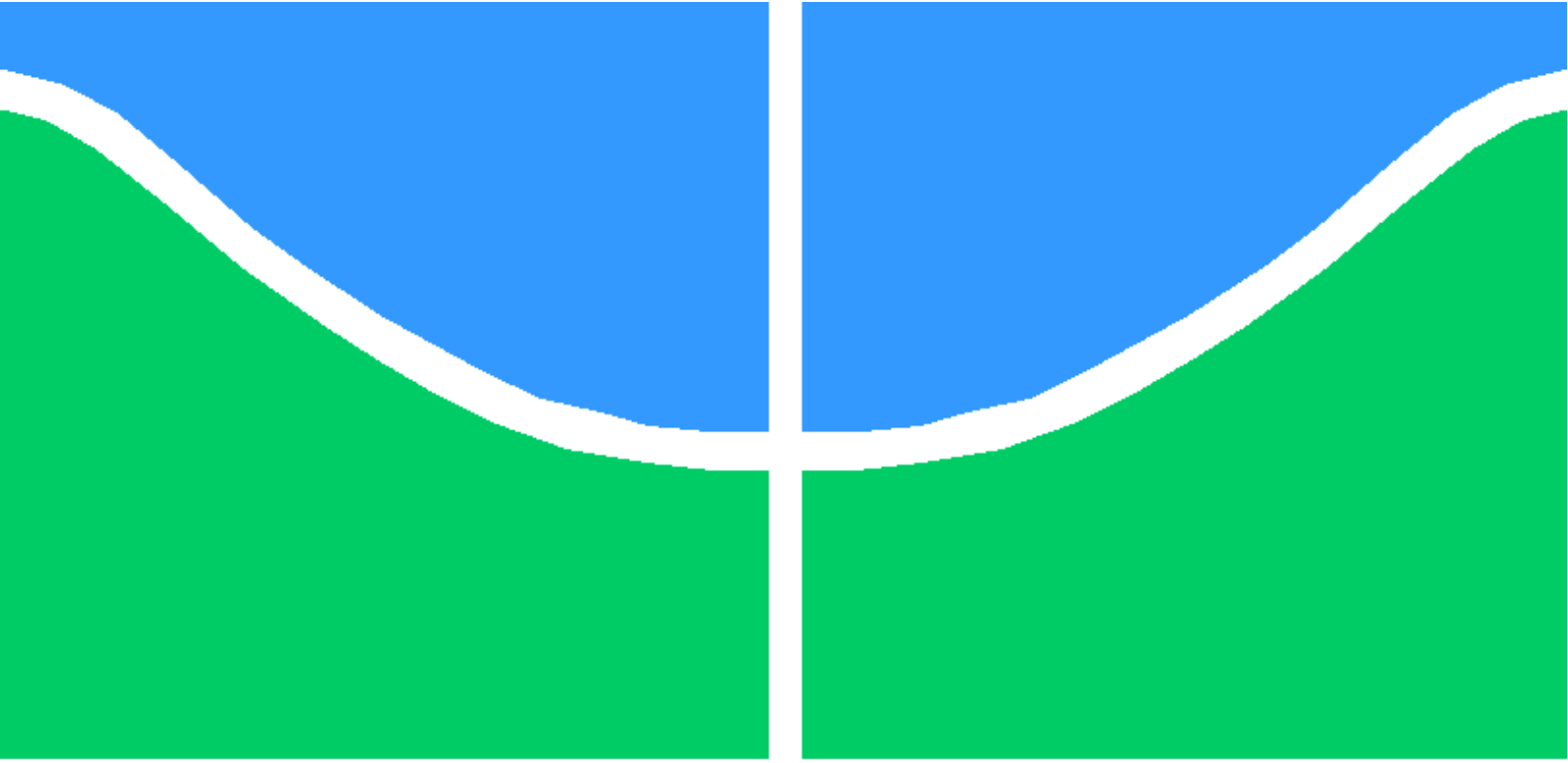

TRABALHO DE CONCLUSÃO DE CURSO

REDES GERADORAS ADVERSÁRIAS: TEORIA E APLICAÇÃO À MODELAGEM GERADORA DE IMAGENS FACIAIS

Alan Assis Pennacchio

Brasília, Julho de 2017

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASÍLIA
Faculdade de Tecnologia

TRABALHO DE CONCLUSÃO DE CURSO

REDES GERADORAS ADVERSÁRIAS: TEORIA E APLICAÇÃO À MODELAGEM GERADORA DE IMAGENS FACIAIS

Alan Assis Pennacchio

Relatório submetido ao Departamento de Engenharia Elétrica como requisito parcial para obtenção do grau de Engenheiro Eletricista

Banca Examinadora

Prof. Alexandre R. S. Romariz, ENE/UnB _____
Orientador

Profª. Mylene C. Q. Farias, ENE/UnB _____
Examinador interno

Prof. Adolfo Bauchspiess, ENE/UnB _____
Examinador interno

Agradecimentos

Agradeço aos meus pais, irmã e amigos, pelo amor, incentivo e apoio incondicional. Também, ao meu orientador, o Prof. Alexandre R. S. Romariz, pela oportunidade e suporte na elaboração deste trabalho. Finalmente, a todos que direta ou indiretamente fizeram parte da minha formação, o meu muito obrigado.

Alan Assis Pennacchio

RESUMO

A tendência para o volume, velocidade e variedade de dados disponíveis no mundo para análise é só crescer. Portanto, faz-se cada vez mais importante o desenvolvimento de algoritmos e modelos que permitam analisar essa quantia significativa de dados inexplorada. Os mesmo são gerados, usualmente, a partir de fenômenos físicos, denominados processos geradores. Uma das formas para se obter maior compreensão acerca de um tipo de dado é, exatamente, por meio do estudo de seu processo gerador. O área que estuda e modela esses fenômenos é denominada modelagem geradora.

Neste trabalho é apresentado um desenvolvimento teórico da classe de modelos geradores chamada redes geradoras adversárias (GANs). Esse desenvolvimento engloba desde a construção do modelo até a formulação de um algoritmo para seu ajuste, sendo apresentado de forma natural e progressiva, baseando-se em conceitos de teoria da informação e probabilidade. Ainda, a fim de verificar a aplicabilidade da teoria introduzida, implementou-se um modelo baseado em GANs para a modelagem geradora de imagens faciais. O mesmo foi ajustado de forma a representar a distribuição das imagens do conjunto de dados CelebA, o qual consiste de aproximadamente 200 mil imagens de rostos de pessoas.

O modelo implementado obteve excelentes resultados, gerando imagens realistas de rostos, bastante parecidas com as encontradas no conjunto de dados CelebA. A qualidade das mesmas foi avaliada subjetivamente a partir da nitidez, detalhamento e organização das diversas estruturas que as compõe. Finalmente, foram exploradas e discutidas algumas propriedades importantes desse modelo, tornando a análise do mesmo mais completa.

Palavras-chave: redes geradoras adversárias, modelagem geradora, redes neurais, imagens naturais, aprendizado de máquina, processamento de imagens.

ABSTRACT

The volume, speed and variety of available data for analysis has been growing considerably. Therefore, it is becoming increasingly important to develop algorithms and models for the analysis of this significant amount of unexplored data. This data is, usually, generated from physical phenomena called generative processes. A smart way to better develop insight on a certain type of data is precisely through the study of its generative process. The area that is concerned with studying and modeling these phenomena is called generative modeling.

In this work, it is presented the theoretical development of a certain class of generative models called generative adversarial networks (GANs). This development encompasses a wide variety of its aspects, ranging from the way it's structured to a formulation of an algorithm for its fitting process, being presented in a natural and progressive style. Also, in order to verify the applicability of the theory introduced, a model based on GANs for the generative modeling of facial images is implemented and adjusted to represent the distribution of images from the CelebA dataset, which consists of approximately 200,000 images of human faces.

The implemented model obtained excellent results, being able to generate realistic human faces, very similar to those found in the CelebA dataset. Their quality was subjectively evaluated based on several aspects, such as sharpness, detail and organization of the various structures that compose them. Finally, some important properties of this model are explored and discussed in greater depth, making the analysis more through.

Keywords: generative adversarial networks, generative modeling, neural networks, natural images, machine learning, image processing.

SUMÁRIO

1 INTRODUÇÃO **1**

2 FUNDAMENTAÇÃO TEÓRICA **5**

2.1 Dados 6

2.1.1 Interpretação probabilística 6

2.2 Modelagem geradora 7

2.2.1 Modelos geradores implícitos e explícitos 8

2.2.2 Ajuste de parâmetros 9

2.3 Redes Neurais 11

2.3.1 Organização em camadas 11

2.3.2 Poder representacional 12

2.3.3 Algumas considerações 13

3 REDES GERADORAS ADVERSÁRIAS **15**

3.1 Construção do modelo 16

3.2 Funções de perda 16

3.2.1 Modelo gerador 16

3.2.2 Modelo discriminador 18

3.3 Otimização das funções de perda 19

3.3.1 Considerações práticas 19

3.3.2 Gradientes 20

3.3.3 Algoritmo para otimização 21

3.4 Influência de π 22

4 APLICAÇÃO **27**

4.1 Imagens naturais 28

4.1.1 Aplicabilidade de GANs 30

4.2 Distribuição geradora de dados 30

4.3 Aspectos técnicos 32

4.3.1 Arquitetura dos modelos 32

4.3.2 Processo de ajuste 34

4.3.3 Aspectos computacionais 36

4.4 Geração de imagens 36

4.5 Interpolação no espaço latente 38

5 CONCLUSÃO **41**

REFERÊNCIAS BIBLIOGRÁFICAS **45**

NOTAÇÃO

x	Escalar
$\boldsymbol{x}$	Vetor
$\mathbb{X}$	Conjunto
$\|\|\boldsymbol{x}\|\|$	Norma de $\boldsymbol{x}$
$f_{\boldsymbol{\theta}}(\cdot)$	Função parametrizada por $\boldsymbol{\theta}$
x	Variável aleatória escalar
$\mathbf{x}$	Variável aleatória vetorial
$\mathbf{x} \sim p(\boldsymbol{x})$	Variável aleatória $\mathbf{x}$ distribuída conforme $p(\boldsymbol{x})$
$p(\cdot\|\cdot)$	Distribuição de probabilidade condicional
$p(\cdot;\boldsymbol{\theta})$	Distribuição de probabilidade parametrizada por $\boldsymbol{\theta}$
$\mathcal{N}(\cdot;\boldsymbol{\mu},\boldsymbol{\Sigma})$	Distribuição gaussiana com média $\boldsymbol{\mu}$ e covariância $\boldsymbol{\Sigma}$
$\mathbb{E}_{p(\boldsymbol{x})}\left[f(\mathbf{x})\right]$	Expectância de $f(\mathbf{x})$ com respeito a $p(\boldsymbol{x})$, $\int_{\mathcal{X}} f(\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x}$
$H(\mathbf{x})$	Entropia de $\mathbf{x}$, $-\mathbb{E}_{p(\boldsymbol{x})}[\log p(\mathbf{x})]$
$H(\mathbf{y}\|\mathbf{x})$	Entropia condicional de $\mathbf{y}$ dado $\mathbf{x}$, $-\mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}[\log p(\mathbf{y}\|\mathbf{x})]$
$D_{KL}\left[p(\boldsymbol{x})\|\|q(\boldsymbol{x})\right]$	Divergência de Kullback-Leibler entre $p(\boldsymbol{x})$ e $q(\boldsymbol{x})$, $\mathbb{E}_{p(\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}\right]$

Capítulo 1

INTRODUÇÃO

Às vezes, não se presta a devida atenção à quantidade de informação facilmente acessível e presente no mundo: há objetos de diversas formas diferentes, com os mais variados atributos; animais que correm, emitem sons, voam e caçam; pessoas que andam, falam, comem e pensam; monitores mostrando informações sobre o ambiente político do país ou sobre os últimos jogos de futebol. Todas essas informações consistem, em sua essência, da interpretação dada a uma quantidade massiva de dados existente, que se encontra em constante fluxo.

Com a crescente utilização e inserção da tecnologia na vida das pessoas, faz-se cada vez mais importante o desenvolvimento de algoritmos e modelos que permitam a análise e entendimento dessa quantia significativa disponível de dados altamente inexplorada. Esses dados tem origem em determinados fenômenos físicos, usualmente extremamente complexos e frutos da interação de diversos fatores, chamados de processos geradores.

Com base na intuição da famosa citação do físico teórico Richard Feynman[1], "o que eu não posso criar, eu não compreendo", pode-se obter maior compreensão acerca de um determinado tipo de dado por meio de seu processo gerador. A área que estuda a modelagem desses fenômenos, de um ponto de vista matemático, é denominada modelagem geradora.

Dentro da área de modelagem geradora, uma classe de modelos que vem ganhando notório destaque, recentemente, é a das redes geradoras adversárias (GANs, do inglês *generative adversarial networks*) [1]. Esse destaque pode ser atrelado tanto à abordagem inovadora na qual esses modelos se baseiam quanto à sinergia que têm com redes neurais, permitindo-se explorar o alto poder representacional destas últimas e levando a resultados extremamente promissores.

A formulação tradicional dessa classe de modelos é baseada em teoria dos jogos, apresentando seu processo de ajuste como um jogo de dois jogadores: o modelo gerador e um modelo discriminador. O papel do primeiro é gerar amostras realistas de dados de um determinado processo gerador, o qual se deseja modelar. O segundo tem função exclusiva de avaliar se essas amostras parecem ser reais, vindas desse processo gerador, ou artificiais, vindas do modelo gerador.

Dessa forma, ajusta-se o modelo gerador de forma a enganar, ao máximo, o modelo discriminador, o qual é, simultaneamente, ajustado com o objetivo de acertar ao máximo em seus palpites. Portanto, pode-se ver esse processo como um jogo entre os dois modelos, no qual um é posto contra o outro, motivando o uso do termo adversárias em seu nome.

Por teoria de jogos ser uma área muito pouco explorada em modelos geradores e aprendizagem de máquinas, no geral [2,3], a formulação de GANs a partir da mesma obscurece diversos aspectos importantes dessa classe de modelos. Dessa forma, não deixa claro os vários paralelos existentes com conceitos já bastante estudados e firmados dessas disciplinas. Essa abordagem, também, introduz suas diversas partes de forma um tanto quanto abrupta, sem um desenvolvimento gradativo que facilite sua compreensão.

Neste trabalho, será feito um desenvolvimento, de um ponto de vista teórico, dessa classe de modelos, englobando seus diversos aspectos, desde sua construção até a determinação de um

[1] https://www.quora.com/What-did-Richard-Feynman-mean-when-he-said-What-I-cannot-create-I-do-not-understand

algoritmo para seu ajuste. Ainda, esse desenvolvimento será feito firmando-se sobre conceitos estabelecidos de aprendizagem de máquinas e modelos geradores, fugindo da abordagem tradicional baseada em teoria de jogos supracitada. Dessa forma, espera-se apresentar GANs de forma bem mais natural e progressiva, tornando muito mais claros seus diversos aspectos.

Uma das formas de interação do homem com o mundo ao seu redor ocorre através do seu sistema visual. Muita da informação que guia seu comportamento e ações está diretamente ligada ao que vê. Dessa forma, cria-se uma motivação e surge um interesse para que computadores tenham um melhor entendimento acerca de imagens naturais, as quais podem ser descritas, de forma simplificada, como fotografias do mundo real.

Ainda, vistas como dados, imagens naturais têm processos geradores extremamente complexos e uma característica inerente de alta dimensionalidade e rica estrutura. Consequentemente, a modelagem geradora desse tipo de dados é extremamente difícil, sendo um desafio que vem sendo estudando há bastante tempo, tendo progredido bastante desde a introdução de GANs [1, 4, 5].

Dito isso, como forma de verificar a aplicabilidade da teoria desenvolvida acerca de GANs, será feita a aplicação desse modelo à modelagem geradora de fotografias de rostos humanos. Devido à dificuldade já mencionada da modelagem geradora de imagens naturais, faz-se uma prova prática perfeita da funcionalidade dessa classe de modelos.

Para tal, será implementado e ajustado um modelo baseado na arquitetura DCGAN [4], a qual consiste de uma das primeiras aplicações bem sucedidas de GANs à modelagem geradora de imagens naturais [5]. A mesma utiliza redes neurais convolucionais na definição de seus modelos gerador e discriminador, as quais consistem de transformações parametrizadas altamente especializadas para o processamento de imagens, trazendo, assim, vários benefícios.

As imagens faciais modeladas serão representadas pelo conjunto de dados CelebA [6], o qual consiste de aproximadamente 200 mil imagens de rostos de 10 mil celebridades diferentes. Espera-se, portanto, obter um modelo capaz de gerar imagens de rostos de pessoas realistas e parecidas com as do conjunto de dados utilizado, o qual terá, também, suas propriedades investigadas em seguida.

O trabalho será organizado em cinco capítulos, sendo este o primeiro. No segundo, será fornecida a fundamentação teórica necessária para o entendimentos dos capítulos seguintes. O terceiro capítulo consistirá do desenvolvimento teórico de GANs, começando pelo básico, na construção do modelo, até a formulação de um algoritmo para seu ajuste.

No quarto capítulo, será feita a aplicação prática dessa teoria, consistindo da implementação do modelo, seu ajuste seguindo os procedimentos desenvolvidos anteriormente e avaliação do mesmo quanto às suas diversas propriedades. No quinto e último capítulo, será feita a conclusão do trabalho.

Capítulo 2

FUNDAMENTAÇÃO TEÓRICA

2.1 Dados

O termo dados se refere, de um modo geral, a um conjunto de valores de variáveis quantitativas ou qualitativas. As anotações de um biólogo sobre o comportamento de um determinado animal, objeto de seu estudo, é um exemplo de dados qualitativos. Por outro lado, digamos que uma empresa do setor imobiliário deseje entender como o tamanho das casas em determinada área se relaciona com seus preços. Dessa forma, essa empresa coleta, para uma parcela de casas nessa área, seus respectivos tamanhos e preços. Esse conjunto de valores consiste de um exemplo de dados quantitativos.

A coleta de dados, usualmente, não é motivada pelos dados em si, mas pelo potencial de os mesmos fornecerem informação através de sua análise. O conhecimento é concebido a partir da experiência humana lidando com informação acerca de um determinado assunto. Dessa forma, fica clara a importância dos dados, visto que estes precedem a informação, a qual precede o conhecimento, objeto de desejo da natureza humana.

2.1.1 Interpretação probabilística

No contexto deste trabalho, o uso do termo dados se restringirá a dados quantitativos. Assim, a partir de agora, pode-se considerar que dados sejam sempre quantitativos. Ainda, será adotada uma formulação matemática desses objetos, construída sobre um ponto de vista probabilístico que fundamentará as análises feitas no restante do trabalho.

Assume-se que dados são originários de um processo gerador, o qual consiste de algum fenômeno físico, usualmente, e que pode ser representado, matematicamente, por uma distribuição de probabilidade. Sob essa perspectiva, a coleta de dados tem seu equivalente matemático na amostragem dessa distribuição, chamada de distribuição geradora de dados.

Assim, estipulado um tipo de dado, assume-se que sua representação matemática é feita por uma variável aleatória $\mathbf{x}$, com valores em $\mathbb{R}^d$, enquanto a de seu processo gerador é feita por uma distribuição geradora de dados $p(\boldsymbol{x})$. Ainda, chama-se d de dimensionalidade dos dados, a qual, usualmente, em casos práticos, está diretamente relacionada com sua complexidade.

Exemplo 2.1.1. *Considere uma sala de aula com 40 alunos numa escola de ensino médio. O professor de biologia, a fim de exemplificar a diferença esperada de altura entre homens e mulheres, pediu a todos os alunos que fornecessem suas respectivas alturas, em metros. Dessa forma, os dados coletados pelo professor podem ser representados, matematicamente, pelo conjunto*

$$\mathbb{X} = \{1.63, 1.51, 1.59, 1.71, 1.53, \ldots, 1.69\} \subset \mathbb{R}.$$

Esse conjunto, denominado conjunto de dados, consiste de uma série de observações acerca de algum fenômeno, ou, em outras palavras, amostras de uma distribuição geradora de dados. Como cada amostra desse conjunto de dados consiste apenas de um número real, diz-se que esses dados têm dimensionalidade um. Caso fossem coletados, também, os pesos de cada aluno, teríamos um conjunto de dimensionalidade dois e assim por diante.

Figura 2.1: Reta marcada com dados relativos às alturas dos alunos.

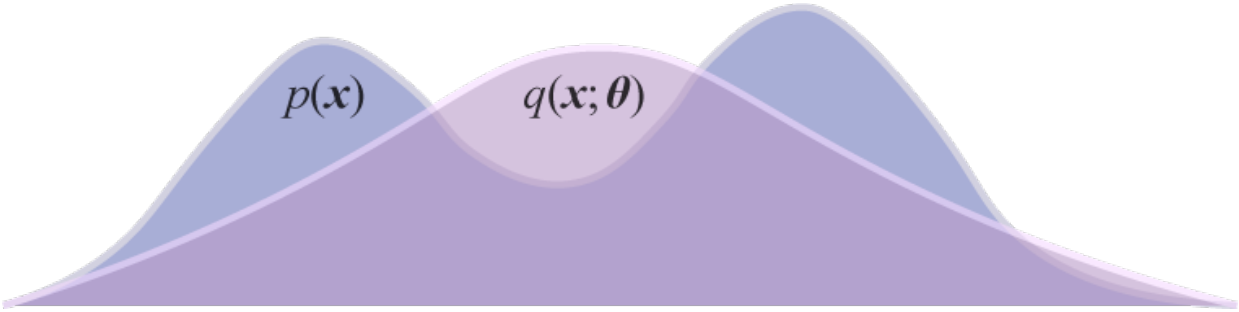

Figura 2.2: Modelos geradores têm seus parâmetros ajustados de forma a representar a distribuição geradora de dados da melhor forma possível, dentro dos critérios estabelecidos. Apesar disso, em alguns casos, não é possível representar $p(\boldsymbol{x})$ satisfatoriamente, como mostrado nessa figura, na qual é utilizado um modelo unimodal para representar uma distribuição bimodal. Nesse caso, a família de distribuições da qual $q(\boldsymbol{x};\boldsymbol{\theta})$ faz parte é imprópria para modelar $p(\boldsymbol{x})$.

O professor marcou os dados obtidos numa reta, como mostrado na Figura 2.1, para que seus alunos pudessem visualizar como suas alturas estavam distribuídas. Assim, pode-se notar, facilmente, que há duas concentrações de amostras, próximas a 1.55 metros e 1.65 metros, referentes às alturas médias das mulheres e homens da sala, respectivamente.

2.2 Modelagem geradora

Uma das formas para se obter melhor compreensão acerca dos dados com os quais se trabalha é através do entendimento de como são gerados. Portanto, é natural questionar-se como se dá esse processo que os origina, o qual é muitas vezes complexo e obscuro, funcionando como uma caixa preta a cujo interior não se tem acesso. Uma forma de iluminar esse processo, tornando-o mais transparente, é através da modelagem do mesmo, a qual é chamada de modelagem geradora.

Mais especificamente, definido um tipo de dado e o contexto sob o qual é obtido, esse termo se refere a representar, de alguma forma, sua distribuição geradora de dados $p(\boldsymbol{x})$. Essa modelagem é feita partindo do pressuposto que, dentro de uma família $\mathcal{Q} = \{q(\boldsymbol{x};\boldsymbol{\theta})\}_{\boldsymbol{\theta}\in\mathbb{R}^k}$ de distribuições parametrizadas, existe uma que a representa bem. Dessa forma, encontrar essa distribuição consiste em encontrar algum $\boldsymbol{\theta} \in \mathbb{R}^k$ tal que $q(\boldsymbol{x};\boldsymbol{\theta})$ seja próximo, sob algum critério, de $p(\boldsymbol{x})$. Esse processo é chamado de ajuste de parâmetros ou ajuste do modelo, podendo ser feito de diversas formas diferentes, sendo tratado mais à frente do trabalho.

2.2.1 Modelos geradores implícitos e explícitos

É interessante evidenciar uma distinção que existe entre duas classes de modelos geradores. Primeiramente, têm-se os chamados modelos explícitos. Essa classe de modelos engloba qualquer modelo que defina uma especificação paramétrica explícita de sua distribuição ou aproximação da mesma, consistindo da maioria dos modelos utilizados em aprendizado de máquinas e estatística.

Por outro lado, existem também os modelos implícitos, os quais definem algum procedimento estocástico capaz de gerar dados diretamente e, dessa forma, induzem uma distribuição, a qual se tem acesso através desse processo de amostragem da mesma. Esses modelos são bastante úteis em problemas onde se deseja simular dados, sendo dominantes em áreas relacionadas a climatologia, genética populacional e ecologia [5, 7].

Exemplo 2.2.1. *Considere que deseja-se modelar uma distribuição geradora de dados $p(x)$, a qual, suspeita-se, seja gaussiana. Dessa forma, um modelo gerador explícito natural a se considerar é $q(x;\boldsymbol{\theta}) = \mathcal{N}(x;\mu,\sigma)$, sendo $\boldsymbol{\theta} = (\mu,\sigma)$ e apresentando a especificação paramétrica, explícita, dada por*

$$q(x;\boldsymbol{\theta}) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right).$$

É interessante notar que esse modelo é tratável computacionalmente, significando que pode-se computar eficientemente $q(x;\boldsymbol{\theta})$ para qualquer $x \in \mathbb{R}^d$. Embora, num primeiro momento, isso não pareça claro, muitos modelos geradores explícitos não usufruem dessa propriedade, apresentando dificuldades para avaliar $q(x;\boldsymbol{\theta})$ em pontos arbitrários ou trabalhando com aproximações da mesma [5].

Exemplo 2.2.2. *Considere, agora, um caso mais geral, onde há uma distribuição geradora de dados $p(\boldsymbol{x})$, a qual é totalmente desconhecida e deseja-se modelar. Não assumindo nada sobre a mesma, inicialmente, é de interesse que o modelo utilizado seja extremamente flexível, capaz de representar, satisfatoriamente, diversos tipos de distribuições diferentes.*

Primeiramente, parte-se de uma variável aleatória $\mathbf{z}$, cuja distribuição $q(\boldsymbol{z})$ é conhecida e da qual pode-se obter amostras conforme desejado. Pode-se definir um modelo gerador mapeando $\mathbf{z}$ para $\mathbf{x}$ pela transformação $g_{\boldsymbol{\theta}} : \mathbb{R}^m \to \mathbb{R}^d$, a qual é parametrizada por $\boldsymbol{\theta}$. Dessa forma, tem-se que $\mathbf{x} \sim q(\boldsymbol{x};\boldsymbol{\theta})$, sendo essa distribuição definida, implicitamente, pelo mapa $\mathbf{z} \mapsto \mathbf{x} = g_{\boldsymbol{\theta}}(\mathbf{z})$. Esse processo é ilustrado na Figura 2.3.

É importante notar que esse modelo gerador é extremamente flexível, podendo ser classificado tanto como explícito ou implícito, em face da escolha, no momento arbitrária, de $\mathbf{z}$ e da transformação $g_{\boldsymbol{\theta}}(\boldsymbol{z})$. Essa flexibilidade tem como contra-peso a dificuldade em especificar explicitamente $q(\boldsymbol{x};\boldsymbol{\theta})$.

Da definição de distribuição transformada como a derivada da função distribuição acumulada [2, 7], tem-se que

$$q(\boldsymbol{x};\boldsymbol{\theta}) = \frac{\partial^d}{\partial x_1 \partial x_2 \dots \partial x_d}\int_{\{g_{\boldsymbol{\theta}}(\boldsymbol{z}) \leq \boldsymbol{x}\}} q(\boldsymbol{z})d\boldsymbol{z}. \tag{2.1}$$

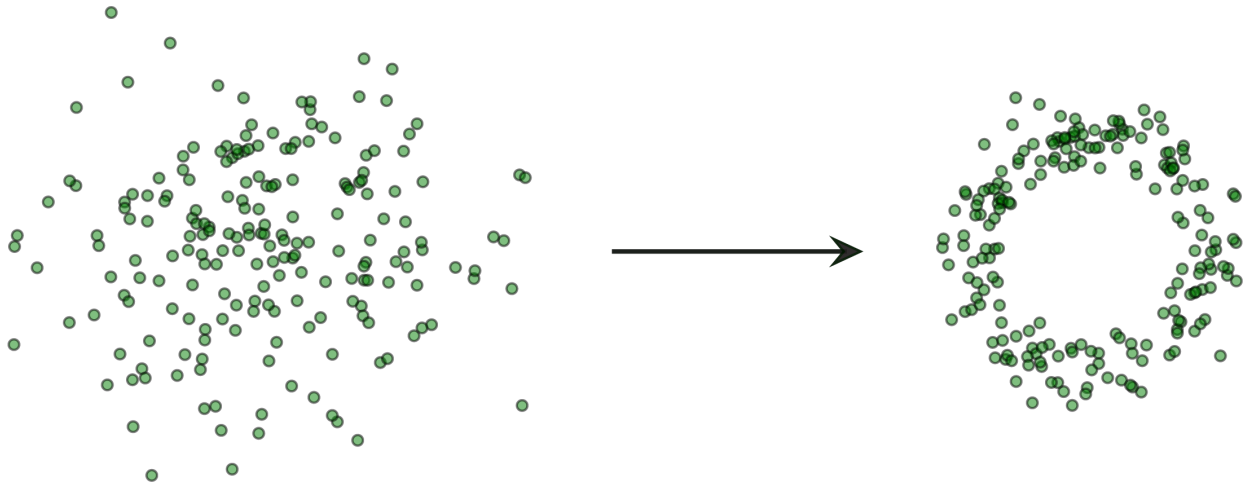

Figura 2.3: Pode-se entender o mapa $g_{\boldsymbol{\theta}}(\boldsymbol{z})$ como levando uma distribuição simples, $q(\boldsymbol{z})$, a qual se compreende bem e tem acesso ilimitado, para uma distribuição de maior complexidade, $q(\boldsymbol{x};\boldsymbol{\theta})$, a qual espera-se aproximar de $p(\boldsymbol{x})$. Essa ideia é ilustrada na figura acima, onde amostras de uma distribuição gaussiana bidimensional padrão são mapeadas, formando um anel, pela transformação $g(\boldsymbol{z}) = \boldsymbol{z}/3 + \boldsymbol{z}/||\boldsymbol{z}||$.

Nos casos de interesse para este trabalho, onde $g_{\boldsymbol{\theta}}$ é usualmente uma transformação não-linear altamente parametrizada, m e d grandes e $\mathbf{z}$ não tem, necessariamente, uma distribuição simples, costuma não ser possível especificar explicitamente $q(\boldsymbol{x};\boldsymbol{\theta})$ de forma útil, no sentido de ser possível extrair qualquer informação dela, mesmo que, como mostrado na Equação 2.1, seja possível defini-la matematicamente.

2.2.2 Ajuste de parâmetros

Como já mencionado, o problema da modelagem geradora, de um ponto de vista exclusivamente matemático, consiste de, definido um tipo de dado e o contexto sob o qual é obtido, encontrar, dentro de uma determinada família $\mathcal{Q} = \{q(\boldsymbol{x};\boldsymbol{\theta})\}_{\boldsymbol{\theta}\in\mathbb{R}^k}$ de distribuições parametrizadas, a que melhor representa sua distribuição geradora de dados $p(\boldsymbol{x})$. Em outras palavras, consiste de determinar $\boldsymbol{\theta} \in \mathbb{R}^k$ tal que $q(\boldsymbol{x};\boldsymbol{\theta})$ seja o mais próximo possível, sob algum critério, de $p(\boldsymbol{x})$.

Logicamente, deve-se especificar, formalmente, essa noção de proximidade ou o que se quer dizer com representar bem. Pode-se fazer isso introduzindo uma medida de desempenho para o modelo, dados parâmetros $\boldsymbol{\theta}$, que indique o quão bem ele representa $p(\boldsymbol{x})$. Ainda, essa medida deve ser numérica e objetiva, de forma a deixar claro, dados dois conjuntos de parâmetros diferentes, se um é melhor que o outro.

Pode-se representar essa medida de desempenho através de uma função de perda $\mathcal{L} : \mathbb{R}^k \to \mathbb{R}$, a qual atribui, para cada conjunto de parâmetros possível para o modelo, um número real estipulando o quão ruim o mesmo se encontra. Assim, dado $\boldsymbol{\theta}$, se o modelo apresenta um bom resultado, $\mathcal{L}(\boldsymbol{\theta})$ deve ser baixo e, no caso contrário, alto. Assim, pode-se aplicar métodos de otimização sob $\mathcal{L}(\boldsymbol{\theta})$, de forma a minimizar essa perda.

Exemplo 2.2.3. *Considere um modelo gerador $q(\boldsymbol{x};\boldsymbol{\theta})$, explícito e tratável, o qual deseja-se ajustar de forma a representar bem a distribuição geradora de dados $p(\boldsymbol{x})$. Uma função de perda interessante para ele é a divergência de Kullback-Leibler [2] entre $p(\boldsymbol{x})$ e $q(\boldsymbol{x};\boldsymbol{\theta})$, a qual consiste, de uma perspectiva de teoria da informação, do quanto de informação se perde quando se aproxima $p(\boldsymbol{x})$ por $q(\boldsymbol{x};\boldsymbol{\theta})$, trazendo consigo uma noção de comparação, desejada, entre essas duas distribuições. A mesma é definida por*

$$D_{KL}\left[p(\boldsymbol{x})||q(\boldsymbol{x};\boldsymbol{\theta})\right] = \mathbb{E}_{p(\boldsymbol{x})}\left[\log \frac{p(\mathbf{x})}{q(\mathbf{x};\boldsymbol{\theta})}\right] \tag{2.2}$$

e é sempre maior ou igual a zero, sendo a igualdade válida somente no caso de $p(\boldsymbol{x}) = q(\boldsymbol{x};\boldsymbol{\theta})$ em quase todos os pontos. Embora não seja possível calculá-la, visto que não se tem acesso a $p(\boldsymbol{x})$, pode-se estimá-la utilizando amostras dessa distribuição.

Primeiramente, deve-se reescrevê-la como

$$D_{KL}\left[p(\boldsymbol{x})||q(\boldsymbol{x};\boldsymbol{\theta})\right] = -\mathbb{E}_{p(\boldsymbol{x})}\left[\log q(\mathbf{x};\boldsymbol{\theta})\right] + \mathbb{E}_{p(\boldsymbol{x})}\left[\log p(\mathbf{x})\right].$$

Como o segundo termo não depende de $\boldsymbol{\theta}$, minimizar a divergência entre as duas distribuições ou só o primeiro termo, $-\mathbb{E}_{p(\mathbf{x})}\left[\log q(\mathbf{x};\boldsymbol{\theta})\right]$, tem o mesmo efeito. Dessa forma, pode-se definir uma nova função de perda para o modelo,

$$\mathcal{L}(\boldsymbol{\theta}) = -\mathbb{E}_{p(\boldsymbol{x})}\left[\log q(\mathbf{x};\boldsymbol{\theta})\right].$$

Embora pareça mais abordável do que a definida na Equação 2.2, o desconhecimento de $p(\boldsymbol{x})$ ainda não permite calculá-la. Por outro lado, assumindo que se tem acesso a um conjunto de dados $\mathbb{X}$ de N amostras i.i.d. de $p(\boldsymbol{x})$, pode-se aproximá-la por

$$\hat{\mathcal{L}}(\boldsymbol{\theta}) = -\frac{1}{N}\sum_{i=1}^{N}\log q(\boldsymbol{x}^{(i)};\boldsymbol{\theta}),$$

onde $\boldsymbol{x}^{(i)}$ denota a i-ésima amostra desse conjunto de dados, visto que, se $N \to \infty$, pela lei dos grandes números, tem-se que $\hat{\mathcal{L}}(\boldsymbol{\theta}) \to \mathcal{L}(\boldsymbol{\theta})$. Pode-se, ainda, interpretar essa aproximação da perda de outra forma, reescrevendo-a como

$$\begin{aligned}\hat{\mathcal{L}}(\boldsymbol{\theta}) &= -\frac{1}{N}\log\prod_{i=1}^{N} q(\boldsymbol{x}^{(i)};\boldsymbol{\theta}) \\ &= -\frac{\log q(\mathbb{X};\boldsymbol{\theta})}{N},\end{aligned}$$

sendo $q(\mathbb{X};\boldsymbol{\theta})$ a probabilidade atribuída pelo modelo ao conjunto de dados $\mathbb{X}$, como um todo. Essa probabilidade, nas áreas de estatística e aprendizado de máquina, é dada um nome especial, denominada verossimilhança de $\boldsymbol{\theta}$ dado $\mathbb{X}$, sendo bastante utilizada para inferência paramétrica [2].

Assim, pode-se minimizar a divergência de Kullback-Liebler entre as duas distribuições, como definida na Equação 2.2, minimizando $\hat{\mathcal{L}}(\boldsymbol{\theta})$, a qual se pode calcular facilmente, visto que o modelo é tratável. Ainda, se $q(\boldsymbol{x};\boldsymbol{\theta})$ é derivável em relação a $\boldsymbol{\theta}$, pode-se utilizar métodos de otimização baseados em gradientes no ajuste do modelo, o que acaba sendo, usualmente, a melhor opção quando se opta por métodos de otimização iterativos, como será discutido mais à frente do trabalho.

2.3 Redes Neurais

O modelo gerador foco do trabalho, como será discutido mais à frente, é definido a partir de um processo de amostragem similar ao mostrado no Exemplo 2.2.2. Assim, pode-se notar que o mesmo é construído sobre uma transformação parametrizada $g_{\boldsymbol{\theta}}(\boldsymbol{x})$ que mapeia uma distribuição simples, $q(\boldsymbol{z})$, para uma de maior complexidade, $q(\boldsymbol{x};\boldsymbol{\theta})$, a qual espera-se tornar o mais próxima possível da distribuição geradora de dados $p(\boldsymbol{x})$.

Sob a óptica desse modelo, partindo do pressuposto do total desconhecimento acerca do processo gerador a ser modelado, é interessante que essa transformação parametrizada seja extremamente flexível e tenha alta capacidade de representação. Dessa forma, assegura-se que o mesmo terá capacidade de representar satisfatoriamente uma vasta gama de processos geradores diferentes, sem impor restrições sobre estes.

Uma família de funções parametrizadas que usufrui dessa propriedade e vem sendo utilizada extensamente na comunidade de aprendizagem de máquinas é a das redes neurais. A palavra *rede* tem origem no fato dessas funções serem representadas através da composição de várias funções mais simples, chamadas de camadas. Essas camadas são tipicamente parametrizadas e acopladas sequencialmente umas às outras, formando, assim, uma cadeia de composições, da qual a rede consiste. Dessa forma, os parâmetros da rede consistem do agrupamento de todos os parâmetros de suas camadas individuais.

2.3.1 Organização em camadas

Como mencionado anteriormente, redes neurais podem ser vistas como funções de alta complexidade construídas a partir da composição de diversas funções mais simples, as quais são chamadas de camadas. Por exemplo, pode-se ter três funções, ou camadas, $f^{(1)}$, $f^{(2)}$ e $f^{(3)}$, acopladas sequencialmente formando uma rede neural de três camadas $f(\boldsymbol{x}) = f^{(3)} \circ f^{(2)} \circ f^{(1)}(\boldsymbol{x})$. Dessa forma, chama-se $f^{(1)}$ da primeira camada da rede, $f^{(2)}$ de segunda e $f^{(3)}$ de terceira.

O comprimento total dessa cadeia de composições é chamado de profundidade do modelo, estando intimamente relacionado com seu capacidade representacional. Quanto mais profunda é a rede, maior é seu poder, no sentido das funções que pode representar. Pode-se entender cada camada como construindo uma representação de maior complexidade sobre sua entrada, a qual, não levando em conta a primeira, consiste da saída da camada anterior [3, 8].

Dessa forma, de um ponto de vista representacional, pode-se interpretar essas várias camadas como criando representações intermediárias que servem como base para as camadas seguintes, que criam representações mais complexas baseadas nessas. Essa estrutura hierárquica de representações é vista em vários tipos de dados e a utilização de redes neurais profundas, nesses casos, é bastante interessante por introduzi-la diretamente na arquitetura modelo [3, 8]. A Figura 2.4 ilustra um caso tradicional onde essa organização em níveis de abstração é encontrada naturalmente.

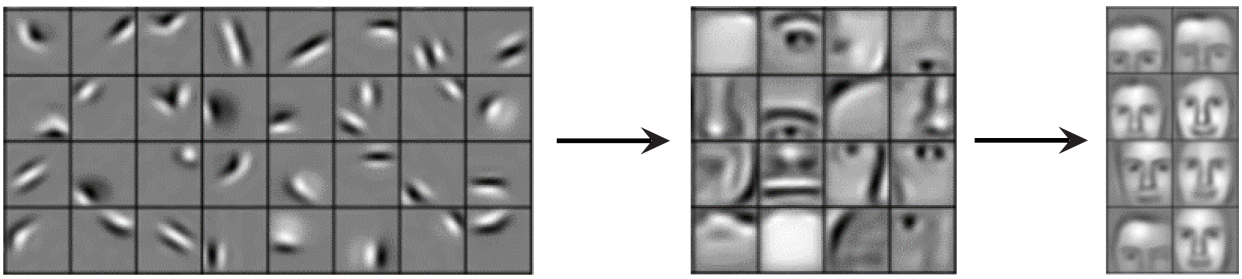

Figura 2.4: Imagens são dados que tipicamente apresentam uma estrutura hierárquica. Por exemplo, no caso de imagens de rostos de pessoas, como ilustrado acima [9], pode-se ver claramente que os mesmos são compostos por bocas, narizes, olhos, orelhas e outras estruturas mais simples, que por sua vez são formadas por estruturas ainda mais básicas, como texturas e contornos. A utilização de redes neurais profundas para trabalhar com esse tipo de dado é extremamente interessante, visto que essa estrutura já se faz presente no modelo, antes mesmo do contato dele com os dados.

2.3.2 Poder representacional

Como já discutido, a profundidade de uma rede neural é fator determinante de sua complexidade e de sua capacidade representacional. Há, ainda, um outro fator que tem importância na determinação desta última: a sua largura. A saída de cada uma de suas camadas tem, tipicamente, uma medida de dimensionalidade associada. No caso de saídas vetoriais, essa medida é exatamente a dimensionalidade do vetor. Em redes neurais convolucionais, utilizadas para o processamento de imagens [3, 10], há camadas com saídas tensoriais tridimensionais, sendo essa medida a profundidade do tensor.

Há muitos teoremas que tratam do poder representacional das mais variadas redes neurais com as mais diversas arquiteturas possíveis [3]. Embora relevantes, não serão tratados a fundo aqui, por questões de escopo. Por outro lado, a conclusão geral que se tira destes, de forma simplificada, é bastante interessante: desde que a rede neural seja profunda e larga o bastante, a mesma é capaz de representar satisfatoriamente qualquer que seja a função que se deseja aproximar.

Por um lado, isso é reconfortante, visto que deixa claro o poder representacional desses modelos, mas é importante notar que não garante que, de fato, será possível aproximar essa função. É necessário, ainda, encontrar um conjunto de parâmetros que aproxima satisfatoriamente a função desejada, não sendo certo que o algoritmo de otimização utilizado irá achá-lo. Dessa forma, embora tenham alto poder representacional, não se pode afirmar que irão, na prática, representar satisfatoriamente qualquer função escolhida.

Por esse motivo, o processo de ajuste de modelos baseados em redes neurais tem, usualmente, um caráter extremamente experimental de tentativa e erro, não havendo uma regra clara e única que permita realizá-lo satisfatoriamente em todos os casos. Afortunadamente, a comunidade de redes neurais vem desenvolvendo, há bastante tempo, uma série de heurísticas e truques que auxiliam no ajuste desses modelos, tornando-o extremamente mais fácil e eficaz [3, 11–13].

2.3.3 Algumas considerações

Como supracitado, há a necessidade de transformações parametrizadas altamente flexíveis para a implementação do modelo gerador desejado. Juntamente com a extensa pesquisa existente em torno de redes neurais, a qual levou ao desenvolvimento de diversas heurísticas e técnicas que permitem um ajuste eficaz de modelos baseados nelas, seu alto poder representacional as tornam uma ótima escolha para suprir essa necessidade.

Embora o funcionamento e formulação das mesmas tenham sido descritos, em parte, há também uma série de outros fatores que foram deixados de lado e são relevantes para sua compreensão [3]. Não foram tratadas das camadas em si, que são as peças fundamentais sobre as quais essas redes são construídas. Ainda, há diversos outros aspectos técnicos relacionados à otimização desses modelos, ao processo de determinação de suas arquiteturas, consistindo da organização de suas camadas e escolhas das mesmas, entre outros, que foram deixados em segundo plano.

Apesar de serem parte essencial para a implementação do modelo gerador desejado, escolheu-se tratar redes neurais apenas como ferramentas para tal. Por trazerem consigo um enorme arcabouço de técnicas e variações existentes, um tratamento adequado das mesmas requer um enfoque em nível muito acima do desejado pelo escopo do trabalho. Ademais, levando em conta a existência de diversas fontes que podem ser facilmente acessadas e discutem esse tema na profundidade que merece [2, 3], se considera desnecessário um tratamento mais aprofundado.

É importante notar que, embora redes neurais sejam utilizadas no modelo gerador implementado, este independe totalmente das mesmas em sua construção, como será evidenciado no capítulo seguinte. Portanto, não se faz necessária, em nenhum momento, a compreensão a fundo dos fundamentos das mesmas, além do básico apresentado acima, para o entendimento do trabalho.

Capítulo 3

REDES GERADORAS ADVERSÁRIAS

O foco deste trabalho é na classe de modelos geradores conhecida como redes geradoras adversárias ou, abreviadamente, GANs [1]. Consistem de modelos geradores implícitos, sendo definidos a partir de um processo de amostragem similar ao ilustrado no Exemplo 2.2.2. O diferencial, nessa classe de modelos, é a introdução de um modelo secundário, discriminador, o qual tem função exclusiva de auxiliar no processo de ajuste de parâmetros do modelo gerador. Naturalmente, esse modelo discriminador tem seus próprios parâmetros, os quais, como será mostrado a seguir, são ajustados de forma adversária aos do modelo gerador.

3.1 Construção do modelo

Como supracitado, GANs são definidas através de um processo de amostragem similar ao ilustrado no Exemplo 2.2.2. Parte-se de uma variável aleatória $\mathbf{z}$, sob $\mathbb{R}^m$, cuja distribuição $q(\boldsymbol{z})$ é conhecida e da qual pode-se obter amostras conforme desejado. Dada uma transformação $g_{\boldsymbol{\theta}} : \mathbb{R}^m \to \mathbb{R}^d$, parametrizada por $\boldsymbol{\theta}$, a variável aleatória $\mathbf{x}$, representando os dados gerados, é definida pelo mapa $\mathbf{z} \mapsto \mathbf{x} = g_{\boldsymbol{\theta}}(\mathbf{z})$, sendo distribuída de acordo com $q(\boldsymbol{x}; \boldsymbol{\theta})$, definida implicitamente por esse processo.

A partir de agora, por motivos de simplicidade, entende-se que, quando se fala de transformações parametrizadas, as mesmas sejam empregadas utilizando redes neurais. Como descrito no capítulo anterior, redes neurais tem alto poder representacional, de forma que $g_{\boldsymbol{\theta}}$ pode se configurar em uma vasta gama de funções diferentes, tornando o modelo gerador altamente flexível.

3.2 Funções de perda

Como já mencionado, é necessária uma escolha sensata de parâmetros para que o modelo gerador represente bem a distribuição geradora de dados. Essa noção do que é representar bem pode ser capturada por uma medida de desempenho posta sobre o modelo. Normalmente, essa medida vem na forma de uma função de perda, a qual representa, numericamente, o quão ruim o mesmo se encontra para um dado conjunto de parâmetros $\boldsymbol{\theta}$.

Pela construção do modelo, não é possível determinar o valor de $q(\boldsymbol{x}; \boldsymbol{\theta})$ para pontos arbitrários, de forma que não se pode utilizar a abordagem empregada na definição da função de perda do Exemplo 2.2.3 ou qualquer outra que necessite disso. A única informação que pode ser extraída de GANs são as amostras produzidas por elas, de forma que deve-se explorar isso na definição de sua perda.

3.2.1 Modelo gerador

Considere um cenário hipotético no qual é escolhida uma distribuição, entre $p(\boldsymbol{x})$ e $q(\boldsymbol{x}; \boldsymbol{\theta})$, e, em seguida, é retirada uma amostra da mesma. Essa amostra é disponibilizada a um observador que tenta determinar qual das duas distribuições deu origem a ela. Se as duas são parecidas, então ele deve ter dificuldade nessa tarefa, visto que, supostamente, devem gerar amostras parecidas.

Assim, analisando esse cenário do ponto de vista do modelo gerador, é interessante que esse observador obtenha um baixo desempenho em distinguir entre as duas distribuições baseado em amostras das mesmas.

Naturalmente, deve-se formalizar esse cenário, de um ponto de vista matemático, para que o mesmo seja de algum uso na definição da perda do modelo. Representa-se essa amostra por uma variável aleatória $\mathbf{x}$ e a distribuição escolhida, da qual foi retirada, pela variável aleatória y, valendo 1, para $p(\boldsymbol{x})$, com probabilidade π, e 0, para $q(\boldsymbol{x};\boldsymbol{\theta})$, com probabilidade $1-\pi$. Dessa forma, define-se a distribuição conjunta $r(\boldsymbol{x}, y)$ entre essas duas variáveis, codificando esse cenário, por

$$
\begin{aligned}
&r(y) = \pi, \text{ se } y = 1, \text{ e } 1-\pi, \text{ se } y = 0,\\
&r(\boldsymbol{x}|y) = p(\boldsymbol{x}), \text{ se } y = 1, \text{ e } q(\boldsymbol{x};\boldsymbol{\theta}), \text{ se } y = 0, \text{ e}\\
&r(\boldsymbol{x}, y) = r(\boldsymbol{x}|y)r(y).
\end{aligned}
$$

Ainda, é necessário representar, matematicamente, o que se quer dizer com esse observador ter dificuldade em distinguir entre as duas distribuições. Pode-se formalizar isso, com base em teoria da informação [14], a partir da diferença entre a informação contida na variável aleatória y antes e depois de observada $\mathbf{x}$, a qual é também chamada de informação mútua entre as duas variáveis. Essa medida representa o quanto de informação $\mathbf{x}$ fornece acerca de y, de forma que, quanto maior for, mais fácil é determinar a segunda observando a primeira ou, em outras palavras, determinar qual distribuição deu origem à amostra.

Dessa forma, essa medida traz consigo a noção desejada do cenário hipotético descrita anteriormente, na forma matemática de uma função de perda. Portanto, define-se como a perda do modelo gerador

$$
\begin{aligned}
\mathcal{L}_g(\boldsymbol{\theta}) &= H(\mathrm{y}) - H(\mathrm{y}|\mathbf{x})\\
&= -\mathbb{E}_{r(y)}\left[\log r(\mathrm{y})\right] + \mathbb{E}_{r(\boldsymbol{x},y)}\left[\log r(\mathrm{y}|\mathbf{x})\right].
\end{aligned}
$$

Proposição 1. *A perda do modelo gerador tem $\boldsymbol{\theta} \in \mathbb{R}^k$ como mínimo global se, e somente se, $q(\boldsymbol{x};\boldsymbol{\theta}) = p(\boldsymbol{x})$ em quase todos os pontos.*

Prova. Aplicando a regra de Bayes, tem-se que $r(y|\boldsymbol{x}) = r(\boldsymbol{x}|y)r(y)/r(\boldsymbol{x})$. Assim, pode-se reescrever a perda do modelo gerador como

$$
\begin{aligned}
\mathcal{L}_g(\boldsymbol{\theta}) &= -\mathbb{E}_{r(y)}\left[\log r(\mathrm{y})\right] + \mathbb{E}_{r(\boldsymbol{x},y)}\left[\log \frac{r(\mathbf{x}|y)r(\mathrm{y})}{r(\mathbf{x})}\right]\\
&= \cancel{-\mathbb{E}_{r(y)}\left[\log r(\mathrm{y})\right]} + \mathbb{E}_{r(\boldsymbol{x},y)}\left[\log \frac{r(\mathbf{x}|y)}{r(\mathbf{x})}\right] + \cancel{\mathbb{E}_{r(\boldsymbol{x},y)}\left[\log r(\mathrm{y})\right]}\\
&= r(\mathrm{y}=1)\mathbb{E}_{r(\boldsymbol{x}|\mathrm{y}=1)}\left[\log \frac{r(\mathbf{x}|\mathrm{y}=1)}{r(\mathbf{x})}\right] + r(\mathrm{y}=0)\mathbb{E}_{r(\boldsymbol{x}|\mathrm{y}=0)}\left[\log \frac{r(\mathbf{x}|\mathrm{y}=0)}{r(\mathbf{x})}\right]\\
&= \pi\mathbb{E}_{p(\boldsymbol{x})}\left[\log \frac{p(\mathbf{x})}{\pi p(\mathbf{x}) + (1-\pi)q(\mathbf{x};\boldsymbol{\theta})}\right]\\
&\qquad + (1-\pi)\mathbb{E}_{q(\boldsymbol{x};\boldsymbol{\theta})}\left[\log \frac{q(\mathbf{x};\boldsymbol{\theta})}{\pi p(\mathbf{x}) + (1-\pi)q(\mathbf{x};\boldsymbol{\theta})}\right]
\end{aligned}
$$

$$
\begin{aligned}
&= \pi D_{KL}\left[p(\boldsymbol{x})||\pi p(\boldsymbol{x}) + (1-\pi)q(\boldsymbol{x};\boldsymbol{\theta})\right] \\
&\qquad\qquad + (1-\pi)D_{KL}\left[q(\boldsymbol{x};\boldsymbol{\theta})||\pi p(\boldsymbol{x}) + (1-\pi)q(\boldsymbol{x};\boldsymbol{\theta})\right] \\
&= JS_{\pi}\left[p(\boldsymbol{x})||q(\boldsymbol{x};\boldsymbol{\theta})\right],
\end{aligned}
$$

sendo o último termo a divergência de Jensen-Shannon generalizada entre $p(\boldsymbol{x})$ e $q(\boldsymbol{x};\boldsymbol{\theta})$, parametrizada por π, a qual é minimizada, se igualando a zero, se, e somente se, $p(\boldsymbol{x}) = q(\boldsymbol{x};\boldsymbol{\theta})$ em quase todos os pontos. □

Como se pode notar na prova da Proposição 1, a perda do modelo gerador consiste da divergência de Jensen-Shannon generalizada entre as distribuições, sendo seu parâmetro dado por π. Apesar de apresentar propriedades teóricas interessantes, as quais serão exploradas mais à frente, não há, da forma como está escrita, como calcular a mesma ou pelo menos seus gradientes em relação a $\boldsymbol{\theta}$, o que é necessário para o ajuste do modelo.

Visto que $H(\mathrm{y})$, a entropia de y ou o tanto de informação que a mesma contém, é constante em função de $\boldsymbol{\theta}$, pode-se ignorar esse termo na minimização da perda do modelo gerador, possibilitando reescrevê-la como

$$
\begin{aligned}
\mathcal{L}_g(\boldsymbol{\theta}) &= \mathbb{E}_{r(\boldsymbol{x},y)}\left[\log r(\mathrm{y}|\mathbf{x})\right] \\
&= r(\mathrm{y}=1)\mathbb{E}_{r(\boldsymbol{x}|\mathrm{y}=1)}\left[\log r(\mathrm{y}=1|\mathbf{x})\right] + r(\mathrm{y}=0)\mathbb{E}_{r(\boldsymbol{x}|\mathrm{y}=0)}\left[\log r(\mathrm{y}=0|\mathbf{x})\right] \\
&= \pi\mathbb{E}_{p(\boldsymbol{x})}\left[\log r(\mathrm{y}=1|\mathbf{x})\right] + (1-\pi)\mathbb{E}_{q(\boldsymbol{x};\boldsymbol{\theta})}\left[\log(1 - r(\mathrm{y}=1|\mathbf{x}))\right].
\end{aligned}
$$

Ainda assim, não é possível determinar $\mathcal{L}_g(\boldsymbol{\theta})$ por dois problemas: (1) as expectâncias sobre $p(\boldsymbol{x})$ e $q(\boldsymbol{x};\boldsymbol{\theta})$; (2) o termo $r(\mathrm{y}=1|\boldsymbol{x})$, o qual requer, para seu cálculo, o conhecimento dessas duas distribuições. Entretanto, há como contorná-los com o uso de aproximações. O primeiro é simples, visto que pode ser utilizada a mesma abordagem do Exemplo 2.2.3, estimando as expectâncias através de amostras das distribuições. O segundo, como será visto a seguir, requer a introdução de um modelo auxiliar para estimar $r(\mathrm{y}=1|\boldsymbol{x})$.

3.2.2 Modelo discriminador

Como mencionado anteriormente, no ajuste dos parâmetros de GANs, é empregado um modelo auxiliar discriminador, cujo papel é, exatamente, estimar $r(\mathrm{y}=1|\boldsymbol{x})$. Pode-se defini-lo por uma transformação $d_{\boldsymbol{\phi}} : \mathbb{R}^d \to [0,1]$, parametrizada por $\boldsymbol{\phi} \in \mathbb{R}^l$, que, dada uma amostra $\boldsymbol{x} \in \mathbb{R}^d$, atribui a ela uma probabilidade da mesma ter vindo de $p(\boldsymbol{x})$, de preferência próxima da real, dada por $r(\mathrm{y}=1|\boldsymbol{x})$.

Logicamente, esse modelo precisa, também, de um ajuste de parâmetros para representar corretamente $r(\mathrm{y}=1|\boldsymbol{x})$. Por sorte, pode-se utilizar quase que exatamente o contrário da perda do modelo gerador para isso, com apenas algumas pequenas alterações, a qual é dada por

$$
\mathcal{L}_d(\boldsymbol{\theta},\boldsymbol{\phi}) = -\pi\mathbb{E}_{p(\boldsymbol{x})}\left[\log d_{\boldsymbol{\phi}}(\mathbf{x})\right] - (1-\pi)\mathbb{E}_{q(\boldsymbol{x};\boldsymbol{\theta})}\left[\log(1 - d_{\boldsymbol{\phi}}(\mathbf{x}))\right].
$$

É importante notar que essa perda é função tanto dos parâmetros do modelo discriminador, quanto do gerador, visto que contém uma expectância sobre $q(\boldsymbol{x};\boldsymbol{\theta})$.

Proposição 2. *Para $\boldsymbol{\theta}$ fixo, a perda do modelo discriminador tem $\boldsymbol{\phi} \in \mathbb{R}^l$ como mínimo global se, e somente se, $d_{\boldsymbol{\phi}}(\boldsymbol{x}) = r(\mathrm{y} = 1|\boldsymbol{x})$ em quase todos os pontos.*

Prova. Define-se como $s(y|\boldsymbol{x}; \boldsymbol{\phi})$ a distribuição induzida pelo modelo discriminador, especificada por $s(\mathrm{y} = 1|\boldsymbol{x}; \boldsymbol{\phi}) = d_{\boldsymbol{\phi}}(\boldsymbol{x})$ e $s(\mathrm{y} = 0|\boldsymbol{x}; \boldsymbol{\phi}) = 1 - d_{\boldsymbol{\phi}}(\boldsymbol{x})$. Assim, pode-se reescrever sua perda como

$$\begin{aligned}
\mathcal{L}_d(\boldsymbol{\theta}, \boldsymbol{\phi}) &= -\pi \mathbb{E}_{p(\boldsymbol{x})} \left[\log s(\mathrm{y} = 1|\mathbf{x}; \boldsymbol{\phi})\right] - (1 - \pi) \mathbb{E}_{q(\boldsymbol{x};\boldsymbol{\theta})} \left[\log s(\mathrm{y} = 0|\mathbf{x}; \boldsymbol{\phi})\right] \\
&= -\mathbb{E}_{r(\boldsymbol{x},y)} \left[\log s(\mathrm{y}|\mathbf{x}; \boldsymbol{\phi})\right] \\
&= \mathbb{E}_{r(\boldsymbol{x})} \left[\mathbb{E}_{r(y|\mathbf{x})} \left[\log \frac{r(\mathrm{y}|\mathbf{x})}{s(\mathrm{y}|\mathbf{x}; \boldsymbol{\phi})}\right]\right] - \mathbb{E}_{r(\boldsymbol{x},y)} \left[\log r(\mathrm{y}|\mathbf{x})\right] \\
&= \mathbb{E}_{r(\boldsymbol{x})} \left[D_{KL} \left[r(y|\mathbf{x})||s(y|\mathbf{x}; \boldsymbol{\phi})\right]\right] + C,
\end{aligned}$$

sendo C constante em função de $\boldsymbol{\phi}$ e, portanto, irrelevante e descartável para o problema de otimização em foco. Desse modo, é necessário minimizar apenas o primeiro termo, visto que é o único sob o qual os parâmetros do modelo discriminador tem influência.

Defina $\mathcal{X}$ como o suporte de $r(\boldsymbol{x})$ em $\mathbb{R}^d$, i.e., o subconjunto de $\mathbb{R}^d$ no qual $r(\boldsymbol{x})$ é positivo, e $\mathcal{X}'$ como o conjunto de pontos em $\mathcal{X}$ nos quais $d_{\boldsymbol{\phi}}(\boldsymbol{x}) \neq r(\mathrm{y} = 1|\boldsymbol{x})$, de forma que $D_{KL} \left[r(y|\boldsymbol{x})||s(y|\boldsymbol{x}; \boldsymbol{\phi})\right] > 0$ para qualquer $\boldsymbol{x} \in \mathcal{X}'$. Assim, $D_{KL} \left[r(y|\boldsymbol{x})||s(y|\boldsymbol{x}; \boldsymbol{\phi})\right] r(\boldsymbol{x})$, é estritamente positivo em $\mathcal{X}'$, logo

$$\begin{aligned}
\mathbb{E}_{r(\boldsymbol{x})} \left[D_{KL} \left[r(y|\mathbf{x})||s(y|\mathbf{x}; \boldsymbol{\phi})\right]\right] &= \int_{\mathcal{X}} D_{KL} \left[r(y|\boldsymbol{x})||s(y|\boldsymbol{x}; \boldsymbol{\phi})\right] r(\boldsymbol{x}) d\boldsymbol{x} \\
&= \int_{\mathcal{X}'} D_{KL} \left[r(y|\boldsymbol{x})||s(y|\boldsymbol{x}; \boldsymbol{\phi})\right] r(\boldsymbol{x}) d\boldsymbol{x} \\
&= 0
\end{aligned}$$

se, e somente se, $\int_{\mathcal{X}'} d\boldsymbol{x} = 0$, ou, em outras palavras, se $d_{\boldsymbol{\phi}}(\boldsymbol{x}) = r(\mathrm{y} = 1|\boldsymbol{x})$ em quase todos os pontos. □

3.3 Otimização das funções de perda

Antes de continuar manipulando a perda do modelo gerador, moldando-a de forma a permitir seu cálculo, é interessante fazer algumas observações quanto à sua otimização, a qual é o objetivo final a ser, de fato, alcançado. Como mencionando anteriormente, assume-se que as transformações $g_{\boldsymbol{\theta}}(\boldsymbol{x})$ e $d_{\boldsymbol{\phi}}(\boldsymbol{x})$ sejam empregadas por redes neurais, devido à flexibilidade e alta capacidade de representação desejada para ambos os modelos, o que impõe algumas restrições quanto aos seus processos de otimização.

3.3.1 Considerações práticas

Devido à característica extremamente não-linear e ao gritante número de parâmetros envolvidos, o uso de redes neurais usualmente leva a problemas de otimização não-convexos que requerem

métodos iterativos, não havendo soluções fechadas ou únicas e quase que garantidamente convergindo para mínimos locais [3]. De um ponto de vista teórico, isso não é interessante, visto que esses mínimos locais não fornecem as garantias desejadas quanto ao desempenho desses modelos.

Por outro lado, empiricamente, constata-se que isso não é prejudicial, visto que, grosseiramente falando, a qualidade desses mínimos locais é, no geral, boa [3,15]. Dessa forma, embora os métodos de otimização iterativos utilizados levem, quase que sempre, a mínimos locais, pode-se tratar estes como mínimos globais sem maiores preocupações.

Métodos de otimização iterativos podem ser categorizados pela informação da função a ser otimizada, também chamada de objetivo, que os mesmos utilizam. Com base na complexidade computacional envolvida e na taxa de convergência, no geral, dos mesmos, pode-se ordená-los, crescentemente, partindo de métodos que utilizam apenas o valor do objetivo, métodos que utilizam os gradientes do mesmo e métodos que utilizam suas hessianas.

Em problemas envolvendo redes neurais, devido à exorbitante quantidade de parâmetros, métodos que se baseiam somente em valores do objetivo são inviáveis pela taxa de convergência extremamente baixa. No outro extremo, métodos que envolvem hessianas tem custo computacional tão alto que são impraticáveis. Portanto, limita-se a métodos que utilizam apenas informação dos gradientes das funções sendo otimizadas, cujo cálculo é, usualmente, factível.

Dessa forma, não se faz necessário calcular a perda dos modelos, em si, para ajustá-los, e sim os gradientes das mesmas em relação aos seus parâmetros. Portanto, muda-se o foco, a partir de agora, em determinar essas perdas, para determinar os gradientes das mesmas.

Há uma série de métodos de otimização baseados em gradientes disponíveis para uso [3] a comparação dos mesmos e eventual escolha de um não é interessante, neste ponto do trabalho. Consequentemente, não será especificado, agora, nenhum método em particular, de forma que o problema de otimização em foco se estende somente até a determinação dos gradientes da funções a serem otimizadas, as quais consistem das perdas dos modelos.

3.3.2 Gradientes

Como argumentado na subseção anterior, o problema em foco, a partir de agora, é a determinação do gradiente da perda do modelo gerador em relação aos seus parâmetros, dado por $\nabla_{\boldsymbol{\theta}}\mathcal{L}_g(\boldsymbol{\theta})$. Primeiramente, assumindo que o modelo discriminador tem capacidade de representação suficiente para modelar $r(\mathrm{y}=1|\boldsymbol{x})$, tem-se que

$$d_{\boldsymbol{\phi}^*(\boldsymbol{\theta})}(\boldsymbol{x}) = r(\mathrm{y}=1|\boldsymbol{x}), \text{ sendo } \boldsymbol{\phi}^*(\boldsymbol{\theta}) = \operatorname*{argmin}_{\boldsymbol{\phi}} \mathcal{L}_d(\boldsymbol{\theta}, \boldsymbol{\phi}).$$

É interessante observar que os parâmetros ótimos para o modelo discriminador são dependentes do modelo gerador, justificando a escrita, pelo menos no momento, de $\boldsymbol{\phi}^*(\boldsymbol{\theta})$ ao invés de somente $\boldsymbol{\phi}^*$. Em seguida, pode-se escrever o gradiente desejado como

$$\begin{aligned}\nabla_{\boldsymbol{\theta}}\mathcal{L}_g(\boldsymbol{\theta}) &= \pi\nabla_{\boldsymbol{\theta}}\mathbb{E}_{p(\boldsymbol{x})}\left[\log d_{\boldsymbol{\phi}^*(\boldsymbol{\theta})}(\mathbf{x})\right] + (1-\pi)\nabla_{\boldsymbol{\theta}}\mathbb{E}_{q(\boldsymbol{x};\boldsymbol{\theta})}\left[\log(1-d_{\boldsymbol{\phi}^*(\boldsymbol{\theta})}(\mathbf{x}))\right] \\ &= \pi\mathbb{E}_{p(\boldsymbol{x})}\left[\nabla_{\boldsymbol{\theta}}\log d_{\boldsymbol{\phi}^*(\boldsymbol{\theta})}(\mathbf{x})\right] + (1-\pi)\mathbb{E}_{q(\boldsymbol{z})}\left[\nabla_{\boldsymbol{\theta}}\log(1-d_{\boldsymbol{\phi}^*(\boldsymbol{\theta})}(g_{\boldsymbol{\theta}}(\mathbf{z})))\right].\end{aligned}$$

Sua determinação envolve dois obstáculos: (1) as expectâncias sobre $p(\boldsymbol{x})$ e $q(\boldsymbol{z})$, as quais requerem a resolução de integrais sobre quantidades bastante complexas; (2) o cálculo dos gradientes das quantidades dentro das expectâncias, as quais são definidas com base em $\boldsymbol{\phi}^*(\boldsymbol{\theta}) = \operatorname{argmin}_{\boldsymbol{\phi}} \mathcal{L}_d(\boldsymbol{\theta}, \boldsymbol{\phi})$. O primeiro problema, assim como no Exemplo 2.2.3, pode ser solucionado facilmente estimando as expectâncias com amostras de $p(\boldsymbol{x})$ e $q(\boldsymbol{z})$. O segundo, por outro lado, requer algumas premissas.

Assim sendo, assume-se que $\boldsymbol{\phi}^*(\boldsymbol{\theta})$ varia suavemente o bastante, para pequenas variações $\boldsymbol{\theta}$, de forma que pode-se considerar seu gradiente, em relação a este último, nulo. É importante notar que isso não é uma premissa absurda, sendo inclusive uma propriedade desejada, visto que uma pequena mudança no modelo gerador não deve resultar em mudanças bruscas no modelo discriminador ótimo para ele. Assim, para fim de cálculos de gradientes que o envolvem, pode-se considerá-lo constante em função de $\boldsymbol{\theta}$, motivando sua reescrita como $\boldsymbol{\phi}^*$.

Com essa nova premissa, pode-se aproximar o gradiente da perda do modelo gerador por

$$\nabla_{\boldsymbol{\theta}} \mathcal{L}_g(\boldsymbol{\theta}) \approx (1-\pi)\mathbb{E}_{q(\boldsymbol{z})} \left[\nabla_{\boldsymbol{\theta}} \log(1 - d_{\boldsymbol{\phi}^*}(g_{\boldsymbol{\theta}}(\mathbf{z})))\right],$$

onde o primeiro termo se iguala a zero por não ser dependente de $\boldsymbol{\theta}$. O gradiente contido dentro da expectância não será determinado, mais especificamente, visto que não há necessidade para tal, já que é extremamente simples e necessita apenas da aplicação da regra da cadeia. Por outro lado, ainda deve-se resolver o problema da expectância sobre $q(\boldsymbol{z})$.

Como, pela construção do modelo, é possível amostrar, irrestritamente, $q(\boldsymbol{z})$, pode-se obter, conforme necessário, conjuntos da forma $\mathbb{Z} = \{\boldsymbol{z}^{(1)}, \boldsymbol{z}^{(2)}, \ldots, \boldsymbol{z}^{(N)}\}$, consistindo de N amostras i.i.d. da mesma. Assim, pode-se aproximar o gradiente, indefinidamente, utilizando essas amostras, por

$$\nabla_{\boldsymbol{\theta}} \mathcal{L}_g(\boldsymbol{\theta}) \approx \frac{1-\pi}{N} \sum_{i=1}^{N} \nabla_{\boldsymbol{\theta}} \log(1 - d_{\boldsymbol{\phi}^*}(g_{\boldsymbol{\theta}}(\boldsymbol{z}^{(i)}))).$$

Pode-se observar que, para utilizar essa aproximação, deve-se ter acesso ao modelo discriminador otimizado, para um dado $\boldsymbol{\theta}$. Assim sendo, deve-se calcular, também, os gradientes de sua perda em relação aos seus parâmetros, $\nabla_{\boldsymbol{\phi}} \mathcal{L}_d(\boldsymbol{\theta}, \boldsymbol{\phi})$, de forma a ser possível ajustá-lo. Aplicando, de certo modo, as mesmas manipulações descritas anteriormente para se obter uma aproximação calculável do gradiente da perda do modelo gerador, tem-se que

$$\begin{aligned}
\nabla_{\boldsymbol{\phi}} \mathcal{L}_d(\boldsymbol{\theta}, \boldsymbol{\phi}) &= -\pi\mathbb{E}_{p(\boldsymbol{x})} \left[\nabla_{\boldsymbol{\phi}} \log d_{\boldsymbol{\phi}}(\mathbf{x})\right] - (1-\pi)\mathbb{E}_{q(\boldsymbol{x};\boldsymbol{\theta})} \left[\nabla_{\boldsymbol{\phi}} \log(1 - d_{\boldsymbol{\phi}}(\mathbf{x}))\right] \\
&\approx -\frac{\pi}{N_p} \sum_{i=1}^{N_p} \nabla_{\boldsymbol{\phi}} \log d_{\boldsymbol{\phi}}(\boldsymbol{x}_p^{(i)}) - \frac{1-\pi}{N_q} \sum_{i=1}^{N_q} \nabla_{\boldsymbol{\phi}} \log(1 - d_{\boldsymbol{\phi}}(\boldsymbol{x}_q^{(i)})),
\end{aligned}$$

sendo $\mathbb{X}_p = \{\boldsymbol{x}_p^{(1)}, \ldots, \boldsymbol{x}_p^{(N_p)}\}$ um conjunto de N_p amostras i.i.d. de $p(\boldsymbol{x})$ e $\mathbb{X}_q = \{\boldsymbol{x}_q^{(1)}, \ldots, \boldsymbol{x}_q^{(N_q)}\}$ um conjunto de N_q amostras i.i.d. de $q(\boldsymbol{x}; \boldsymbol{\theta})$, utilizados na aproximação das expectâncias sobre as duas distribuições.

3.3.3 Algoritmo para otimização

Da forma como foram desenvolvidos os métodos para se calcular, mesmo que aproximadamente, os gradientes $\nabla_{\boldsymbol{\theta}} \mathcal{L}_g(\boldsymbol{\theta})$ e $\nabla_{\boldsymbol{\phi}} \mathcal{L}_d(\boldsymbol{\theta}, \boldsymbol{\phi})$, chega-se, naturalmente, num algoritmo para otimização de

ambos os modelos. Antes, é importante ressaltar que, embora fique parecendo que se busca a otimização dos dois, na verdade, o foco é no modelo gerador, sendo o discriminador otimizado somente para possibilitar o cálculo dos gradientes da perda do primeiro.

No momento, espera-se, intuitivamente, o seguinte algoritmo para ajuste desses, descrito informalmente: (1) otimiza-se o modelo discriminador o máximo possível com o uso dos gradientes de sua perda, de forma que o mesmo seja ótimo para um dado $\boldsymbol{\theta}$; (2) atualiza-se $\boldsymbol{\theta}$, através de uma iteração do método de otimização sendo utilizado, com o gradiente da perda do modelo gerador, o qual requer o modelo discriminador otimizado para seu cálculo.

Embora teoricamente embasada, essa abordagem introduz um problema, do ponto de vista prático: é inviável realizar a otimização completa do modelo discriminador a cada iteração do processo de otimização do modelo gerador, devido ao custo computacional envolvido e simples questões de tempo. Por sorte, voltando à premissa que o modelo discriminador ótimo varia suavemente o bastante para pequenas variações de $\boldsymbol{\theta}$, não há necessidade de otimizá-lo exaustivamente a cada atualização deste.

Dessa forma, assumindo que cada iteração do processo de otimização do modelo gerador corresponda a pequenas variações de $\boldsymbol{\theta}$, são realizadas apenas algumas iterações do processo de otimização do modelo discriminador. Com essa abordagem, tenta-se manter o modelo discriminador sempre próximo do ótimo, o qual deve se deslocar pouco a cada atualização de $\boldsymbol{\theta}$. Formalmente, esse processo é representado no Algoritmo 1, sendo *opt* uma função que toma parâmetros e gradientes como argumentos e retorna os valores atualizados desses parâmetros, representando o método de otimização utilizado, e, para cada atualização do modelo gerador, são realizadas k iterações do modelo discriminador.

3.4 Influência de π

Voltando à Proposição 1, pode-se notar que, assumindo que as premissas introduzidas ao longo do capítulo são verdadeiras, o Algoritmo 1 minimiza a divergência de Jensen-Shannon generalizada, parametrizada por π, entre $p(\boldsymbol{x})$ e $q(\boldsymbol{x};\boldsymbol{\theta})$. Dessa forma, como a mesma só é nula para $p(\boldsymbol{x}) = q(\boldsymbol{x};\boldsymbol{\theta})$ em quase todos os pontos, vê-se que, de fato, minimizá-la leva ao resultado desejado e, ainda, não depende do valor de π para isso. Pode-se escrevê-la como

$$JS_\pi\left[p(\boldsymbol{x})||q(\boldsymbol{x};\boldsymbol{\theta})\right] = \pi D_{KL}\left[p(\boldsymbol{x})||\pi p(\boldsymbol{x}) + (1-\pi)q(\boldsymbol{x};\boldsymbol{\theta})\right] + (1-\pi)D_{KL}\left[q(\boldsymbol{x};\boldsymbol{\theta})||\pi p(\boldsymbol{x}) + (1-\pi)q(\boldsymbol{x};\boldsymbol{\theta})\right].$$

Considere a otimização do modelo para os limites $\pi \to 0$ e $\pi \to 1$. Pode-se notar que, nesses limites, $JS_\pi\left[p(\boldsymbol{x})||q(\boldsymbol{x};\boldsymbol{\theta})\right] \to 0$, de forma que não se pode utilizá-los na prática. Por outro lado, tem-se que [16]

$$\lim_{\pi\to 0}\frac{JS_\pi\left[p(\boldsymbol{x})||q(\boldsymbol{x};\boldsymbol{\theta})\right]}{\pi} = D_{KL}\left[p(\boldsymbol{x})||q(\boldsymbol{x};\boldsymbol{\theta})\right]$$

e, por questões de simetria,

$$\lim_{\pi\to 1}\frac{JS_\pi\left[p(\boldsymbol{x})||q(\boldsymbol{x};\boldsymbol{\theta})\right]}{\pi} = D_{KL}\left[q(\boldsymbol{x};\boldsymbol{\theta})||p(\boldsymbol{x})\right].$$

Algoritmo 1: Ajuste dos modelos gerador e discriminador utilizando algum método de otimização baseado em gradientes.

para *número de iterações* **faça**

  para *k iterações* **faça**

    Amostre $\mathbb{X}_p = \{\boldsymbol{x}_p^{(1)}, \boldsymbol{x}_p^{(2)}, \ldots, \boldsymbol{x}_p^{(N_p)}\}$ de $p(\boldsymbol{x})$.

    Amostre $\mathbb{X}_q = \{\boldsymbol{x}_q^{(1)}, \boldsymbol{x}_q^{(2)}, \ldots, \boldsymbol{x}_q^{(N_q)}\}$ de $q(\boldsymbol{x}; \boldsymbol{\theta})$.

    Calcule o gradiente da perda do modelo discriminador em relação aos seus parâmetros:

$$\nabla_{\boldsymbol{\phi}} \leftarrow \frac{\pi}{N_p} \sum_{i=1}^{N_p} \nabla_{\boldsymbol{\phi}} \log d_{\boldsymbol{\phi}}(\boldsymbol{x}_p^{(i)}) - \frac{1-\pi}{N_q} \sum_{i=1}^{N_q} \nabla_{\boldsymbol{\phi}} \log(1 - d_{\boldsymbol{\phi}}(\boldsymbol{x}_q^{(i)})).$$

    Atualize os parâmetros do modelo discriminador:

$$\boldsymbol{\phi} \leftarrow opt(\boldsymbol{\phi}, \nabla_{\boldsymbol{\phi}}).$$

  fim

  Amostre $\mathbb{Z} = \{\boldsymbol{z}^{(1)}, \boldsymbol{z}^{(2)}, \ldots, \boldsymbol{z}^{(N)}\}$ de $q(\boldsymbol{z})$.

  Calcule o gradiente da perda do modelo gerador em relação aos seus parâmetros:

$$\nabla_{\boldsymbol{\theta}} \leftarrow \frac{1-\pi}{N} \sum_{i=1}^{N} \nabla_{\boldsymbol{\theta}} \log(1 - d_{\boldsymbol{\phi}}(g_{\boldsymbol{\theta}}(\boldsymbol{z}^{(i)}))).$$

  Atualize os parâmetros do modelo gerador:

$$\boldsymbol{\theta} \leftarrow opt(\boldsymbol{\theta}, \nabla_{\boldsymbol{\theta}}).$$

fim

A implicação disso é que, para valores de π próximos a 0 e 1, $JS_\pi\left[p(\boldsymbol{x})||q(\boldsymbol{x};\boldsymbol{\theta})\right]$ tem os mesmos gradientes de $D_{KL}\left[p(\boldsymbol{x})||q(\boldsymbol{x};\boldsymbol{\theta})\right]$ e $D_{KL}\left[q(\boldsymbol{x};\boldsymbol{\theta})||p(\boldsymbol{x})\right]$, respectivamente, desconsiderando um fator de escala. Assim, de um ponto de vista de otimização, a variação de π de 0 a 1 permite interpolar entre o comportamento dessas duas perdas distintas.

De um ponto de vista teórico, assumindo que os modelos gerador e discriminador utilizados tem capacidade de representação infinita e que pode-se amostrar indefinidamente de $p(\boldsymbol{x})$, entre outras premissas introduzidas anteriormente, isso se torna irrelevante, visto que pode-se garantir que o Algoritmo 1 leva $q(\boldsymbol{x};\boldsymbol{\theta})$ para $p(\boldsymbol{x})$, eventualmente, independente de π. Por outro lado, na prática, quando os modelos tem capacidade de representação limitada, se tem acesso apenas a um conjunto finito de amostras de $p(\boldsymbol{x})$ e, além dessas, se perdem outras garantias, o valor de π influência no processo de otimização.

Observa-se, empiricamente, que, nessas circunstâncias, $D_{KL}\left[p(\boldsymbol{x})||q(\boldsymbol{x};\boldsymbol{\theta})\right]$ tende a favorecer modelos geradores que generalizam em excesso a distribuição geradora de dados. Exemplificando, se esta última é multi-modal, o modelo gerador ajustado tende a cobrir todos seus modos, até mesmo ao custo de introduzir probabilidades positivas em pontos onde a mesma é nula [16, 17]. O efeito prático disso é a geração de amostras implausíveis pelo modelo gerador, as quais não parecem ter sido tiradas de $p(\boldsymbol{x})$.

No outro extremo, $D_{KL}\left[p(\boldsymbol{x})||q(\boldsymbol{x};\boldsymbol{\theta})\right]$ favorece modelos geradores que são mais conservadores, deixando de cobrir partes importantes de $p(\boldsymbol{x})$ em troca de não gerar amostras inverossímeis. Ilustrativamente, se $p(\boldsymbol{x})$ é multi-modal, $q(\boldsymbol{x};\boldsymbol{\theta})$ tende a cobrir bem alguns de seus modos, se concentrando bastante nestes e, às vezes, ignorando completamente outros [16, 17]. Esse comportamento é chamado de colapso modal, no qual o modelo gerador colapsa em alguns modos da distribuição geradora de dados, deixando, assim, de capturar outros aspectos importantes da mesma.

Assim sendo, variando π no intervalo aberto de 0 a 1, pode-se interpolar entre processos de otimização que favorecem modelos que trocam a qualidade de suas amostras, eventualmente gerando algumas inverossímeis, pela diversidade das mesmas, valorizando cobrir toda a distribuição geradora de dados. Em outras palavras, pode-se ver π como uma medida de conservadorismo do processo de otimização, de forma que, quanto mais próximo é de 1, mais o modelo ajustado se permite deixar de capturar aspectos importantes de $p(\boldsymbol{x})$ em troca de não produzir amostras de baixa qualidade, que não pareçam ter vindo desta última. Esse comportamento é ilustrado, graficamente, na Figura 3.1.

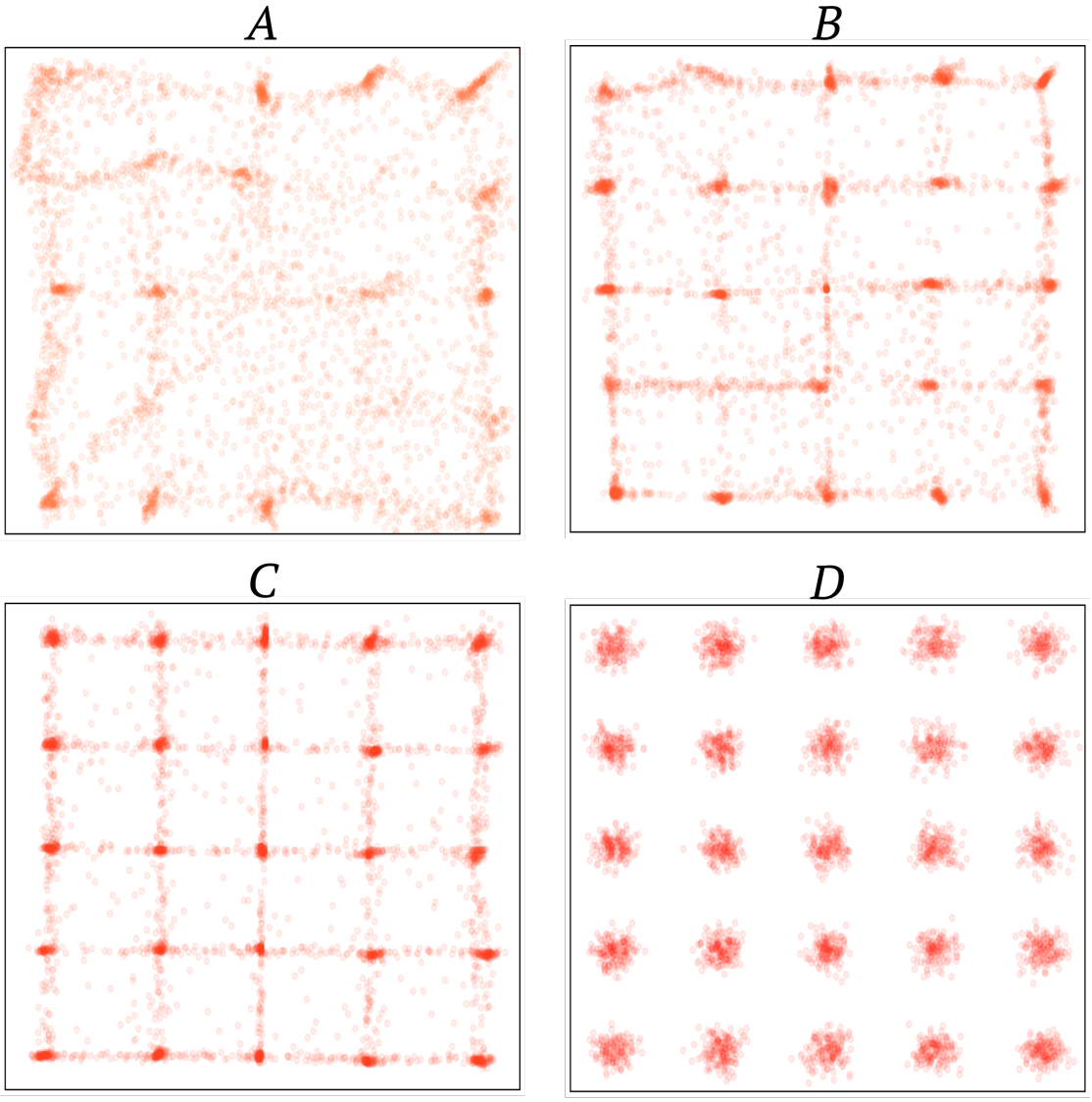

Figura 3.1: Ilustração da influência de π sobre o processo de ajuste do modelo gerador. Em (A), (B) e (C), tem-se amostras geradas por três modelos exatamente iguais ajustados com as amostras de $p(\boldsymbol{x})$ mostradas em (D), com a única diferença de terem utilizado π igual a 0.1, 0.5 e 0.9, respectivamente. Em (A), nota-se uma generalização excessiva por parte do modelo, falhando em cobrir com precisão alguns modos de $p(\boldsymbol{x})$ e gerando amostras inverossímeis. Em (C), no outro extremo, observa-se uma distribuição extremamente concentrada nos modos de $p(\boldsymbol{x})$, ao custo de ignorar aspectos importantes da mesma, como a variância existente em torno desses modos. Finalmente, em (B), pode-se observar claramente um comportamento intermediário entre esses dois extremos.

Capítulo 4

APLICAÇÃO

Pode-se observar, pela leitura do capítulo anterior, diversas premissas, tomadas como verdadeiras, que foram introduzidas de forma a chegar na versão final do modelo, juntamente com um algoritmo para seu ajuste. Essas premissas foram apresentadas ao lado de argumentos que as motivam, embora não haja prova formal da validade de algumas delas. Por esse motivo, é importante e necessário realizar experimentos com o intuito de avaliar se a teoria apresentada é, de fato, funcional e se traduz nas aplicações práticas desejadas.

Como será discutido em maior profundidade a seguir, a modelagem geradora de imagens naturais consiste de um enorme desafio, pois trata de distribuições de alta dimensionalidade e complexidade. Ainda, a mesma traz consigo algumas aplicações práticas potenciais bastante atraentes. A geração de imagens naturais realistas, por si só, por exemplo, pode ser usada no auxílio a criação de conteúdo visual de diversos tipos. Dessa forma, consiste de um problema extremamente interessante para se aplicar a teoria desenvolvida.

Dito isso, será implementado um modelo baseado em GANs para modelar uma distribuição de imagens naturais de rostos de pessoas. Essa distribuição é representada pelo conjunto de dados CelebA [6], o qual consiste de aproximadamente 200 mil imagens de rostos de celebridades e será apresentado em maior detalhe mais à frente. Em seguida, serão realizados alguns experimentos com o modelo ajustado a esse conjunto de dados, de forma a clarear algumas de suas características e funcionalidades. Assim, espera-se por em prática a teoria desenvolvida no capítulo anterior num caso de complexidade considerável, tornando evidente a aplicabilidade da mesma.

4.1 Imagens naturais

Imagens naturais podem ser descritas, de forma simplificada, como fotografias do mundo real. Vistas como dados, são originárias de um processo gerador bastante obscuro e enigmático. Isso se dá por esses processos terem, por trás, um fenômeno físico extremamente complexo e que envolve uma quantidade exorbitante de fatores que interagem entre si, tornando-os incrivelmente difíceis de se modelar.

Sob uma perspectiva probabilística, pode-se ver o ato de tirar uma foto, seja dentro de um contexto, como cachorros ou paisagens, ou não, como equivalente a amostrar de uma distribuição $p(\boldsymbol{x})$, a qual representa como essas imagens são encontradas no mundo. Por ter um caráter inerente de enorme complexidade, essa distribuição está longe de ser compreendida em seu todo, mas são observadas diversas regularidades estatísticas em suas amostras que indicam haver uma rica estrutura por trás da mesma [18–20].

Por exemplo, imagens naturais costumam conter objetos, no sentido de coisas materiais que podem ser percebidas pelos sentidos. Dessa forma, como objetos, usualmente, tem superfícies suaves, regiões próximas de imagens tendem a ser parecidas e, portanto, surgem correlações locais entre as intensidades de seus pixels. Há uma série de outras regularidades estatísticas presentes nesses tipos de imagens além dessa que vem já vem sendo estudadas extensivamente há bastante tempo [18–20], tornando evidente a complexidade por trás desse tipo de dado.

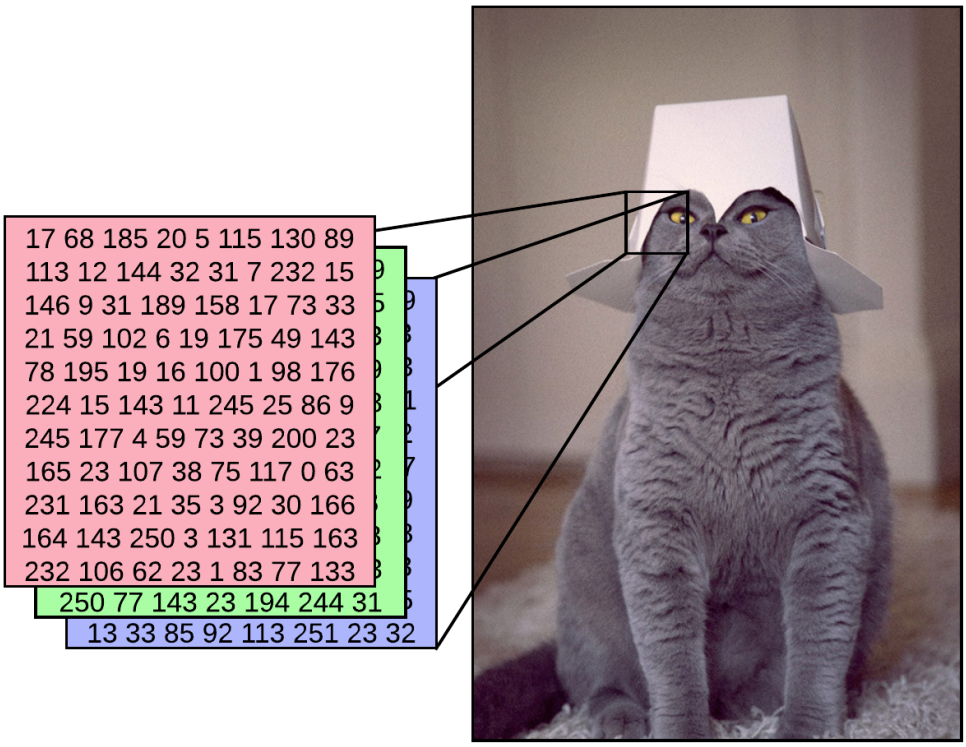

Figura 4.1: Exemplo ilustrativo mostrando como uma imagem colorida pode ser representada, computacionalmente, através de um *array* tri-dimensional. Nota-se que há três canais de cores: vermelho, verde e azul. Dessa forma, o valor de cada elemento desse *array* mede a intensidade de um pixel em particular, variando de 0 a 255, nesse caso. Suas primeiras duas dimensões são relativas à localização espacial dos pixels, enquanto a terceira, à cor dos mesmos. Portanto, se essa imagem tem 500 pixels de largura e 750 pixels de altura, ela consiste de $500 \times 750 \times 3 = 1125000$ inteiros entre 0 e 255.

Ainda, pode-se olhar para elas como pontos num espaço de alta dimensionalidade, onde as dimensões correspondem aos valores individuais de cada pixel. Por exemplo, imagens coloridas de 500×750 pixels podem ser vistas como elementos de $\mathbb{R}^{1125000}$, como ilustrado na Figura 4.1. Outro indicador de sua rica estrutura é que a parte desse espaço que é preenchida, de fato, por elas, é mínima. Considere um ponto escolhido aleatoriamente nele. Não é difícil perceber que a chance de se parecer com uma imagem natural é quase que nula.

A modelagem geradora de imagens, se faz, portanto, interessante, por tratar da modelagem de distribuições extremamente complexas sobre dados de alta dimensionalidade e bastante estruturados, os quais são objetos de interesse para várias áreas da matemática aplicada e engenharia [5]. Dessa forma, constitui um desafio de imensa dificuldade que, embora longe de ser resolvido, vem progredindo bastante nos últimos tempos [4].

4.1.1 Aplicabilidade de GANs

Em comparação a outros modelos geradores, imagens geradas por GANs são consideradas muito mais nítidas e de qualidade extremamente superior às de outros modelos geradores tradicionais [5], embora não se saiba, ao certo, o que as levam a isso. Conjectura-se que, principalmente, seu sucesso em produzir imagens realistas se deve ao uso da divergência de Jensen-Shannon ao invés da perda baseada em verossimilhança usual, a qual leva a imagens tipicamente borradas [16].

A geração de imagens naturais realistas tem uma vasta gama de aplicações potenciais que está em constante crescimento e longe de ter sido explorada em seu todo [5]. Um exemplo trivial é no desenvolvimento de gráficos ultra-realistas para jogos. Juntamente com tecnologias de realidade virtual, há o potencial para se criar ambientes de extrema imersão com aspecto realista. De um modo geral, há aplicações relacionadas a criação de praticamente qualquer tipo de conteúdo visual, visto que possibilita a automação desse processo.

4.2 Distribuição geradora de dados

Primeiramente, é importante notar que, embora o Algoritmo 1 esteja formulado com base na premissa que se pode amostrar $p(\boldsymbol{x})$, a distribuição a qual se deseja modelar, isso não ocorre, de fato, na prática. Por outro lado, isso também não é preocupante, visto que, dado um conjunto grande o bastante de amostras de $p(\boldsymbol{x})$, pode-se aproximar satisfatoriamente o processo de amostragem da mesma amostrando uniformemente dele. Como mencionado anteriormente, esse conjunto é chamado de conjunto de dados.

Dessa forma, escolheu-se o conjunto de dados CelebA [6] para representar a distribuição desejada. O mesmo consiste de aproximadamente 200 mil imagens coloridas de rostos de 10 mil celebridades. Na Figura 4.2 são mostradas algumas das imagens pertencentes a esse conjunto de dados. Elas consistem, em sua grande maioria, de retratos de pessoas tirados por fotógrafos profissionais, tendo, assim, um padrão bem definido entre si. Com exceção dos rostos em si, incluindo cabelo e aparatos como chapéus ou óculos, e das poses dos mesmos, há poucos fatores que

Figura 4.2: Exemplos de imagens de rostos encontradas no conjunto de dados CelebA, já redimensionadas para 64×64 pixels, utilizadas no ajuste do modelo gerador implementado.

introduzem variância nas mesmas.

Imagens naturais de rostos de pessoas são interessantes por possuírem um padrão bastante claro, o qual é facilmente perceptível para o observador humano e fascinante de ver um modelo aprender. Embora transparente, esse padrão tem uma rica estrutura por trás, com uma hierarquia bastante complexa. Essas imagens são compostas, primeiramente, por um fundo qualquer e um rosto. Este último, num nível abaixo, é composto de olhos, boca, nariz, orelhas, cabelo, entre outros. Por sua vez, estes são formados por estruturas ainda mais simples, como texturas e contornos. Assim, torna-se clara a complexidade por trás das mesmas, motivando sua escolha.

Todas as imagens foram redimensionadas para 64×64 pixels, devido ao seu tamanho original tornar inviável a implementação dos modelos, devido ao poder computacional limitado disponível para tal. Também, tiverem os valores de seus pixels ajustados para o intervalo $[-1, 1]$, visto que facilita o processo de ajuste [3, 11]. Finalmente, o conjunto de dados foi ampliado, artificialmente, aplicando-se transformações que preservassem as características originais das imagens que as definem como rostos. A cada vez que uma imagem era exposta ao modelo para seu ajuste, a mesma era submetida às seguintes transformações:

- Com 50% de chance, era espelhada horizontalmente;
- Aumento ou diminuição de saturação em até 5%, escolhido uniformemente;
- Aumento de brilho em até 5%, escolhido uniformemente;
- Aumento ou diminuição de contraste em até 5%, escolhido uniformemente;
- Deslocamentos de até 8 pixels para a direita ou esquerda e cima ou baixo, escolhidos, também, uniformemente.

Todas essas transformações mudam essas imagens consideravelmente de uma perspectiva quantitativa. Por outro lado, qualitativamente, as mesmas quase não sofrem alteração. Assim, aumenta-se imensamente a variabilidade dos dados e o tamanho efetivo do conjunto de dados utilizado, tornando a aproximação da amostragem direta de $p(\boldsymbol{x})$ muito mais fiel [3].

4.3 Aspectos técnicos

4.3.1 Arquitetura dos modelos

Tanto o modelo gerador quanto o modelo discriminador são construídos em cima de transformações parametrizadas. Como argumentado anteriormente, é interessante que essas sejam extremamente flexíveis e tenham alta capacidade de representação. Isso se dá por não ser possível determinar, a priori, o poder necessário para que as mesmas possibilitem a modelagem do processo gerador desejado satisfatoriamente. Dessa forma, as mesmas serão implementadas através de redes neurais.

Mais especificamente, escolheu-se implementá-las com um tipo de rede neural bastante específico para o processamento de imagens, chamado de rede neural convolucional [3, 10]. Tendo inspirações biológicas no córtex visual primário de mamíferos, estrutura cerebral responsável pelo processamento inicial de estímulos visuais [3], se faz especialmente apropriada para aplicações no contexto de imagens.

Apesar do nome particular, não passam de redes neurais comuns que utilizam camadas convolucionais em sua arquitetura. Essas camadas são encontradas em várias formas diferentes [3, 21–23], havendo diversas variações e tecnicalidades que fogem do escopo do trabalho e, portanto não serão discutidas em mais detalhes. Apesar dessas variações, todas elas têm em comum a aplicação, de alguma forma, da operação de convolução em suas entradas.

O uso de convoluções leva a uma série de características bastante interessantes [3, 10]. Por exemplo, as mesmas consistem de operações invariantes a translações. No contexto de imagens, esse é um atributo bastante útil, visto que o deslocamento das mesmas por alguns pixels não costuma mudar as mesmas de um ponto de vista qualitativo. Ainda, tipicamente, camadas convolucionais utilizam um número muito menor de parâmetros, possibilitando a construção de redes neurais mais profundas e, portanto, mais poderosas [3].

Usualmente, o ajuste de modelos baseados em redes neurais convolucionais costuma ser bem mais complicado e trabalhoso do que de redes neurais tradicionais. Isso se dá, principalmente, pela profundidade desses modelos, que costuma ser bem maior, e pela característica de alta dimensionalidade típica dos dados com as quais operam. Essa dificuldade é contornada, muitas vezes, utilizando algumas arquiteturas padrões que tenham se provado efetivas na resolução de um determinado problema que seja similar ao problema em foco.

No contexto da modelagem geradora de imagens naturais com GANs, uma das primeiras arquitetura de redes neurais convolucionais a ser proposta e utilizada com sucesso foi a DCGAN [4,5]. Após extensa experimentação dos pesquisadores por trás da mesma com diversas variações de arquitetura diferentes, chegou-se a certos princípios que nortearam seu desenvolvimento, levando a um processo de ajuste mais estável e rápido, além de geração de imagens com maior qualidade [4,5]. Seguindo a prática supracitada de reaproveitamento de arquiteturas que tenham provado seu sucesso, escolheu-se implementar a DCGAN, visto que a mesma já se mostrou extremamente eficaz para o problema em questão.

Essa arquitetura consiste de duas redes neurais convolucionais, uma para a transformação parametrizada do modelo gerador e outra para o modelo discriminador, detalhadas nas Tabelas 4.1 e 4.2, respectivamente. A primeira leva uma entrada $\boldsymbol{z} \in \mathbb{R}^{100}$ para uma imagem $\boldsymbol{x} \in \mathbb{R}^{64\times64\times3}$, enquanto a segunda leva uma imagem $\boldsymbol{x} \in \mathbb{R}^{64\times64\times3}$ para um número em $[0, 1]$ representando a probabilidade da mesma ter vindo do conjunto de dados e não do modelo gerador.

A única diferença entre a arquitetura implementada e a original foi a alteração da distribuição da variável latente $\mathbf{z}$. A dimensão da mesma é mantida em 100, mas foi distribuída conforme uma gaussiana padrão ao invés de uniformemente em $[-1, 1]$. Essa mudança foi motivada pela observação subjetiva de melhora na qualidade das imagens geradas.

Tabela 4.1: Arquitetura DCGAN para o modelo gerador.

Camada	Dimensões de saída
Entrada	100
Densa (BN, ReLU)	16384
Reshape	$4 \times 4 \times 1024$
Convolução 2D Transposta (BN, ReLU, filtros 5×5 e strides 2×2)	$8 \times 8 \times 512$
Convolução 2D Transposta (BN, ReLU, filtros 5×5 e strides 2×2)	$16 \times 16 \times 256$
Convolução 2D Transposta (BN, ReLU, filtros 5×5 e strides 2×2)	$32 \times 32 \times 128$
Convolução 2D Transposta (tanh, filtros 5×5 e strides 2×2)	$64 \times 64 \times 3$

Tabela 4.2: Arquitetura DCGAN para o modelo discriminador.

Camada	Dimensões de saída
Entrada	$64 \times 64 \times 3$
Convolução 2D (LeakyReLU, filtros e strides 2×2)	$32 \times 32 \times 128$
Convolução 2D (BN, LeakyReLU, filtros 5×5 e strides 2×2)	$16 \times 16 \times 256$
Convolução 2D (BN, LeakyReLU, filtros 5×5 e strides 2×2)	$8 \times 8 \times 512$
Convolução 2D (BN, LeakyReLU, filtros 5×5 e strides 2×2)	$4 \times 4 \times 1024$
Flatten	16384
Densa (sigmóide)	1

4.3.2 Processo de ajuste

O ajuste dos modelos gerador e discriminador foram feitos seguindo o procedimento descrito no Algoritmo 1. Pode-se observar que o mesmo depende de uma série de parâmetros que regulam seu funcionamento. Não há regra clara para escolha destes, sendo, usualmente, um processo altamente experimental e intuitivo de tentativa e erro [3, 11, 13]. Por outro lado, há algumas recomendações para arquitetura DCGAN [4] que auxiliam na determinação de seus valores [4].

Primeiramente, escolheu-se como inicializar os parâmetros das redes neurais utilizadas. Há diversas formas existentes para realizar essa inicialização, algumas embasadas teoricamente e outras frutos de observação experimental. Uma das mais simples e eficazes consiste da simples inicialização a partir de uma gaussiana centrada em zero e com desvio padrão baixo o bastante [3], usualmente da ordem de 10^{-2} ou menor. Seguindo recomendações para a DCGAN [4], utilizou-se essa técnica com um desvio padrão de 0.0002.

Em seguida, determinou-se o método de otimização a ser utilizado. Foi escolhido o Adam, o qual consiste de um algoritmo de otimização baseado em gradientes bastante apropriado para problemas com grande número de parâmetros, objetivos não-estacionários e que envolvem gradientes ruidosos [24], todas características do problema em questão. Na linha das sugestões para a DCGAN [4], utilizou-se uma taxa de aprendizagem igual a 0.0002 e termo de momento, β, igual a 0.5.

O número de atualizações do modelo discriminador para cada atualização do modelo gerador,

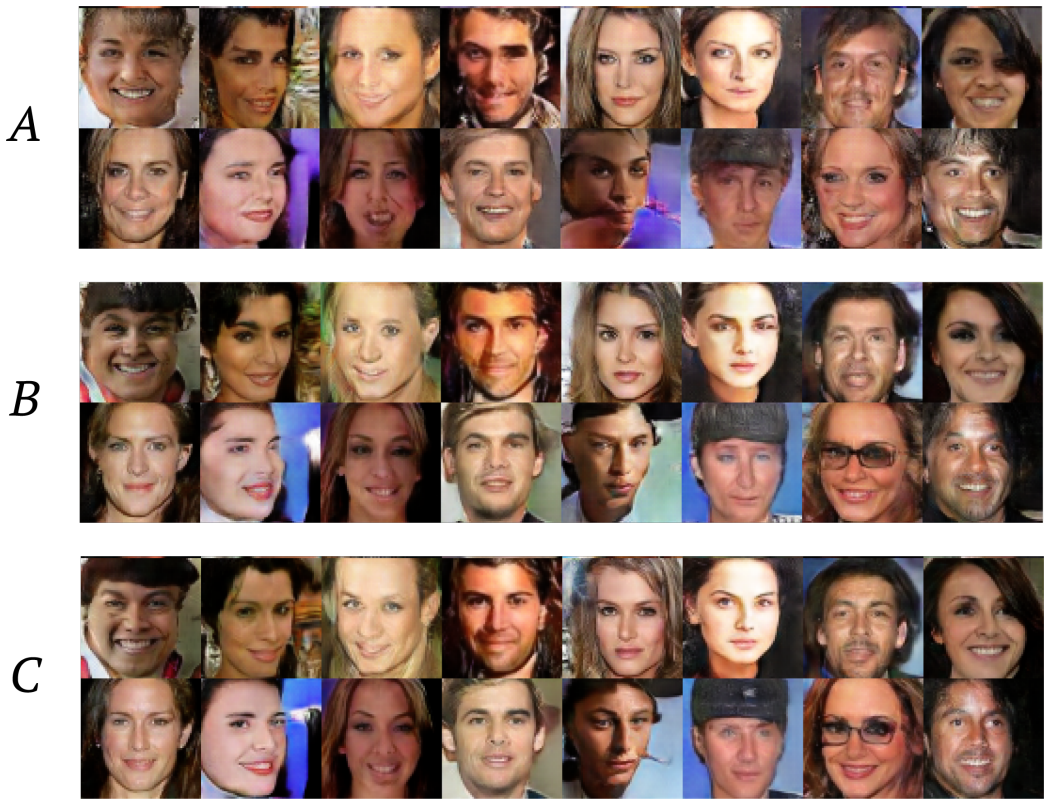

Figura 4.3: Imagens geradas pelo modelo gerador para um conjunto fixo de pontos em seu espaço latente após 40 mil (A), 120 mil (B) e 200 mil (C) iterações de seu processo de ajuste. Pode-se observar uma evolução significativa na qualidade de todos os rostos.

k, foi fixado em 1. Escolheu-se esse valor simplesmente por ter funcionado bem e ser o menor possível, tornando cada iteração do processo de ajuste, dessa forma, mais rápida. É importante notar que essa escolha é uma extremamente comum no ajuste de GANs, sendo utilizada em diversas implementações diferentes [1, 4, 5].

Finalmente, escolheu-se N_p, N_q e N todos iguais a 128. Observou-se que, utilizando valores muito baixos, as estimativas dos gradientes utilizados no ajuste dos modelos eram muito ruidosas, prejudicando-os dessa forma. Por outro lado, valores muito altos aumentam a complexidade computacional envolvida no processo, tornando-o lento sem necessidade. Assim, escolheu-se esse valor por ser um intermediário entre esses dois comportamentos indesejados, o que foi observado experimentalmente.

O desempenho do modelo gerador ao longo do seu processo de ajuste foi avaliado a partir da qualidade das imagens geradas pelo mesmo. A cada mil iterações, eram geradas imagens a partir de um conjunto fixo de pontos no espaço latente, de forma que fosse possível avaliar a evolução das mesmas no decorrer desse processo. O algoritmo foi executado por 200 mil iterações, ponto a partir do qual não se observou melhora significativa em relação à qualidade das imagens geradas. Na Figura 4.3, são mostradas algumas imagens geradas pelo modelo para um conjunto fixo de pontos em seu espaço latente depois de 40 mil, 120 mil e 200 mil iterações, ilustrando sua evolução.

4.3.3 Aspectos computacionais

A implementação dos modelos foi feita na linguagem de programação Python 3, utilizando, como base, a biblioteca de computação numérica TensorFlow [25]. Essa biblioteca foi desenvolvida, inicialmente, pela Google, tendo sido posteriormente liberada no formato *open source* para o público em geral. É extremamente útil na implementação de modelos baseados em redes neurais por permitir que o processamento seja feito em placas gráficas. Assim, devido a característica extremamente paralelizável desses modelos, há ganhos consideráveis de velocidade em seu uso [3].

É importante notar que há uma série de outras opções de bibliotecas que também permitem processamento em placas gráficas [26], como Theano [27], por exemplo. Todas elas possibilitariam, da mesma forma, a implementação dos modelos desejados. A escolha do TensorFlow foi motivada, principalmente, pela experiência prévia e familiaridade já existente do autor com a ferramenta. Dessa forma, o processo de implementação, em si, se tornou extremamente mais rápido e menos suscetível a erros.

Os aspectos computacionais evolvidos na implementação desses modelos são extremamente interessantes e longe de triviais. Por outro lado, um tratamento detalhado dos mesmos, além de extenso e trabalhoso, não é relevante para os objetivos do trabalho. Dessa forma, escolheu-se tratá-los apenas como meios para verificação prática da teoria apresentada, que é exatamente a visão que se tem dos mesmos. Assim, serão deixados em segundo plano, não se prolongando em sua discussão mais do que o necessário[1].

4.4 Geração de imagens

O processo de geração de imagens a partir do modelo gerador é bastante simples: obtém-se uma amostra $\boldsymbol{z}$ de $p(\boldsymbol{z})$, a qual é fornecida como entrada para a rede neural $g_{\boldsymbol{\theta}}$ do modelo gerador, resultando em uma imagem $\boldsymbol{x} = g_{\boldsymbol{\theta}}(\boldsymbol{z})$. Como já mencionado, a distribuição da variável latente $\mathbf{z}$ foi tomada como uma gaussiana padrão de dimensionalidade igual a 100. Dessa forma, esse processo faz-se extremamente eficiente, visto que a amostragem de $p(\boldsymbol{z})$ e a passagem pela rede neural $g_{\boldsymbol{\theta}}$ tem baixo custo computacional.

Na Figura 4.4, são mostradas algumas imagens geradas pelo modelo gerador a partir de pontos no espaço latente amostrados de $p(\boldsymbol{z})$. No geral, as imagens geradas pelo modelo são, claramente, de rostos de pessoas, havendo todas as características que os definem como tais. Por outro lado, em alguns casos, há algumas falhas evidentes, como distorções no formato dos rostos ou partes que os compõe e perda de nitidez total ou parcial das imagens, ficando borradas.

A maioria dos rostos utilizados para ajustar o modelo gerador se encontram virados para frente, usualmente com olhos, cabelo, boca e nariz bastante visíveis. Nota-se que os defeitos encontrados nas imagens geradas são mais acentuados para rostos que se encontram de lado, usando algum

[1]Para os interessados, pode-se entender melhor esses aspectos a partir da análise do código utilizado na implementação desses modelos. O mesmo se encontra disponível em um repositório do github acessível em: https://github.com/pennacchio/tcc

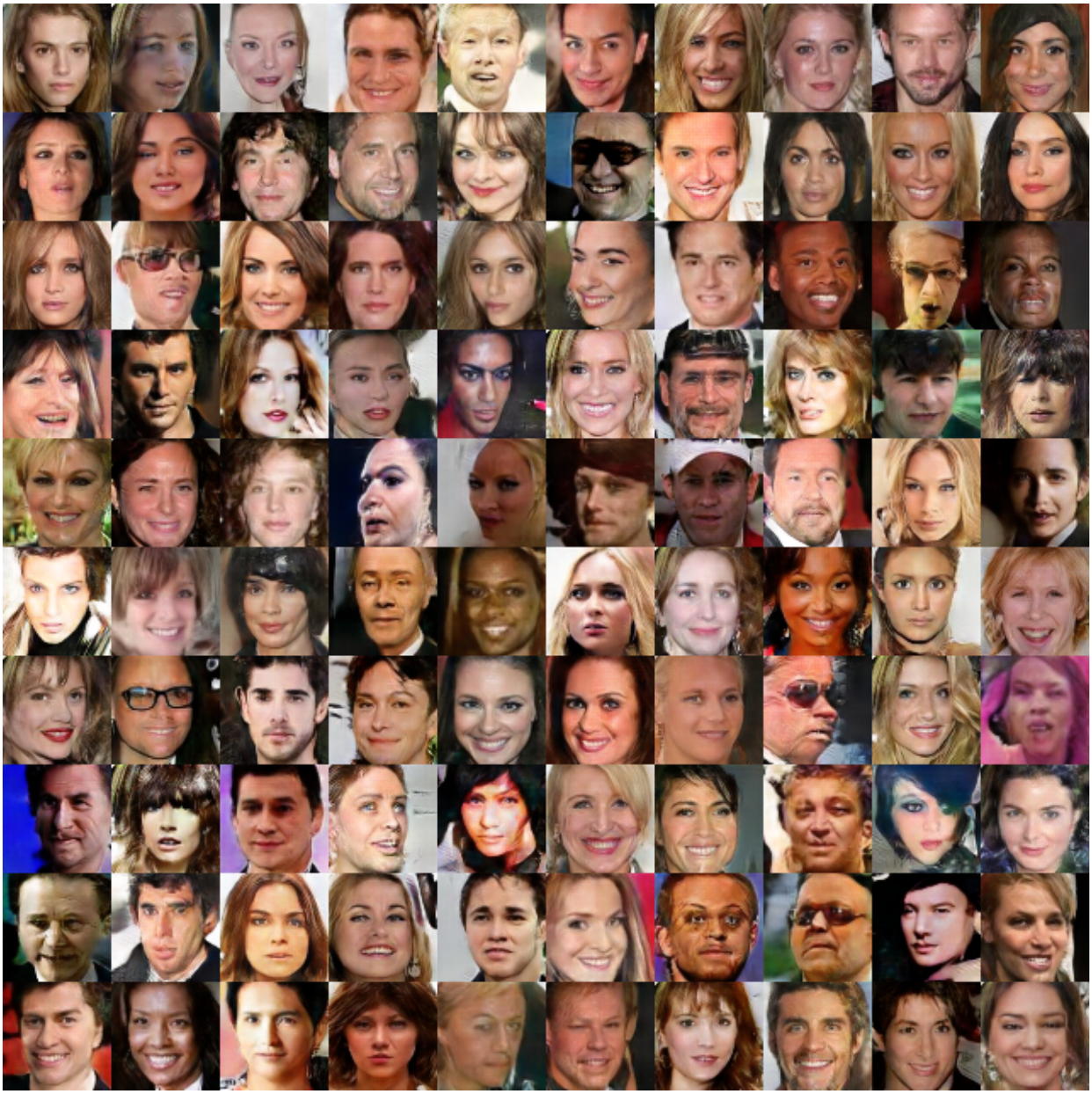

Figura 4.4: Grade de 10×10 imagens de rostos de pessoas artificiais geradas pelo modelo gerador implementado. A qualidade das imagens é, no geral, muito boa, podendo ser notada pela organização coerente e detalhe das estruturas que compõem os rostos, como olhos, nariz e boca. Ainda, percebe-se o uso correto de cores e captura de elementos secundários como expressões e iluminação. Por outro lado, em alguns casos, embora minoria, nota-se claramente severas falhas, principalmente consistindo de distorções no formato do rosto, chegando ao caso extremo de ficarem irreconhecíveis, como na última imagem da sétima linha.

acessório como óculos ou chapéu ou, no geral, que tenham alguma característica que fuja do padrão posto acima. Isso se dá, exatamente, pelo modelo ter menos exposição a imagens que fogem desse padrão, menos frequentes no conjunto de dados CelebA, tornando-o, naturalmente, pior na geração das mesmas.

Não há uma medida objetiva para a qualidade de imagens geradas por modelos geradores, sendo essa avaliada, usualmente, a partir de critérios subjetivos [17]. Dessa forma, as imagens geradas pelo modelo gerador implementado serão avaliadas, em relação à sua qualidade, a partir de aspectos qualitativos, como os detalhes da estruturas que a compõe e organização geral destas.

Nota-se, claramente, uma grande diversidade de rostos gerados, variando-se vários aspectos como cor de pele, sexo, expressão, idade, cabelo, pose, presença de barba, óculos ou chapéus, entre outros. No geral, percebe-se que a qualidade das imagens geradas é muito boa. Nos cabelos, nota-se claramente a presença de estruturas que lembram fios. Há imensa nitidez nas diversas outras estruturas que compões os rostos, como nariz, olhos, boca e sobrancelhas.

Em algumas imagens, observa-se a presença de rugas, usualmente relacionadas às expressões dos rostos, como o aparecimento de covinhas quando estão sorrindo. Também, nota-se claramente que o modelo gerador implementado captura bem a iluminação natural existente nessas imagens, sendo perceptíveis até reflexos de luz em algumas, como na oitava imagem da primeira linha.

Em relação a outros aspectos dessas imagens fora os rostos em si, o modelo parece não se comprometer tanto em modelá-los precisamente. No geral, os fundos são bastante simples, usualmente sendo planos e contendo poucas cores, sem nenhuma estrutura complexa. Os pescoços e roupas das pessoas são, em alguns casos, bastante nítidos e facilmente identificáveis, mas, no geral, de baixa qualidade, às vezes inexistentes e com formatos que não correspondem aos rostos das mesmas.

Embora não seja possível afirmar, com certeza, o porquê disso, conjectura-se que seja devido a uma propensão do modelo discriminador a focar em estruturas dos rostos em si para distinguir entre as imagens geradas e as reais. Dessa forma, não há incentivo para o modelo gerador em modelar com precisão as características secundárias dessas imagens, como o fundo que as compões ou as roupas das pessoas nelas.

4.5 Interpolação no espaço latente

Já foi discutido que o papel da rede neural do modelo gerador pode ser visto como mapear uma distribuição simples, $p(\boldsymbol{z})$, sobre um espaço de baixa dimensionalidade, para uma distribuição complexa, $q(\boldsymbol{x};\boldsymbol{\theta})$, num espaço de alta dimensionalidade, a qual, no caso, espera-se gerar imagens de rostos de pessoas. A mesma consiste da parte determinística do modelo, enquanto a parte estocástica é representada pela variável latente $\mathbf{z}$.

Dessa forma, qualquer imagem obtida do modelo pode ser associada a um ponto no espaço latente que, fornecido como entrada para sua rede neural, a gerou. Portanto, dadas duas imagens $\boldsymbol{x}_1 = g_{\boldsymbol{\theta}}(\boldsymbol{z}_1)$ e $\boldsymbol{x}_2 = g_{\boldsymbol{\theta}}(\boldsymbol{z}_2)$, pode-se interpolar entre as duas através da interpolação entre $\boldsymbol{z}_1$ e $\boldsymbol{z}_2$

seguida do mapeamento por $g_{\boldsymbol{\theta}}$ dos pontos obtidos. Assim, pode-se transformar, de certa forma, uma imagem na outra, onde os passos entre as mesmas consistem das imagens intermediárias obtidas a partir desse processo de interpolação.

A interpolação entre dois pontos $\boldsymbol{z}_1$ e $\boldsymbol{z}_2$ pode ser resumida em uma função $i(\boldsymbol{z}_1, \boldsymbol{z}_2, t)$, sendo t uma variável entre 0 e 1 que representa o quanto se deseja distanciar de $\boldsymbol{z_1}$ em prol de $\boldsymbol{z}_2$, de forma que $\boldsymbol{z}_1 = i(\boldsymbol{z}_1, \boldsymbol{z}_2, 0)$ e $\boldsymbol{z}_2 = i(\boldsymbol{z}_1, \boldsymbol{z}_2, 1)$. A forma mais simples de se fazer isso é via interpolação linear, dada pela função $i(\boldsymbol{z}_1, \boldsymbol{z}_2, t) = t\boldsymbol{z}_1 + (1-t)\boldsymbol{z}_2$.

Embora seja extremamente intuitiva e simples, observou-se que há alternativas melhores que a interpolação linear, devido ao caráter esférico da distribuição da variável da latente $\mathbf{z}$ [28]. Portanto, utiliza-se a chamada interpolação esférica linear, a qual observou-se produzir interpolações, no espaço de imagens, visualmente mais atrativas, sendo descrita pela função

$$i(\boldsymbol{z}_1, \boldsymbol{z}_2, t) = \frac{\sin((1-t)\Omega)}{\sin\Omega}\boldsymbol{z}_1 + \frac{\sin(t\Omega)}{\sin\Omega}\boldsymbol{z}_2, \text{ sendo } \Omega = \arccos\frac{\boldsymbol{z}_1 \cdot \boldsymbol{z}_2}{||\boldsymbol{z}_1||||\boldsymbol{z}_2||}.$$

Na Figura 4.5, são mostradas algumas interpolações feitas para 10 pares de imagens diferentes. Cada linha é referente a interpolação entre duas imagens em particular, enquanto as colunas se referem aos valores de t utilizados, os quais foram tomados como 0, para a primeira, 0.1, para a segunda, 0.2, para a terceira, e assim em diante, até 1, para a última. Como mencionado acima, utilizou-se interpolação esférica linear para se obter os pontos intermediários no espaço latente entre cada par de imagens.

Primeiramente, essas interpolações são uma forma de averiguar se o modelo apenas memoriza alguns rostos do conjunto de dados, ao invés de aprender sua distribuição, que é o que se deseja. Transições muito acentuadas indicam que há uma parte significativa do espaço latente sendo mapeada para uma mesma imagem, o que serve como um indicativo de memorização [4, 28]. Por outro lado, na Figura 4.5, pode-se observar transições suaves, indicando uma modelagem bem sucedida, não havendo esse colapso de partes do espaço latente para imagens específicas.

Ainda, as mesmas permitem uma maior compreensão acerca da existência de uma estrutura semântica no espaço latente. Em outras palavras, essa estrutura refere-se a como certas partes do mesmo são mapeadas para imagens com características específicas. Dessa forma, pode-se olhar para ele como uma representação, em baixa dimensionalidade, dessas imagens, a qual codifica as propriedades existentes nas mesmas. A observação da a aparição ou remoção de certos atributos no decorrer dessas interpolações é sinal da existência dessa estrutura [4].

Na quinta linha da Figura 4.5, nota-se a mudança progressiva da pose do rosto, a qual começa levemente para a direita e termina virada para a esquerda. Simultaneamente, nota-se o desaparecimento do sorriso, diminuição de iluminação, envelhecimento e feminização, de um modo geral. Esse comportamento de mudança em certos atributos pode ser observado, também, de forma mais isolada. Por exemplo, nas últimas três imagens da sexta linha, observa-se, quase que unicamente, o surgimento dos óculos. Já nas primeiras imagens da oitava linha, percebe-se quase que isoladamente uma feminização, mantendo-se as outras características constantes.

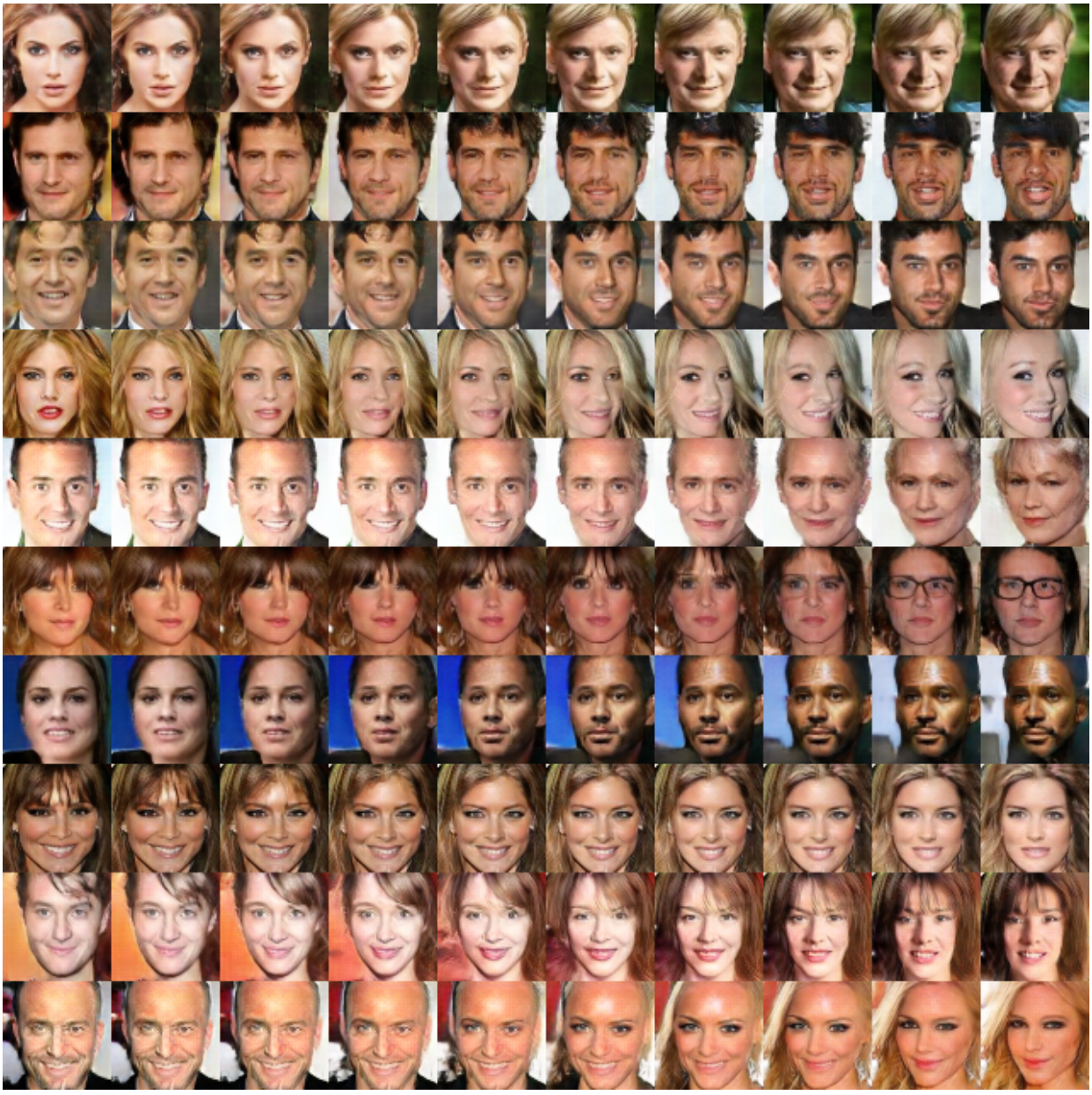

Figura 4.5: Conjunto de interpolações feitas para 10 pares diferentes de imagens geradas pelo modelo gerador implementado. Nota-se, claramente, transições suaves, indicando uma modelagem bem sucedida, não havendo sinais de memorização por parte do modelo. Ainda, a mudança progressiva em certos atributos das imagens, como expressão facial ou cor de pele, indica haver uma estrutura semântica no espaço latente, no sentido ter certas partes associadas a determinado conjuntos de atributos.

Capítulo 5

CONCLUSÃO

Assume-se que dados são gerados a partir de processos geradores, os quais consistem, usualmente, de algum fenômeno físico. Esses processos são muitas vezes obscuros e complexos, funcionando como uma caixa preta a cujo interior não se tem acesso. Uma das formas para se obter melhor compreensão acerca de um determinado tipo de dado é através do entendimento desse fenômeno que o gera.

Processos geradores podem ser associados, de um ponto de vista matemático, a distribuições de probabilidade, denominadas distribuições geradoras de dados. A modelagem dessas distribuições, usualmente a partir de amostras das mesmas, é chamada de modelagem geradora. A mesma é feita, usualmente, utilizando modelos cujos parâmetros são ajustados de forma a representarem, da melhor maneira possível, a distribuição geradora de dados em foco.

No caso de total desconhecimento acerca da distribuição que se deseja modelar, são necessários modelos extremamente flexíveis e com alta capacidade de representação. Uma classe de modelos com essas propriedades, que não impõe praticamente nenhuma restrição sobre as distribuições modeladas, é chamada de GANs [1]. Esses modelos consistem de duas partes, uma distribuição simples, $p(\boldsymbol{z})$, e uma transformação parametrizada, $g_{\boldsymbol{\theta}}$, que a mapeia para outra distribuição de maior complexidade, $q(\boldsymbol{x};\boldsymbol{\theta})$, com a qual deseja-se aproximar a distribuição geradora de dados em questão.

Utilizando redes neurais para implementar essas transformações parametrizadas, pode-se obter modelos com alto poder representacional e flexibilidade, aplicáveis a uma vasta gama de tipos de dados distintos. Ainda, GANs utilizam, em seu processo de ajuste, um modelo discriminador, auxiliar, que tem função de distinguir entre amostras reais, vindas da distribuição geradora de dados, e artificiais, vindas do modelo gerador. Dessa forma, o ajuste do modelo consiste, de forma simplificada, em enganar, ao máximo possível, esse modelo discriminador.

O objetivo do trabalho consistia, primeiramente, em realizar um desenvolvimento teórico de GANs, de forma natural e progressiva, firmando-se sobre conceitos estabelecidos de modelos geradores e aprendizagem de máquinas. Também, em segundo lugar, em aplicar a teoria apresentada à modelagem geradora de imagens faciais, levando a um modelo capaz de gerar imagens realistas de rostos humanos, cujas propriedades deseja-se explorar.

Foi apresentada a construção de GANs, juntamente com a formulação da perda utilizada para essa classe de modelos, levando, eventualmente, a um algoritmo para seu ajuste. A abordagem utilizada no trabalho para o desenvolvimento teórico de GANs diferiu um pouco da tradicional. Esta última, embora longe de trivial e bastante interessante, introduz conceitos teóricos importantes e centrais para o modelo sem muita explicação [1]. Dessa forma, acaba se utilizando muito pouco do imenso arcabouço teórico já existente para modelos geradores, visto que é formulada a partir de conceitos de teoria dos jogos, área pouquíssimo explorada em aprendizagem de máquinas [2].

Formulando essa classe de modelos sobre conceitos de teoria da informação, foi possível introduzir suas diferentes partes de forma, acredita-se, muito mais natural e progressiva. O desenvolvimento de cada aspecto da teoria foi feito com o intuito de explicitar a necessidade por trás do mesmo antes de introduzi-lo, objetivando-se tornar claro de onde surgiu. Com essa abordagem, espera-se que a estrutura teórica por trás desses modelos tenha se apresentado muito mais firme

e embasada.

Ainda, por teoria da informação ser uma área extensivamente estudada e utilizada em modelos geradores e aprendizagem de máquina, ao contrário de teoria dos jogos [2,3], pôde-se analisar GANs sob uma perspectiva muito mais clara, evidenciando aspectos que antes bastante mais obscuros. Consequentemente, foi possível traçar paralelos extremamente interessantes e que auxiliam na compreensão desses modelos, como, por exemplo, a influência que o parâmetro π exerce sobre os mesmos.

Naturalmente, necessita-se de experimentos para avaliar a validade da teoria apresentada. No desenvolvimento da mesma, foram feitas diversas aproximações e introduzidas várias premissas que, embora tenham motivação e sejam embasadas, não são acompanhadas de demonstrações e provas formais. Portanto, é interessante a prática da teoria desenvolvida, de forma a demonstrar sua coerência e aplicabilidade.

Dito isso, escolheu-se aplicar o modelo gerador apresentado na modelagem geradora de imagens naturais de rostos de pessoas. Por consistir de um tipo de dado extremamente complexo, de alta dimensionalidade e rica estrutura, notável pela grande dificuldade em sua modelagem, consistem de uma prova prática perfeita para a aplicabilidade de GANs.

Escolheu-se modelar imagens de rostos de pessoas, cuja distribuição foi representada pelo conjunto de dados CelebA [6]. Esse consiste de aproximadamente 200 mil retratos de 10 mil celebridades, tirados por fotógrafos profissionais e, consequentemente, sendo de alta qualidade e visibilidade. Essa escolha foi motivada pelo padrão facilmente identificável de rostos de pessoas, ao mesmo tempo que contém uma rica estrutura envolvendo as diversas partes que os compõe, expressões, entre outros.

Implementaram-se os modelos gerador e discriminador, computacionalmente, utilizando-se da biblioteca de processamento numérico TensorFlow [25], disponibilizada pela Google, na linguagem de programação Python 3. A arquitetura escolhida para o modelo foi baseada na DCGAN [4], a qual consiste de uma das primeiras aplicações de GANs para modelagem geradora de imagens com redes neurais convolucionais.

O processo de ajuste do modelo foi feito seguindo os procedimentos descritos no Algoritmo 1, desenvolvido a partir da teoria apresentada. A escolha de seus parâmetros foi altamente influenciada pelas recomendações existentes para a arquitetura DCGAN [4], não apresentando, dessa forma, nenhuma dificuldade notável, fora o tempo necessário para executá-lo, devido ao alto custo computacional. A evolução desse ajuste foi mensurada com base na qualidade das imagens geradas pelo modelo, a qual era avaliada subjetivamente em intervalos regulares.

O modelo, após ajustado, foi avaliado, primeiramente, em relação à qualidade das imagens geradas pelo mesmo. Nesse quesito, apresentou ótimo desempenho. Eventualmente, gerava imagens incompreensíveis, como formas distorcidas que mal lembravam rostos. Porém, no geral, gerava rostos extremamente bem definidos, com alto nível de detalhe e nitidez, capturando com precisão a estrutura dos mesmos e a organização das partes que os compões, como olhos, boca e nariz. Ainda, capturava diversos aspectos secundários dessas imagens com extrema exatidão, como a

iluminação e expressão dos rostos.

Em seguida, foram investigadas as característica de seu espaço latente e do mapeamento do mesmo para o espaço das imagens. Através do uso de interpolações entre imagens geradas pelo modelo, pôde-se avaliá-lo quanto à memorização do conjunto de dados, comportamento indesejado para o mesmo, que se mostrou não ocorrer. Ainda, observou-se a existência de uma estrutura e organização semântica em seu espaço latente, no sentido de suas diferentes partes corresponderem a atributos específicos dos rostos gerados pelo mesmo.

Dessa forma, mostrou-se que o ajuste do modelo, de acordo com o algoritmo desenvolvido, foi realizado com sucesso, visto que o mesmo gerou imagens de boa qualidade, capturando todos aspectos sutis da estrutura por trás das mesmas. Ainda, observou-se imensa diversidade entre as suas amostras, sendo extremamente rara a geração de duas imagens muito similares. Finalmente não houve memorização por parte do modelo, o que indicaria um processo de ajuste ineficaz, havendo inclusive uma estruturação semântica clara do seu espaço latente.

Sumarizando, foi introduzida a classe de modelos geradores GANs de uma perspectiva teórica baseada em teoria da informação, tornando seu desenvolvimento extremamente natural e progressivo. Ainda, a teoria apresentada foi avaliada, experimentalmente, com a modelagem geradora de imagens faciais, obtendo excelentes resultados.

Para futuros trabalhos, sugere-se explorar melhor a seguinte premissa utilizada na derivação do algoritmo para ajuste do modelo: $\boldsymbol{\phi}^*(\boldsymbol{\theta})$ varia suavemente o bastante, para pequenas variações $\boldsymbol{\theta}$, de forma que se pode considerar seu gradiente, em relação a este último, nulo. Embora motivada, não foi apresentada nenhuma demonstração formal de sua validade. Seria interessante uma análise do efeito que seu uso causa na aproximação dos gradientes do modelo gerador. Já foram desenvolvidas formas de calcular esses gradientes sem a introdução dessa premissa [29], mas essas aumentam consideravelmente o custo computacional do processo de ajuste do modelo, tornando-o inviável em alguns casos.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] GOODFELLOW, I. et al. Generative adversarial nets. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2014. p. 2672–2680.

[2] BISHOP, C. M. *Pattern recognition and machine learning*. [S.l.]: Springer, 2006.

[3] GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016.

[4] RADFORD, A.; METZ, L.; CHINTALA, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[5] GOODFELLOW, I. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.

[6] LIU, Z. et al. Deep learning face attributes in the wild. In: *Proceedings of International Conference on Computer Vision (ICCV)*. [S.l.: s.n.], 2015.

[7] MOHAMED, S.; LAKSHMINARAYANAN, B. Learning in implicit generative models. *arXiv preprint arXiv:1610.03483*, 2016.

[8] LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. *Nature*, Nature Research, v. 521, n. 7553, p. 436–444, 2015.

[9] LEE, H. et al. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In: ACM. *Proceedings of the 26th annual international conference on machine learning*. [S.l.], 2009. p. 609–616.

[10] LECUN, Y. et al. Generalization and network design strategies. *Connectionism in perspective*, Zurich, Switzerland: Elsevier, p. 143–155, 1989.

[11] LECUN, Y. A. et al. Efficient backprop. In: *Neural networks: Tricks of the trade*. [S.l.]: Springer Berlin Heidelberg, 2012. p. 9–48.

[12] SALIMANS, T. et al. Improved techniques for training gans. In: *Advances in Neural Information Processing Systems*. [S.l.: s.n.], 2016. p. 2234–2242.

[13] ARJOVSKY, M.; BOTTOU, L. Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862*, 2017.

[14] COVER, T. M.; THOMAS, J. A. *Elements of information theory*. [S.l.]: John Wiley & Sons, 2012.

[15] CHOROMANSKA, A. et al. The loss surfaces of multilayer networks. In: *Artificial Intelligence and Statistics*. [S.l.: s.n.], 2015. p. 192–204.

[16] HUSZÁR, F. How (not) to train your generative model: Scheduled sampling, likelihood, adversary? *arXiv preprint arXiv:1511.05101*, 2015.

[17] THEIS, L.; OORD, A. v. d.; BETHGE, M. A note on the evaluation of generative models. *arXiv preprint arXiv:1511.01844*, 2015.

[18] SIMONCELLI, E. P.; OLSHAUSEN, B. A. Natural image statistics and neural representation. *Annual review of neuroscience*, Annual Reviews 4139 El Camino Way, PO Box 10139, Palo Alto, CA 94303-0139, USA, v. 24, n. 1, p. 1193–1216, 2001.

[19] GERHARD, H. E.; THEIS, L.; BETHGE, M. Modeling natural image statistics. *Biologically-inspired Computer Vision—Fundamentals and Applications. Wiley VCH*, 2015.

[20] TORRALBA, A.; OLIVA, A. Statistics of natural image categories. *Network: computation in neural systems*, Taylor & Francis, v. 14, n. 3, p. 391–412, 2003.

[21] DUMOULIN, V.; VISIN, F. A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285*, 2016.

[22] CHEN, L.-C. et al. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *arXiv preprint arXiv:1606.00915*, 2016.

[23] CHOLLET, F. Xception: Deep learning with depthwise separable convolutions. *arXiv preprint arXiv:1610.02357*, 2016.

[24] KINGMA, D.; BA, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[25] ABADI, M. et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[26] ERICKSON, B. J. et al. Toolkits and libraries for deep learning. *Journal of Digital Imaging*, Springer, p. 1–6, 2017.

[27] BERGSTRA, J. et al. Theano: A cpu and gpu math compiler in python. In: *Proc. 9th Python in Science Conf.* [S.l.: s.n.], 2010. p. 1–7.

[28] WHITE, T. Sampling generative networks: Notes on a few effective techniques. *arXiv preprint arXiv:1609.04468*, 2016.

[29] METZ, L. et al. Unrolled generative adversarial networks. *arXiv preprint arXiv:1611.02163*, 2016.