

به نام خدا

گزارش پژوهه اول درس یادگیری ماشین

بازسازی و بهبود کیفیت تصاویر چهره با استفاده از شبکه‌های عصبی کانولوشنی (CNN)

هدی عطاری - 401243068

۱. مقدمه

پردازش تصاویر دیجیتال و بینایی ماشین (Computer Vision) نقش بسزایی در کاربردهای مختلف از جمله امنیت، سرگرمی و پزشکی ایفا می‌کنند. یکی از چالش‌های اساسی در این حوزه، وجود تصاویر با کیفیت پایین است که ممکن است ناشی از نویز محیطی، محدودیت‌های سخت‌افزاری دوربین (رزولوشن پایین) یا تاری ناشی از حرکت (Motion Blur) باشد. بازسازی چهره (Face Restoration) به فرآیند بازیابی یک تصویر چهره با کیفیت بالا از یک ورودی تخریب‌شده (Degraded) اطلاق می‌شود.

این پژوهه با هدف طراحی و پیاده‌سازی یک مدل یادگیری عمیق (Deep Learning) برای حل این چالش تعریف شده است. مرکز اصلی بر استفاده از شبکه‌های کانولوشنی (CNN) و معماری‌های Encoder-Decoder (مانند U-Net) است تا بتوانیم جزئیات از دست رفته چهره را بازیابی کنیم. داده‌های مورد استفاده از مجموعه داده مشهور CelebA انتخاب شده‌اند که شامل تنوع بالایی از چهره‌هاست.

۲. اهداف پژوهه

بر اساس نیازمندی‌های تعریف شده، اهداف اصلی و مراحل اجرایی این پژوهه عبارتند از:

۱. پیش‌پردازش و همترازی (Alignment): شناسایی چهره و همتراز کردن آن بر اساس نقاط شاخص صورت (Landmarks) برای ایجاد یک ساختار یکنواخت در داده‌های آموزشی.
۲. تولید داده‌های تخریب‌شده (Degradation Simulation): شبیه‌سازی مصنوعی خرابی‌ها شامل:
 - کاهش رزولوشن (Downsampling) با ضرایب ۱.۲ و ۱.۴.
 - افزودن نویز گاووسی (Gaussian Noise).
 - اعمال تاری حرکتی (Motion Blur).
۳. مدل‌سازی: طراحی و آموزش شبکه عصبی U-Net برای نگاشت تصویر تخریب‌شده به تصویر اصلی.
۴. توابع هزینه ترکیبی: استفاده همزمان از خطای پیکسلی (Pixel-wise Loss) و خطای هویت (Identity/Perceptual Loss) برای حفظ ویژگی‌های هویتی فرد.
۵. ارزیابی: سنجش عملکرد مدل با معیارهای PSNR، SSIM و فاصله Embedding و تست تعمیم‌پذیری روی داده‌های خارج از توزیع (Out-of-Distribution).

۳. آماده‌سازی مجموعه داده (Data Preparation)

۱-۳. جمع‌آوری داده‌ها (Data Acquisition)

برای آموزش مدل، از مجموعه داده مشهور CelebA استفاده شده است که شامل بیش از ۲۰۰ هزار تصویر چهره با ویژگی‌های متنوع است. با توجه به حجم بالای داده‌ها و محدودیت‌های پهنای باند در محیط Google Colab (به‌ویژه محدودیت‌های Quota در Google Drive)، در این پروژه از یک استراتژی جایگزین برای دریافت داده‌ها استفاده شد. به جای دانلود مستقیم از سرور اصلی یا درایو، داده‌ها از سرور آینه‌ای (Mirror Udacity) که پایداری و سرعت بالاتری دارد، دریافت شدند.

۲-۳. پیش‌پردازش اولیه و بارگذاری (Preprocessing & Loading)

برای مدیریت کارآمد داده‌ها در حافظه، یک کلاس اختصاصی با نام CustomCelebA با ارث‌بری از کلاس پایه Dataset در کتابخانه PyTorch پیاده‌سازی شد. این کلاس وظایف زیر را بر عهده دارد:

1. جستجوی فایل‌ها: یافتن تمام تصاویر با فرمت jpg در مسیر استخراج شده.
2. بارگذاری تبل (Lazy Loading): تصاویر تنها در زمان نیاز (هنگام درخواست توسط DataLoader) در حافظه بارگذاری می‌شوند تا از پر شدن حافظه RAM جلوگیری شود.
3. تغییر اندازه (Resizing): تمامی تصاویر ورودی به ابعاد استاندارد 128×128 پیکسل تغییر اندازه داده می‌شوند تا ورودی شبکه یکنواخت باشد.
4. تبدیل به تانسور: تبدیل تصاویر از فرمت PIL به torch.Tensor با مقادیر نرمال شده بین [0,1] در نهایت، با استفاده از DataLoader، داده‌ها دسته‌بندی (Batching) شده و با فعال‌سازی قابلیت shuffle، ترتیب آن‌ها برای جلوگیری از بایاس در آموزش، به صورت تصادفی برهم زده می‌شود. خروجی این مرحله (تصویر زیر) صحت بارگذاری و کیفیت اولیه تصاویر را تایید می‌کند

۳-۳. مدیریت هوشمند مسیر داده‌ها (Data Path Handling)

با توجه به اینکه ساختار فایل‌های فشرده دیتاست‌ها ممکن است در محیط‌های مختلف متفاوت باشد، یک الگوریتم جستجوی خودکار برای یافتن مسیر تصاویر پیاده‌سازی شد. تابع find_image_directory با پیمایش بازگشتی پوشش‌ها، اولین مسیری که حاوی تصاویر با فرمت jpg باشد را به عنوان ریشه (Root) انتخاب می‌کند. این رویکرد از بروز خطاهای FileNotFoundError جلوگیری کرده و اجرای کد را در محیط‌های مختلف (Local یا Google Colab) تضمین می‌کند.

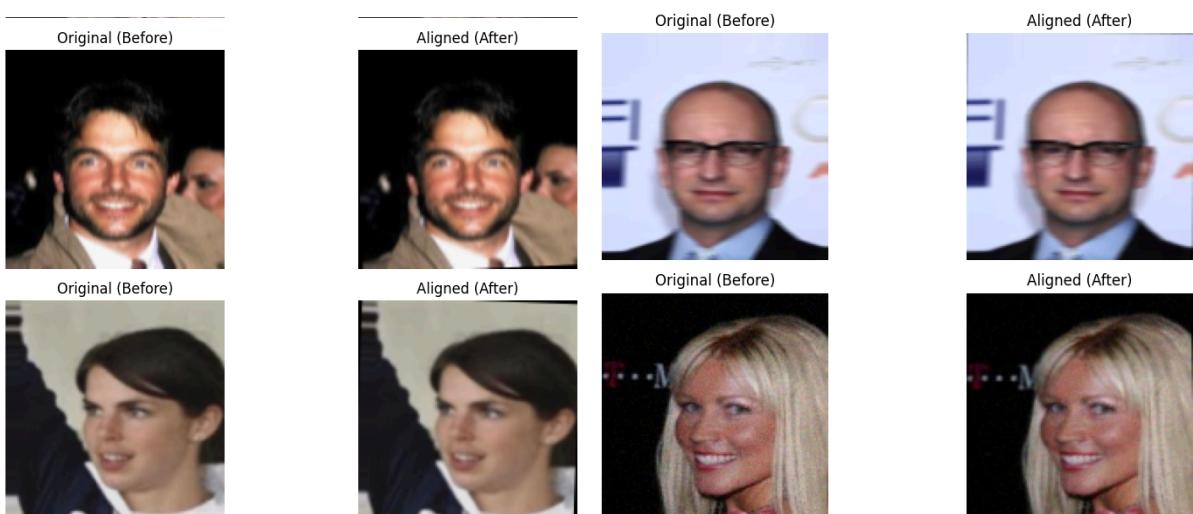
۴-۳. همترازی چهره (Face Alignment)

بر اساس توصیه اکید در بخش ۱ مستندات پروژه، پیش از هرگونه پردازش، عملیات همترازی چهره انجام شد. این فرآیند با هدف یکنواختسازی هندسی ورودی‌های شبکه صورت می‌گیرد تا مدل بتواند تمرکز خود را بر روی بازسازی بافت متمرکز کند نه یادگیری تغییرات مکانی.

برای این منظور از کتابخانه MTCNN و مدل facenet-pytorch استفاده شد. مراحل انجام شده برای هر تصویر عبارتند از:

۱. تشخیص نقاط کلیدی: استخراج مختصات چشم چپ و راست.
۲. محاسبه زاویه انحراف: محاسبه زاویه خط وصل بین دو چشم نسبت به افق با استفاده ازتابع آرکتانزانت ($\theta = \arctan(\frac{\Delta y}{\Delta x})$)
۳. تصیح هندسی: اعمال تبدیل آفین (Affine Transformation) شامل چرخش و جابجایی به گونه‌ای که چشم‌ها در یک راستای افقی قرار گیرند.

همان‌طور که در نتایج بصری مشخص است، این عملیات باعث می‌شود ساختار صورت در تمام نمونه‌ها از یک الگوی ثابت پیروی کند.



۴-۴. پیاده‌سازی کلاس داده و خط لوله تخریب (Degradation Pipeline)

در این مرحله، کلاس اختصاصی FaceRestorationDataset برای مدیریت بارگذاری و پردازش تصاویر CelebA طراحی و پیاده‌سازی شد. هدف اصلی این کلاس، اعمال تغییرات لازم طبق مستندات پروژه (صفحات ۲ و ۳ فایل PDF) جهت تولید داده‌های آموزشی مقاوم است.

۱. داده‌افزایی (Data Augmentation) روی تصاویر هدف:

طبق خواسته مرحله دوم پروژه، برای افزایش تنوع داده‌ها و جلوگیری از بیشبرازش (Overfitting)، روی تصاویر سالم (Target) تغییرات زیر اعمال می‌شود:

- قرینه‌سازی افقی (Random Horizontal Flip): با احتمال ۵۰٪.
- چرخش تصادفی (Random Rotation): چرخش جزئی تا ۱۰ درجه.
- تغییر رنگ و روشنایی (Color Jitter): تغییر در روشنایی و کنترast به میزان ۲۰٪.

۲. توابع تخریب (Degradation Functions)

برای شبیه‌سازی شرایط واقعی و تولید ورودی شبکه، سه نوع تابع تخریب پیاده‌سازی شد است:

- کاهش رزولوشن (Downsampling):

طبق دستورالعمل پروژه (صفحه ۲)، تابع `apply_downsampling` تصویر را با ضرایب ۱.۲ یا ۱.۴ کوچک کرده (Downsample) و سپس مجدداً به ابعاد اصلی بازمی‌گرداند (Upsample). این کار باعث از بین رفتن جزئیات فرکانس بالا و مات شدن تصویر می‌شود.

- نویز گاوسی (Gaussian Noise):

تابع `add_gaussian_noise` نویز سفید را با انحراف معیار مشخص به تصویر اضافه می‌کند تا نویز سنسور دوربین شبیه‌سازی شود.

- تاری حرکتی (Motion Blur):

طبق بند ۲ بخش امتیازی (صفحه ۳ PDF)، تابع `apply_motion_blur` پیاده‌سازی شد. این تابع با ایجاد کرنل‌های خطی در جهات مختلف (افقی، عمودی و قطری) و کانوالو کردن آن با تصویر، اثر لرزش دست یا حرکت سوزه را شبیه‌سازی می‌کند.

۳. ساختار کلاس `__getitem__`:

در هر بار فراخوانی داده، فرآیند زیر طی می‌شود:

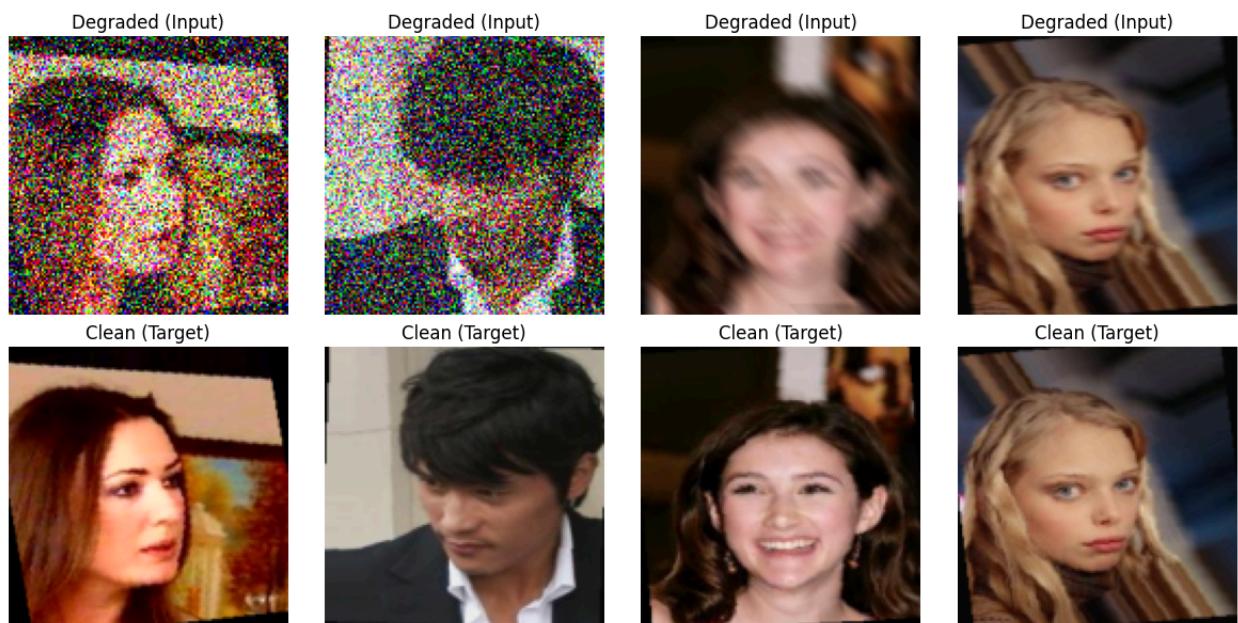
۱. تصویر اصلی بارگذاری شده و داده‌افزایی‌های اولیه روی آن انجام می‌شود (تصویر Target).
۲. به صورت تصادفی (Probabilistic) یکی از سه روش تخریب (نویز، کاهش رزولوشن یا تاری حرکتی) انتخاب شده و روی کپی تصویر اعمال می‌شود (تصویر Input).
۳. خروجی نهایی به صورت یک زوج تنسور (Input, Target) برگردانده می‌شود.

۴. تقسیم‌بندی داده‌ها (Data Split)

برای ارزیابی صحیح مدل، کل دیتاست با نسبت ۹۰٪ آموزش (Train) و ۱۰٪ اعتبارسنجی (Validation) تقسیم شد تا معیارهای ارزیابی روی داده‌هایی که مدل ندیده است محاسبه شوند.

۵. نمایش نمونه‌ها:

در انتهای کد، یک دسته (Batch) از داده‌ها فراخوانی و نمایش داده شد تا صحت عملکرد توابع تخریب و تطابق ابعاد تansورها قبل از شروع آموزش تایید شود.



۴. طراحی معماری مدل و توابع هزینه

در این مرحله، معماری شبکه عصبی و استراتژی آموزش مدل تعیین گردید. طبق خواسته پروژه (بخش ۳)، از ساختار Encoder-Decoder مبتنی بر U-Net استفاده شد که به دلیل داشتن اتصالات پرش (Skip Connections)، در حفظ جزئیات مکانی تصاویر بسیار کارآمد است. همچنین، [تابع هزینه ترکیبی](#) پیاده‌سازی شد.

۱. معماری شبکه (U-Net):

مدل UNet پیاده‌سازی شده شامل بخش‌های زیر است:

- مسیر انکودر (Contracting Path): شامل ۴ بلوک کاهش دهنده (Down) است. هر بلوک از دو لایه کانولوشن 3×3 ، نرمال‌سازی دسته (Batch Norm) و تابع فعال‌ساز ReLU تشکیل شده و سپس با Max Pooling ابعاد تصویر را نصف می‌کند. تعداد کanal‌ها در هر مرحله دو برابر می‌شود (از ۶۴ تا ۵۱۲).
- مسیر دیکودر (Expansive Path): شامل ۴ بلوک افزاینده (Up) است. در هر بلوک، تصویر با استفاده از Upsampling دو برابر شده و با ویژگی‌های منتظر از مسیر انکودر (Skip Connection) ترکیب می‌شود تا اطلاعات از دست رفته مکانی بازیابی شوند.
- لایه خروجی: یک کانولوشن 1×1 برای کاهش کanal‌ها به ۳ (RGB) و اعمال تابع فعال‌ساز Sigmoid.
- استفاده از Sigmoid در لایه آخر حیاتی بود تا خروجی مدل در بازه استاندارد $[0,1]$ محدود شود و با بازه داده‌های ورودی همخوانی داشته باشد.

۲. تابع هزینه (Loss Functions):

برای آموزش مدل، از یک رویکرد ترکیبی استفاده شد تا هم کیفیت پیکسلی و هم هویت چهره حفظ شود:

الف) تابع هزینه بازسازی (Restoration Loss): مطابق بخش ۴ پروژه، از معیار L1 Loss (میانگین قدر مطلق خطای استفاده شد. تجربه نشان می‌دهد L1 نسبت به L2 (MSE تصاویر واضح‌تری تولید می‌کند و کمتر باعث تاری (Blurring) می‌شود.

ب) تابع هزینه تشخیص هویت (Identity Loss): مطابق بخش موارد امتیازی، کلاس IdentityLoss طراحی شد. این کلاس از مدل پیش‌آموزش دیده InceptionResnetV1 (آموزش دیده روی VGGFace2) استفاده می‌کند.

نحوه عملکرد: تصویر بازسازی شده و تصویر اصلی به ابعاد 160×160 تغییر اندازه داده شده و به مدل Inception داده می‌شوند. سپس فاصله بین بردارهای ویژگی (Embeddings) استخراج شده محاسبه می‌شود. این کار شبکه را مجبور می‌کند چهره‌ای بسازد که از نظر ادراکی و هویتی شبیه به فرد اصلی باشد، نه صرفاً از نظر پیکسلی.

۳. ترکیب نهایی:

مدل بر روی GPU بارگذاری شد و بهینه‌ساز Adam با نرخ یادگیری $1e-4$ برای بهروزرسانی وزن‌ها انتخاب گردید. متغیر use_id_loss وضعیت فعلی بودن تابع هزینه امتیازی را کنترل می‌کند.

۵. حلقه آموزش، ارزیابی و ذخیره‌سازی مدل

در این مرحله، فرآیند آموزش مدل (Training Loop) با استفاده از داده‌های آماده‌سازی شده و معماری طراحی شده اجرا می‌شود. هدف نهایی، کمینه‌سازی تابع هزینه ترکیبی و افزایش معیارهای کیفیت تصویر است.

۱. پیکربندی و تقسیم داده‌ها (Data Split):

مطابق اصول استاندارد یادگیری ماشین، دیتاست به دو بخش تقسیم شد:

- مجموعه آموزشی (Train Set): شامل ۹۰٪ داده‌ها برای بهروزرسانی وزن‌های شبکه.
- مجموعه اعتبارسنجی (Validation Set): شامل ۱۰٪ داده‌ها برای نظارت بر عملکرد مدل روی تصاویر دیده نشده و جلوگیری از بیش‌برازش (Overfitting).

۲. تعریف معیارها (Metrics):

برای ارزیابی کمی کیفیت بازسازی، از کتابخانه `torchmetrics` استفاده شد:

- PSNR (Peak Signal-to-Noise Ratio): معیاری برای سنجش کیفیت بازسازی سیگنال نسبت به نویز. مقدار بالاتر نشان‌دهنده شباهت پیکسلی بیشتر به تصویر اصلی است.
- SSIM (Structural Similarity Index): معیاری که شباهت ساختاری (روشنایی، کنترast و ساختار) را می‌سنجد و به درک انسان از کیفیت تصویر نزدیک‌تر است.

۳. پیاده‌سازی تابع هزینه ترکیبی (Bonus Task Integration):

تابع هزینه نهایی به صورت مجموع وزن‌دار دو تابع تعریف شد:

$$\text{LossTotal} = \alpha \times \text{LossPixel} + \beta \times \text{Identity Loss}$$

- ضریب $\alpha=1.0$: وزن خطای پیکسلی (L1) برای حفظ ساختار کلی چهره.
- ضریب $\beta=0.05$: وزن خطای هویت (Identity Loss). این ضریب کمتر در نظر گرفته شد تا از ناپایداری گرادیان‌ها جلوگیری شود، اما همچنان مدل را مجبور می‌کند ویژگی‌های هویتی را حفظ کند.

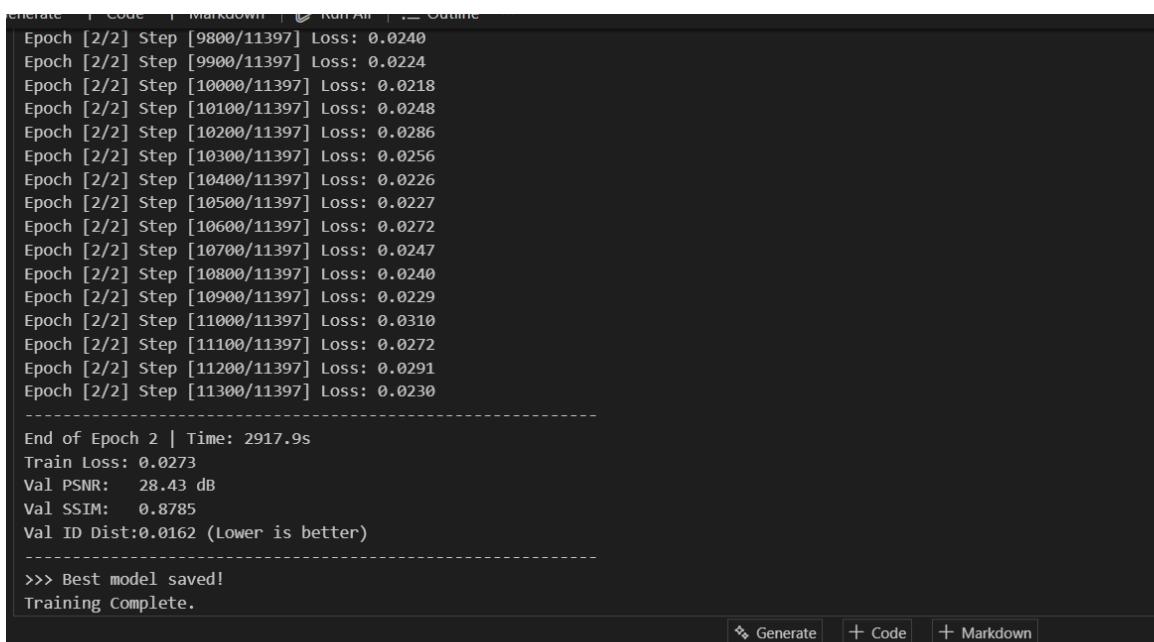
۴. استراتژی بهینه‌سازی (Optimization Strategy):

- بهینه‌ساز: از الگوریتم Adam با نرخ یادگیری اولیه $2e-4$ استفاده شد.

- زمانبند نرخ یادگیری (Scheduler) ReduceLROnPlateau استفاده شد. این ابزار معیار PSNR را در داده‌های اعتبارسنجی پایش می‌کند و اگر برای ۲ دوره (Epoch) متوالی بهبود نیابد، نرخ یادگیری را نصف می‌کند تا مدل بتواند همگرایی دقیق‌تری داشته باشد.

۵. ذخیره‌سازی بهترین مدل:

در انتهای هر Epoch، عملکرد مدل روی داده‌های اعتبارسنجی سنجیده می‌شود. اگر معیار PSNR نسبت به بهترین رکورد قبلی بالاتر باشد، وزن‌های مدل در فایل best_unet_face_restoration.pth ذخیره می‌شوند. این کار تضمین می‌کند که در نهایت، بهترین نسخه مدل (و نه لزوماً آخرین نسخه) برای تست نهایی انتخاب شود.



```

Generate + Code + Markdown Run All ▾ Outline
Epoch [2/2] Step [9800/11397] Loss: 0.0240
Epoch [2/2] Step [9900/11397] Loss: 0.0224
Epoch [2/2] Step [10000/11397] Loss: 0.0218
Epoch [2/2] Step [10100/11397] Loss: 0.0248
Epoch [2/2] Step [10200/11397] Loss: 0.0286
Epoch [2/2] Step [10300/11397] Loss: 0.0256
Epoch [2/2] Step [10400/11397] Loss: 0.0226
Epoch [2/2] Step [10500/11397] Loss: 0.0227
Epoch [2/2] Step [10600/11397] Loss: 0.0272
Epoch [2/2] Step [10700/11397] Loss: 0.0247
Epoch [2/2] Step [10800/11397] Loss: 0.0240
Epoch [2/2] Step [10900/11397] Loss: 0.0229
Epoch [2/2] Step [11000/11397] Loss: 0.0310
Epoch [2/2] Step [11100/11397] Loss: 0.0272
Epoch [2/2] Step [11200/11397] Loss: 0.0291
Epoch [2/2] Step [11300/11397] Loss: 0.0230
-----
End of Epoch 2 | Time: 2917.9s
Train Loss: 0.0273
Val PSNR: 28.43 dB
Val SSIM: 0.8785
Val ID Dist: 0.0162 (Lower is better)
-----
>>> Best model saved!
Training Complete.

```

The screenshot shows a Jupyter Notebook cell with the following content:
 - Header: Generate + Code + Markdown Run All ▾ Outline
 - Log output:
 Epoch [2/2] Step [9800/11397] Loss: 0.0240
 Epoch [2/2] Step [9900/11397] Loss: 0.0224
 Epoch [2/2] Step [10000/11397] Loss: 0.0218
 Epoch [2/2] Step [10100/11397] Loss: 0.0248
 Epoch [2/2] Step [10200/11397] Loss: 0.0286
 Epoch [2/2] Step [10300/11397] Loss: 0.0256
 Epoch [2/2] Step [10400/11397] Loss: 0.0226
 Epoch [2/2] Step [10500/11397] Loss: 0.0227
 Epoch [2/2] Step [10600/11397] Loss: 0.0272
 Epoch [2/2] Step [10700/11397] Loss: 0.0247
 Epoch [2/2] Step [10800/11397] Loss: 0.0240
 Epoch [2/2] Step [10900/11397] Loss: 0.0229
 Epoch [2/2] Step [11000/11397] Loss: 0.0310
 Epoch [2/2] Step [11100/11397] Loss: 0.0272
 Epoch [2/2] Step [11200/11397] Loss: 0.0291
 Epoch [2/2] Step [11300/11397] Loss: 0.0230
 - Summary:
 End of Epoch 2 | Time: 2917.9s
 Train Loss: 0.0273
 Val PSNR: 28.43 dB
 Val SSIM: 0.8785
 Val ID Dist: 0.0162 (Lower is better)
 - Message:
 >>> Best model saved!
 - Status:
 Training Complete.
 - Footer buttons: Generate, Code, Markdown

۶. ارزیابی کیفی و مصورسازی (Visualization)

در این مرحله، عملکرد مدل بر روی دسته‌ای از داده‌های اعتبارسنجی (که در حین آموزش دیده نشده‌اند) به صورت بصری ارزیابی شد. هدف از این کار، بررسی قدرت مدل در حذف نویز، افزایش رزولوشن و رفع تاری ضمن حفظ هویت چهره است.

روند اجرا:

۱. حالت ارزیابی (Evaluation Mode) مدل با دستور model.eval() به حالت تست رفت تا رفتارهایی مانند Dropout غیرفعال شوند.

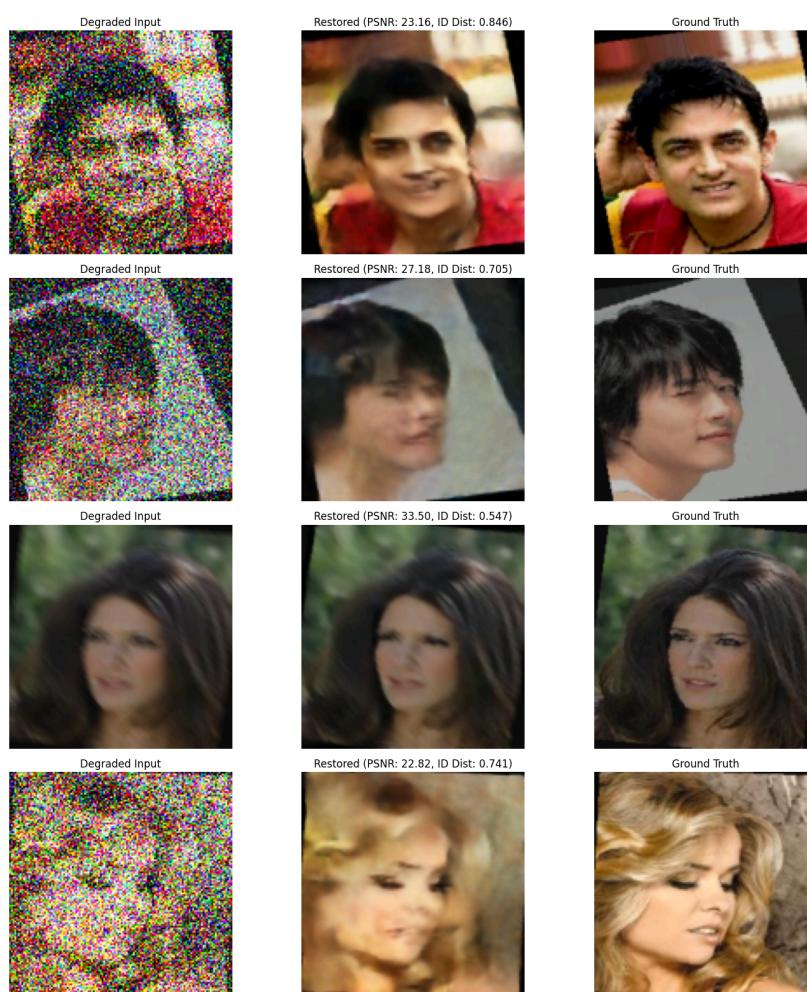
۲. استنتاج (Inference): یک دسته (Batch) از تصاویر تخریب شده وارد شبکه شده و خروجی بازسازی شده تولید گردید.

۳. محاسبه متريک های موردي: برای هر تصویر به صورت جداگانه:

- PSNR: جهت سنجش بهبود کيفيت سيگنال تصویر.
- Identity Distance: با استفاده از شبکه InceptionResnetV1، فاصله اقلیدسي بين ويزگي هاي چهره تصویر بازسازی شده و تصویر اصلی محاسبه شد. (عدد کمتر = شباهت هویتی بیشتر).

۴. نمایش خروجی: تصاویر در سه ستون کنار هم نمایش داده شدند:

- ستون اول: تصویر ورودی تخریب شده (Degraded).
- ستون دوم: تصویر بازسازی شده توسط مدل (Restored) به همراه امتیازات کسب شده.
- ستون سوم: تصویر اصلی و باکیفیت (Ground Truth).

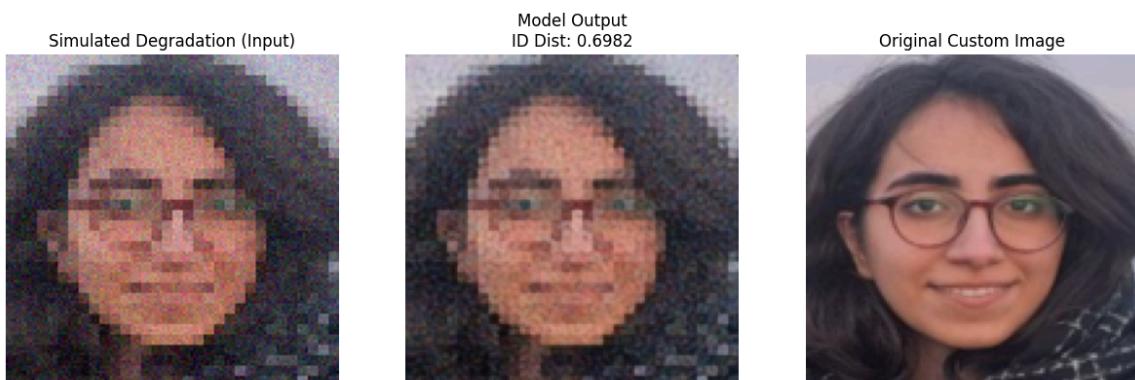


۷. تست تعمیم‌پذیری (Generalization Test)

طبق بخش ۵ مستندات پروژه، مدل علاوه بر دیتاست CelebA باید روی "یک دیتاست چهره دلخواه" یا تصاویر واقعی کاربران نیز ارزیابی شود تا قدرت تعمیم‌پذیری (Generalization) آن سنجیده شود.

شرح فرآیند:

۱. آپلود تصویر: کاربر یک تصویر چهره دلخواه (مثلًا تصویر خودش یا یک چهره اینترنتی) را آپلود می‌کند.
۲. پیش‌پردازش و تخریب مصنوعی: از آنجا که تصویر آپلود شده معمولاً باکیفیت است، برای تست مدل باید ابتدا آن را تخریب کنیم. تصویر ابتدا به ابعاد 128×128 تغییر سایز داده شده، سپس رزولوشن آن به شدت کاهش می‌یابد (Downsampling) و نویز به آن اضافه می‌شود تا شبیه به ورودی‌های مدل شود.
۳. بازسازی (Restoration): تصویر تخریب شده به مدل داده می‌شود تا نسخه باکیفیت آن تولید شود.
۴. مقایسه هویت: فاصله هویت بین تصویر اصلی کاربر و تصویر بازسازی شده توسط مدل محاسبه می‌شود. این معیار نشان می‌دهد که آیا مدل توانسته است هویت فرد را در فرآیند بازسازی حفظ کند یا خیر.



Please upload a face image (jpg/png) to test generalization:
 photo_2026...0-22-02.jpg
photo_2026-02-09_10-22-02.jpg(image/jpeg) - 24342 bytes, last modified: 2/9/2026 - 100% done
Saving photo_2026-02-09_10-22-02.jpg to photo_2026-02-09_10-22-02.jpg

Noisy Input (Generalization)	Restored by Model	Original Uploaded