به نام خدا

علی قاسم زاده

۴۰۱۱۰۶۳۳۹

مدل های مولد

تمرین ۳

سوال اول

الف) می دانیم که در $reparametrization trick$ بجای سمپل برداری از $\mathcal{N}(\mu,1)$ از $\mathcal{N}(0,1)$ سمپل بر می داریم و سپس سمپل ها را با μ جمع می کنیم، پس داریم که :

$$\mathbb{E}_{z\sim\mathcal{N}(\mu,1)}[f(z)] = \mathbb{E}_{\epsilon\sim\mathcal{N}(0,1)}[f(\mu+\epsilon)] \approx \frac{1}{N}\sum_{i=1}^{N} f(\mu+\epsilon_i)$$

$$\eta(\mu) = \nabla_\mu \mathbb{E}_{\epsilon\sim\mathcal{N}(0,1)}[f(\mu+\epsilon)] \approx \nabla_\mu \frac{1}{N}\sum_{i=1}^{N} f(\mu+\epsilon_i), \quad \epsilon_i \sim \mathcal{N}(0,1)$$

حالا این برآورد را α می نامیم :

$$\alpha(\mu) = \frac{1}{N}\sum_{i=1}^{N} \nabla_\mu f(\mu+\epsilon_i)$$

حالا اگر قرار دهیم که $f(z) = mz$ خواهیم داشت که :

$$\eta(\mu) = m\nabla_\mu \mathbb{E}_{\epsilon\sim\mathcal{N}(0,1)}[(\mu+\epsilon)] = m\mathbb{E}_{\epsilon\sim\mathcal{N}(0,1)}[\nabla_\mu(\mu+\epsilon)] =$$

$$m\mathbb{E}_{\epsilon\sim\mathcal{N}(0,1)}[1] = m$$

پس واریانس آن صفر خواهد بود زیرا نسبت به μ ثابت است. حالا اگر قرار دهیم که $f(z) = (mz)^2$ خواهیم داشت که :

$$\eta(\mu) = m^2\nabla_\mu \mathbb{E}_{\epsilon\sim\mathcal{N}(0,1)}[(\mu+\epsilon)^2] = m^2\mathbb{E}_{\epsilon\sim\mathcal{N}(0,1)}[\nabla_\mu(\mu^2+\epsilon^2+2\mu\epsilon)] =$$

$$m^2\mathbb{E}_{\epsilon\sim\mathcal{N}(0,1)}[2\mu+2\epsilon] = 2m^2\mu$$

ب) ابتدا می دانیم که :

$$\nabla \log P(z;\mu) = \frac{\nabla_\mu P(z;\mu)}{P(z;\mu)}$$

حال داریم که :

$$p(z;\mu)\nabla_\mu \log P(a;\mu) = \nabla_\mu P(z;\mu)$$

حالا داشتیم که :

$$\eta(\mu) = \nabla_\mu \mathbb{E}_{z\sim\mathcal{N}(\mu,۱)}[f(z)]$$

می دانیم که P هم تابع توزیع $\mathcal{N}(\mu,۱)$ است پس داریم که :

$$\eta(\mu) = \nabla_\mu \mathbb{E}_{z\sim P}[f(z)] = \nabla_\mu \int_{-\infty}^{\infty} P(z;\mu) f(z) dz =$$

$$\int_{-\infty}^{\infty} f(z)(\nabla_\mu P(z;\mu))dz = \int_{-\infty}^{\infty} f(z)P(z;\mu)(\nabla_\mu \log P(z;\mu))dz =$$

$$\mathbb{E}_{z\sim P(z;\mu)}[f(z)\nabla_\mu \log P(z;x)] = \mathbb{E}_{z\sim\mathcal{N}(\mu,۱)}[f(z)\nabla_\mu \log P(z;x)] \approx$$

$$\frac{۱}{N}\sum_{i=۱}^{N} f(z_i)\nabla_\mu \log P(z_i;\mu), \quad z_i \sim \mathcal{N}(\mu,۱)$$

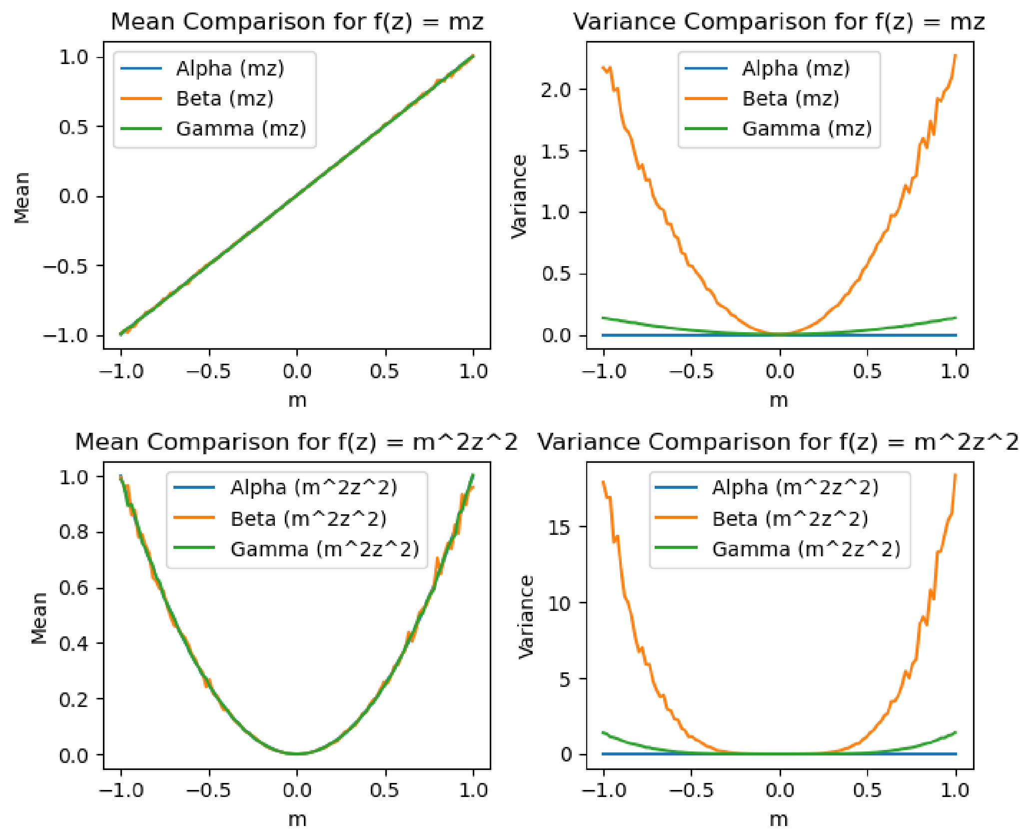
و $\beta(\mu)$ هم دقیقا همین تخمین بالا است.

$$\beta(\mu) = \frac{۱}{N}\sum_{i=۱}^{N} f(z_i)\nabla_\mu \log P(z_i;\mu), \quad z_i \sim \mathcal{N}(\mu,۱)$$

همچنین داریم که :

$$\nabla_\mu \log P(z_i;\mu) = -\frac{۱}{۲}\nabla_\mu (z_i-\mu)^۲ = \mu - z_i$$

پ) نمودار را برای $N = ۱۰۰۰$ سمپل رسم می کنیم :



با توجه به نتایج کد پایتون داریم که میانگین هر سه نمودار تقریبا یکسان است ولی واریانس روش اول صفر است پس بهترین انتخاب است در هر دو حالت هم واریانس روش دوم بیشتر از سایرین است پس بعد از روش اول روش سوم بهتر است چون واریانسش بین روش اول و دوم است.

روش اول زمان هایی که توزیع z قابل بازسازی($reparametrization$) باشد خوب است.
روش دوم گرایش به تولید واریانس زیاد دارد.
روش سوم هم پیچیدگی محاسباتی بیشتری نسبت به سایرین دارد.
در این مسئله از آنجایی که می توانیم $reparametrization$ انجام دهیم پس روش اول مناسب ترین است.

سوال دوم

الف) ابتدا $D_{KL}(p||q)$ و $D_{KL}(q||p)$ را حساب می کنیم، می دانیم که داریم :

$$p(x) = \frac{۱}{\sqrt{۲\pi}\sigma} e^{-\frac{(x-\mu_۱)^۲}{۲\sigma^۲}}$$

$$q(x) = \frac{۱}{\sqrt{۲\pi}\sigma} e^{-\frac{(x-\mu_۲)^۲}{۲\sigma^۲}}$$

$$\longrightarrow \frac{p(x)}{q(x)} = e^{-\frac{(x-\mu_۱)^۲}{۲\sigma^۲}+\frac{(x-\mu_۲)^۲}{۲\sigma^۲}}$$

$$D_{KL}(p||q) = \int_{-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)} dx = \int_{-\infty}^{\infty} p(x) \log e^{-\frac{(x-\mu_۱)^۲}{۲\sigma^۲}+\frac{(x-\mu_۲)^۲}{۲\sigma^۲}} dx =$$

$$\int_{-\infty}^{\infty} p(x)(\frac{-x^۲ + ۲x\mu_۱ + \mu^۲ + x^۲ - ۲x\mu_۲\mu_۲^۲}{۲\sigma^۲})dx = \int_{-\infty}^{\infty} p(x)(\frac{۲x\mu_۱ + \mu^۲ - ۲x\mu_۲\mu_۲^۲}{۲\sigma^۲})dx =$$

$$-\frac{\mu_۱^۲}{\sigma^۲}\int_{-\infty}^{\infty} p(x)dx + \frac{\mu_۲^۲}{\sigma^۲}\int_{-\infty}^{\infty} p(x)dx + \frac{۲\mu_۱}{۲\sigma^۲}\int_{-\infty}^{\infty} xp(x)dx - \frac{۲\mu_۲}{۲\sigma^۲}\int_{-\infty}^{\infty} xp(x)dx =$$

$$\frac{\mu_۱^۲ - ۲\mu_۱\mu_۲ + \mu_۲^۲}{۲\sigma^۲} = \frac{(\mu_۱ - \mu_۲)^۲}{۲\sigma^۲} \quad ***$$

از طرفی می دانیم که $p(x)$ نسبت به نقطه ی $\mu_۱$ متقارن است زیرا اگر داشته باشیم $\mu_۱+\epsilon$ و $\mu_۱-\epsilon$ آنگاه داریم که $p(\mu_۱+\epsilon) = p(\mu_۱-\epsilon)$ پس حالا اگر در انتگرال داشتیم که : $\int_{-\infty}^{\infty}(x-\mu_۱)p(x)dx$ حاصا این انتگرال صفر بود پس داریم که

$$\int_{-\infty}^{\infty} xp(x)dx = \mu_۱ \int_{-\infty}^{\infty} p(x)dx = \mu_۱$$

حالا پس رابطه ی نوشته شده ی بالا درست است. از طرفی اگر جای $\mu_۱$ و $\mu_۲$ را عوض کنیم انگار توزیع های p و q را عوض کرده ایم و در نتیجه $D_{KL}(q||p)$ بدست می آید حالا با توجه به مقدار KL دراولی ، مقدار KL دومی هم با جابه جایی p و q همان می شود و مینیممشان هم همان می شود.
پس داریم که :

$$D_{Kl}(p||q) = D_{KL}(q||p) = \frac{(\mu_۱ - \mu_۲)^۲}{۲\sigma^۲}$$

حالا انتگرال $\int_{-\infty}^{\infty}$ را حساب می کنیم :

$$\int_{-\infty}^{\infty} p(x)q(x)dx = \int_{-\infty}^{\infty} \frac{۱}{۲\sigma^۲} e^{-\frac{(x-\mu_۱)^۲}{۲\sigma^۲}-\frac{(x-\mu_۲)^۲}{۲\sigma^۲}} dx = \int_{-\infty}^{\infty} \frac{۱}{۲\sigma^۲} e^{-\frac{(x-\frac{\mu_۱+\mu_۲}{۲})^۲}{\sigma^۲}} =$$

$$\frac{۱}{۲\sqrt{\pi}\sigma}e^{-\frac{(\mu_۱-\mu_۲)^۲}{۴\sigma^۲}}\int_{-\infty}^{\infty}\mathcal{N}(x;\frac{\mu_۱+\mu_۲}{۲},\frac{\sigma^۲}{۲})dx=\frac{۱}{۲\sqrt{\pi}\sigma}e^{-\frac{(\mu_۱-\mu_۲)^۲}{۴\sigma^۲}}$$

حالا اگر خود p را بجای q بزاریم داریم که عبارت نمایی ۱ می شود و فقط عبارت کسری می ماند. ئر نهایت داریم که :

$$D_{cs}(p||q)=-\log\frac{\int_{-\infty}^{\infty}p(x)q(x)}{\sqrt{(\int_{-\infty}^{\infty}p(x)^۲dx)}\sqrt{(\int_{-\infty}^{\infty}q(x)^۲dx)}}=-\frac{(\mu_۱-\mu_۲)^۲}{۴\sigma^۲}$$

این عبارتی منفی است در حالی که KL بزرگتر مساوی ۰ است پس این نامساوی برقرار است.

ب)

$$\mathcal{L}_{cs}=\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]-\lambda D_{cs}(q_\phi(z|x)||p(z))\longrightarrow$$

$$\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]=\int_{-\infty}^{\infty}q_\phi(z|x)\log p_\theta(x|z)dz=$$

$$\int_{-\infty}^{\infty}q_\phi(z|x)\log\frac{p_\theta(x)p_\theta(z|x)}{p_\theta(z)}dz=$$

$$\int_{-\infty}^{\infty}q_\phi(z|x)\log p_\theta(x)dz+\int_{-\infty}^{\infty}q_\phi(z|x)\log(\frac{p_\theta(z|x)}{q_\phi(z|x)}\frac{q_\phi(z|x)}{p_\theta(z)})dz=$$

$$\log p_\theta(x)-KL(q_\phi(z|x)||p_\theta(z|x))+KL(q_\phi(z|x)||p_\theta(z))-\lambda D_{cs}(q_\phi(z|x)||p(z))$$

در نتیجه حکم ثابت می شود.
- $\log p(x)$ که لگاریتم احتمال دیتای مشاهده شده ی x را نشان می دهد. به $objective$ برای vae است که بتواند خوب سمپل تولید کند (سمپل هایی نزدیک به توزیع ورودی).
- $-KL(q_\phi(z|x)||p_\theta(z|x))$ این ترم هر چقدر بزرگتر باشد نشان می دهد که توزیع شرطی q به توزیع شرطی p نزدیک تر است.(p توزیع واقعی دیتا)
- $KL(q_\phi(z|x)||p_\theta(z))$ این ترم اجازه می دهد تا توزیع q از توزیع $p(z)$ فاصله بگیرد.
- $\lambda D_{cs}(q_\phi(z|x)||p(z))$ مدل را مجبور می کند تا نزدیک به توزیع $p(z)$ بماند.

اگر مقدار λ را افزایش دهیم فضای پنهان محدود تر می شود و ممکن است کیفیت توزیع $prior$ بهتر شود ولی باعث می شود تا کیفیت $reconstrcution$ کاهش یابد.

سوال سوم

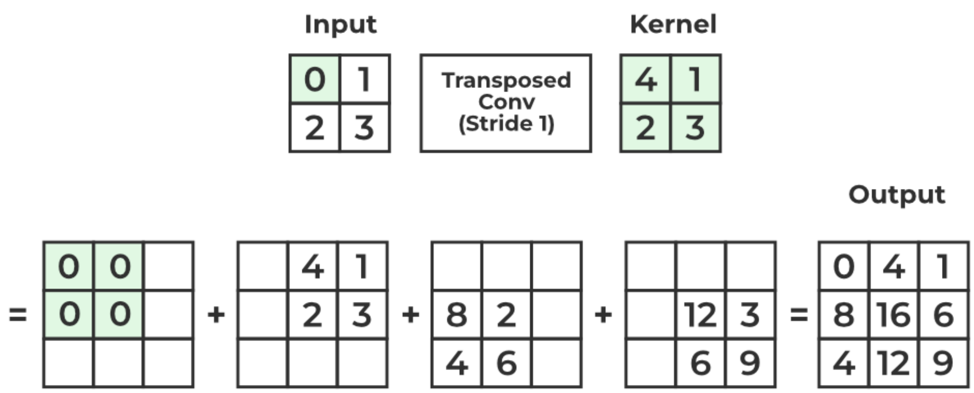
توضیخاتی نیز در آخر فایل جوپیتر قرار داده شده اند، علاوه بر آن اینجا هم موارد لازم گفته شده اند.

الف) مدل ما اینگونه عمل می کند که یک ورودی از $\mathbb{R}^{۲۸x۲۸}$ می گیرد و آنرا به $\mathbb{R}^{۱۰}$ می برد. همچنین داریم که

$$q_\phi(z|x) = \mathcal{N}(z|\mu, diag(\sigma^۲)), \quad \log \sigma^۲ = g_\phi^\sigma(x)$$

ب) یه سری لایه ی خطی داریم که $\mathbb{R}^{۱۲۸}$ را به $\mathbb{R}^{۱۰}$ می برند. برای هرکدام از متغییر های μ و $\sigma^۲$ یکی از این لایه ها داریم.

ج) $ConvTranspose۲d$ برای $UPsampling$ به کار می رود (افزایش ارتفاع و طول عکس ها) نحوه ی کار آن به صورت زیر است :



به صورت نقطه به نقطه ضرب داخلی می کند و با مقادیر قبلی پیکسل متناظر جمع می کند. فرمول آن هم از رابطه ی زیر بدست می آید :

$$h_{new} = s(h_{old} - ۱) + k - ۲p + p' w_{new}$$

$$s \rightarrow stride, \quad k \rightarrow kernel_size, \quad p \rightarrow padding, \quad p' \rightarrow added\ to\ theside\ of\ the\ output$$

د) از KL داشتیم که :

$$\log \frac{q(z|x)}{p(z)} = \log \frac{\frac{۱}{\sqrt{۲\pi\sigma}} exp(-\frac{(z-\mu)^۲}{۲\sigma^۲})}{\frac{۱}{\sqrt{۲\pi\sigma}} exp(-\frac{z^۲}{۲})} = -\log\sigma - \frac{۱}{۲} + \frac{۱}{۲} z^۲$$

$$\mathbb{E}[z] = \mu, \quad \mathbb{E}[(z-\mu)^۲] = \sigma^۲ \longrightarrow \mathbb{E}[z^۲] = \sigma^۲ + \mu^۲$$

$$D_{KL} = \mathbb{E}[\log\sigma - \frac{۱}{۲} + \frac{۱}{۲}z^۲] = -\frac{۱}{۲}(\log\sigma^۲ + ۱ - \sigma^۲ - \mu^۲) = -\frac{۱}{۲}(logvar + ۱ - exp(logvar) - \mu^۲)$$

ر) معیار FID از دو مرحله ی اصلی تشکیل شده است :
۱. استخراج ویژگی ها : ابتدا ویژگی های تصاویر توسط یک شبکه ی از پیش آموزش داده شده مانند $Inceptionv۳$ استخراج می شوند. این شبکه برای هر تصویر، یک بردار ویژگی در فضای ویژگی تعریف شده توسط لایه های داخلی شبکه ی $inception$ ایجاد می کند.
۲. مقایسه توزیع ها : بردارهای ویژگی تصاویر واقعی و تصاویر تولیدی به دست آمده از مدل به کار رفته جمع‌آوری شده و توزیع آماری آن‌ها محاسبه می‌شود. سپس، فاصله فرشه بین دو توزیع ویژگی (که به طور معمول با استفاده از میانگین و کوواریانس بردارهای ویژگی تعیین می‌شود) محاسبه می‌شود. این فاصله نشان‌دهنده تفاوت آماری بین دو گروه تصویر است.
فرمول آن عبارت است از :

$$FID = \|\mu_۱ - \mu_۲\|^۲ + Tr(\sigma_۱ + \sigma_۲ - ۲\sqrt{\sigma_۱ * \sigma_۲})$$

معیار FID اطلاعات مهمی در مورد کیفیت تصاویر تولید شده توسط مدل‌هایی مانند VAE ارائه می‌دهد:
۱.تنوع: FID به خوبی توانایی مدل در تولید تصاویر متنوع را ارزیابی می‌کند. اگر تصاویر تولیدی دارای تنوع کمی باشند و شبیه به هم باشند، مقدار FID بالا خواهد رفت زیرا توزیع ویژگی‌های تصاویر تولیدی تفاوت زیادی با توزیع تصاویر واقعی خواهد داشت.
۲.واقع‌گرایی: FID همچنین کیفیت و واقع‌گرایی تصاویر تولیدی را ارزیابی می‌کند. یک مقدار پایین FID نشان‌دهنده آن است که تصاویر تولیدی در سطح ویژگی‌های دیداری نزدیک به تصاویر واقعی هستند. فرمول بالا را می توان به صورت زیر برای مدل مان بنویسیم :

$$FID = \|\mu_r - \mu_g\|^۲ + Tr(\sum_r + \sum_g - ۲(\sum_r \sum_g)^{\frac{۱}{۲}})$$

که میانگین و ماتریس کواریانس مربوط به تصاویر و تصاویر تولید شده هستند.

ه) همانطور که از خروجی های چاپ شده ی بالا پیدا است، FID در داده های $reconstructed$ کمتر از داده های $randomgenerated$ است و علتش هم واضح است که زیرا که در اولی دیتایی دارد که از روی آن بازسازی کند ولی در دومی دیتای رندومی با دیکودر داده شده است و

ع) مقایسه با FID
تنوع vs واقع‌گرایی: FID به طور مستقیم واقع‌گرایی تصاویر تولیدی را با توجه به تصاویر واقعی اندازه‌گیری می‌کند و به خوبی تنوع را در نظر می‌گیرد. در حالی که IS بیشتر بر تنوع تمرکز دارد و ممکن است در ارزیابی واقع‌گرایی دقیق کوتاهی کند.
کاربرد در تحلیل فضای $PPL, PRD\ latent$ اطلاعات بیشتری در مورد رفتار فضای $latent$ و تناسب آن با داده‌های واقعی ارائه می‌دهند، که می‌تواند برای درک بهتر دینامیک‌های مدل و بهبود ساختارهای $latent$ مفید باشد. در کل هر معیاری نقاط ضعف و قوت خود را دارد و بسته به هدفی که مد نظر داریم هر کدام را استفاده می کنیم.

سوال چهارم

الف) ما به $CGAN$ ها یک ورودی x هم می دهیم که می تواند $text$ یا $label$ یا $image$ باشد و سپس داریم که :

$$\mathcal{L}_{GAN} = \mathbb{E}_{x,y}[\log D(x,y)] + \mathbb{E}_{x,z}[\log(1 - D(x, G(x,z))]$$

ب)
۱. انتقال سبک $(Style\ Transfer)$ این تکنیک به شما این امکان را می‌دهد که سبک یک تصویر (مانند نقاشی‌های ون‌گوگ یا مونه) را به تصویر دیگری منتقل کنید، در حالی که محتوای اصلی تصویر حفظ می‌شود. برای مثال، یک عکس ساده می‌تواند شبیه یک اثر هنری نقاشی شود.
کاربردش هم خلق آثار هنری دیجیتال و ویرایش تصاویر است.
A Neural Algorithm of Artistic Style
۲. رنگ‌آمیزی تصاویر $(ImageColorization)$ این روش به تصاویر سیاه و سفید یا خاکستری رنگ اضافه می‌کند. این کاربرد به‌ویژه برای ترمیم عکس‌های قدیمی و افزایش جذابیت بصری تصاویر مفید است. کاربردش هم ترمیم تصاویر تاریخی و بهبود داده‌های تصویری در تحقیقات است.
Deep Learning for Image Colorization
۳. ارتقای وضوح تصویر $(Super - Resolution)$ در این روش، وضوح یک تصویر با جزئیات بیشتر ارتقا پیدا می‌کند. این تکنیک می‌تواند برای بهبود کیفیت تصاویر کم‌وضوح، مانند تصاویر دوربین‌های امنیتی، استفاده شود. کاربردش هم بهبود کیفیت ویدیوها و تصاویر در کاربردهای پزشکی یا امنیتی است.

Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network

ج) Pix۲Pix یک مدل $Image-to-ImageTranslation$ مبتنی بر GAN است که به داده‌های جفت‌شده نیاز دارد. داده‌های جفت‌شده شامل ورودی و خروجی‌های مربوطه هستند. هدف این مدل تولید تصویری است که کاملاً مرتبط با ورودی باشد. در واقع یادگیری آن $supervised$ است.

برای مثال : تبدیل عکس‌های نقشه به تصاویر واقعی یا عکس‌های اسکچ به تصاویر رنگی.

نوع $Generator$ و $Discriminator$:

$Generator$: از معماری $U-Net$ استفاده می‌کند. این معماری به دلیل حفظ جزئیات در تصویر خروجی مناسب است.

$Discriminator$:

از معماری $PatchGAN$ استفاده می‌کند که در آن تصاویر به صورت "پچ‌های" کوچک بررسی می‌شوند تا شباهت‌های محلی بهبود پیدا کنند.

کاربرد ها :

رنگ‌آمیزی تصاویر ($Image\ Colorization$).

تولید تصاویر واقع‌گرایانه از طرح‌های ساده.

بازسازی تصاویر ($Image\ Restoration$).

CycleGAN نیز یک مدل $Image-to-ImageTranslation$ مبتنی بر GAN است، اما بر خلاف $Pix۲Pix$، به داده‌های جفت‌شده نیاز ندارد. این مدل می‌تواند بدون داشتن داده‌های تطبیقی، تصاویر را از یک دامنه به دامنه‌ای دیگر تبدیل کند.در واقع یادگیری آن $unsupervised$ است.

برای مثال: تبدیل تصاویر اسب به تصاویر گورخر یا تصاویر تابستانی به زمستانی.

نوع $Generator$ و $Discriminator$:

$Generator$:

از معماری ResNet استفاده می‌کند. این معماری برای انتقال ویژگی‌های پیچیده مناسب است.

$Discriminator$:

مانند $Pix۲Pix$ از معماری $PatchGAN$ استفاده می‌کند، اما تفاوت‌هایی در تنظیمات آن وجود دارد. ویژگی منحصربه‌فرد $CycleGAN$ این است که دو $Generator$ و دو $Discriminator$ دارد. یکی برای تبدیل A به B و دیگری برای تبدیل B به A.

کاربردها:

انتقال سبک بین تصاویر ($StyleTransfer$).

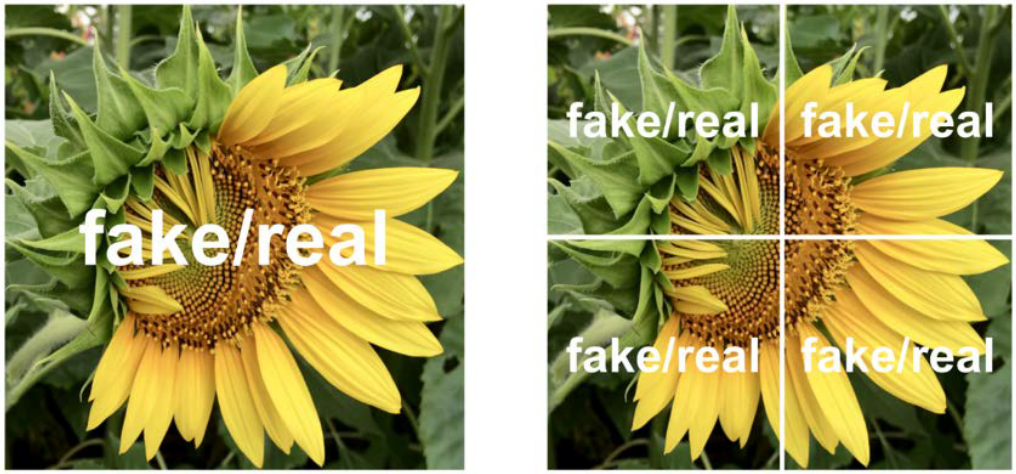
تبدیل تصاویر شب به روز یا بالعکس.

تغییر دامنه‌های بدون تطبیق ($Unpaired\ Image\ Translation$).

د) خواندن مقاله

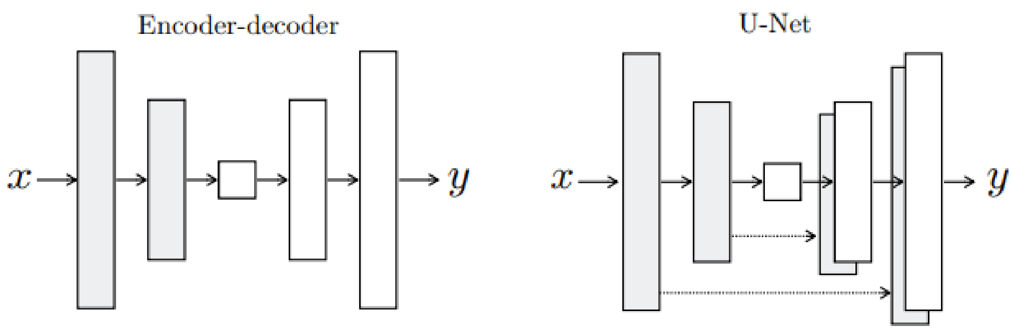
ه) تفاوت آن ین است که در این حالت discriminator و generator به ورودی x نیز وابسته اند، در نتیجه generator با یک z و x شروع می کند و خروجی y روی x، conditioned می شود. در نتیجه باید یک ترم دیگه هم objective مان اضافه کنیم تا فاصله ی y و $G(x, z)$ را اندازه بگیرد.

و)
۱. تفاوت اصلی در معماری Discriminator در GAN عادی:
هدف اصلی Discriminator در GAN عادی، تشخیص واقعی یا مصنوعی بودن یک تصویر کامل است. این Discriminator کل تصویر ورودی را به عنوان یک واحد در نظر می‌گیرد و سعی می‌کند تصویر را به صورت کلی ارزیابی کند.
Discriminator در PatchGAN :
در معماری PatchGAN، به جای ارزیابی کل تصویر، تصویر به پچ‌های کوچک NxN تقسیم می‌شود. هر پچ به صورت مستقل بررسی شده و طبقه‌بندی می‌شود. خروجی نهایی این Discriminator میانگین نتایج تمام پچ‌ها است.
۲. تفاوت در نوع پردازش GAN عادی:
تمرکز اصلی بر کلیت تصویر است و ممکن است به جزئیات کوچک تصویر توجه کافی نداشته باشد.
PatchGAN :
طبقه‌بندی محلی روی پچ‌ها باعث می‌شود به جزئیات ظریف‌تر تصویر مانند بافت‌ها، لبه‌ها و جزئیات کوچک توجه بیشتری شود.
۳. بهبود عملکرد مدل با توجه به جزئیات محلی، PatchGAN قادر است تصاویری با کیفیت بالاتر و واقعی‌تر تولید کند. این رویکرد از تولید تصاویر غیرواقعی در بخش‌های کوچک تصویر جلوگیری می‌کند. همچنین این معماری باعث می‌شود مدل برای کاربردهایی که به جزئیات بالا نیاز دارند (مانند انتقال سبک یا ترمیم تصاویر) بهتر عمل کند.
در زیر تفاوت آنها در شکل آمه است :



GAN discriminator 2x2 PatchGAN discriminator

ز)



در این حالت generator ما یک Unet است و یه سری skip connection از encoder به decoder در آن اضافه شده اند که جزئیات اطلاعات قدیمی را هم نگه دارند برای همین عکس هایی با کیفیت بهتر تولید می کند و kernel size در همه جا ۴ است و pooling ای نداریم.

همچنین در بخش downsample ابعاد عکس را کاهش می دهیم با استفاده از conv۲d و در upsample ابعاد عکس را افزایش می دهیم با استفاده از convtrans-pose۲d

همچنین متغییر رندوم z برای ایجاد تنوع کافی نیست و برای همین از dropout برابر با ۰.۵ استفاده می کنیم. با این احتمال بالا dropout تنوع خوبی در generator ایجاد می کند.

ح)
discriminator loss به دو بخش تقسیم می شود :

$$\mathcal{L}_D = -\frac{۱}{۲}(\mathbb{E}_{x,y}[\log D(x,y)] + \mathbb{E}_{x,y}[\log(۱ - D(x, G(x,z)))])$$

همچنین در generator بجای $\log(۱ - D)$ از $-\log D$ استفاده می کنیم همچنین ترم نرمالایزر را اگر نرم دو برداریم تصاویر مات می شوند برای همین از نرم یک استفاده می کنیم :

$$\mathcal{L}_G = -\mathbb{E}_{x,z}[\log D(x, G(x,z))] + \lambda\mathbb{E}_{x,y,z}[||y - G(x,z)\|_۱]$$

برای آپدیت هم روی همین loss ها backward می زنیم و discriminator و generator را بر اساس تابع loss شان آپدیت می کنیم.

سوال پنجم

لینک ها را در زیر مشاهده می کنید :
github

paperswithcode
با نام :
Alighasemzadeh۸۳/Pix۲Pix-implementation-from-scratch

kaggle با نام :
notebookb۲a۵۵ac۰۶f

medium
