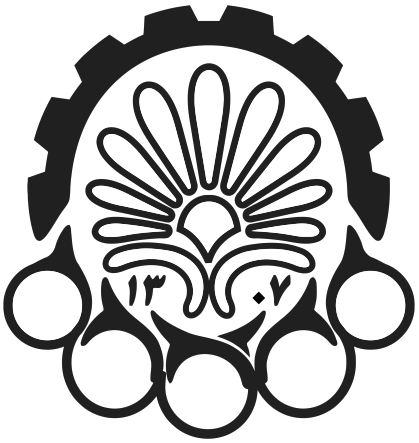
شبکه‌های عصبی و یادگیری عمیق
دکتر صفابخش



دانشگاه صنعتی امیرکبیر

(پلی تکنیک تهران)

دانشکده مهندسی کامپیوتر

رضا آدینه پور ۴۰۲۱۳۱۰۵۵

تمرین ششم
شبکه GAN

۱۵ تیر ۱۴۰۳

دانشکده مهندسی کامپیوتر

شبکه‌های عصبی و یادگیری عمیق

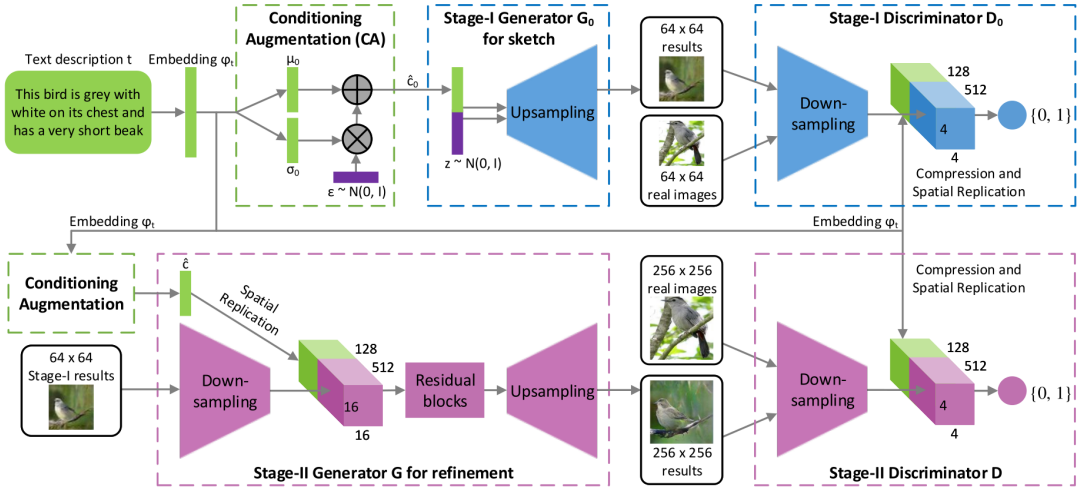
تمرین ششم

رضا آدینه پور ۴۰۲۱۳۱۰۵۵

سوال اول - شبکه‌های مولد تقابلی

شبکه‌های مولد تقابلی[۱] همانطور که در کلاس با آنها آشنا شدید شامل دو زیرشبکه‌ی تولیدکننده[۲] و تمایزگر[۳] هستند که به صورت تقابلی آموزش داده می‌شوند تا داده‌های جدید تولید کنند. تولید جدید هدفی است که در تمامی مدل‌های مولد مد نظر قرار دارد و به شکل‌های مختلف از جمله ترجمه‌ی تصویر به تصویر، تبدیل دامنه و تولید شرطی صورت می‌گیرد. یکی از این اشکال، تولید تصویر با دریافت فرمان زبانی است که امروزه نیز نمونه‌های کاربردی آن همچون Imagen و Dall-E در دسترس عموم قرار دارند. در این تمرین به طور خاص به پیاده سازی این وظیفه با شبکه‌ی مولد تقابلی پشته‌ای یا SatckGAN می‌پردازیم.

۱. به مراجعه به مقاله StackGAN کلیت ساختار و چگونگی عملکرد این شبکه را توضیح دهید. توضیح دهید که شبکه‌ی تعریف شده در هر گام[۴] به چه منظور استفاده می‌شود. به طور خاص ذکر کنید که ورودی شبکه‌ی تولیدکننده در هر دو گام چه تفاوتی با ورودی یک شبکه‌ی مولد تقابلی ساده[۵] دارد؟ همچنین بررسی کنید که آموزش این شبکه به چه صورت انجام می‌شود.



شکل ۱: معماری کلی شبکه مولد تقابلی پشته ای

[۱] NetworksAdverserial Generative
[۲] Generator
[۳] Discriminator
[۴] Stage
[۵] Vanilla GAN

پاسخ

همانطور که در توضیحات سوال بیان شد، تولید تصاویر با کیفیت بالا از متن، چالشی است که این مقاله آن را حل نموده است. این مقاله با شکستن کار به دو مرحله:

(a) Stage-I GAN

(b) Stage-II GAN

چالش تولید تصویر با وضوح و دقت بالا از داده‌های متنی را حل می‌کند.

(آ) **مرحله اول:**

- **هدف:** مرحله اول GAN مسئول تولید یک تصویر (64x64) با وضوح پایین ست که شکل و رنگ اساسی شیء توصیف شده توسط متن را ترسیم می‌کند.
- **ورودی:**
  - **بردار تبدیل شده متن:** توضیحات متنی با استفاده از یک انکودر از پیش آموزش داده شده ϕ_t به یک بردار عددی تبدیل می‌شوند.
  - **متغیرهای شرطی گوسی:** متغیرهای شرطی از توزیع گوسی که توسط تعبیه متنی پارامتره شده‌اند، نمونه‌برداری می‌شوند.
    متغیرهای شرطی گوسی $\hat{c}_0$ برای تعبیه متنی از $N(\mu_0(\phi_t), \Sigma_0(\phi_t))$ نمونه‌برداری می‌شوند تا معنای ϕ_t را با تغییرات مختلف به تصویر بکشند. این می‌تواند پایداری آموزش و تنوع نمونه را فراهم کند، به عنوان مثال: انواع مختلف حالات و ظاهرها.
  - **بردار نویز:** یک بردار نویز تصادفی z با متغیرهای شرطی ترکیب می‌شود.
- **خروجی:** یک تصویر با وضوح پایین که شکل اصلی و رنگ‌های اساسی متناظر با توضیحات متنی را به تصویر می‌کشد.
  مشروط بر $\hat{c}_0$ و متغیر تصادفی z، مرحله اول GAN متمایزکننده D_0 و تولیدکننده G_0 را با جایگزینی حداکثر کردن L_{D_0} در معادله (۱) و حداقل کردن L_{G_0} در معادله (۲) آموزش می‌دهد:

(۱)
$$L_{D_0} = \mathbb{E}_{(I_0,t)\sim p_{data}}[\log D_0(I_0, \phi_t)] + \mathbb{E}_{z\sim p_z, t\sim p_{data}}[\log(1 - D_0(G_0(z, \hat{c}_0), \phi_t))],$$

(۲)
$$L_{G_0} = \mathbb{E}_{z\sim p_z, t\sim p_{data}}[\log(1 - D_0(G_0(z, \hat{c}_0), \phi_t))] + \lambda D_{KL}(N(\mu_0(\phi_t), \Sigma_0(\phi_t)) \| N(0, I)),$$

تصویر واقعی I_0 و توضیح متنی t از توزیع داده‌های واقعی p_{data} هستند.

(ب) **مرحله دوم:**

- **هدف:** مرحله دوم GAN تصویر با وضوح پایین تولید شده در مرحله اول را تصحیح می‌کند تا یک تصویر (256x256) با وضوح بالا با جزئیات بیشتر و واقع‌گرایانه‌تر تولید کند.

پاسخ

- **ورودی:**
  - **خروجی مرحله اول:** تصویر با وضوح پایین از مرحله اول.
  - **تعبیه متنی:** تعبیه متنی دوباره برای شرطی‌سازی فرآیند تولید استفاده می‌شود، که اطمینان حاصل کند جزئیات تصویر با متن همخوانی دارد یا خیر.
  - **متغیرهای شرطی گوسی:** متغیرهای شرطی جدید از توزیع گوسی که توسط تعبیه متنی پارامتره شده‌اند، نمونه‌برداری می‌شوند.
- **خروجی:** یک تصویر با وضوح بالا و جزئیات دقیق که واقع‌گرایانه است که توضیحات متنی را به درستی منعکس می‌کند.

$$L_D = \mathbb{E}_{(I,t)\sim p_{data}}[\log D(I, \phi_t)] + \mathbb{E}_{s_0\sim p_{G_0}, t\sim p_{data}}[\log(1 - D(G(s_0, \hat{c}), \phi_t))], \quad (۳)$$

$$L_G = \mathbb{E}_{s_0\sim p_{G_0}, t\sim p_{data}}[\log(1 - D(G(s_0, \hat{c}), \phi_t))] + \lambda D_{KL}(N(\mu(\phi_t), \Sigma(\phi_t))\|N(0, I)), \quad (۴)$$

مرحله دوم GAN یاد می‌گیرد تا اطلاعات مفید در تعبیه متنی که توسط مرحله اول **GAN** نادیده گرفته شده است را به دست آورد.

- تولیدکننده مرحله دوم یک شبکه انکودر–دکودر با بلوک‌های residual است.
- ویژگی‌های تصویر و ویژگی‌های متن در طول بعد کانال ترکیب می‌شوند تا نمایش‌های چندوجهی در سراسر ویژگی‌های تصویر و متن را یاد بگیرند.
- در نهایت، از یک سری لایه‌های دیکودر برای تولید یک تصویر با وضوح بالا $W \times H$ استفاده می‌شود.
- به جای استفاده از متمایزکننده ساده، از متمایزکننده matching-aware برای هر دو مرحله استفاده می‌شود.

تفاوت‌های ورودی در مقایسه با GAN های ساده:

(آ) **GAN های ساده:**

- **ورودی:** یک بردار نویز ساده به عنوان ورودی برای شبکه تولیدکننده استفاده می‌شود.
- **خروجی:**یک فرآیند تولید تک‌مرحله‌ای که به طور مستقیم سعی در ایجاد تصویر با وضوح بالا دارد.

(ب) **StackGAN:**

- **ورودی مرحله اول:** ترکیبی از بردار نویز و متغیرهای شرطی مشتق شده از تعبیه‌های متنی برای تولید یک تصویر با وضوح پایین.
- **ورودی مرحله دوم:** از خروجی مرحله اول استفاده می‌کند و تعبیه‌های متنی را با متغیرهای شرطی جدید برای تصحیح تصویر و افزودن جزئیات مجدداً استفاده می‌کند.

۲. شبکه‌های مولد تقابلی در مقایسه با سایر شبکه‌ها از سه مشکل اساسی رنج می‌برند: این سه مشکل عبارتند از فروپاشی مد[۶]، عدم همگرایی و ناپدید شدن گرادیان. به طور مختصر توضیح دهید که هر کدام به چه صورتی و چه راهکارهایی برای رفع آنها مطرح شده است؟

پاسخ

(آ) **فروپاشی مد:**
فروپاشی مد زمانی رخ می‌دهد که مولد تعداد محدودی خروجی تولید می‌کند و عملاً به چند مد (یا حتی یک مد) از توزیع داده‌ها فرو می‌ریزد. این به این معناست که GAN نمی‌تواند تمام تنوع توزیع داده‌های واقعی را به تصویر بکشد. برای حل این مشکل می‌توان از موارد زیر استفاده نمود:

- **تمایز دسته‌ای (Minibatch Discrimination):** این تکنیک به تشخیص‌دهنده اجازه می‌دهد تا نمونه‌های متعدد را با هم بررسی و مقایسه کند، و مولد را تشویق می‌کند تا خروجی‌های متنوع‌تری تولید کند.
- **GAN های بدون بازگشت (Unrolled GANs):** این رویکرد فرآیند آموزش را تغییر می‌دهد تا بهینه‌سازی تشخیص‌دهنده را Unrolled کند و اثر به‌روزرسانی‌های آینده را در نظر بگیرد که به جلوگیری از فروپاشی مد کمک می‌کند.
- **مجموعه‌ای از مولدها (Ensemble of Generators):** استفاده از مولدهای متعدد می‌تواند به پوشش مدهای مختلف توزیع داده‌ها کمک کند و بنابراین فروپاشی مد را کاهش دهد.

(ب) **عدم همگرایی:**
عدم همگرایی یا همگرایی ناکامل زمانی رخ می‌دهد که فرآیند آموزش GAN نوسان کند یا نتواند به یک حالت پایدار برسد. این ناپایداری ناشی از این است که مولد و تشخیص‌دهنده دائماً سعی در شکست دادن یکدیگر دارند. برای اینکه شبکه ناپایدار نشود می‌توان راهکار های زیر را انجام داد:

- **معماری‌های بهبود یافته:** استفاده از تکنیک‌هایی مانند نرمال‌سازی طیفی که با کنترل ثابت Lipschitz تشخیص‌دهنده، آموزش را پایدار می‌کند.
- **توابع هدف اصلاح شده:** توابع زیان جایگزین، مانند زیان Wasserstein (استفاده شده در Wasserstein GANs یا WGANs)، می‌تواند همگرایی را با ارائه یک گرادیان معنادارتر برای مولد حتی زمانی که تشخیص‌دهنده نزدیک به بهینه است، بهبود بخشد.
- **تکنیک‌های تنظیم (Regularization):** تکنیک‌هایی مانند gradient penalty (استفاده شده در WGAN-GP) می‌تواند با جریمه کردن نُرم گرادیان تشخیص‌دهنده، آموزش را پایدار تر کند و از ناپایداری جلوگیری کند.

(ج) **محوشدن گرادیان:**
ناپدید شدن گرادیان زمانی رخ می‌دهد که گرادیان‌های بازگشتی از طریق شبکه بسیار کوچک می‌شوند و منجر به یادگیری آهسته یا عدم یادگیری شبکه مولد می‌شوند. این مشکل زمانی که تشخیص‌دهنده بسیار قوی شود و به راحتی بین داده‌های واقعی و جعلی تمایز قائل شود، بیشتر می‌شود. برای قلبه به این مشکل راه‌حل های زیر پیشنهاد می‌شود:

[۶] Mode Collapse

پاسخ

- **استفاده از WGAN:** چارچوب Wasserstein GAN مشکل ناپدید شدن گرادیان‌ها را با اطمینان از اینکه گرادیان‌ها ناپدید نمی‌شوند، حل می‌کند. این کار را با استفاده از تابع زیان متفاوت بر اساس فاصله Wasserstein انجام می‌دهد که سیگنال گرادیان پایدارتر و مفیدتری ارائه می‌دهد.
- **تطبیق ویژگی (Feature Matching):** این تکنیک شامل آموزش مولد برای تطبیق آمار ویژگی‌های استخراج شده از یک لایه واسطه تشخیص‌دهنده است، به جای اینکه به طور مستقیم سعی کند تشخیص‌دهنده را فریب دهد.
- **تعادل در آموزش (Training Balance):** اطمینان از اینکه مولد و تشخیص‌دهنده ظرفیت‌های یادگیری متعادلی دارند و به طور متعادل آموزش داده می‌شوند می‌تواند از قوی شدن بیش از حد تشخیص‌دهنده نسبت به مولد جلوگیری کند و یادگیری هردو شبکه را در یک حد نگه دارد.

۳. یک ایده‌ی رایج برای بهبود عملکرد شبکه‌های مولد تقابلی استفاده از عملگر PixelShuffle است. نحوه‌ی عملکرد این عملگر و تأثیر آن را بررسی کنید. بررسی کنید که این عملگر اولین بار در چه وظیفه‌ای و به چه منظور تعریف شد؟ همچنین بررسی کنید که به طور خاص در معماری StackGAN در کدام زیرشبکه‌ها قابل استفاده است و چه عملکردی خواهد داشت؟

پاسخ

عملگر PixelShuffle که به عنوان هم‌ترازی زیرپیکسلی نیز شناخته می‌شود، تکنیکی برای بهبود وضوح مکانی تصویر است. این عملگر عناصر یک تنسور را به گونه‌ای بازآرایی می‌کند که یک تنسور با وضوح مکانی بالاتر و عمق کانال کمتر تولید شود. این عملگر یک تنسور ورودی با شکل $(B, C \times r^2, H, W)$ را دریافت می‌کند و آن را به تنسوری با شکل $(B, C, H \times r, W \times r)$ تبدیل می‌کند، که در آن B اندازه دسته، C تعداد کانال‌ها، H و W ارتفاع و عرض تصویر و r ضریب بزرگ‌نمایی است.

روابط ریاضی آن را می‌توان به‌صورت زیر بیان نمود:

با توجه به تنسور ورودی X با شکل $(B, C \times r^2, H, W)$، عملگر PixelShuffle آن را به تنسور خروجی Y با شکل $(B, C, H \times r, W \times r)$ بازآرایی می‌کند:

$$Y(b, c, h \times r + i, w \times r + j) = X(b, c \times r^2 + i \times r + j, h, w) \quad (۵)$$

برای بازه‌های:

(a) $b \in [0, B)$

(b) $c \in [0, C)$

(c) $h \in [0, H)$

(d) $w \in [0, W)$

(e) $i, j \in [0, r)$

عملگر PixelShuffle تاثیرات مفید زیادی بر GANها دارد، به‌ویژه در وظایفی که نیاز به تولید تصاویر با وضوح بالا دارند برای مثال می‌توان به موارد زیر اشاره نمود:

پاسخ

(آ) **وضوح بهبود یافته:** با بازآرایی موثر عناصر تنسور، PixelShuffle وضوح تصویر را افزایش می‌دهد که برای تولید تصاویر با کیفیت بالا حیاتی است.

(ب) **کارایی:** PixelShuffle از لحاظ محاسباتی کارا است زیرا نیاز به لایه‌های بزرگ‌نمایی و کانولوشن‌های اضافی را کاهش می‌دهد و شبکه را سریع‌تر و با منابع کمتر می‌سازد.

(ج) **استفاده بهتر از ویژگی‌ها:** این عملگر به استفاده بهتر از ویژگی‌های یادگرفته شده توسط شبکه کمک می‌کند زیرا مستقیماً وضوح مکانی را بدون از دست دادن جزئیات یادگرفته شده بهبود می‌بخشد.

عملگر PixelShuffle ابتدا در زمینه وظایف سوپررزولوشن، به‌ویژه در مقاله Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network" (ESPCN) توسط Wenzhe Shi و همکاران تعریف شد. هدف این بود که با یادگیری یک عملگر کانولوشن زیرپیکسلی کارا، وضوح تصویر را بدون هزینه محاسباتی قابل توجهی افزایش دهد.

استفاده در معماری StackGAN در زیرشبکه دوم:

(آ) در مرحله دوم معماری PixelShuffle هدف بهبود رزولوشن تصاویر تولید شده در مرحله اول است حال اگر با عملگر PixelShuffle ادغام شود، مدل می‌تواند تصاویر با جزئیات بیشتر و کیفیت بالاتری تولید کند.

(ب) PixelShuffle پیچیدگی محاسباتی عملیات بزرگ‌نمایی را کاهش می‌دهد که منجر به زمان‌های آموزش سریع‌تر می‌شود.

۴. معیار FID (Frechet Inception Score) یک معیار برای ارزیابی کیفیت و تنوع تصاویر تولید شده توسط مدل‌های مولد است. توضیح دهید که این معیار به چه صورت محاسبه می‌شود، به چه ویژگی‌هایی از مدل و یا داده وابسته است و آیا معیار قابل اتکایی برای مقایسه‌ی مدل‌های مولد محسوب می‌شود؟

پاسخ

امتیاز FID میزان شباهت بین توزیع تصاویر واقعی و توزیع تصاویر تولید شده را اندازه‌گیری می‌کند. مراحل محاسبه FID به شرح زیر است:

(آ) **استخراج ویژگی‌ها:**
از مدل Inception v3 پیش‌آموزش دیده برای استخراج ویژگی‌ها از تصاویر واقعی و تصاویر تولید شده استفاده می‌شود. فرض می‌کنیم بردارهای ویژگی به‌صورت $\{f_i\}_{i=1}^{N}$ برای تصاویر واقعی و $\{g_j\}_{j=1}^{M}$ برای تصاویر تولید شده باشد

(ب) **برازش توزیع گاوسی چندمتغیره:**
فرض می‌شود که ویژگی‌های استخراج شده از یک توزیع گاوسی چندمتغیره پیروی می‌کنند. میانگین و کواریانس این توزیع‌ها برای تصاویر واقعی و تصاویر تولید شده محاسبه می‌شود:

برای تصاویر واقعی: میانگین μ_r و کواریانس Σ_r

برای تصاویر تولید شده: میانگین μ_g و کواریانس Σ_g

و درنهایت امتیاز FID به‌صورت زیر محاسبه می‌شود:

پاسخ

$$\text{FID} = ||\mu_r - \mu_g||^2 + \text{Tr}(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2})$$

در اینجا، $||\mu_r - \mu_g||^2$ اختلاف مربعی بین میانگین‌های توزیع‌های ویژگی‌های واقعی و تولید شده است، و $\text{Tr}(\cdot)$ قطر ماتریس را نشان می‌دهد.

(آ) **ویژگی‌هایی که lrFID به آن‌ها بستگی دارد:**
امتیاز FID به چند ویژگی مدل و داده بستگی دارد:

- **کیفیت تصاویر تولید شده:**
تصاویر تولید شده با کیفیت بالا که مشابه تصاویر واقعی هستند منجر به امتیاز FID پایین‌تر می‌شوند که نشان‌دهنده عملکرد بهتر مدل مولد است.
- **تنوع تصاویر تولید شده:**
امتیاز FID همچنین تنوع تصاویر تولید شده را نیز در نظر می‌گیرد. اگر تصاویر تولید شده طیف وسیعی از توزیع تصاویر واقعی را پوشش دهند، امتیاز کمتر خواهد بود. فروپاشی مد (که در آن مدل تصاویر کمتر متنوع تولید می‌کند) منجر به افزایش امتیاز FID می‌شود.
- **مدل Inception پیش‌آموزش دیده:**
انتخاب مدل پیش‌آموزش دیده (Inception v3) و لایه‌ای که ویژگی‌ها از آن استخراج می‌شود، تاثیر زیادی بر امتیاز FID دارد. ویژگی‌ها باید به اندازه کافی تمایزپذیر باشند تا جنبه‌های مهم تصاویر را ثبت کنند.
- **تعداد تصاویر:**
تعداد تصاویر استفاده شده برای محاسبه میانگین‌ها و کواریانس‌ها نیز می‌تواند بر امتیاز FID تاثیر بگذارد. اندازه نمونه بزرگتر می‌تواند برآورد دقیق‌تری از توزیع‌های زیرین ارائه دهد.

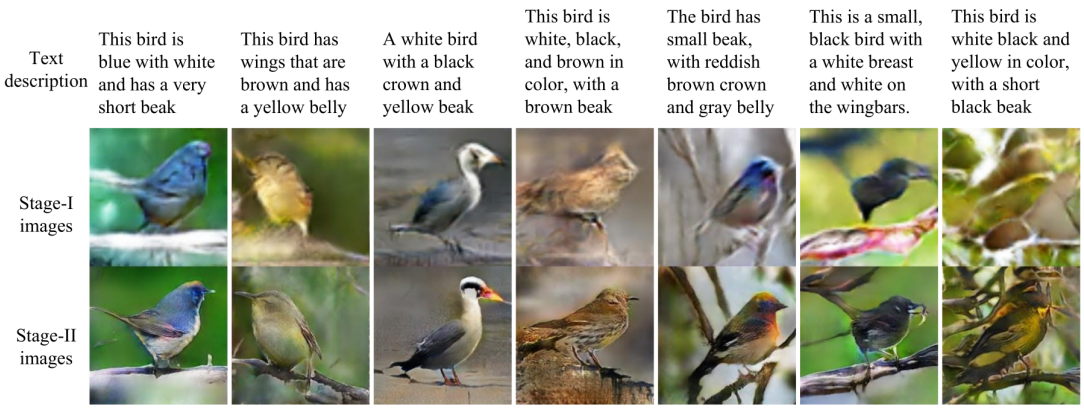
(ب) **قابلیت اطمینان FID برای مقایسه مدل‌های مولد:**
امتیاز FID به طور کلی به عنوان معیاری قابل اعتماد برای مقایسه مدل‌های مولد شناخته می‌شود، اما دارای محدودیت‌هایی نیز هست:

- **حساسیت به انتخاب مدل:**
انتخاب شبکه پیش‌آموزش دیده و لایه‌ای که برای استخراج ویژگی‌ها استفاده می‌شود می‌تواند بر امتیاز تاثیر بگذارد. لایه‌های مختلف ممکن است جنبه‌های مختلف تصاویر را ثبت کنند.
- **وابستگی به مجموعه داده:**
FID وابسته به مجموعه داده خاص و توزیع ویژگی‌های درون آن مجموعه داده است. بنابراین، مقایسه‌ها باید در بستر همان مجموعه داده انجام شود.
- **سوگیری با اندازه نمونه کوچک:**
FID می‌تواند با اندازه نمونه کوچک مغرضانه باشد. نمونه‌های بزرگتر و نماینده‌تر منجر به امتیازهای قابل اعتمادتر می‌شوند.
- **نادیده گرفتن کیفیت ادراکی:**
در حالی که FID شباهت‌های آماری بین توزیع‌ها را ثبت می‌کند، ممکن است کیفیت ادراکی را به طور کامل منعکس نکند. دو مجموعه تصاویر با امتیازهای FID مشابه می‌توانند سطح‌های مختلفی از کیفیت ادراکی داشته باشند.

۵. مدل را بر روی این داده‌ها آموزش دهید. معماری نهایی هر یک از چهار زیرشبکه به همراه نمودار خطای تولیدکننده و

تمایزگر در هر گام آموزش را در گزارش خود بیاورید. پس از پایان آموزش ۱۰ تصویر را به صورت تصادفی از خروجی مدل در stage اول و دوم تولید کنید.

برای این پروژه از مجموعه داده‌ی CUB-2011 استفاده می‌کنیم که شامل یازده هزار تصویر از ۲۰۰ گونه پرنده می‌باشد و به ازای هر تصویر یک توصیف متنی نیز وجود دارد. مجموعه‌ی داده در سایت Kaggle و توصیفات متنی نیز در این لینک موجود است. همچنین برای توصیفات متنی از پیش تعبیه‌ی آماده شده در فایل char-CNN-RNN-embeddings.pickle وجود دارد که می‌تواند جایگزین ساختن داده باشد. استفاده از پیش تعبیه‌ها نیز که منجر به کارایی بهتر مدل شوند دارای ۵ امتیاز اضافی می‌باشد.



شکل ۲: نمونه خروجی مدل StackGAN برای مجموعه داده CUB

پاسخ

شبکه طراحی شده است و در ۶۰۰ دوره آموزش داده شده است.
خروجی های تولید شده این شبکه به ازای متن های مشخص به صورت زیر است:

شکل ۳: خروجی ۱

شکل ۴: خروجی ۲

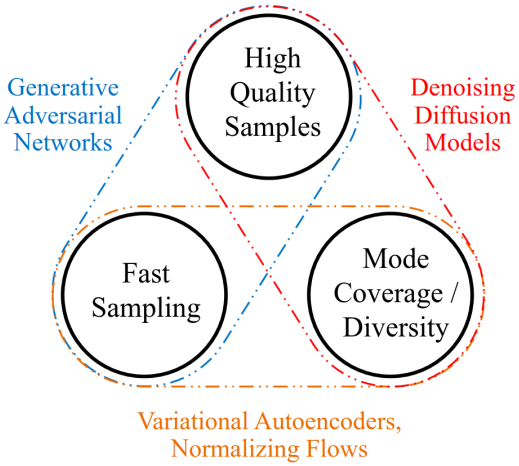
پاسخ

شکل ۵: خروجی ۳

سوال دوم - مدل‌های پخشی (امتیازی)

مدل‌های مولد حوزه‌ی تصویر به چهارچوب‌های مختلف تقسیم می‌شوند. وجه مشترک همه‌ی این مدل‌ها این است که تلاش می‌کنند تا با یادگیری توزیع داده‌ها نمونه‌های جدیدی از آن تولید کنند. تاکنون مدل‌های مولد حوزه‌ی تصویر را می‌توان در چهار قالب کلی شامل خودکدگذارهای تغییراتی[۷]، شبکه‌های مولد تقابلی، جریان‌های نرمال‌ساز[۸] و مدل‌های پخشی[۹] دسته‌بندی کرد که در هر قالب انواع مختلفی از پیاده سازی‌ها وجود دارد.

۱. در این مقاله سه نیازمندی کلی برای کارایی یک مدل مولد حوزه‌ی تصویر ذکر می‌شود که عبارتند از: تولید نمونه‌های با کیفیت، سرعت بالای تولید نمونه و تنوع نمونه‌های تولیدی. و نیز اشاره می‌شود که هر مدل مولدی که تاکنون در یکی از قالب‌هایی که بالاتر ذکر شد ارائه شده است در یکی از این سه نیازمندی ضعیف عمل می‌کند. با بررسی مقاله توضیح دهید که هر مدل در چه زمینه‌ای و به چه علتی ضعیف عمل می‌کند؟



شکل ۶: مشکل سه‌گانه مدل‌های مولد

پاسخ

در این مقاله، ضعف‌های شبکه‌های مولد نام‌برده به‌صورت زیر بیان شده است:

۱. **شبکه GAN:**

- ضعف: پوشش مُد و تنوع نمونه‌ها
- دلیل: شبکه‌های GAN به تولید نمونه‌های با کیفیت بالا و سریع معروف هستند، اما اغلب از پوشش مُد ضعیف رنج می‌برند که به آن پدیده فروپاشی مُد گفته می‌شود. این به این معناست که GANها ممکن است نمونه‌های بسیار مشابهی را به طور مکرر تولید کنند و نتوانند تنوع کامل داده‌های آموزشی را به تصویر بکشند

۲. **VAE ها و Normalizing Flows:**

- ضعف: پوشش مود و تنوع نمونه‌ها

[۷] Variational Autoencoder
[۸] Flows Normalizing
[۹] Diffusion Models

پاسخ

- دلیل: در حالی که VAEs و جریان‌های نرمال‌ساز می‌توانند مُدهای داده را به خوبی پوشش دهند و تنوع در نمونه‌های تولید شده را تضمین کنند، اغلب نمونه‌هایی با کیفیت پایین‌تر نسبت به GANها و مدل‌های انتشار تولید می‌کنند. از دست دادن بازسازی که در VAEs استفاده می‌شود، معمولاً منجر به خروجی‌های مبهم می‌شود و جریان‌های نرمال‌ساز، در حالی که قابل برگشت و انعطاف‌پذیر هستند، می‌توانند در تولید تصاویر با fidelity بالا نیز مشکل داشته باشند

۳. **مدل‌های پخشی**

- ضعف: سرعت نمونه‌گیری پایین
- دلیل: مدل‌های انتشار قادر به تولید نمونه‌های با کیفیت بالا و متنوع هستند اما به دلیل فرآیند نمونه‌گیری آهسته خود دچار محدودیت می‌شوند. این امر به دلیل نیاز به بسیاری از مراحل نویززدایی تکراری است که هر مرحله شامل محاسبات پیچیده‌ای می‌شود. این امر آنها را از نظر محاسباتی گران و زمان‌بر می‌کند که کاربرد عملی آنها را محدود می‌کند

این مقاله (DD-GANs) را برای رفع این ضعف‌ها پیشنهاد می‌کند که نقاط قوت GANها و مدل‌های انتشار را ترکیب می‌کند. هدف DD-GANها کاهش هزینه نمونه‌گیری مدل‌های انتشار در حالی که کیفیت بالا و تنوع نمونه‌ها را حفظ می‌کنند.

مدل‌های پخشی دسته‌ای از مدل‌های مولد هستند که در حال حاضر بهترین مدل تولید تصویر شناخته می‌شوند. در مدل‌های پخشی احتمالاتی از یک ذخیره‌ی مارکف برای مدل کردن فرآیند نویززدایی و نویز افزایی استفاده می‌شود و دو مسیر کلی رو به جلو[۱۰] و رو به عقب[۱۱] در نظر گرفته می‌شود. در مسیر رو به جلو داده‌ی اولیه مرحله به مرحله با نویز تخریب می‌شود تا به یک نویز گاوسی تبدیل شود و در فرآیند رو به عقب نیز نویز زدایی با شروع از یک نویز تصادفی اولیه انجام می‌شود تا به نمونه‌ای جدید از توزیع داده‌ها برسیم. این موضوع در تصویر «۷» آورده شده است.

شکل ۷: فرایند کلی مدل‌های پخشی

۱. با توجه به مقاله‌ی مدل‌های پخشی احتمالاتی، در مسیر رو به جلو نیازی به اضافه کردن نویز به صورت مرحله به مرحله نیست و می‌توان نویز اضافه شونده به هر مرحله را به صورت مستقیم و با استفاده از رابطه‌ی زیر به دست آورد:

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\overline{\alpha}_t}\mathbf{x}_0, (1-\overline{\alpha}_t)\mathbf{I})$$

با استفاده از خاصیت نویز گاوسی رابطه‌ی بالا را اثبات کنید. (راهنمایی: اگر دو متغیر نرمال مستقل داشته باشیم جمع آنها نیز نرمال است.)

[۱۰] Forward
[۱۱] Backward

پاسخ

برای اثبات رابطه:

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\overline{\alpha}_t}\mathbf{x}_0, (1-\overline{\alpha}_t)\mathbf{I})$$

باید از ویژگی نویز گوسی و ماهیت بازگشتی فرآیند انتشار استفاده کنیم.
فرآیند انتشار به گونه‌ای تعریف می‌شود که در هر گام t، نویز به حالت قبلی $\mathbf{x}_{t-1}$ اضافه می‌شود. به طور خاص، فرآیند پیشرو به صورت زیر است:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{\alpha_t}\mathbf{x}_{t-1}, (1-\alpha_t)\mathbf{I})$$

که در آن α_t پارامتری است که میزان نویز اضافه شده در هر گام را کنترل می‌کند.
حال اگر فرآیند را از $\mathbf{x}_0$ تا $\mathbf{x}_t$ در نظر بگیریم. می‌توانیم $\mathbf{x}_t$ را بر حسب $\mathbf{x}_0$ و نویز اضافه شده بنویسیم. با استفاده از ویژگی بازگشتی، داریم:

$$\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\epsilon_t$$

که در آن $\epsilon_t \sim \mathcal{N}(0, \mathbf{I})$ نویزی است که در گام t اضافه شده است.
برای یافتن رابطه بین $\mathbf{x}_t$ و $\mathbf{x}_0$، $\mathbf{x}_{t-1}$ را به‌صورت بازگشتی بسط می‌دهیم:

$$\mathbf{x}_{t-1} = \sqrt{\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-\alpha_{t-1}}\epsilon_{t-1}$$

این بسط را تا $\mathbf{x}_0$ ادامه می‌دهیم:

$$\mathbf{x}_t = \sqrt{\alpha_t}(\sqrt{\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-\alpha_{t-1}}\epsilon_{t-1}) + \sqrt{1-\alpha_t}\epsilon_t$$

$$= \sqrt{\alpha_t\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{\alpha_t(1-\alpha_{t-1})}\epsilon_{t-1} + \sqrt{1-\alpha_t}\epsilon_t$$

این روند را به صورت تکراری ادامه می‌دهیم:

$$\mathbf{x}_t = \left(\prod_{s=1}^{t}\sqrt{\alpha_s}\right)\mathbf{x}_0 + \sum_{s=1}^{t}\sqrt{1-\alpha_s}\left(\prod_{u=s+1}^{t}\sqrt{\alpha_u}\right)\epsilon_s$$

تعریف می‌کنیم $\overline{\alpha}_t = \prod_{s=1}^{t}\alpha_s$. سپس داریم:

$$\mathbf{x}_t = \sqrt{\overline{\alpha}_t}\mathbf{x}_0 + \sum_{s=1}^{t}\sqrt{1-\alpha_s}\left(\prod_{u=s+1}^{t}\sqrt{\alpha_u}\right)\epsilon_s$$

عبارت‌های نویز ϵ_s گاوسی‌های مستقل هستند. جمع متغیرهای تصادفی گاوسی مستقل نیز گاوسی است با میانگین برابر با جمع میانگین‌های آنها و واریانس برابر با جمع واریانس‌های آنها.
میانگین $\mathbf{x}_t$:

$$\mathbb{E}[\mathbf{x}_t] = \sqrt{\overline{\alpha}_t}\mathbf{x}_0$$

واریانس $\mathbf{x}_t$:

$$\text{Var}(\mathbf{x}_t) = \sum_{s=1}^{t}(1-\alpha_s)\left(\prod_{u=s+1}^{t}\alpha_u\right) = (1-\overline{\alpha}_t)\mathbf{I}$$

بنابراین، $\mathbf{x}_t$ با توجه به $\mathbf{x}_0$ از توزیع گاوسی پیروی می‌کند:

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\overline{\alpha}_t}\mathbf{x}_0, (1-\overline{\alpha}_t)\mathbf{I})$$

۲. با توجه به مقاله‌ی سوال قبل، فرآیند آموزش و نمونه برداری مدل‌های پخشی را توضیح دهید. در فرآیند رو به عقب یک فرض مهم این است که توزیع $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ گاوسی است. در چه صورتی این فرض درست است؟

پاسخ

(آ) **فرآیند آموزش:**

- فرآیند پیشرو (انتشار):
فرآیند پیشرو به تدریج نویز را در طول T گام به داده‌ها اضافه می‌کند. از داده اصلی $\mathbf{x}_0$ شروع کرده و نویز به صورت تکراری به آن اضافه می‌شود طبق رابطه:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{\alpha_t}\mathbf{x}_{t-1}, (1-\alpha_t)\mathbf{I})$$

– در اینجا، α_t پارامتری است که میزان نویز اضافه شده در هر گام را کنترل می‌کند. – با انباشته کردن این گام‌ها، می‌توانیم $\mathbf{x}_t$ را به صورت مستقیم بر حسب $\mathbf{x}_0$ بیان کنیم:

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\overline{\alpha}_t}\mathbf{x}_0, (1-\overline{\alpha}_t)\mathbf{I})$$

– این رابطه به ما اجازه می‌دهد که $\mathbf{x}_t$ را در هر گام t از داده تمیز $\mathbf{x}_0$ نمونه‌برداری کنیم.

- فرآیند معکوس (نویززدایی):
– فرآیند معکوس هدف دارد تا داده اصلی $\mathbf{x}_0$ را با حذف تکراری نویز از $\mathbf{x}_T$ به $\mathbf{x}_0$ بازسازی کند.
– فرآیند معکوس توسط مدلی به نام p_θ که توزیع پسین را تقریب می‌زند، پارامتریزه می‌شود:

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t))$$

– پارامترهای μ_θ و Σ_θ در طول آموزش یاد گرفته می‌شوند با بهینه‌سازی کران پایین واریانسی (KL) روی احتمال داده‌ها.

- تابع هزینه:
– هدف آموزشی به حداقل رساندن واگرایی (KL) بین توزیع پسین واقعی $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$ و مدل $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ است:

$$\mathbb{E}_q \left[\sum_{t=1}^{T} D_{KL}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)) \right]$$

(ب) **فرآیند نمونه‌برداری:**

- شروع:
– شروع با یک نمونه از توزیع پیشین $p(\mathbf{x}_t)$ که معمولاً یک توزیع گاوسی استاندارد $\mathcal{N}(0, \mathbf{I})$ است.

- نویززدایی تکراری:
– به صورت ترتیبی $\mathbf{x}_{t-1}$ را از $\mathbf{x}_t$ با استفاده از فرآیند معکوس یادگرفته شده p_θ نمونه‌برداری می‌کنیم:

$$\mathbf{x}_{t-1} \sim p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$$

– این فرآیند تکرار می‌شود تا به $\mathbf{x}_0$ برسیم.

پاسخ

پ **فرض توزیع گاوسی در فرآیند معکوس**

در فرآیند معکوس، فرض مهم این است که توزیع $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ گاوسی باشد. این فرض تحت شرط اندازه گام‌های بی‌نهایت کوچک معتبر است:

- اندازه گام‌های بی‌نهایت کوچک:
  - هنگامی که اندازه گام β_t بسیار کوچک است، فرآیند پیشرو به دقت یک فرآیند انتشار پیوسته زمانی را تقریب می‌زند که در هر گام یک مقدار بی‌نهایت کوچک نویز گاوسی اضافه می‌کند. این اطمینان می‌دهد که فرآیند معکوس که شامل حذف این نویز است، می‌تواند به خوبی توسط یک توزیع گاوسی تقریب زده شود.
  - به صورت ریاضی، این به دلیل این است که حاصلضرب بسیاری از انحراف‌های گاوسی کوچک همچنان گاوسی باقی می‌ماند و نویز انباشته شده در بسیاری از گام‌های کوچک می‌تواند با یک توزیع گاوسی توصیف شود.
- β_t به اندازه کافی کوچک:
  - برای گام‌های متناهی، این فرض زمانی معتبر است که β_t (واریانس نویز در هر گام) به اندازه کافی کوچک باشد تا اطمینان حاصل شود که توزیع شرطی معکوس به گاوسی نزدیک باقی می‌ماند. این معمولاً نیاز به تعداد زیادی از گام‌ها دارد تا به صورت پیوسته از $\mathbf{x}_T$ به $\mathbf{x}_0$ منتقل شود.

۳. یک مساله‌ی اساسی در مدل‌های پخشی این است که در هیچ یک از گام‌ها محاسبات در بعد کوچکتری صورت نمی‌گیرند در نتیجه در صورت بزرگ بودن دیتاست و اندازه‌ی تصاویر ورودی مدل‌های پخشی بسیار پرهزینه و حجیم خواهد شد. مدل Latent Stable Diffusion برای حل این چالش از چه رویکردی استفاده می‌کند؟ به نظر شما چرا برای کاهش حجم نیاز به مدل‌های تغییراتی[۱۲] داریم؟

پاسخ

مدل Latent Stable Diffusion برای کاهش هزینه‌های محاسباتی و حجیم بودن در مجموعه داده‌های بزرگ و تصاویر با وضوح بالا، فرآیند انتشار را در فضای نهان با ابعاد پایین‌تر انجام می‌دهد. این روش با استفاده از یک رمزگذار، تصاویر را به فضای نهان تبدیل کرده و سپس فرآیند انتشار و حذف نویز را در این فضا انجام می‌دهد و در نهایت با رمزگشا تصاویر را بازسازی می‌کند. مدل‌های تغییراتی، مانند خودرمزگذارهای تغییراتی (VAE)، با کاهش ابعاد و تنظیم‌سازی، فضای نهان ساختاریافته‌ای ایجاد می‌کنند که برای این مدل‌ها مفید است. این روش باعث می‌شود تا مدل بتواند داده‌های بزرگ و تصاویر با وضوح بالا را با کارایی بیشتری مدیریت کند.

۴. یک مدل پخشی را بر روی داده‌های MNIST آموزش دهید. تعداد پارامترهای مدل، تصاویر میان فرآیند رو به جلو و فرآیند رو به عقب را برای یک تصویر گزارش کنید. همچنین ۵ تصویر تولید شده توسط این مدل را نمایش دهید.

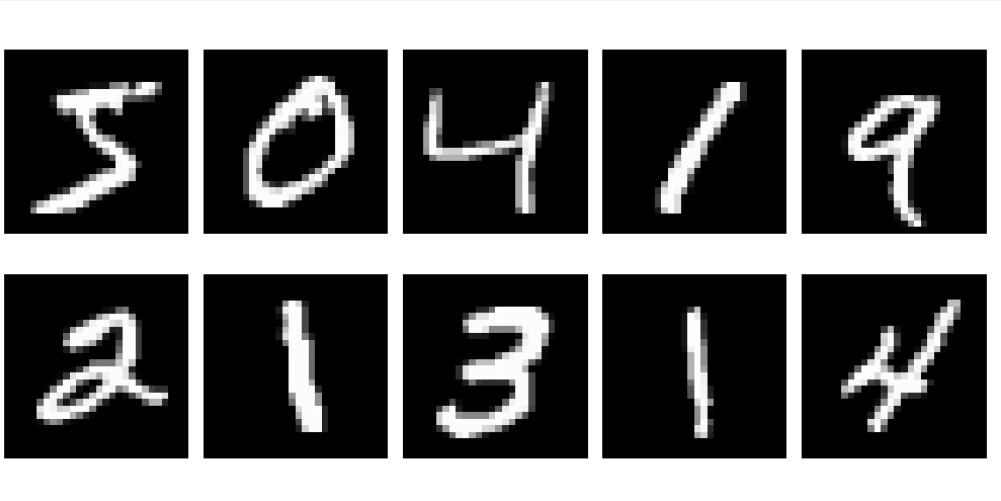
[۱۲] Models Variational

پاسخ

مدل استفاده‌شده در این پیاده‌سازی U-Net است که به طور خاص به عنوان UNet2DModel از کتابخانه diffusers تعریف شده است. در ادامخ مشخصات آن آورده شده است:

```
model = diffusers.UNet2DModel(
sample_size=config.image_size,
in_channels=1,
out_channels=1,
layers_per_block=2,
block_out_channels=(128, 128, 256, 512),
down_block_types=(
"DownBlock2D",
"DownBlock2D",
"AttnDownBlock2D",
"DownBlock2D",
),
up_block_types=(
"UpBlock2D",
"AttnUpBlock2D",
"UpBlock2D",
"UpBlock2D",
),
)
```

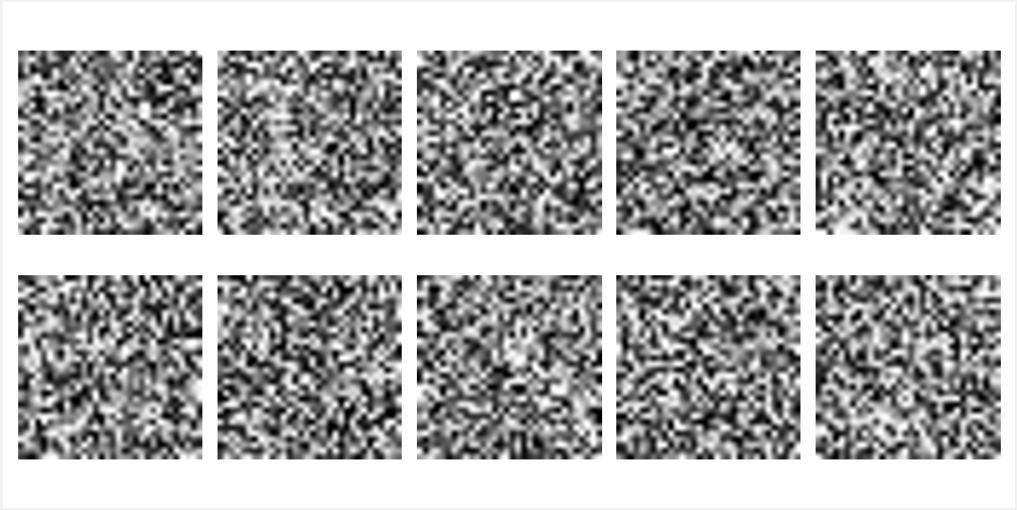
یکی از ورودی‌های شبکه قبل از اعمال نویز به‌صورت زیر است:



شکل ۸: تصویر بدون نویز ورودی

پس از اعمال نویز گوسی تصویر به‌صورت زیر می‌شود:

پاسخ



شکل ۹: تصویر نویزی خروجی

خروجی‌های مرحله معکوس به صورت زیر است:

شکل ۱۰: خروجی‌های تولید شده در مرحله معکوس