Controlando a Geração de Faces através de *Generative Adversarial Networks*

Felipe Fischer da Silva, Carine Geltrudes Webber

***Abstract*—Generative Advesarial Networks (GANs) are a relatively recent machine learning approach. As a framework, it has been used in several computer vision successful projects. Studies about face synthesis commonly use GANs in order to produce new images of human faces. The possibilities range from changing small face characteristics to recreating a completely new one. This particular study aims to do an exploratory research, introducing the topic and investigating possible applications for face synthesis. To begin with, a systematic review has been done. Next, it was proposed the use of a GAN to generate random faces using controllable attributes from CelebA dataset. In order to evaluate such scenario, FID and IS metrics were emploied. A method based on deep learning principles was followed along this research. Relevant methods as noisy scale-space and one-sided label smoothing were applied in order to guarantee GAN convergence and better results. To conclude, the best model has performed similarly to models described in literature, reaching a FID value of 23.08 (±1.35).**

***Index Terms*—Generative Adversarial Networks, Face Generation, CelebA, cGAN, DCGAN, cDCGAN.**

I. Introdução

O Aprendizado de Máquina (AM) (ou *machine learning*) refere-se a algoritmos que aprendem a partir da experiência a tratar de uma classe de tarefas. Seu desempenho nessas tarefas é aperfeiçoado com a experiência [1]. Pode-se subdividir os algoritmos de AM em: aprendizado supervisionado, e não supervisionado. No aprendizado supervisionado é onde se encaixam os problemas de classificação (envolvendo classes conhecidas e categóricas) e os problemas de regressão (envolvendo predição de valores numéricos) [2]. Com o aprendizado não-supervisionado, pretende-se através dos dados de entrada, codificar as características desses dados para que se possam gerar novas classes [3], seja para agrupar os dados (algoritmos de *clustering*) ou para gerar novas instâncias de dados (modelos generativos).

Dentro do aprendizado não-supervisionado, os modelos generativos têm como principal exemplo as Redes Generativas Adversárias *Generative Adversarial Networks* (GAN), que têm se destacado desde que apresentadas em 2014 por Goodfellow *et al.* [4]. Desde então, percebe-se grandes progressos realizados em relação à sua teoria e aplicação, além de diversas variações dessas redes terem sido introduzidas [5].

As GANs são definidas como um *framework* para treinar um modelo generativo, composto por duas redes neurais denominadas Gerador e Discriminador. O Discriminador é responsável por classificar os dados que recebe entre reais e falsos, enquanto o Gerador é responsável por tentar produzir novas instâncias dentro do mesmo domínio [4].

Uma das principais razões que tornaram as GANs amplamente estudadas, desenvolvidas e utilizadas é o seu sucesso em problemas de visão computacional [6]. As implementações de GAN comumente utilizam métodos de *deep learning* [6]. Conforme seus criadores [7], a maioria das aplicações é baseada na arquitetura *Deep Convolutional GAN* (DCGAN) [8], onde são utilizadas as redes neurais convolucionais ou *Convolutional Neural Networks* (CNN). A DCGAN é considerada um ponto de partida para o desenvolvimento de novas GANs voltadas para tarefas de síntese de imagens [6].

Em 2018, pesquisadores da Nvidia introduziram a StyleGAN, capaz de ser treinada com fotos de rostos de pessoas reais e produzir rostos totalmente novos, muito próximos de pessoas reais [9]. Outro trabalho de destaque foi realizado por um grupo de pesquisadores da Samsung, que demonstrou em 2019 um sistema baseado em GAN, que produz um vídeo de uma pessoa falando, apenas usando uma foto [10].

Analisando os estudos e aplicações recentes, percebe-se que é uma área complexa, os avanços notáveis ainda são recentes, entretanto, estão ganhando relevância na área da Computação e da Inteligência Artificial. Dentro desse contexto, o presente trabalho realiza uma pesquisa de natureza exploratória, a fim de apresentar os conceitos da área através de um referencial teórico, uma revisão sistemática para identificar métricas, *datasets*, e aspectos em comum a trabalhos da área. A partir da pesquisa realizada, trazer uma aplicação dessas redes para a síntese de faces condicionada por atributos, capaz de produzir imagens similares as de um conjunto de dados conhecido, neste caso, o *dataset* CelebA. Com isso, detalhar o processo de treinamento e testes, mostrando como elas se comportam em relação a diferentes parâmetros, técnicas para estabilizar o treinamento e métricas de avaliação.

II. Referencial Teórico

A. Generative Adversarial Networks

A GAN é descrita como um *framework* para estimar modelos generativos por meio de um processo adversarial [4]. Nesse processo, os modelos Gerador G e Discriminador D competem entre si, onde D faz uma estimativa da probabilidade de uma instância pertencer ou não aos dados de treino. Enquanto isso G tenta produzir novas instâncias que estejam dentro da distribuição desses dados.

Considera-se que D atua como um classificador binário e G é uma rede que, ao receber como entrada um ruído de dados aleatório e produz novas instâncias de dados em sua

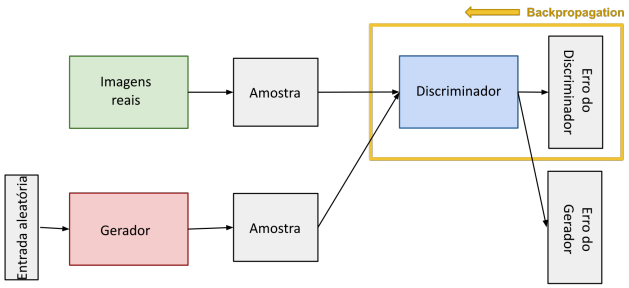


Fig. 1: Treinamento do discriminador. [11]

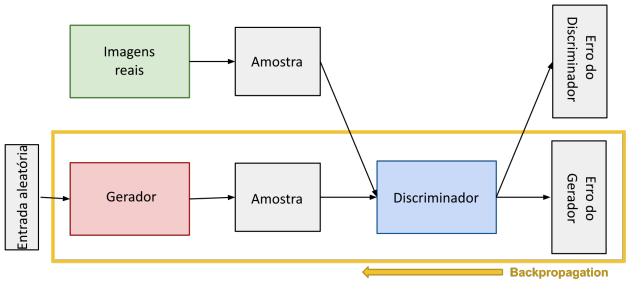


Fig. 2: Treinamento do gerador. [11]

saída. O treinamento de ambos é feito dentro de um mesmo laço, executado por n iterações em cada época, onde n é o número de iterações, o valor resultante da divisão do total de instâncias no *dataset* pelo tamanho do *batch* definido. O tamanho do *batch* define quantas instâncias serão utilizadas em cada iteração.

No início de cada iteração, considera-se um *batch* de imagens reais, e uma mesma quantidade de instâncias, que é produzida por G a partir do ruído de dados. O treinamento do Discriminador (ilustrado na Figura 1) é feito através dos seguintes passos:

1) D recebe e classifica ambos os conjuntos entre reais e falsos.
2) A partir dessas classificações, calcula-se a função de Erro do Discriminador.
3) Propaga-se o erro calculado através de D pelo algoritmo *backpropagation*, atualizando os pesos de suas conexões.

Ainda na mesma iteração, G produz novamente a mesma quantidade de instâncias, para realizar o treinamento do Gerador (ilustrado na Figura 2), que é feito pelos passos a seguir:

1) D recebe e classifica as instâncias geradas por G.
2) A partir da classificação, calcula-se o Erro do Gerador.
3) Aplica-se então o algoritmo *backpropagation* em D e G, para que se obtenham os gradientes.
4) Usa-se os gradientes para alterar apenas os pesos de G.

Dessa forma, o treinamento de D é feito visando maximizar a probabilidade de atribuir corretamente o rótulo, real ou falso, tanto para os dados de treinamento, quanto os dados gerados por G. Simultaneamente, G é treinado para minimizar o valor da expressão $log(1 - D(G(z)))$ produzindo exemplos que recebam classificação "real" por D. Para que G aprenda a distribuição p_g sobre os dados x, é definido um ruído como

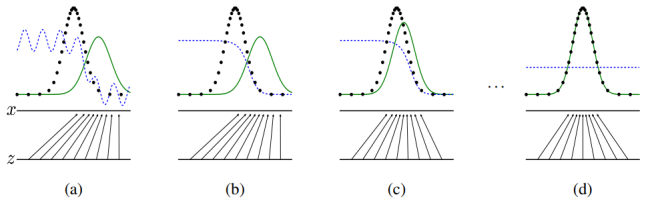


Fig. 3: Treinamento da GAN. [4]

variável de entrada $p_z(z)$, então o mapeamento para o estado de dados é representado pela função $G(z;\theta_g)$ e a função $D(x;\theta_d)$ retorna um valor escalar. Este valor representa a probabilidade de x ter vindo de outra distribuição de dados que não p_g. Considera-se p_data a distribuição de dados reais. Este processo é formalmente descrito como um jogo *minimax* (Equação 1) de 2 jogadores sendo eles D e G, e é representado através da expressão:

$$\min_G \max_D V(D,G) = E_{x\ p_{data}(x)}[logD(x)] + E_{z\ p_z(z)}[log(1 - D(G(z)))] \quad (1)$$

A Figura 3 exemplifica como as distribuições se organizam durante o treinamento de ambos os modelos. A linha tracejada azul representa a distribuição do discriminador D, a linha verde contínua representa p_g, que é a distribuição dos dados gerados por G e a linha preta pontilhada representa a distribuição real dos dados (p_{data}). Na Figura 3.a é representada uma situação onde os modelos estão próximos de uma convergência. D possui certa precisão em suas classificações e p_g já se encontra em um estado similar a p_{data}. Em Figura 3.b D é treinado por seu algoritmo e converge para $D^*(x) = \frac{p_{data(x)}}{p_{data(x)}+p_{g(x)}}$, nesse ponto a compreensão de D da distribuição é bem próxima da real, mas ainda com interferência de G. Em Figura 3.c G é otimizado e $G(z)$, guiado pelo gradiente de D, já passa a se direcionar para regiões muito próximas dos dados reais. Em Figura 3.d após vários passos do treinamento, ambos D e G encontram um ponto onde não são mais capazes de aperfeiçoamento. Os dados gerados por G já estão totalmente dentro da distribuição, ou seja, $p_g = p_{data}$, e com isso o discriminador não as consegue mais diferenciar, sendo assim $D(x) = \frac{1}{2}$.

B. GANs Convolucionais Profundas

Em sua definição inicial, as redes que compõem a GAN (D e G) são implementadas como *multilayer perceptron*. Para melhor trabalhar dados de imagens, foi introduzida a *Deep Convolutional* GAN (GAN Convolucional Profunda). Sendo esta, implementada via redes neurais convolucionais CNNs, assim gerando (G) novas imagens e classificando (D) as imagens como reais/falsas ou pertencentes/não pertencentes a um domínio. Dessa forma esse tipo de rede é um ponto de partida para o desenvolvimento de novas GANs para tarefas de síntese de imagens.

Nesse contexto, a rede geradora (representada na Figura 4) recebe uma distribuição uniforme z de tamanho 100 como entrada. A conversão dessa entrada para uma representação

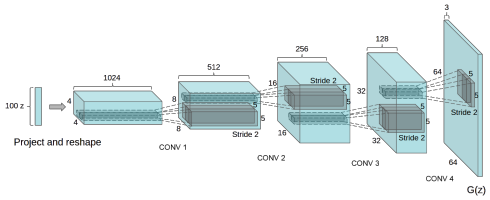


Fig. 4: Gerador de uma DCGAN [8]

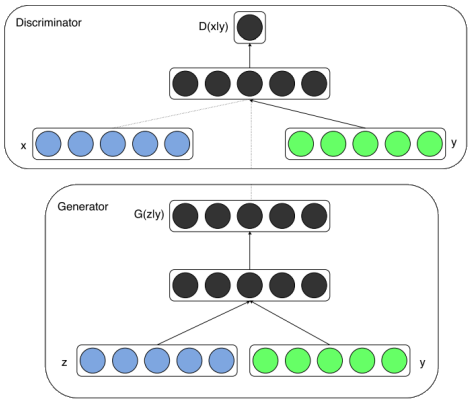


Fig. 5: Estrutura simplificada de uma cGAN [14]

em imagem é feita através de uma pequena representação convolucional, de extensão espacial com muitos mapas de características[1] e uma série de convoluções transpostas[2].

C. GANs Condicionais

A GAN como formulada originalmente é capaz de gerar exemplos plausíveis dentro de uma dada distribuição de um conjunto de dados, porém, não possui formas de controlar como será a instância gerada. Apenas busca que essa instância seja coerente à distribuição para a qual foi treinada. A *Conditional GAN* (cGAN) é uma extensão das GANs para um modelo condicional. Nessa abordagem tanto Gerador quanto Discriminador são condicionados com alguma informação y através de uma camada de entrada adicional (Figura 5). Essa informação pode ser qualquer dado auxiliar, como rótulos ou atributos [14]. Dessa forma a geração de imagens, por exemplo, pode ser controlada para que produza resultados com determinadas características desejadas, ou com um rótulo [6]. Sendo assim, o modelo possibilita que através do treinamento condicionado por y tenha-se melhor controle de como as gerações serão compostas.

III. Trabalhos Relacionados

Nesta seção, através de uma revisão sistemática, analisaram-se trabalhos relacionados. Por meio do conhecimento adquirido com o estudo das diferentes aplicações, definiu-se o escopo da proposta do presente artigo.

[1]Mapeamentos presentes nas camadas convolucionais das CNNs responsáveis por associar áreas da imagem a uma característica ou atributo [12].

[2]Convoluções realizadas para realizar as transformações na direção oposta, invertendo entradas e saídas. [13]

A. Revisão Sistemática

Esta seção é referente à pesquisa realizada a fim de encontrar trabalhos relacionados à síntese de faces utilizando GANs. As subseções seguintes explicam como a pesquisa foi realizada, e relatam as informações encontradas. Essas informações ajudam a compreender alguns aspectos que compõem a concepção de um modelo GAN para essa aplicação. O conteúdo analisado nos trabalhos serviu de guia para que se pesquisassem detalhes sobre métricas e *datasets* abordados. Essa pesquisa teve como objetivo obter uma melhor compreensão sobre tais aspectos, que são comuns a trabalhos da área, bem como direcionar o desenvolvimento do trabalho.

Para mapear o estado da arte das GANs foram realizadas pesquisas por meio da biblioteca digital *IEEE Xplore*[3]. Ela provê acesso a uma grande quantidade de artigos nas áreas de, Engenharia Elétrica, Ciência da Computação e Eletrônica. Para as pesquisas foram utilizadas as palavras chave *"generative adversarial network face generation"*, com o intuito de encontrar os trabalhos relacionados à sintetização/geração de imagens faciais, uma das aplicações mais populares das GANs. A busca limitou-se a publicações no intervalo entre os anos de 2018 e 2021, e neste processo, os filtros foram sendo redefinidos com base em palavras chave presentes nos resultados da pesquisa inicial. Por fim, os filtros aplicados foram os seguintes:

- face recognition
- neural nets
- learning (artificial intelligence)
- feature extraction
- image coding
- computer vision
- image reconstruction
- image resolution
- convolutional neural nets
- realistic images

Obteve-se 33 resultados, que após analisados por seus títulos, 9 foram desconsiderados, por sugerirem outras aplicações das GANs. Nos 24 trabalhos restantes, fez-se uma leitura no *abstract* com a intenção de identificar aqueles que melhor representassem os estudos envolvendo sintetização, alteração ou geração de faces. Após esta análise, 14 artigos foram selecionados para realizar a extração de informações. Para fins de comparação dos dados, um dos artigos foi desconsiderado, por se tratar de uma revisão geral da área e não uma nova proposta de aplicação.

Os artigos foram analisados afim de identificar características como: *datasets* utilizados, tema/aplicação do trabalho e métricas de avaliação utilizadas nos modelos propostos.

B. Aplicações e objetivos

Primeiramente procurou-se identificar as aplicações/temas de cada um dos artigos estudados. A Tabela I relaciona os diferentes trabalhos analisados na revisão sistemática e suas respectivas aplicações. As aplicações identificadas

[3]https://ieeexplore.ieee.org/

demonstram que ainda no processamento de imagens de faces, existem variados nichos procurando trabalhar ou otimizar algum aspecto na geração dessas imagens, como por exemplo: síntese de faces, edição de atributos faciais, aumento de dados e frontalização de faces. A partir dessas informações, observou-se as aplicações mais recorrentes afim de identificar um nicho a se explorar. A Síntese de faces é presente na maioria dos artigos, e as demais aplicações mais recorrentes na pesquisa fazem parte deste amplo tema. Conforme a Tabela II, a edição de atributos faciais tem maior representatividade dentro dos resultados encontrados, sendo assim uma boa aplicação a ser explorada no trabalho.

C. Datasets utilizados em aplicações de síntese facial

Com a pesquisa realizada buscou-se atentar também aos *datasets* utilizados, com o intuito de descobrir quais os comumente utilizados nessas aplicações e escolher um deles para utilizar no desenvolvimento do presente trabalho. A Tabela III lista a ocorrência dos diferentes *datasets* identificados nessa pesquisa.

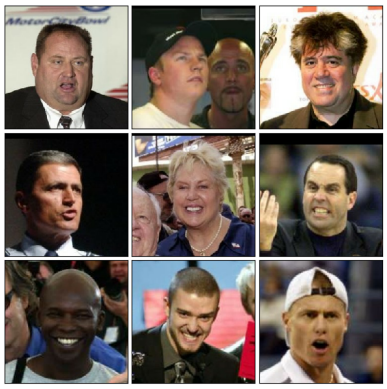
Constatou-se que os *datasets* CelebA [28] e LFW [29] são os mais utilizados. O CelebA consiste em um conjunto de mais de 200 mil imagens de celebridades, cada imagem possui 40 atributos de valor binário, 5 atributos referentes a coordenadas de algumas características específicas da face como olhos, nariz, boca [28]. A Figura 6a contém exemplos do CelebA relacionando com alguns atributos. O LFW conta com mais de 13 mil imagens de rostos de pessoas em ambientes variados, cada imagem possui um rótulo para identificar a pessoa [29]. Algumas instâncias do *Labeled Faces in the Wild* (LFW) são demonstradas na Figura 6b. Nota-se que ambos consistem em amplos conjuntos de imagens de rostos, destinados ao estudo e treinamento de modelos para reconhecimento facial. Sua riqueza de dados e a frequente ocorrência na pesquisa realizada, aponta que ambos são bons candidatos para o treinamento de modelos generativos.

D. Métricas de avaliação de modelos

Atualmente não existe um consenso sobre quais são as métricas mais adequadas para avaliar quantitativamente o desempenho das GANs [5]. São variadas as métricas existentes com objetivo de avaliar diferentes cenários. A Tabela IV associa o número de ocorrências com as métricas identificadas na literatura, sendo elas: *Frechét Inception Distance* (FID), *Structural Similarity* (SSIM), *Inception Score* (IS), *Peak signal-to-noise ratio* (PSNR). Elas são comumente utilizadas quando o objetivo é avaliar a qualidade das imagens geradas por modelos voltados à síntese de imagem facial. A Tabela IV relaciona as diferentes métricas utilizadas pelos vários autores e suas ocorrências. Observa-se que a métrica mais utilizada nos trabalhos selecionados foi a FID, que foi proposta como uma métrica mais consistente que o IS, que também aparece entre as mais utilizadas.

O IS [30] quantifica a qualidade de imagens geradas baseando-se na classificação de um modelo pré-treinado *Inception v3* [31]. O modelo captura a distribuição dos rótulos, então obtém-se a classificação de cada imagem e a variedade

(a) Exemplos do CelebA



(b) Exemplos do *LFW*

Fig. 6: *Datasets* com maior ocorrência na pesquisa.

das imagens geradas dentro dessa distribuição. O valor do IS é calculado a partir desses dois elementos, através da entropia relativa, ou *Kullback–Leibler divergence (KL divergence)*. Esta métrica consegue quantificar a correlação entre a qualidade e a diversidade das imagens geradas de forma razoável [30], e quanto maior o valor calculado do IS maior a qualidade das imagens avaliadas.

A FID é uma métrica proposta para avaliar as imagens de forma mais consistente, se comparada com o *Inception Score*. Seu objetivo é capturar a similaridade das imagens geradas em relação às imagens reais. A métrica é calculada usando o modelo pré-treinado *Inception v3*, assim como no IS. A camada de codificação do modelo (última camada de *pooling* antes da classificação da saída) é utilizada para capturar características específicas de visão computacional de uma imagem de entrada. As ativações desta camada são calculadas pra um conjunto de imagens geradas e imagens reais. Para o cálculo da métrica, a média e covariância das imagens são calculadas a partir das ativações, sumarizando-as em uma distribuição Gaussiana multivariada [32]. A *Frechét Inception Distance* é calculada através da diferença entre as distribuições (imagens reais e imagens geradas), usando a *Frechét Distance* ou *Wasserstein-2 distance* [6]. Os autores da métrica [33] demonstram como ela captura bem o nível de "perturbação" das imagens conforme elas se distanciam e diferem das originais, demonstrando que quanto menor a FID, melhor a qualidade das imagens produzidas. A Figura 7 ilustra como a métrica se comporta em relação à distorção na imagem. Nela é demonstrado como cada tipo de distorção

TABELA I: Aplicações dos trabalhos pesquisados.

Trabalho	Tema/Aplicação
Feature-Improving Generative Adversarial Network for Face Frontalization [15]	Frontalização de faces, Reconhecimento de faces pose-invariante, Reconhecimento de faces
A Realistic Image Generation of Face From Text Description Using the Fully Trained Generative Adversarial Networks [16]	Texto para imagem, Síntese de faces, Aumento de dados
RAG: Facial Attribute Editing by Learning Residual Attributes [17]	Edição de atributos faciais
LAUN Improved StarGAN for Facial Emotion Recognition [18]	Aumento de dados, Reconhecimento de expressão facial
FEGAN: Flexible and Efficient Face Editing With Pre-Trained Generator [19]	Edição de atributos faciais
Component Semantic Prior Guided Generative Adversarial Network for Face Super-Resolution [20]	Super resolução de face, Síntese de faces
Generating Users' Desired Face Image Using the Conditional Generative Adversarial Network and Relevance Feedback [21]	Síntese de faces
Adversarially Regularized U-Net-based GANs for Facial Attribute Modification and Generation [22]	Edição de atributos faciais, Síntese de faces
Geometry-Aware GAN for Face Attribute Transfer [23]	Edição de atributos faciais, Transferência de atributo facial
Unpaired Domain Transfer for Data Augment in Face Recognition [24]	Aumento de dados, Reconhecimento de faces
From Eyes to Face Synthesis: a New Approach for Human-Centered Smart Surveillance [25]	Síntese de faces, Reconhecimento de faces, Preenchimento de faces (face inpainting)
CP-GAN: A Cross-Pose Profile Face Frontalization Boosting Pose-Invariant Face Recognition [26]	Frontalização de faces, Reconhecimento de faces pose-invariante, Reconhecimento de faces, Síntese de faces
Multi-Pose Facial Expression Recognition Based on Generative Adversarial Network [27]	Frontalização de faces, Reconhecimento de expressão facial pose-invariante, Reconhecimento de expressão facial

TABELA II: Ocorrências de cada aplicação nos trabalhos pesquisados.

Tema/Aplicação	Ocorrências
Síntese de faces	6
Edição de atributos faciais	4
Reconhecimento de faces	4
Aumento de dados	3
Frontalização de faces	3

TABELA III: Ocorrências de cada aplicação nos trabalhos pesquisados.

Dataset	Ocorrências
CelebFaces Attributes Dataset (CelebA)	6
LFW (*Labeled Faces in the Wild*)	6
CelebA-HQ	2
Celebrities in Frontal-Profile (CFP)	2
CMU Multi-PIE Face Database (Multi-PIE)	2
CASIA 3D Face	2

TABELA IV: Número de ocorrências das métricas

Métrica	Ocorrências
FID	5
SSIM	4
IS	3
PSNR	3

aplicado as distanciam das imagens originais conforme o nível da distorção aumenta, fazendo assim a FID subir, indicando que a qualidade das imagens está diminuindo.

A SSIM [34] é uma métrica que visa medir a similaridade entre duas imagens, buscando descartar aspectos da imagem irrelevantes à percepção humana. A métrica é calculada através da comparação de três características principais das imagens: luminância, contraste e estrutura.

A PSNR (relação sinal-ruído de pico) avalia a qualidade

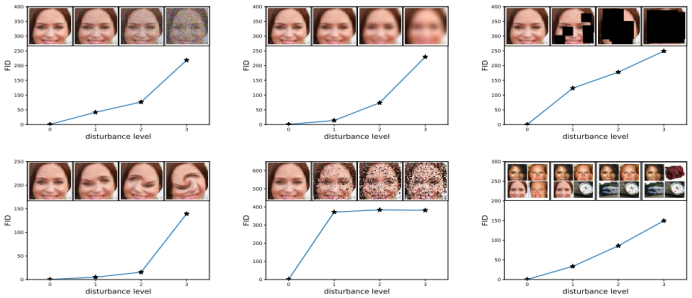


Fig. 7: Variação da métrica FID conforme as distorções das imagens

de uma imagem gerada em relação à sua imagem real correspondente. É medida em decibéis (dB), quanto maior o PSNR, maior a qualidade da imagem gerada [5].

Em questões de relevância, identifica-se que a FID deve ser utilizada em projetos que envolvam uso das GANs e a IS normalmente é utilizada em conjunto na avaliação. Essas duas métricas avaliam as imagens em relação ao conjunto de imagens reais, o que é adequado para um contexto onde se quer gerar novas instâncias, enquanto que as demais (SSIM e PSNR) dependem da comparação direta entre uma imagem gerada e imagem real correspondente.

E. Considerações sobre a pesquisa

As variações da GAN propostas pelos autores dos trabalhos envolvidos nesta pesquisa apresentam formalizações e conceitos diferentes entre si por tratarem de propósitos distintos. Exemplifica-se a seguir a complexidade do trabalhos descritos na literatura por meio de uma amostra dos modelos encontrados.

Para iniciar, a *Feature Improving GAN* (FI-GAN) [15] propõe uma solução para gerar faces frontalizadas a partir de

uma foto de perfil em ângulos variados, buscando encontrar um mapeamento entre uma imagem de face frontal e uma face de perfil, utilizando outros elementos como a extração dos atributos das imagens, e posteriormente avaliar as imagens usando dois discriminadores, um para as imagens, e outro para os atributos que foram extraídos. A *Residual Attributes GAN* (RAG) [17] é voltada para a alteração de atributos nas imagens existentes, mudando características faciais. Para isso foram utilizados conceitos de outra arquitetura: a CycleGAN e a *cycle-consistency loss* [35], função de perda com objetivo de tornar os mapeamentos entre dois domínios (X e Y) consistente em ambas as direções ($X \rightarrow Y$ e $Y \rightarrow X$), de forma que se pudesse obter a mesma imagem ao submetê-la aos dois mapeamentos consecutivamente. Foi introduzido também o conceito de atributos residuais: a diferença entre os atributos desejados e os atributos da imagem a ser alterada. A FEGAN [19] é voltada a realizar a edição de atributos de uma imagem alterando um vetor de atributos no espaço latente. Para isso os autores focam em treinar uma CNN para extrair o código latente das imagens como um vetor a ser utilizado pelo Gerador (pré-treinado). Utilizando uma segunda rede calcula-se a diferença entre a imagem real e a imagem produzida, utilizando o resultado como *feedback* para a otimizar a CNN de extração, para que assim ela passe a mapear corretamente os atributos das imagens para o vetor de atributos, possibilitando que ao alterar atributos no vetor, as imagens serão geradas de acordo com essas alterações.

Considerando os trabalhos de pesquisa analisados, verificou-se que a tarefa de alteração de atributos faciais tem maior ocorrência, mas possui grande complexidade e escopo. Para fins deste trabalho optou-se por direcionar os objetivos para uma aplicação similar a alteração de atributos faciais: controlar a geração das imagens a partir dos atributos desejados. Esta tarefa aplica os conceitos da DCGAN e da cGAN, pois envolve a geração e avaliação de dados de imagens, onde as gerações serão condicionadas por uma informação auxiliar. Neste caso, os atributos que se deseja alterar na face em questão. Para tal aplicação, é necessário um *dataset* que forneça tais atributos pelos quais a geração será controlada, dentro dos mais recorrentes na pesquisa, o CelebA mostra-se ideal nesse quesito. Para as imagens geradas pelo modelo, é necessário avaliá-las através de uma métrica apropriada que represente a qualidade da geração. Conforme mencionado anteriormente, não existe um total consenso na área sobre as melhores métricas para tal, então o presente trabalho focará em representar os resultados apenas utilizando métricas dentro das mais recorrentes na pesquisa.

A revisão sistemática realizada nesta seção reuniu conhecimento sobre aspectos comuns na aplicação de GANs, cumprindo o caráter exploratório do presente trabalho. Nas seções seguintes, as ferramentas utilizadas serão apresentadas, a aplicação proposta é desenvolvida e explicada, e por fim, tem seus resultados apresentados.

IV. Materiais e Métodos

Nesta seção são descritos os materiais e métodos utilizados na realização do presente trabalho.

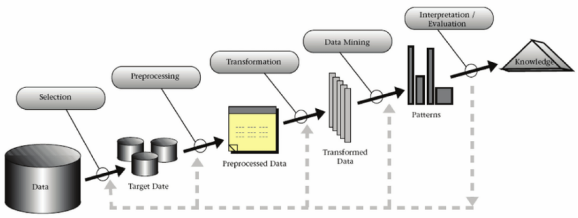


Fig. 8: Processo KDD

A. Descrição do Estudo de Caso

A GAN é um avanço relativamente recente, tendo atualmente menos de uma década de existência. O presente artigo tem o objetivo de trazer uma aplicação dessas redes para a síntese de faces condicionada por atributos, detalhar o processo de treinamento e testes, mostrando como elas se comportam em relação a diferentes parâmetros, técnicas para estabilizar o treinamento e métricas de avaliação. A aplicação será uma DCGAN que aplica também o conceito condicional da cGAN. O objetivo é treinar as redes para gerar novas imagens de faces usando o *dataset* escolhido, controlando atributos específicos na geração dessa imagem. Com isso pretende-se gerar variações dos rostos, apesar de sua aleatoriedade, através da alteração dos atributos.

B. Descrição do Método

O presente trabalho constitui uma pesquisa de natureza qualitativa e exploratória, a qual visa investigar, compreender e aplicar as GANs no contexto da síntese de imagens. O trabalho seguirá um processo comumente utilizado para guiar a aplicação de algoritmos de *machine learning*, chamado *Knowledge Discovery and Data mining* (KDD) [36]. Neste processo tem-se uma série de etapas que, a partir dos dados, permitem a obtenção de conhecimento como também a validação de métodos de IA para tarefas de aprendizado de máquina. O processo KDD é ilustrado na Figura 8, que começa a partir de uma seleção dos dados. Nesse caso o *dataset*, seguido de um pré-processamento para retirar possíveis ruídos e dados incompletos, passa posteriormente pela transformação dos dados, ou seja, os adequa para serem usados nos algoritmos de aprendizado de máquina. O processo então segue com a etapa denominada *Data-Mining*, onde no caso deste trabalho será aplicado o algoritmo de aprendizado e finalizando com a etapa de avaliação dos resultados para se chegar nas conclusões.

Partindo-se deste processo, este trabalho foi desenvolvido em cinco etapas:

1) Etapa 1: identificação da base de dados (*dataset*).
2) Etapa 2: pré-processamento dos dados.
3) Etapa 3: transformação dos dados.
4) Etapa 4: aplicação da GAN pra geração de faces.
5) Etapa 5: geração e avaliação do modelo.

C. Materiais

Para a realização do trabalho descrito, as seguintes ferramentas foram utilizadas:

- Ambiente Google Colab[4] para codificação e execução dos testes. Em função dos tempos prolongados de treinamento da rede, uma assinatura Colab Pro foi necessária para execuções duradouras na plataforma, nesta categoria o ambiente pode fornecer as GPUs Nvidia Tesla P100 ou Nvidia Tesla T4;
- Linguagem de programação Python na versão 3.7.13;
- Pytorch[5] utilizado como framework de *machine learning*, assim como Pytorch Ignite[6] para estruturar a execução do código para o treinamento e cálculo das métricas utilizadas na avaliação as imagens.

V. Desenvolvimento

Nesta seção são apresentadas as etapas do desenvolvimento, definidas na descrição do método.

A. Etapa 1 - identificação da base de dados (dataset)

Conforme estabelecido, o objetivo do presente trabalho é realizar a geração de imagens para que respeitem os atributos desejados, portanto opta-se pela utilização do *dataset* CelebA, pois possui diversos atributos que remetem a características faciais presentes nas imagens, que serão utilizados como condição para o treinamento dos modelos. O conjunto de dados possui maior quantidade de usos/citações em aplicações de GANs na síntese facial, se comparado a outros *datasets* com propósitos similares, o que o categoriza como uma boa escolha para utilização no modelo proposto neste trabalho.

O CelebA possui 40 atributos referentes às diferentes características dos rostos. Estes atributos são de valor binário, indicando a presença daquela característica específica na imagem. Possui também atributos de localização do nariz, olhos esquerdo e direito e extremidades esquerda e direita da boca, através de coordenadas x e y de onde se encontram na imagem [28]. É constituído por 202599 imagens contendo rostos de 10177 celebridades diferentes.

B. Etapa 2: pré-processamento dos dados

Para o treinamento do modelo proposto foram selecionados os atributos *Male* (sexo masculino) e *Smiling* (sorrindo). Estes atributos são exemplificados na Figura 9. Outros atributos foram testados, porém, estes foram os que produziram os melhores resultados tanto na análise visual quanto em termos de disponibilidade dos dados (verificar Tabela V). Sendo assim, os demais atributos não são carregados nem utilizados para controlar o treinamento. Assim, cada imagem fica apenas com esses dois atributos associados a ela. Todas as instâncias do dataset são utilizadas no treinamento. Um subconjunto com as primeiras 3000 imagens do *dataset* é utilizado para o cálculo das métricas FID e IS, conforme a implementação disponibilizada para o cálculo das métricas [37] [38].

[4]https://colab.research.google.com/
[5]https://pytorch.org
[6]https://pytorch-ignite.ai

TABELA V: Quantidade de instâncias verdadeiras e falsas dos atributos selecionados

valor/Atributo	Male	Smiling
verdadeiro	84434	104930
falso	118165	97669

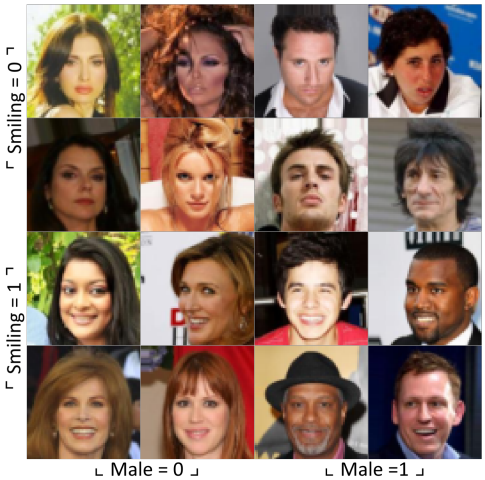


Fig. 9: Valores dos atributos selecionados nas instâncias do CelebA.

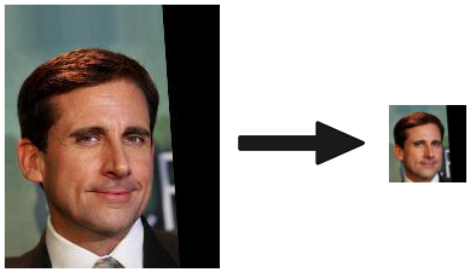


Fig. 10: Exemplo de uma imagem após o pré-processamento.

C. Etapa 3: transformação dos dados

Para realizar o treinamento do modelo, as imagens do dataset são transformadas primeiramente aplicando um corte quadrado no centro, de 178 x 178 *pixels*, e então redimensionadas para o tamanho de 64x64 pixels. Os valores de RGB de cada pixel são normalizados para que sejam representados na faixa de valores entre -1 e 1, conforme a implementação da DCGAN [8]. Esse tratamento é realizado pois os modelos Gerador e Discriminador estão dimensionados para trabalhar com as imagens na resolução de 64x64 pixels. A Figura 10 exibe uma imagem do *dataset* antes e depois deste redimensionamento.

Após a carga do *dataset*, cada instância é composta por 1 imagem de 64x64 pixels, e os valores dos atributos *Male* e *Smiling*.

D. Etapa 4 - aplicação da GAN pra geração de faces

Nesta etapa, foram feitos os ajustes necessários no código, partindo da implementação da DCGAN [8], adicionando a entrada adicional para os atributos [39], categorizando-a como

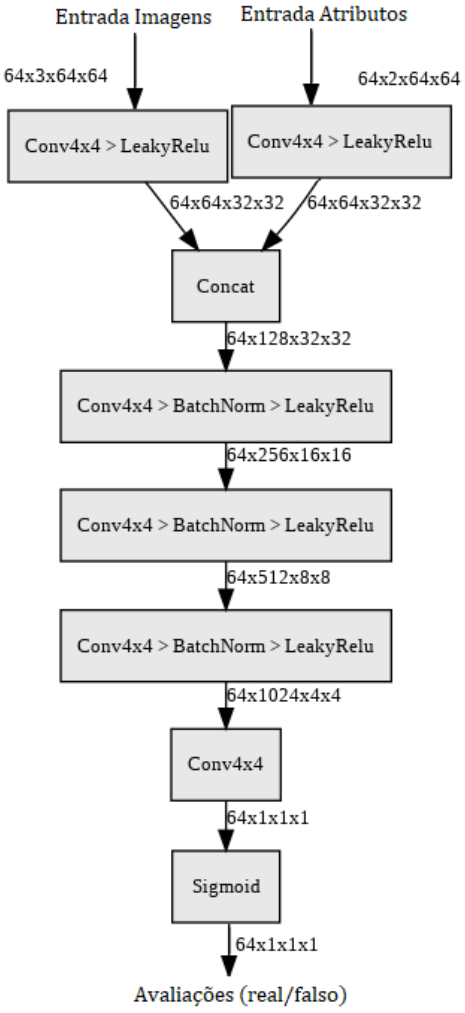


Fig. 11: Modelo Discriminador

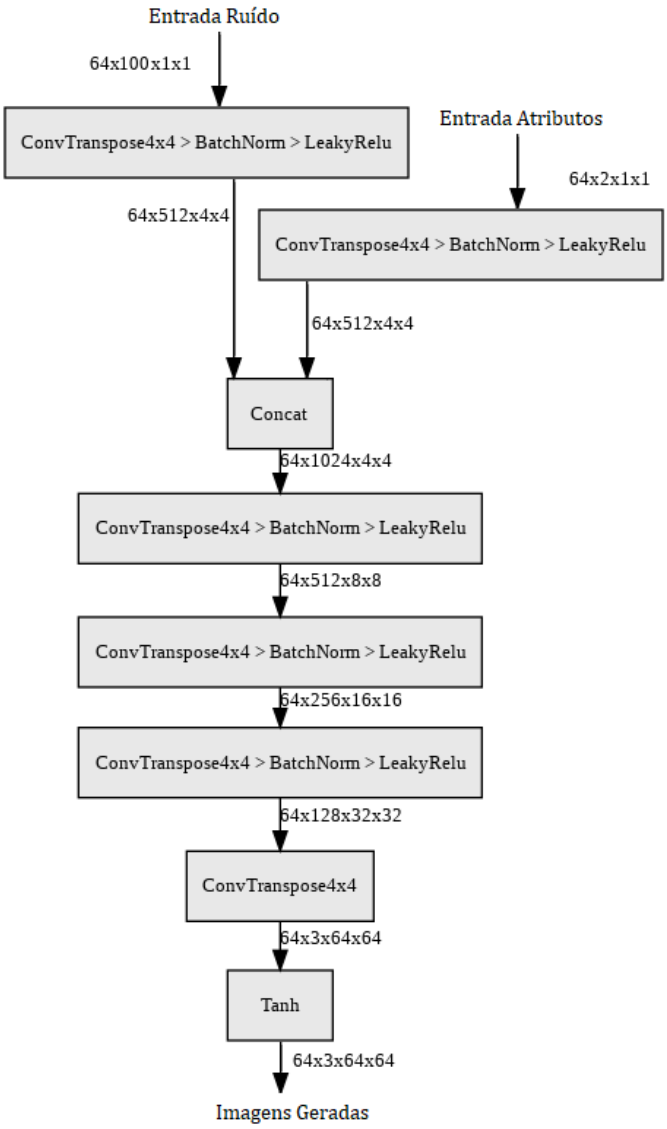


Fig. 12: Modelo Gerador

uma cDCGAN. A partir disso propõe-se modificar parâmetros, treinar os modelos, avaliar métricas e salvar os resultados. No discriminador, são recebidas as imagens do *batch* atual e o vetor de atributos respectivos à aquelas imagens no *dataset*. No gerador, é recebida a entrada de tamanho 100, conforme explicado na Seção II-B, e o mesmo vetor de atributos das imagens do *batch* da iteração corrente. Para o treinamento do Discriminador, a entrada das imagens e a entrada de atributos inicialmente vão para duas camadas convolucionais de entrada separadas. As saídas de ambas são então concatenadas para a entrada da próxima camada convolucional. Na sequência, passa por mais 3 camadas de convolução sendo a última ligada a função sigmoide como ativação para se ter o resultado da classificação de real/falso. O modelo é representado na Figura 11.

O Gerador possui a entrada para o vetor ruído de tamanho 100, e os mesmos atributos do *batch* atual, que são recebidos em duas camadas de entrada distintas, para em seguida concatenar as saídas dessas camadas para a entrada de uma próxima camada. Na sequência passa por mais 3 convoluções transpostas, sendo a saída da última ligada a função da tangente hiperbólica, como ativação para se ter o resultado da geração. O modelo é representado na Figura 12.

O *Pytorch Ignite* foi utilizado para o cálculo das métricas de avaliação FID e IS. Utilizou-se o código fornecido pelos criadores da biblioteca [38] para realizar o cálculo. O código em questão segue os conceitos de treinamento do *Ignite*, que se baseiam em três componentes principais: *Engine*, *Events*, *Handlers*. O *Engine* ou motor, considerando um *batch size* e um número de épocas definido, executa uma função atribuída pra ele, nesse caso, a função responsável por fazer uma iteração do *loop* de treinamento. Os *Events* são eventos que o motor emite em pontos específicos do treinamento, por exemplo: época concluída ou iteração concluída. Por fim, os *Handlers* são funções que podem ser associadas a eventos do motor. No código fornecido, dois motores são utilizados: um motor de treino (que executará o *loop* de treinamento), e um de avaliação (que executará o cálculo das métricas). Para este trabalho, ao final de cada iteração, um *handler* armazena os valores de perda do Gerador e do Discriminador para posterior análise, e ao final de cada época, conforme proposto no código

fornecido, as métricas FID e IS são calculadas pelo motor de avaliação.

O motor de avaliação utiliza o subconjunto definido na Etapa 2 deste trabalho como *dataset*. A função associada a ele recebe o *batch* da iteração (as instâncias são carregadas aleatoriamente) e gera um *batch* de mesmo tamanho através do Gerador. Essa função então redimensiona as imagens de ambos os *batches* real e falso para 299x299 pixels, tamanho mínimo necessário para que o *InceptionV3* possa receber as imagens e então as métricas sejam calculadas. Foi necessário realizar um ajuste nas dimensões do modelo *InceptionV3* utilizado pela métrica *FID*, pois a versão Pytorch da métrica utiliza um modelo levemente diferente daquele utilizado pela implementação original da métrica. Este detalhe faz com que os valores de *FID* calculados não fossem diretamente comparáveis ao da implementação original. Através do repositório oficial do *Pytorch Ignite*, foi disponibilizado o ajuste a ser feito no código antes de carregar a métrica (apresentado no Algoritmo 1).

Algoritmo 1: Ajuste de dimensões do *InceptionV3* utilizado para o cálculo da FID

```
from pytorch_fid.inception import InceptionV3
import torch.nn as nn

dims = 2048
block_idx = InceptionV3.BLOCK_INDEX_BY_DIM[dims]
model = InceptionV3([block_idx]).to(device)

class WrapperInceptionV3(nn.Module):

  def __init__(self, fid_incv3):
    super().__init__()
    self.fid_incv3 = fid_incv3

  @torch.no_grad()
  def forward(self, x):
    y = self.fid_incv3(x)
    y = y[0]
    y = y[:, :, 0, 0]
    return y

wrapper_model = WrapperInceptionV3(model)
wrapper_model.eval();

fid_metric = FID(num_features=dims,
     feature_extractor=wrapper_model)
```

E. Etapa 5 - geração e avaliação do modelo

Durante as primeiras execuções, notou-se certa dificuldade do Gerador em manter-se estável durante o treinamento, enquanto o valor da perda do Discriminador decaía rapidamente com o treinamento, a perda do Gerador dificilmente abaixava. Isso acontece quando o Discriminador evolui acima do esperado em estágios iniciais. Aprendendo assim a rotular as imagens do Gerador como falsas com uma acurácia alta antes que o gerador evolua o suficiente na tarefa de geração.

Após algumas pesquisas, identificou-se que uma técnica comum para tentar chegar a uma estabilização do treinamento é adicionar ruído nas imagens [40] [41] que são avaliadas pelo Discriminador. Fazendo com que ele evolua de forma mais lenta em estágios iniciais, permitindo ao Gerador alcançar gerações melhores.

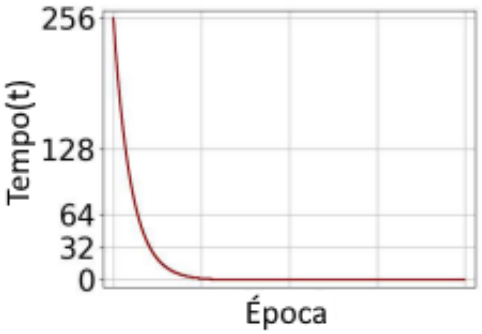


Fig. 13: Curva do valor de t para a aplicação do *Noisy Scale-Space* [41]

Para este trabalho, duas formas de aplicar ruído às imagens são testadas, para que se possa comparar seus efeitos no cenário da aplicação proposta:

1) Ruído simples: um vetor de valores aleatórios multiplicado a uma variável chamada $noisepercent$, que começa com valor 0.5 e decai 0.2 nas duas primeiras épocas e vai para 0 após a terceira época. Este vetor é somado à entrada das imagens a serem avaliadas pelo discriminador.

2) Noisy Scale-Space: [41] aplica uma função ϕ sobre 50% das imagens reais e 50% das geradas a serem avaliadas pelo discriminador, a função é aplicada por número t de vezes. O valor de t é obtido a partir de uma função exponencial que faz o valor de t decair conforme as iterações acontecem:

$$t^{(i)} := T \cdot exp(i/\beta) \tag{2}$$

Onde i é o número da iteração dentro da época atual. O autor utiliza $\beta = 20$ e $T = 256$, estes valores serão mantidos para os testes neste trabalho. A Figura 13 mostra como a curva de t decai conforme as iterações são realizadas.

A função ϕ descrita pelo autor trata-se da aplicação de um filtro gaussiano com tamanho de *kernel* 3x3 e também um ruído gaussiano com desvio padrão 0.15 sobre as imagens.

Uma técnica conhecida para ajudar no treinamento das GANs [30], a qual é aplicada neste trabalho, é o *Label Smoothing* (suavização de rótulo) unilateral, que consiste em considerar valores próximos de 1 (como por exemplo 0.9) ao invés de 1 para o rótulo verdadeiro. Dessa forma, o gerador não é penalizado em casos onde chega a resultados próximos do que seria rotulado como verdadeiro para o discriminador. Portanto o discriminador é penalizado assim que uma avaliação de imagem real passa de 0.9. Isso evita causar o *overconfidence*, efeito que pode ocorrer nas GANs quando o discriminador depende de um conjunto pequeno de características para determinar se uma imagem é real ou não, descartando assim, instâncias falsas com confiança demais [42].

VI. Resultados

Esta seção apresenta os resultados dos testes executados e em quais condições foram obtidos. Cada teste traz o valor obtido das métricas de avaliação definidas para este trabalho. Buscando alcançar quais parâmetros produzem melhores resultados em relação às métricas, foram testadas variadas

TABELA VI: Cenários de teste e seus respectivos parâmetros variados

Cenário	Aplicação Ruído	LS
1	Nenhum	Não
2	Nenhum	Sim
3	RS	Não
4	RS	Sim
5	NSS	Não
6	NSS	Sim

configurações. Primeiramente vários testes foram realizados para escolher os atributos com os quais o treinamento das redes seria condicionado para este trabalho, até definir os atributos *Male* e *Smiling*, mantendo-os em todos os testes apresentados nesta seção.

Para apresentar os testes, os cenários variam: Aplicação de ruído e uso de *Label Smoothing* (LS). As aplicações de ruído possíveis são:

- Nenhum: nenhum ruído aplicado às imagens durante o treinamento.
- Ruído Simples (RS): conforme descrito na Seção V-E1 para 50% das imagens reais e 50% das imagens geradas.
- *Noisy Scale-Space* (NSS): conforme descrito na Seção V-E2 para 50% das imagens reais e 50% das imagens geradas.

Para o *Label Smoothing* unilateral, é utilizado o valor 0.9 para considerar uma instância verdadeira, ao invés de 1.

Considerando estes parâmetros, foram definidos 6 cenários, com variações dos mesmos para testar diferentes situações. Com isso tem-se o intuito de comparar o efeito do uso de ruídos e/ou *Label Smoothing* nos valores das métricas FID e IS. Os cenários são identificados de 1 a 6, e apresentados na Tabela VI.

Os testes apresentados a seguir foram todos executados com os seguintes parâmetros fixos:

- Otimizador Adam para ambos os modelos, com taxa de aprendizagem = 0,0002, $\beta 1$ = 0,5 e $\beta 2$ = 0,999
- Taxa de aprendizagem decai nas épocas 10 e 15, tendo seu valor dividido por 10.
- Tamanho do *batch* = 64

Para cada cenário, calcularam-se as métricas FID e IS nas épocas 5, 10, 15, 20, 25 e 30, a fim de verificar os momentos do treinamento onde cada cenário produziu os melhores resultados. Cada época de treinamento leva em média 14 minutos de execução. O tempo de avaliação (cálculo das métricas) durou em média de 25 minutos para cada época, visto que o *InceptionV3* trabalha com as imagens em uma dimensão maior.

Considerando que cada cenário é constituído de 30 épocas de treinamento, sendo que a avaliação é realizada após 6 delas (épocas 5, 10, 15, 20, 25, 30), o tempo total para a execução de um cenário foi de aproximadamente 9h30min. Dentro do tempo de execução deste trabalho, foi possível executar cada cenário definido por 4 vezes. Portanto o tempo total para as execuções dos cenários foi em média de 228 horas. Este tempo desconsidera os demais testes que foram realizados para compreensão de cada parâmetro, incluindo-se os parâmetros fixos. Estima-se que durante o desenvolvimento deste trabalho, houve cerca de 150 horas adicionais em que o ambiente executou testes para validar escolhas de parâmetros e atributos, ou ainda re-execuções para salvar informações adicionais durante o treinamento.

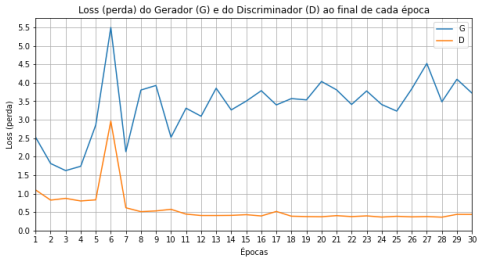


Fig. 14: Perdas do Gerador e Discriminador para o melhor cenário (Cenário 6)

Os resultados de cada cenário são apresentados na Tabela VII. Ela apresenta os valores de média e desvio padrão para a FID e o IS calculados nas épocas pré-definidas (5, 10, 15, 20, 25 e 30). Os valores são referentes a 4 execuções para cada cenário. Nesta tabela pode-se observar que os cenários 5 e 6 produziram os melhores valores para a métrica FID, atingindo uma média de 23,08 nas épocas 25 e 30 do Cenário 6. O valor do IS teve a melhor média na época 30 do Cenário 1, atingindo 2,62. Conforme constatado anteriormente, a FID representa melhor a qualidade das imagens se comparada à IS. Com isso pode-se dizer, com base nas métricas calculadas, que os Cenários 5 e 6 são capazes de produzir imagens de maior qualidade dentre os demais.

Os cenários 1 e 2 atingiram valores similares ou até melhores aos cenários 3 e 4, nas épocas finais os valores calculados para a FID dos Cenários 1 e 2 foram menores do que os cenários 3 e 4. Com isso conclui-se que a técnica de ruído simples, aplicada em 3 e 4 causou um efeito negativo na avaliação. Nestes 4 primeiros cenários, a suavização de rótulo unilateral não causou uma melhora nos valores. Entre os cenários 5 e 6, a diferença pequena entre os valores não indica que a suavização de rótulo tenha proporcionado alguma melhora significativa no cenário 6 em comparação ao 5, onde não é aplicada.

Por meio dos resultados obtidos, constata-se que a utilização do *Noisy Scale-Space* como forma de aplicação de ruído possibilitou ao modelo Gerador atingir os melhores resultados dentre as configurações. As perdas do Gerador nesta configuração também se mantiveram mais estáveis se comparadas às outras configurações, apesar de não acompanharem a evolução do Discriminador que, em etapas finais, tem uma perda muito pequena. No cenário 6, os valores de perda do Gerador permaneceram entre 3 e 4 por um grande período do treinamento (Figura 14). Pode-se perceber que entre as épocas 15 e 25, os valores das métricas tem pouca variação, o que é bem representado pelas imagens produzidas nestas épocas (Figura 15), onde as mudanças são pouco perceptíveis.

Para cada vetor ruído de entrada, o Gerador produz imagens diferentes. Portanto, para ilustrar a evolução do Gerador foram produzidas imagens a cada época de treinamento,

TABELA VII: Valores FID e IS atingidos para cada cenário e respectiva época.

Época →	5		10		15		20		25		30	
Cenário ↓	FID	IS	FID	IS	FID	IS	FID	IS	FID	IS	FID	IS
1	37,90±3,51	2,48±0,06	32,35±4,18	2,51±0,05	25,85±0,88	2,56±0,04	25,01±0,85	2,56±0,02	24,77±1,05	2,59±0,04	24,46±0,66	**2,62**±0,04
2	41,46±1,03	2,47±0,07	35,26±3,12	2,56±0,09	27,26±0,78	2,59±0,04	26,42±1,17	2,55±0,01	26,60±1,06	2,57±0,02	26,23±0,91	2,58±0,02
3	43,15±3,57	2,54±0,07	36,94±5,07	2,52±0,09	26,81±0,99	2,60±0,03	26,33±0,68	2,57±0,02	25,56±0,63	2,60±0,03	25,81±0,57	2,58±0,04
4	43,84±3,03	2,48±0,06	35,57±4,18	2,47±0,06	28,53±1,62	2,56±0,01	27,00±1,51	2,55±0,05	27,47±1,41	2,58±0,02	26,41±1,41	2,56±0,02
5	37,40±2,36	2,54±0,06	31,01±1,68	2,54±0,04	24,70±0,84	2,59±0,04	24,09±0,98	2,57±0,01	23,96±0,98	2,54±0,04	23,63±0,97	2,58±0,02
6	37,79±2,17	2,50±0,05	32,46±3,75	2,50±0,04	24,33±0,42	2,58±0,03	23,66±0,93	2,60±0,03	23,08±1,52	2,60±0,04	**23,08**±1,35	2,58±0,03

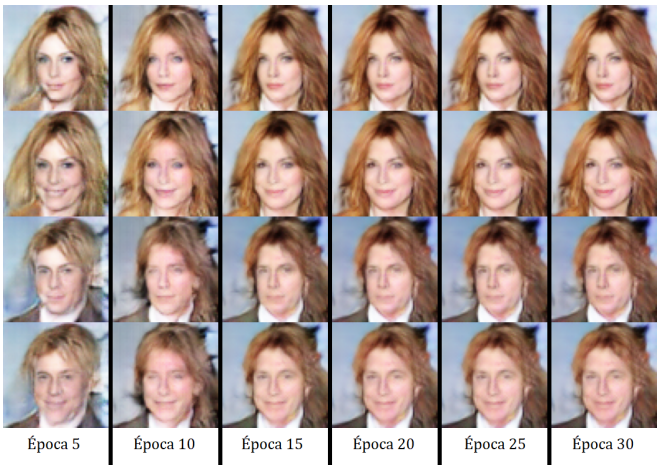


Fig. 15: Evolução das imagens produzidas pelo Gerador. Para a geração das imagens, definiu-se o mesmo ruído de entrada para todas. Nas linhas 1 e 2 *Male = 0*, 3 e 4 *Male = 1*. Nas linhas 1 e 3 *Smiling = 0*, 2 e 4 *Smiling= 1*

utilizando para isso um valor fixo de ruído de entrada do Gerador. Assim, as imagens representam apenas as variações produzidas a partir dos atributos. Essas imagens estão reunidas na Figura 16, mostrando a evolução das gerações a cada época. Para esta visualização foram geradas 4 imagens por época para demonstração, com as variações de atributos *Male* = 0 e 1, e *Smiling* = 0 e 1. Por meio das imagens é possível verificar a evolução das gerações e como as imagens se aproximam da realidade em relação à percepção humana, o que é refletido na melhora das métricas calculadas a cada época. Os resultados alcançados por este trabalho revelam um caminho promissor. O melhor cenário foi atingido por meio do uso do *Noisy Scale-Space*. Propõe-se comparar os valores das métricas obtidos pelos autores desta técnica [41], assumindo-os como o estado da arte (*baseline*), e os valores alcançados pelo presente trabalho. Ao comparar a média FID de 23,08 obtida aqui com a média 19,45 (*baseline*), constata-se que os autores do *Noisy Scale-Space* atingiram um melhor valor de FID, porém, este valor não contempla a complexidade adicional de controle dos atributos.

Ao alcançar uma média da FID próxima ao estado da arte, com diferença de 3,63 no valor, mesmo ao adicionar o condicionamento dos atributos, este trabalho cumpre seu propósito exploratório, demonstrando o comportamento de uma GAN condicional para geração de faces com atributos controláveis em relação a diferentes configurações.

VII. Considerações Finais e Trabalhos Futuros

A pesquisa realizada neste trabalho teve o objetivo de, por meio de uma pesquisa exploratória, compreender o funcionamento das *Generative Adversarial Networks* no contexto da síntese de faces e propor uma aplicação para gerar faces aleatórias controláveis via atributos. Para isso apresentou primeiramente uma breve fundamentação teórica que constroi conhecimento para a compreensão das etapas seguintes. O conhecimento para compreender o funcionamento dessas redes, precisou ser construído por camadas de complexidade, desde a formalização clássica [4], indo para modelos que melhoram a capacidade de controlar o que será gerado. A aplicação de técnicas para estabilizar o treinamento também foi investigada. Realizou-se uma revisão sistemática de trabalhos relacionados à síntese de imagens de faces humanas através das *gan*s, para tomar conhecimento sobre *datasets*, métricas e dentre outros aspectos. Por fim, foi proposta uma aplicação voltada à síntese de faces condicionada por atributos, xpresentando o comportamento das redes em relação a parâmetros e métricas de avaliação.

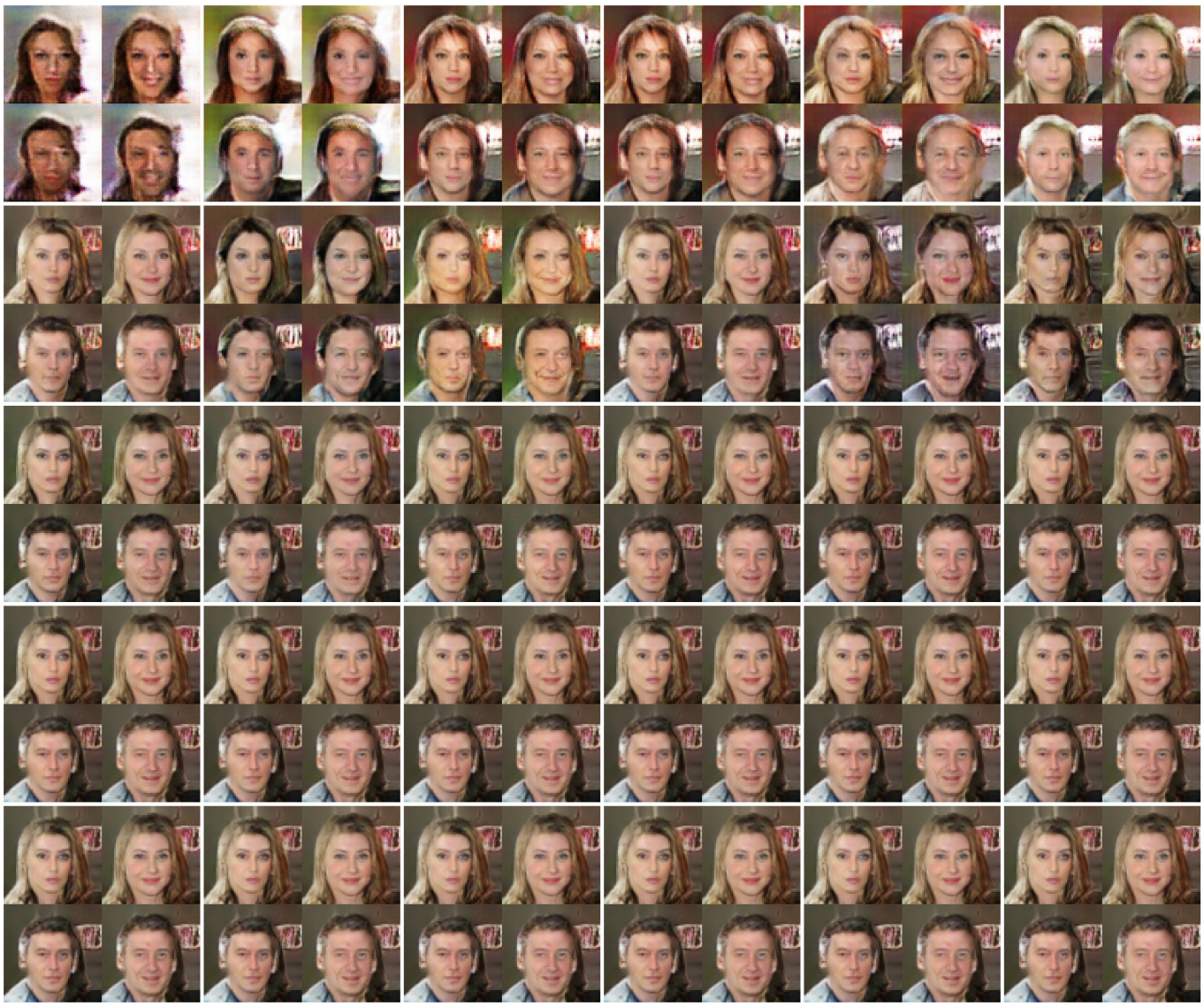
Os resultados obtidos atingem o objetivo exploratório deste trabalho. Apesar de produzir uma métrica FID maior que outros trabalhos na área, demonstra como técnicas para estabilização do treinamento podem melhorar os resultados dentro de um contexto onde os atributos da imagem são controláveis. Esta tarefa vai além de apenas gerar dados pertencentes a um domínio, mas envolve a complexidade de controlar como essas instâncias serão geradas pelas redes.

Em algumas imagens produzidas, identificam-se áreas, características e distorções que visualmente não remetem a características humanas, mas este efeito é minimizado conforme o avanço das épocas. O *dataset* CelebA possui faces em diferentes ângulos e em diferentes ambientes, e algumas das instâncias possuem objetos não relacionados ao rosto nas imagens, o que poderia atrapalhar os modelos na compreensão das características humanas contidas nelas. Nas imagens produzidas, percebe-se que há algumas distorções em volta das faces geradas, geralmente o rosto é a parte mais nítida e reconhecível da imagem, enquanto que os elementos ao redor, incluindo parte do cabelo, aparecem borrados ou distorcidos.

Para trabalhos futuros, sugere-se investigar e experimentar mais formas de melhorar e variar os resultados aqui apresentados, como por exemplo:

- Identificar e filtrar do *dataset*, instâncias que possuam objetos ou distorções não relacionados aos rostos.
- Adicionar outros atributos juntamente com os escolhidos para este trabalho.

Fig. 16: Imagens produzidas pelo modelo Gerador do Cenário 6. Cada grupo de 4 imagens representa a geração de uma época para as variações dos atributos. Representa as épocas de 1 a 30.



- Utilizar uma CNN pré-treinada com o dataset, capaz de reconhecer os atributos, para avaliar a presença destes na imagem, visto que neste trabalho, o modelo consegue seguir os atributos para a geração, mas nenhuma avaliação sobre a presença dos atributos é feita nas imagens.
- Experimentar e estudar o impacto de outras funções de perda, pois existem diversas funções propostas pelos pesquisadores na área, mas que não foram exploradas no escopo deste trabalho.

REFERENCES

[1] T. M. Mitchell, *Machine Learning*. New York: McGraw-Hill, 1997.

[2] M. C. Monard and J. A. Baranauskas, "Conceitos sobre aprendizado de máquina," in *Sistemas Inteligentes Fundamentos e Aplicações*, pp. 89–114, Barueri-SP: Manole Ltda, 1 ed., 2003.

[3] S. Haykin, *Redes Neurais: Princípios e Prática*. Porto Alegre: Bookman, 2 ed., 2007.

[4] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014.

[5] A. Borji, "Pros and cons of gan evaluation measures," 2018.

[6] J. Brownlee, *Generative Adversarial Networks with Python: Deep Learning Generative Models for Image Synthesis and Image Translation*. Machine Learning Mastery, 2019.

[7] I. Goodfellow, "Nips 2016 tutorial: Generative adversarial networks," 2017.

[8] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2016.

[9] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," 2018.

[10] E. Zakharov, A. Shysheya, E. Burkov, and V. Lempitsky, "Few-shot adversarial learning of realistic neural talking head models," 2019.

[11] Google, "The discriminator | generative adversarial networks." https://developers.google.com/machine-learning/gan/discriminator?hl=pt-br, May 2019.

[12] M. A. Nielsen, *Neural networks and deep learning*, vol. 25. Determination press San Francisco, CA, 2015.

[13] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," 2018.

[14] M. Mirza and S. Osindero, "Conditional generative adversarial nets," 2014.

[15] C. Rong, X. Zhang, and Y. Lin, "Feature-improving generative adversarial network for face frontalization," *IEEE Access*, vol. 8,

pp. 68842–68851, 2020.
[16] M. Z. Khan, S. Jabeen, M. U. G. Khan, T. Saba, A. Rehmat, A. Rehman, and U. Tariq, "A realistic image generation of face from text description using the fully trained generative adversarial networks," *IEEE Access*, vol. 9, pp. 1250–1260, 2021.
[17] H. Zhang, W. Chen, J. Tian, H. He, and Y. Jin, "Rag: Facial attribute editing by learning residual attributes," *IEEE Access*, vol. 7, pp. 83266–83276, 2019.
[18] X. Wang, J. Gong, M. Hu, Y. Gu, and F. Ren, "Laun improved stargan for facial emotion recognition," *IEEE Access*, vol. 8, pp. 161509–161518, 2020.
[19] X. Ning, S. Xu, W. Li, and S. Nie, "Fegan: Flexible and efficient face editing with pre-trained generator," *IEEE Access*, vol. 8, pp. 65340–65350, 2020.
[20] L. Liu, S. Wang, and L. Wan, "Component semantic prior guided generative adversarial network for face super-resolution," *IEEE Access*, vol. 7, pp. 77027–77036, 2019.
[21] C. Xu, Y. Tang, M. Toyoura, J. Xu, and X. Mao, "Generating users' desired face image using the conditional generative adversarial network and relevance feedback," *IEEE Access*, vol. 7, pp. 181458–181468, 2019.
[22] J. Zhang, A. Li, Y. Liu, and M. Wang, "Adversarially regularized u-net-based gans for facial attribute modification and generation," *IEEE Access*, vol. 7, pp. 86453–86462, 2019.
[23] D. Huang, X. Tao, J. Lu, and M. N. Do, "Geometry-aware gan for face attribute transfer," *IEEE Access*, vol. 7, pp. 145953–145969, 2019.
[24] J. Liu, Q. Li, P. Zhang, G. Zhang, and M. Liu, "Unpaired domain transfer for data augment in face recognition," *IEEE Access*, vol. 8, pp. 39349–39360, 2020.
[25] X. Chen, L. Qing, X. He, J. Su, and Y. Peng, "From eyes to face synthesis: a new approach for human-centered smart surveillance," *IEEE Access*, vol. 6, pp. 14567–14575, 2018.
[26] J. Liu, Q. Li, M. Liu, and T. Wei, "Cp-gan: A cross-pose profile face frontalization boosting pose-invariant face recognition," *IEEE Access*, vol. 8, pp. 198659–198667, 2020.
[27] D. Li, Z. Li, R. Luo, J. Deng, and S. Sun, "Multi-pose facial expression recognition based on generative adversarial network," *IEEE Access*, vol. 7, pp. 143980–143989, 2019.
[28] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild.," in *ICCV*, pp. 3730–3738, IEEE Computer Society, 2015.
[29] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," Tech. Rep. 07-49, University of Massachusetts, Amherst, October 2007.
[30] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," 2016.
[31] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," *CoRR*, vol. abs/1512.00567, 2015.
[32] Y. L. Tong, *The Multivariate Normal Distribution*. Springer New York, 1990.
[33] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," 2018.
[34] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *Image Processing, IEEE Transactions on*, vol. 13, pp. 600 – 612, 05 2004.
[35] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," 2020.
[36] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, *et al.*, "Knowledge discovery and data mining: Towards a unifying framework.," in *KDD*, vol. 96, pp. 82–88, 1996.
[37] V. Fomin, J. Anmol, S. Desroziers, J. Kriss, and A. Tejani, "High-level library to help with training neural networks in pytorch." https://github.com/pytorch/ignite, 2020.
[38] V. Fomin, J. Anmol, S. Desroziers, J. Kriss, and A. Tejani, "Gan evaluation : the frechet inception distance and inception score metrics." https://github.com/pytorch-ignite/pytorch-ignite.ai/blob/gh-pages/blog/2021-08-11-GAN-evaluation-using-FID-and-IS.ipynb, 2021.
[39] E. Hu, "Using conditional deep convolutional gans to generate custom faces from text descriptions," Jun 2021.
[40] J. Brownlee, "Tips for training stable generative adversarial networks," Jun 2019.
[41] K. Nakamura, S. Korman, and B.-W. Hong, "Stabilization of generative adversarial networks via noisy scale-space," 2021.
[42] J. Hui, "Gan — ways to improve gan performance," Mar 2020.

Felipe Fischer da Silva Bacharelando em Ciência da Computação pela Universidade de Caxias do Sul. A 5 anos atua como desenvolvedor de software para desktop e aplicações em nuvem utilizando a linguagem C#.

Carine Geltrudes Webber Doutora em Ciência da Computação pela École Doctorale Mathématiques et Informatiques, da Université de Grenoble I Joseph Fourier, França, Mestre (UFRGS) e Graduada (UCS) em Ciência da Computação. Atua como Professor Titular na Área de Conhecimento de Exatas e Engenharias da UCS. Integra o Programa de Pós-graduação em Ensino de Ciências e Matemática, desenvolvendo a linha de pesquisa de IA aplicada ao Ensino. Atua em diversos cursos e projetos de pesquisa nas áreas de IA, Ciência de Dados e Indústria 4.0.