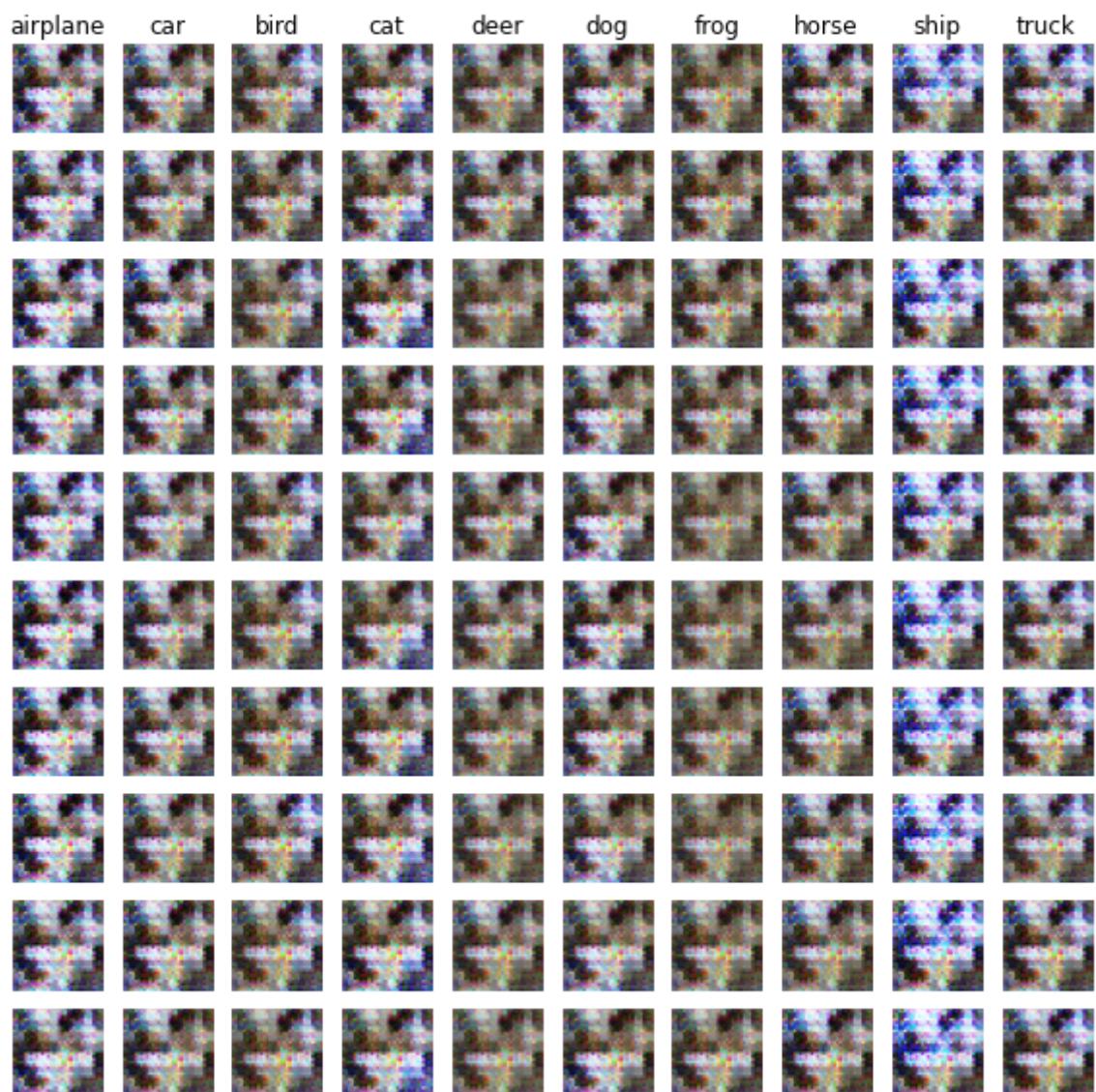
Operation	Kernel	Strides	Feature maps	BN?	Dropout	Nonlinearity
$G_x(z) - 110 \times 1 \times 1$ input						
Linear	N/A	N/A	384	×	0.0	ReLU
Transposed Convolution	5 × 5	2 × 2	192	✓	0.0	ReLU
Transposed Convolution	5 × 5	2 × 2	96	✓	0.0	ReLU
Transposed Convolution	5 × 5	2 × 2	3	×	0.0	Tanh
$D(x) - 32 \times 32 \times 3$ input						
Convolution	3 × 3	2 × 2	16	×	0.5	Leaky ReLU
Convolution	3 × 3	1 × 1	32	✓	0.5	Leaky ReLU
Convolution	3 × 3	2 × 2	64	✓	0.5	Leaky ReLU
Convolution	3 × 3	1 × 1	128	✓	0.5	Leaky ReLU
Convolution	3 × 3	2 × 2	256	✓	0.5	Leaky ReLU
Convolution	3 × 3	1 × 1	512	✓	0.5	Leaky ReLU
Linear	N/A	N/A	11	×	0.0	Soft-Sigmoid
Generator Optimizer	Adam ($\alpha = [0.0001, 0.0002, 0.0003]$, $\beta_1 = 0.5$, $\beta_2 = 0.999$)					
Discriminator Optimizer	Adam ($\alpha = [0.0001, 0.0002, 0.0003]$, $\beta_1 = 0.5$, $\beta_2 = 0.999$)					
Batch size	100					
Iterations	50000					
Leaky ReLU slope	0.2					
Activation noise standard deviation	[0, 0.1, 0.2]					
Weight, bias initialization	Isotropic gaussian ($\mu = 0$, $\sigma = 0.02$), Constant(0)					

نتایج مورد نیاز پس از اجرای شبکه:

۱. نمونه تصویر تولید شده

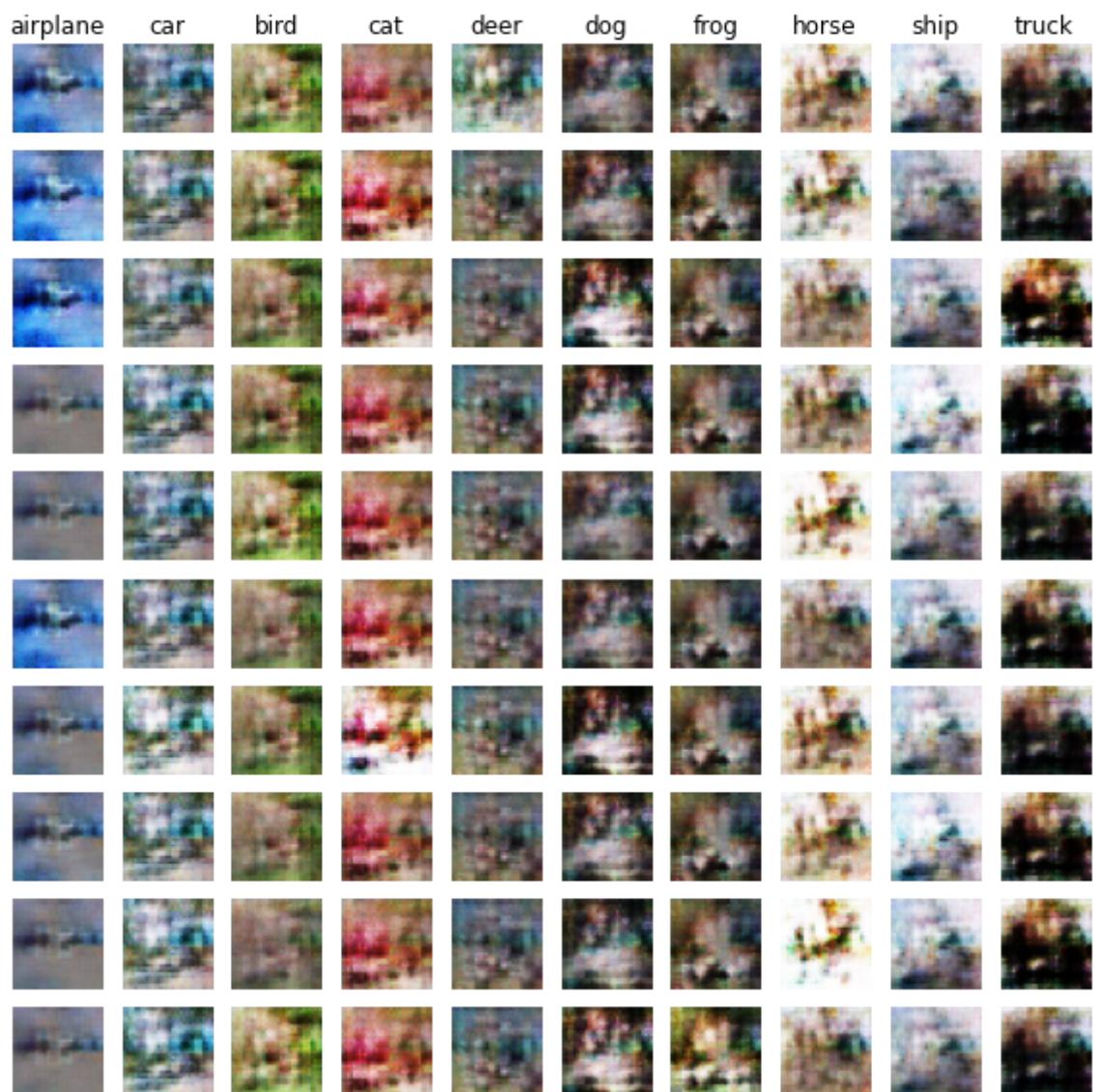
در ادامه تصاویر تولید شده برای ۵ زمان انتخابی آورده شده است. در مقایسه با تصاویر تولیدی از CGAN اشاره به این نکته مناسب است که به علت axillary classifier در بخش تشخیص گیر، با مشروط کرد ورودی به کلاس خاص تصاویر مشابه تر هستند.

epoch 1



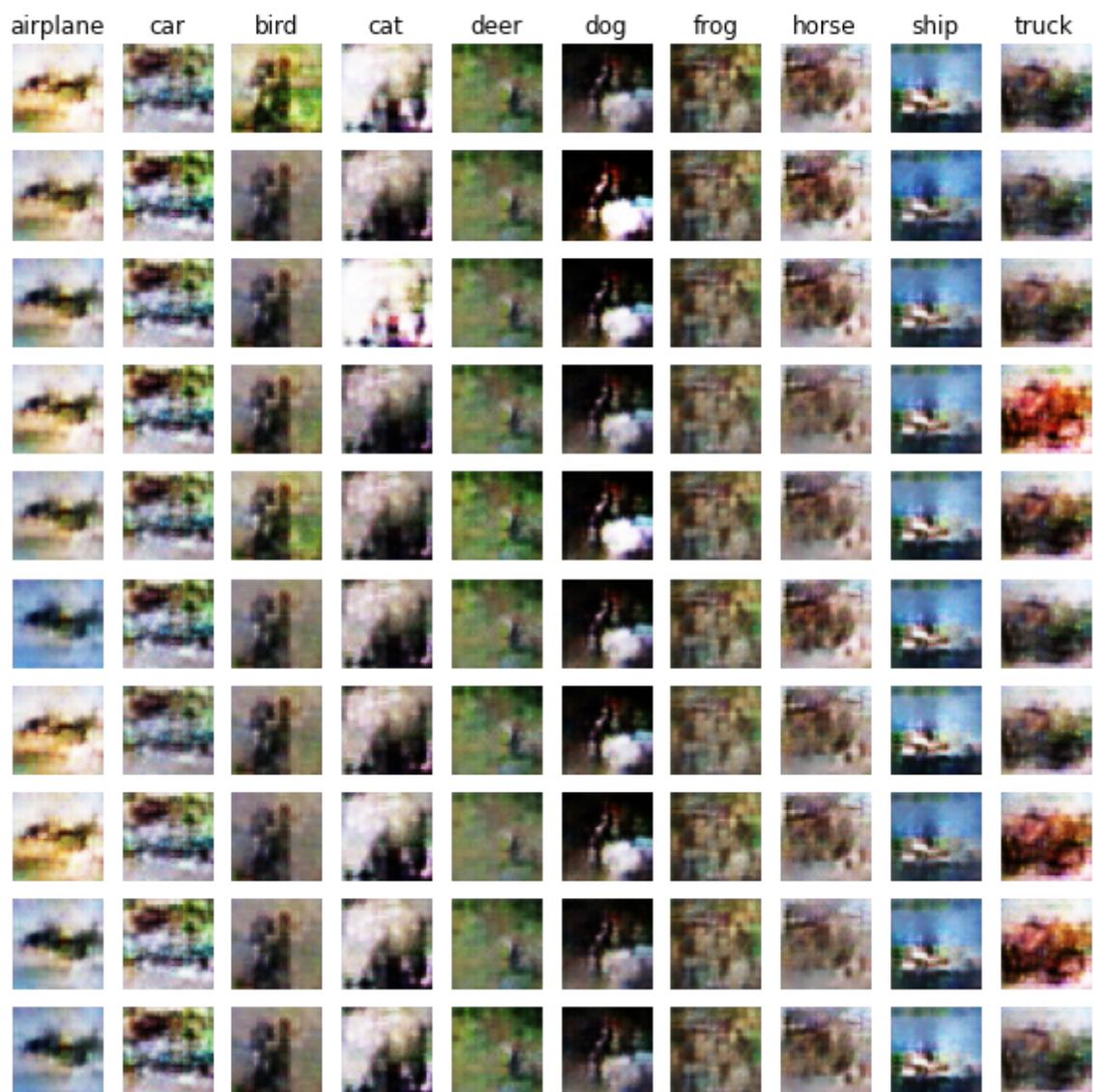
شکل ۲۲: نمونه تصاویر ایجاد شده توسط شبکه ACGAN در مرحله ۱

epoch 6



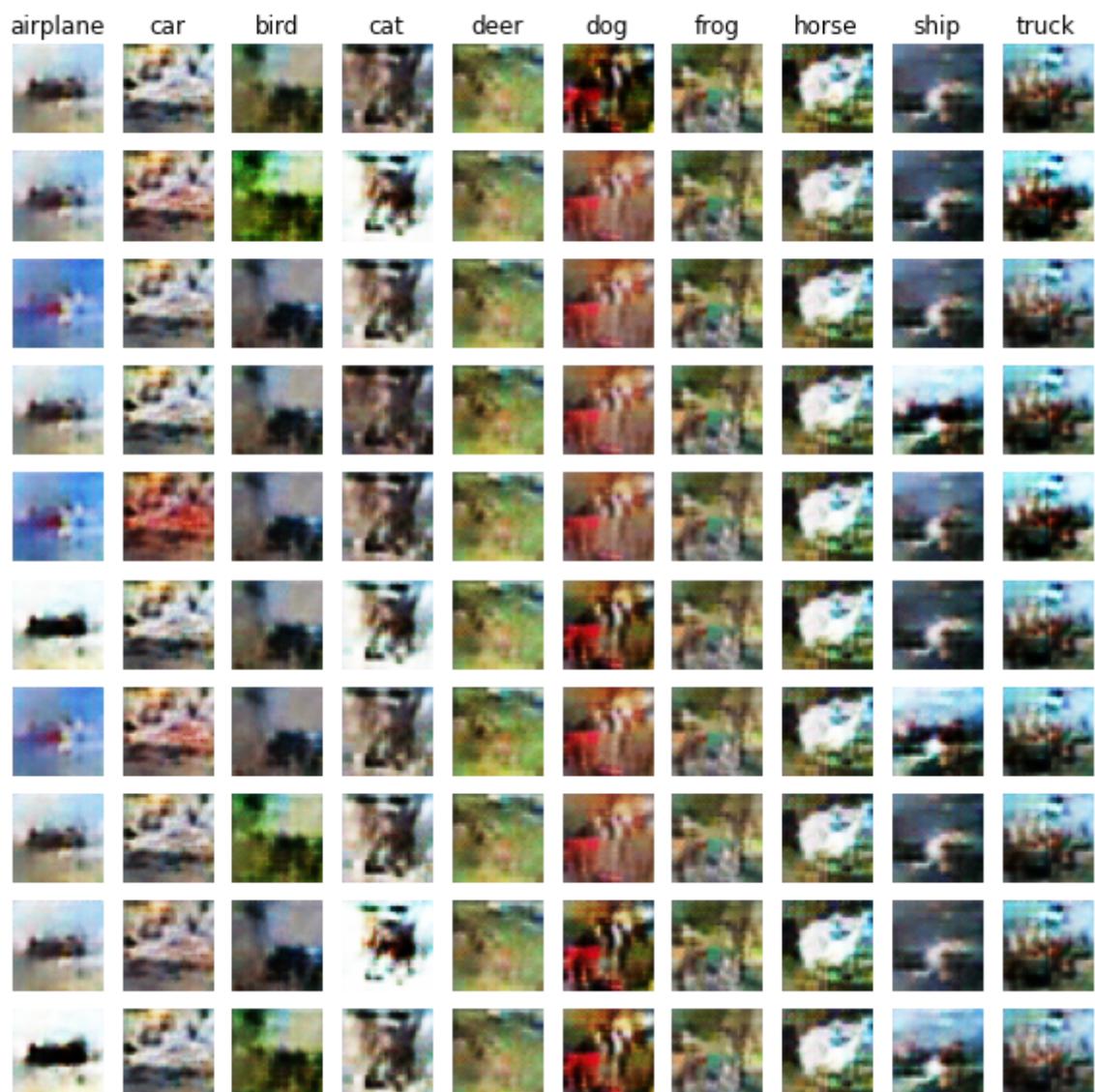
شکل ۲۳: نمونه تصاویر ایجاد شده توسط شبکه ACGAN در مرحله ۶

epoch 9

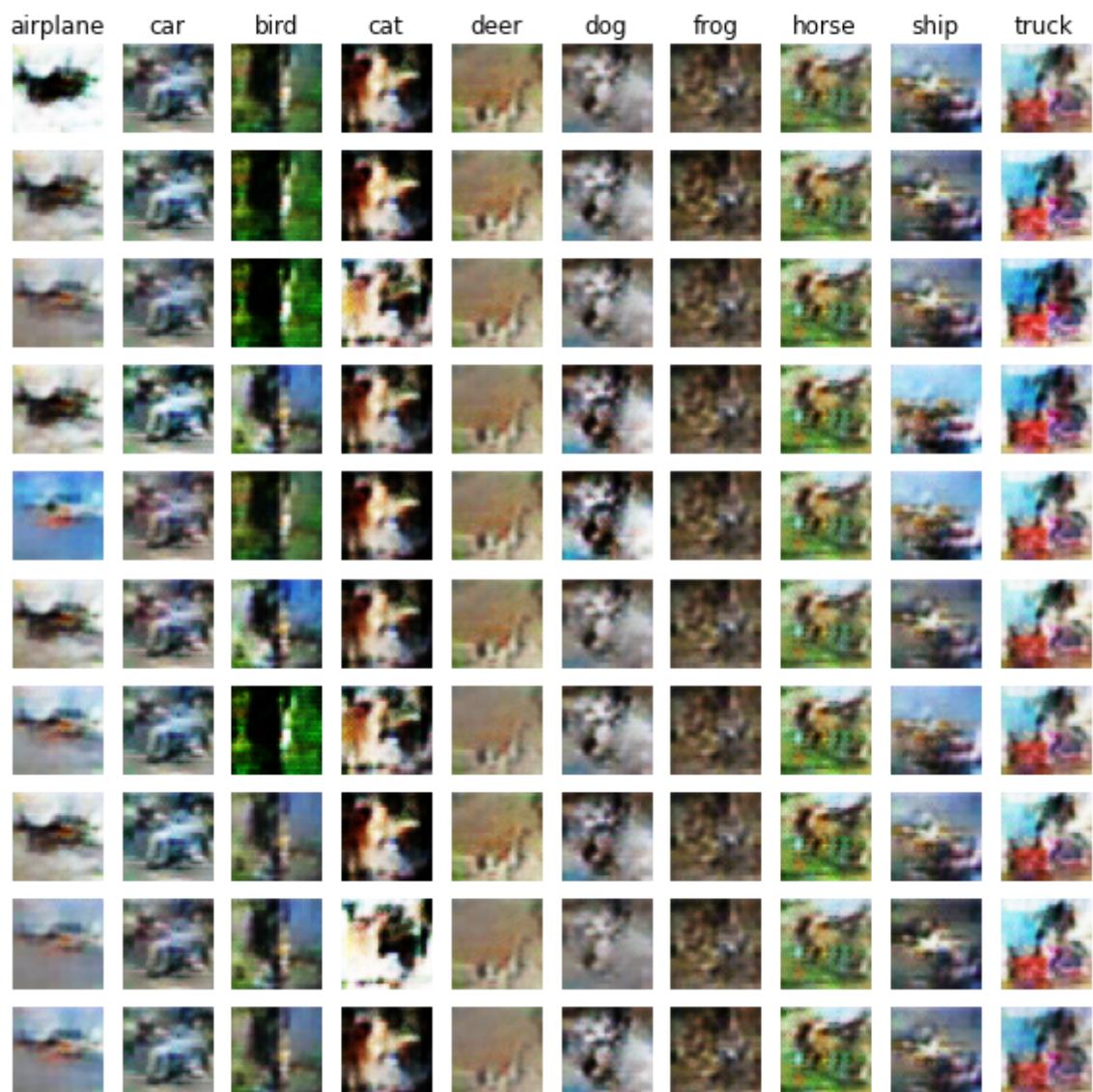


شکل ۲۴: نمونه تصاویر ایجاد شده توسط شبکه ACGAN در مرحله ۹

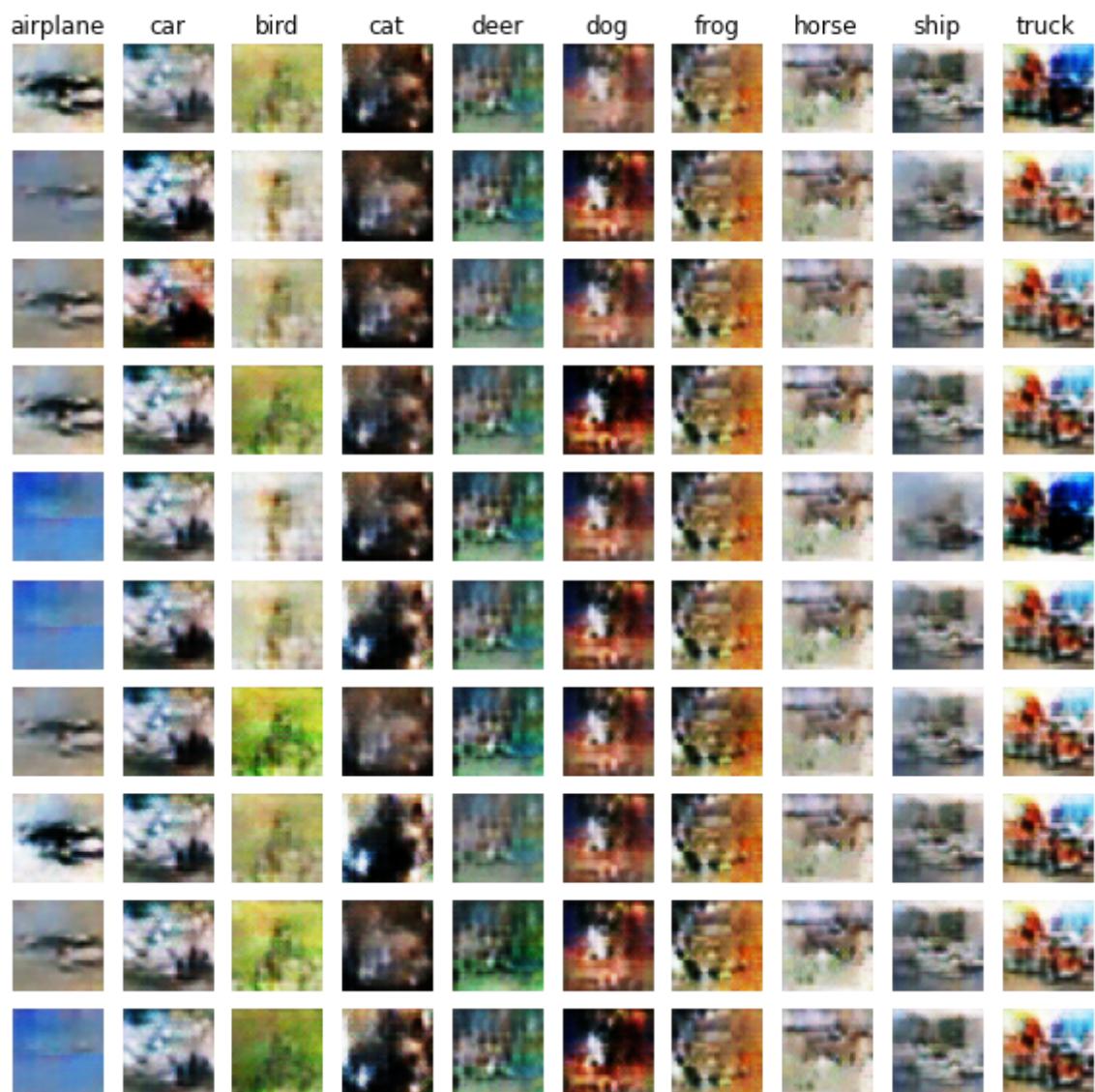
epoch 13



epoch 17



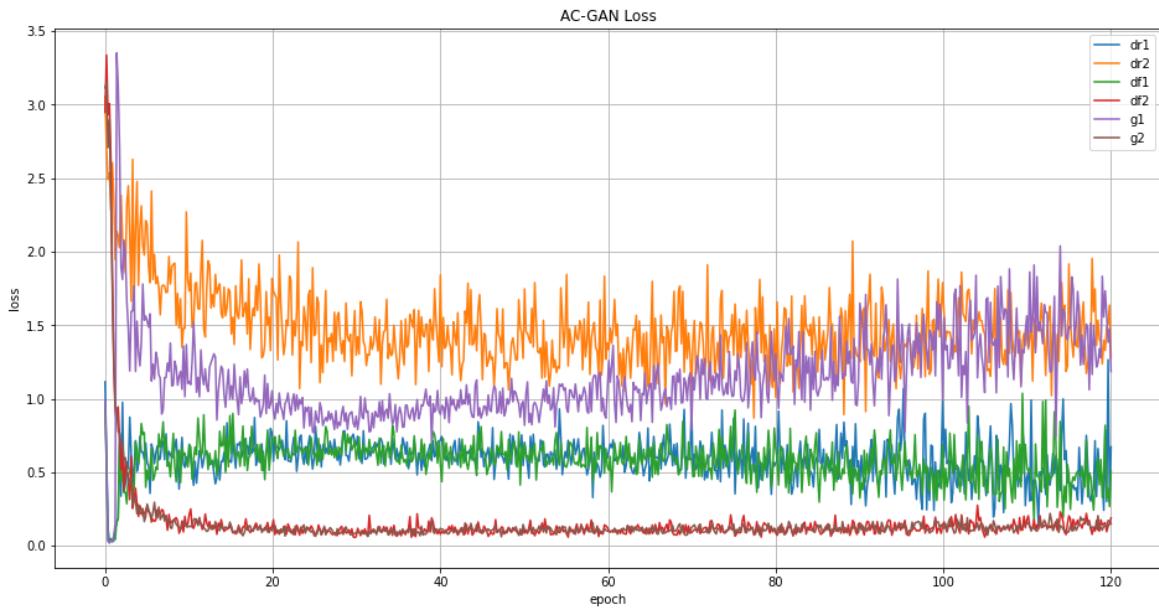
epoch 18



شکل ۲۷: نمونه تصاویر ایجاد شده توسط شبکه ACGAN در مرحله ۱۸

۲. نمودار دقت و هزینه

در شکل زیر نمودار هزینه برای شبکه در زمان آموزش آورده شده است. اشاره به این نکته خارج از لطف نیست که دقت مولد و تشخیص‌گر مرتبه کاهش و افزایش می‌یابندکه ناشی از ذات رقابتی این دو شبکه است.



شکل ۲۸: نمودار هزینه مدل