Universidade de São Paulo
Instituto de Matemática e Estatística
Bacharelado em Ciência da Computação

Uso de imagens geradas artificialmente como uma alternativa à imagens reais no treinamento de modelos classificadores de imagens

Luca Lopes Barcelos

Monografia Final

mac 499 — Trabalho de Formatura Supervisionado

Supervisora: Prof. Dr. Denis Deratani Mauá
Cossupervisor: Samuel Gales Guimarães

São Paulo
2023

Resumo

Luca Lopes Barcelos. **Uso de imagens geradas artificialmente como uma alternativa à imagens reais no treinamento de modelos classificadores de imagens**. Monografia (Bacharelado). Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2023.

Para obter bons resultados, as técnicas de aprendizado de máquina profundo requerem uma grande quantidade de dados representativos da tarefa a ser resolvida. Em muitos domínios, bons conjuntos de dados são difíceis de serem obtidos em grande quantidade, devido ao custo, à operacionalização e, mais recentemente, a aspectos legais de privacidade e segurança. Uma alternativa é a utilização de conjuntos de dados artificiais gerados automaticamente, como, por exemplo, imagens foto-realistas geradas por redes neurais profundas (p.ex., DALL-E). Esse projeto pretende analisar como o uso de imagens geradas artificialmente afeta a eficácia de um sistema classificador de imagens. O trabalho está relacionado à pesquisa de mestrado no tema pelo aluno Samuel Gales Guimarães.

Palavras-chave: Inteligência artificial. Redes neurais. Aprendizado profundo. Aprendizado de máquina. Modelos de Difusão. Modelos generativos. Classificadores.

Abstract

Luca Lopes Barcelos. **Usage of artificially generated images as an alternative to real images in the training of image classifier models**. Capstone Project Report (Bachelor). Institute of Mathematics and Statistics, University of São Paulo, São Paulo, 2023.

To obtain good results, deep learning techniques require a large amount of data representative of the task to be solved. In many domains, good and large enough datasets are hard to aquire due to reasons such as cost, logistics, and, more recently, legal aspects with regards to privacy and security. A possible alternative is using artificially generated images, for example, photo-realistic pictures generated by deep neural networks (e.g., DALL-E). This project aims to analyze how the usage of artificially generated images affects the performance and accuracy of an image classifying system. The work is related to the ongoing Master's research on the topic by Samuel Gales Guimarães.

Keywords: Artificial Intelligence. Neural networks. Deep learning. Machine learning. Diffusion models. Generative models. Classifiers.

Lista de abreviaturas

ANN	Rede Neural Artificial (*Artificial Neural Network*)
GAN	Rede Adversária Generativa (*Generative Adversarial Network*)
MSE	Erro quadrático médio (*Mean squared error*)
ReLU	Unidade Linear Retificada (*Rectified Linear Unit*)
NLP	Processamento de Linguagem Natural (*Natural Language Processing*)
RNN	Rede Neural Recorrente (*Recurrent Neural Network*)
CNN	Rede Neural Convolucional (*Convolutional Neural Network*)
ResNet	Redes Residuais (*Residual Network*)
DDPM	*Denoising Diffusion Probabilistic Models*
AE	*Autoencoder*
ELBO	Limite Inferior de Evidência (*Evidence Lower Bound*)
LDM	Modelo de Difusão Latente (*Latent Diffusion Model*)
DM	Modelo de Difusão (*Diffusion Model*)
VAE	Autoencoder Variacional (*Variational Autoencoder*)
LAION	*Large-scale Artificial Intelligence Open Network*
NSFW	*Not safe for work*
CLIP	*Contrastive Language-Image Pre-Training*
GPU	Unidade de Processamento Gráfico (*Graphics Processing Unit*)
IA	Inteligência Artificial
Adam	*Adaptive Moment Estimation*

Lista de figuras

Lista de tabelas

Sumário

Introdução

Nos últimos anos, o campo da Ciência da Computação tem testemunhado avanços significativos na área de processamento de imagens e aprendizado de máquina. Um dos desafios mais prementes neste domínio é o treinamento eficiente de modelos de reconhecimento de imagem, que frequentemente requer um grande conjunto de dados de imagens. No entanto, a obtenção de um conjunto de dados extenso e diversificado pode ser um grande obstáculo, devido a questões como privacidade, custo e acessibilidade. Uma solução potencial para este desafio vem na forma de imagens geradas artificialmente, um campo que tem visto um crescimento exponencial, especialmente com o advento de modelos de difusão.[Zhou *et al.*, 2023]

Este trabalho aborda a utilização de imagens geradas por modelos de difusão para o treinamento de modelos classificadores no contexto do reconhecimento de imagem. Modelos de difusão funcionam adicionando ruído a uma imagem e aprendendo a reverter esse processo para criar novas amostras realistas a partir do ruído, e tendem a gerar imagens de maior qualidade e com menos artefatos em comparação às GANs (Generative adversarial networks), outra técnica utilizada para geração de imagens [Dhariwal e Nichol, 2021]. A pesquisa foca em explorar até que ponto as imagens artificialmente geradas podem substituir ou complementar as imagens reais em conjuntos de treinamento, e como isso impacta a eficácia e eficiência dos modelos de reconhecimento de imagem. A análise centra-se na performance dos modelos de reconhecimento quando expostos a imagens reais, após terem sido treinados do zero (com os pesos inicializados aleatoriamente) com determinadas proporções entre imagens reais e imagens geradas por modelos de difusão.

Foram selecionados dois conjuntos de dados reais diferentes, ambos advindos do *Kaggle*, uma das maiores plataformas online para ciência de dados e aprendizado de máquina. O primeiro conjunto de dados, *Animals-10*, contém imagens de 10 classes diferentes de animais, e o segundo conjunto de dados, *Fruits Classification*, contém imagens de 5 classes diferentes de frutas. Para a geração das imagens artificiais, quatro modelos generativos diferentes foram utilizados, esses sendo: *Stable Diffusion v2.1, Realistic Vision v1.4, Openjourney v4, Kandinsky v2.2.* Todos os modelos generativos podem ser encontrados na plataforma *Hugging Face* uma plataforma e comunidade online *open source* para trabalhos relacionados à aprendizado de máquina.

Para contextualizar, este trabalho começará com uma revisão da teoria sobre os modelos classificadores, redes neurais e suas técnicas, *transformers* e modelos de difusão, detalhando seu funcionamento, evolução e como eles se comparam a outras técnicas de geração de imagens. Em seguida, discute-se o estado atual dos modelos de reconhecimento de imagem, incluindo os desafios enfrentados no treinamento desses modelos e as vantagens potenciais

do uso de imagens sintéticas. O cerne da pesquisa envolve um experimento prático, no qual um conjunto de imagens geradas por modelos de difusão é utilizado para treinar modelos classificadores, seguido por uma avaliação de sua performance em um conjunto de dados de imagens reais, além de uma comparação entre diferentes modelos de difusão.

O trabalho visa analisar a viabilidade e eficácia do uso de imagens geradas artificialmente no treinamento de modelos classificadores de imagens. Além disso, busca-se avaliar se essa abordagem pode ser uma solução viável para as limitações atuais enfrentadas no treinamento de tais modelos, proporcionando um caminho para futuras pesquisas e aplicações práticas nesta área dinâmica e em rápida evolução.

A análise revelou que, embora imagens artificiais não possam substituir imagens reais no treinamento de classificadores, com os modelos treinados puramente em imagens artificiais tendo o desempenho muito inferior àqueles treinados em imagens reais, as imagens geradas pelos modelos de difusão se mostraram capazes de complementar um conjunto de dados real que possa ser pequeno demais para a tarefa a ser concluída. A qualidade do modelo generativo também se mostrou relevante para o desempenho do classificador treinado.

O projeto se baseia em uma pesquisa de mestrado no mesmo tema pelo aluno Samuel Gales Guimarães. A motivação por trás do tema é um problema real enfrentado hoje no Brasil, que, por conta das leis de proteção à privacidade e segurança da população, não é possível encontrar um conjunto de dados de boa qualidade e tamanho grande o suficiente para treinar modelos de reconhecimento de placas de veículos. Esse trabalho, no entanto, não aborda o tema de placas de veículos, mas sim tem como objetivo dar uma visão generalizada sobre como podemos utilizar imagens artificiais para treinar modelos classificadores que serão utilizados em imagens reais.

Capítulo 1

Revisão Teórica

1.1 Redes Neurais Artificiais

Redes Neurais Artificiais (ANNs) são um paradigma de processamento de informações inspirado na maneira como o sistema nervoso biológico, como o cérebro humano, processa informações. A principal característica das ANNs é a capacidade de aprender e melhorar a execução de tarefas através da experiência. Elas são compostas de um grande número de elementos de processamento altamente interconectados, trabalhando em uníssono para resolver problemas específicos. ANNs, assim como as pessoas, aprendem por exemplos. Elas são configuradas para uma aplicação específica, como reconhecimento de padrões ou classificação de dados, através de um processo de aprendizagem.

ANNs são semelhantes aos seres humanos na forma como aprendem: observando e repetindo. Elas aprendem a partir de exemplos históricos e usam essa aprendizagem para prever ou classificar novos eventos. Essa capacidade de aprender e generalizar a partir de exemplos históricos torna as ANNs uma ferramenta valiosa e versátil em muitas áreas de aplicação, desde a medicina até as finanças e a engenharia.

1.1.1 Estrutura Básica das ANNs

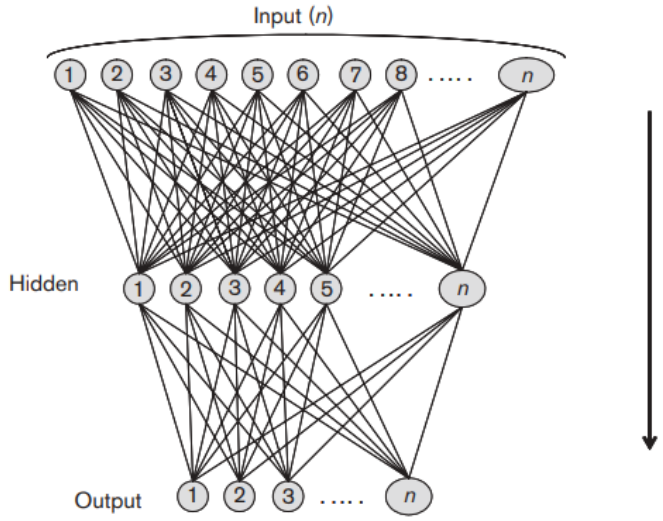


Figura 1.1: *Ilustração das camadas de entrada, ocultas, e de saída em uma ANN. Imagem retirada de Grossi e Buscema, 2008*

Os elementos base das ANNs são os nós, ou neurônios, e suas conexões. Cada neurônio tem sua própria entrada, advinda de outros neurônios e/ou do ambiente, e sua própria saída, pela qual se comunica com outros neurônios ou com o ambiente. Cada neurônio tem uma função f através da qual transforma sua própria entrada global em saída [Grossi e Buscema, 2008]. Os neurônios são organizados em camadas: a camada de entrada, que recebe o sinal do mundo exterior; as camadas ocultas, que são o principal motor de processamento da rede; e a camada de saída, que fornece a saída do modelo, como podemos ver na Figura 1.1.

Cada neurônio nas camadas ocultas transforma os valores de entrada de acordo com seus pesos sinápticos, que são ajustados durante o processo de treinamento da rede, de acordo com a equação:

$$y_i = f\left(\sum_j (w_{ij} \cdot x_j) + b_i\right)$$

Onde y_i é a saída do neurônio i na camada oculta, f é a função de ativação do neurônio, w_{ij} é o peso sináptico entre o neurônio i e a entrada j, x_j é o valor da entrada j e b_i é o viés do neurônio i. A somatória $\sum_j$ representa a combinação linear dos valores de entrada ponderados pelos pesos sinápticos, seguida pela adição do viés. A quantidade de camadas ocultas e o número de neurônios em cada uma dessas camadas determinam a complexidade da rede e sua capacidade de aprender padrões sofisticados. Redes com um grande número de camadas e neurônios são chamadas de redes neurais profundas e são capazes de aprender padrões extremamente complexos.

1.1.2 Funcionamento das ANNs

O funcionamento de uma ANN envolve a integração de diversos componentes e processos. No coração da rede neural artificial está o processo de aprendizado, que ajusta os pesos sinápticos dos neurônios com base na entrada e na saída desejada. Esse processo é frequentemente realizado por meio do algoritmo de retropropagação, uma técnica que permite que o gradiente do erro de saída seja distribuído de volta pela rede, ajustando assim os pesos para minimizar esse erro.

Durante o processo de treinamento, a rede é exposta a um grande conjunto de dados de entrada, juntamente com as saídas correspondentes. A rede processa as entradas e compara sua saída com a saída desejada. Qualquer diferença entre as duas é considerada um erro, que é então usado para ajustar os pesos da rede de maneira a reduzir esse erro nas próximas iterações. A função de erro e o algoritmo utilizado para atualizar os pesos da rede variam de rede para rede, uma das funções de erro mais populares é a MSE (Mean squared error), que é calculada segundo a equação:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Onde n é o número de amostras Y_i é o valor real da i-ésima amostra, e $\hat{Y}_i$ é o valor previsto pela rede para a i-ésima amostra. O algoritmo utilizado para atualizar os pesos mais comum é o de Descida de Gradiente, dado pela equação:

$$W_{\text{novos}} = W - \eta \cdot \nabla_W$$

Onde W são os pesos atuais da rede, η é a taxa de aprendizado e ∇_W é o gradiente da função de erro utilizada em relação aos pesos W.

A função de ativação em cada neurônio desempenha um papel crucial, pois determina como os sinais de entrada ponderados serão transformados em uma saída. Funções de ativação comuns incluem a unidade linear retificada (ReLU) e as funções sigmoidal e tangente hiperbólica. Essas funções introduzem não-linearidades na rede, permitindo que ela aprenda padrões complexos e realize tarefas que vão além do escopo dos modelos lineares simples.

Em resumo, as ANNs funcionam através da recepção de entradas, processamento dessas entradas através de uma rede de neurônios interconectados e ajuste contínuo dos pesos sinápticos com base no erro de saída, permitindo que a rede aprenda e melhore seu desempenho ao longo do tempo. A capacidade de uma ANN de aprender e adaptar-se a novas informações faz dela uma ferramenta poderosa em diversas aplicações, desde o reconhecimento de padrões complexos até a previsão de eventos futuros.

1.2 A Arquitetura Transformer

A arquitetura Transformer, desde sua introdução por Vaswani et al. em 2017, tem revolucionado o campo do processamento de linguagem natural (NLP) e mais recentemente, da visão computacional. Esta inovação se distancia significativamente das abordagens anteriores baseadas em redes neurais recorrentes (RNNs) e convolucionais (CNNs), introduzindo uma metodologia centrada no mecanismo de atenção, em particular a atenção multi-cabeça. Essa nova arquitetura permite uma compreensão mais profunda e uma representação mais rica de sequências de dados, seja em texto ou imagem.

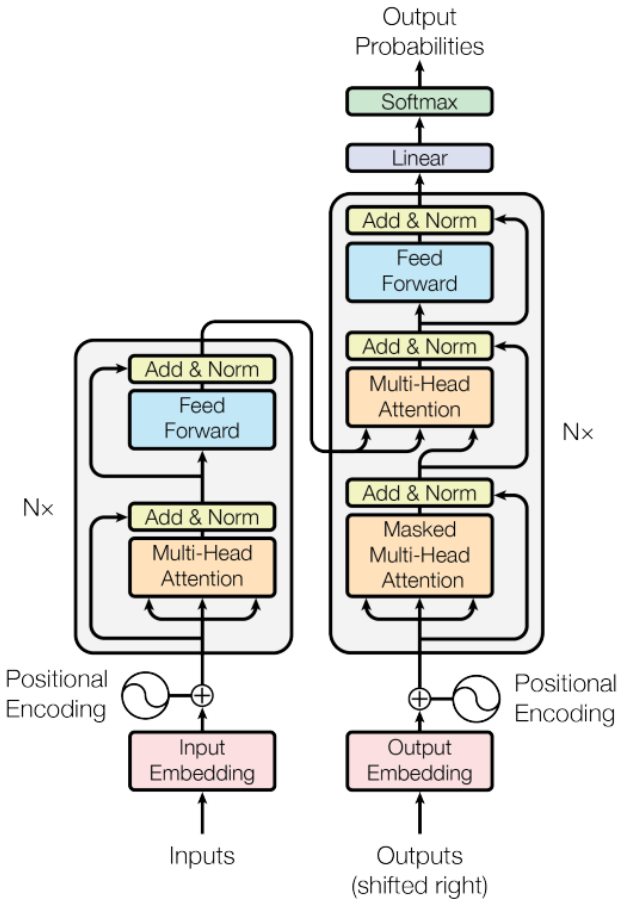


Figura 1.2: *A arquitetura transformer. Imagem retirada de* Vaswani *et al., 2023*

O Transformer é composto por dois componentes principais: o codificador e o decodificador, cada um contendo múltiplas camadas de atenção multi-cabeça e redes neurais feed-forward posicionais. A inclusão de codificação posicional é um aspecto crucial, permitindo que o Transformer mantenha a ordem das palavras ou pixels na sequência. Esta arquitetura é particularmente eficaz na modelagem de dependências de longo alcance, um desafio em abordagens anteriores, e oferece vantagens em termos de paralelismo e escala.

1.2.1 Mecanismo de Atenção

O mecanismo de atenção é o elemento central da arquitetura Transformer, concebido para superar as limitações das redes neurais recorrentes e convolucionais em tarefas de modelagem de sequência, como a ineficiência em computação paralela, o a incapacidade de focar em partes específicas da entrada, já que processam a sequência de entrada inteira, o que pode diluir a importância de informações críticas. Inspirado pela capacidade humana de focar seletivamente em aspectos específicos de uma informação, o mecanismo de atenção nos Transformers permite a modelagem eficaz de dependências sem considerar a distância entre elementos na sequência de entrada ou saída.

O mecanismo de atenção opera com três componentes principais: consultas (queries), chaves (keys) e valores (values). Cada elemento da sequência de entrada é transformado em um conjunto de consultas, chaves e valores. O mecanismo calcula a atenção aplicando uma função de peso, como a softmax, sobre os produtos escalares das consultas e chaves, o que determina a importância relativa de cada valor na saída, segundo a equação:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Onde Q, K, e V são as matrizes de consulta (query), chave (key), e valor (value), respectivamente e d_k é a dimensão das chaves. [Vaswani *et al.*, 2023]

Atenção Multi-cabeça

A implementação específica do mecanismo de atenção no Transformer é conhecida como 'atenção multi-cabeça'. Essa abordagem divide a atenção em várias "cabeças", permitindo que o modelo atenda simultaneamente a diferentes partes da sequência de várias maneiras. Cada cabeça de atenção pode aprender a se concentrar em diferentes aspectos da entrada, oferecendo uma representação mais abrangente e detalhada. O mecanismo de atenção multi-cabeça é definido por:

$$\begin{aligned}\text{MultiHead}(Q, K, V) &= \text{Concat}(\text{head}_1, ..., \text{head}_h)W^O \\ \textit{onde } \text{head}_i &= \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)\end{aligned}$$

Onde W_i^Q, W_i^K, W_i^V são matrizes de pesos distintas para cada cabeça de atenção i, W^O é uma matriz de pesos para combinar as saídas das diferentes cabeças, h é o número de cabeças de atenção e *Concat* representa a concatenação das saídas de cada cabeça de atenção. [Vaswani *et al.*, 2023]

1.2.2 Transformer em Visão Computacional

O sucesso do Transformer em NLP inspirou sua aplicação em visão computacional, um campo tradicionalmente dominado por CNNs. Adaptar o Transformer para proces-

sar imagens envolveu transformar dados 2D em uma sequência de pixels ou blocos de pixels, permitindo que a rede aplique seu mecanismo de atenção poderoso para capturar dependências complexas e globais em imagens.

Em tarefas de classificação e análise de imagens, os Transformers oferecem uma alternativa superior às CNNs. Eles são capazes de capturar contextos mais amplos e dependências de longo alcance em imagens, o que é particularmente benéfico em tarefas complexas de reconhecimento de imagem. Essa abordagem representa um desvio significativo das operações locais e convolucionais das CNNs, permitindo uma compreensão mais holística e integrada das imagens.

1.2.3 Geração de Imagens com Transformers

Os Transformers foram adaptados não apenas para reconhecimento, mas também para a geração de imagens. Ao contrário das GANs, que geram imagens a partir de ruído aleatório, os Transformers abordam a geração de imagens como um problema de sequência, predizendo pixels ou blocos de pixels com base em partes existentes da imagem.

Esta abordagem tem demonstrado ser particularmente eficaz na geração de imagens detalhadas e de alta resolução. Utilizando o mecanismo de atenção para entender as relações entre diferentes regiões da imagem, os Transformers podem gerar imagens que são não apenas visualmente impressionantes, mas também coerentes em termos de estilo e contexto. Essa capacidade abre novas possibilidades para a geração automatizada de arte, design gráfico e outras aplicações que exigem um entendimento sofisticado das relações espaciais e estéticas em imagens.

1.3 Redes Neurais Convolucionais

Redes Neurais Convolucionais (CNNs) são uma classe de redes neurais profundas, especialmente projetadas para processar dados que têm uma topologia de grade, como imagens. Inspiradas pelo sistema visual dos seres humanos e outros animais, as CNNs são capazes de detectar padrões visuais complexos através de múltiplas camadas de processamento. Essas redes são fundamentais na área de visão computacional, onde são utilizadas para interpretar e analisar imagens e vídeos. O princípio central das CNNs é a capacidade de aprender automaticamente os filtros e características em vez de depender de algoritmos de detecção de características manuais, permitindo uma abordagem mais flexível e robusta para a análise de imagens.

1.3.1 Estrutura

A estrutura típica de uma CNN inclui várias camadas, cada uma com uma função específica:

- **Camadas Convolucionais:** Estas camadas aplicam um conjunto de filtros aprendíveis às imagens para extrair características, como bordas, texturas e padrões mais complexos. Cada filtro produz um mapa de características que representa a presença dessas características na imagem.
- **Camadas de Pooling (Agrupamento):** Seguem as camadas convolucionais e são usadas para reduzir a dimensionalidade espacial dos dados. O pooling ajuda a tornar a representação mais compacta e menos sensível à localização exata das características na imagem.
- **Camadas Totalmente Conectadas:** No final da rede, as camadas totalmente conectadas combinam todas as características aprendidas para realizar tarefas específicas, como classificação. Essas camadas são semelhantes às encontradas em redes neurais tradicionais.

1.3.2 Funcionamento das CNNs

O funcionamento das CNNs envolve a passagem dos dados de entrada através de cada camada da rede:

- **Extração de Características:** Nas camadas convolucionais, cada filtro desliza sobre a imagem de entrada, realizando uma operação de convolução. Os filtros aprendem a reconhecer características específicas, e os mapas de características resultantes representam a resposta do filtro em diferentes regiões da imagem.
- **Redução de Dimensionalidade:** As camadas de pooling reduzem a dimensionalidade espacial dos mapas de características, agrupando pixels semelhantes. Isso torna a rede mais eficiente e menos propensa a overfitting, mantendo as características essenciais.
- **Classificação:** As camadas totalmente conectadas integram as características extraídas para realizar tarefas como classificação. Durante o treinamento, a rede ajusta

os pesos de cada camada para minimizar o erro na tarefa de classificação.

1.3.3 CNNs em Classificação de Imagens

As CNNs são amplamente utilizadas na classificação de imagens, uma tarefa que envolve atribuir uma etiqueta ou categoria a uma imagem com base em seu conteúdo visual. Em cenários como reconhecimento facial, diagnóstico médico a partir de imagens e detecção de objetos em vídeos, as CNNs demonstraram ser extremamente eficientes. Sua capacidade de aprender características hierárquicas, desde bordas simples até estruturas complexas, permite que elas identifiquem e classifiquem objetos em imagens com alta precisão. Além disso, a natureza adaptável das CNNs permite que elas sejam treinadas para tarefas específicas em diversos domínios, tornando-as ferramentas versáteis e poderosas na análise de dados visuais.

1.3.4 ResNets

As Redes Neurais Residuais, conhecidas como ResNets, surgiram como uma resposta inovadora aos desafios enfrentados pelas redes neurais convencionais à medida que aumentavam em profundidade. Antes das ResNets, o aumento da profundidade das redes frequentemente resultava em saturação e até mesmo degradação do desempenho, um fenômeno não atribuído ao *overfitting*, mas sim às dificuldades inerentes ao treinamento de redes profundas. As ResNets, introduzidas por Kaiming He e colaboradores, propuseram uma nova abordagem para a construção de redes mais profundas sem comprometer o desempenho.

O problema central que as ResNets visam resolver é o desaparecimento do gradiente, um desafio técnico onde o gradiente necessário para atualizar os pesos da rede torna-se muito pequeno, impedindo a rede de aprender efetivamente. Com o design inovador das ResNets, foi possível treinar redes com centenas de camadas, algo impensável com as arquiteturas anteriores. Isso abriu um novo horizonte em termos de capacidade e complexidade dos modelos de aprendizado profundo.

Arquitetura das ResNets

A arquitetura das ResNets é caracterizada pela introdução de conexões de atalho, que permitem que o sinal do gradiente seja transmitido diretamente através da rede sem passar por todas as camadas lineares. Isso é realizado através do mapeamento residual, onde a saída de uma camada é somada à sua entrada, efetivamente "pulando"uma ou mais camadas. Matematicamente, isso é representado pela equação $y = F(x, \{W_i\}) + x$, onde $F(x, \{W_i\})$ representa a transformação residual aplicada à entrada x, e $\{W_i\}$ são os pesos da rede. [He *et al.*, 2015]

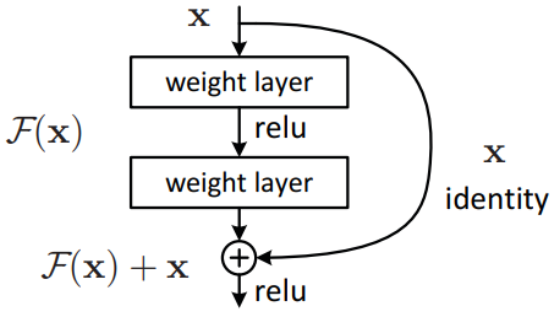


Figura 1.3: *Bloco básico de construção de uma ResNet. Imagem retirada de He et al., 2015*

Essa estrutura simplifica o treinamento da rede, pois permite que o gradiente flua diretamente através das conexões de atalho, facilitando a propagação e a atualização efetiva dos pesos durante o treinamento. Além disso, essa arquitetura permite que a rede aprenda identidades, caso em que o mapeamento residual se torna zero, o que é crucial para evitar o aumento desnecessário da complexidade do modelo. As ResNets mostraram um desempenho excepcional em diversas tarefas de visão computacional, estabelecendo um novo padrão para arquiteturas de redes neurais profundas.

1.3.5 U-net

A U-net é uma arquitetura de Rede Neural Convolucional (CNN) projetada inicialmente para a segmentação de imagens médicas, um desafio particular em modelos de difusão. Sua capacidade de trabalhar com conjuntos de dados limitados, uma característica comum em imagens médicas, a torna ideal para aplicações onde os dados são escassos ou altamente especializados. A U-net é notável por sua eficiência em aprender a segmentar imagens de maneira precisa e detalhada, tornando-se uma ferramenta valiosa para melhorar a qualidade dos dados em tarefas de modelagem de difusão.

Em um contexto mais amplo, a U-net tem aplicações que transcendem a medicina, sendo utilizada em qualquer tarefa de segmentação de imagens onde a precisão é fundamental. Sua arquitetura especializada permite que ela capture tanto o contexto quanto os detalhes finos das imagens, uma capacidade essencial para a preparação e o refinamento de dados em processos de difusão.

Arquitetura da U-net

A arquitetura da U-net é caracterizada por sua forma em U, que consiste em duas partes principais: um caminho de contração e um caminho expansivo. O caminho de contração segue a arquitetura típica de uma CNN, capturando contexto e reduzindo as dimensões espaciais da imagem através de operações de convolução e pooling. Matematicamente, este processo pode ser representado por operações de convolução $C(x, W)$ e pooling $P(x)$, onde x é a entrada, e W são os pesos da rede. O caminho expansivo utiliza operações de convolução transposta para aumentar as dimensões espaciais e refinar os detalhes, representado por $C^T(x, W)$.

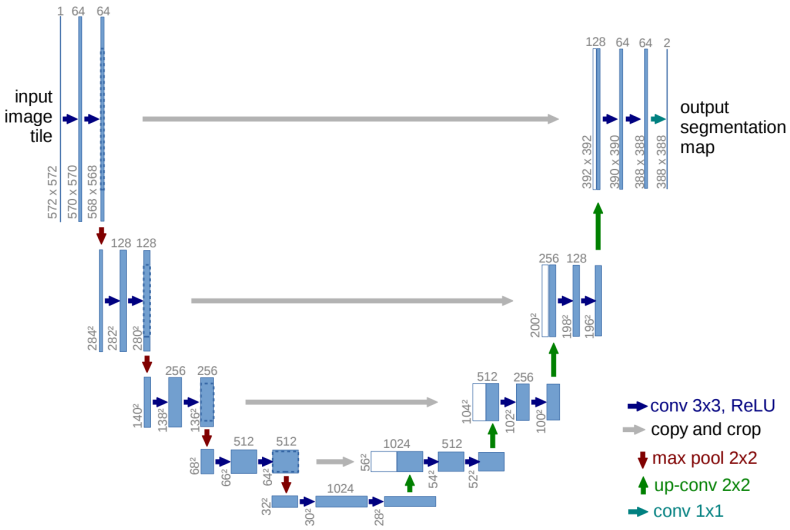


Figura 1.4: *Arquitetura da U-net. Imagem retirada de* Ronneberger *et al., 2015*

As conexões de cópia na U-net são um aspecto crucial, conectando diretamente o caminho de contração ao caminho expansivo. Isso permite que a rede preserve informações espaciais críticas, uma funcionalidade importante para a reconstrução precisa em tarefas de difusão. Essas conexões podem ser descritas como operações de concatenação $\oplus$ entre os recursos do caminho de contração e os do caminho expansivo. Portanto, em cada etapa do caminho expansivo, temos $y = C^T(x \oplus C(x, W), W)$, onde y é a saída reconstruída. Esta abordagem assegura que a U-net não apenas capture o contexto global, mas também mantenha a precisão local, tornando-a ideal para detalhar e aprimorar dados em modelos de difusão. [Ronneberger *et al.*, 2015]

1.4 Modelos de Difusão (DDPMs)

Denoising Diffusion Probabilistic Models (DDPMs) representam uma classe de modelos de cadeia de Markov parametrizados, utilizados para gerar amostras que correspondem a dados após um tempo finito. Esses modelos são treinados usando inferência variacional para reverter um processo de difusão, que é uma cadeia de Markov que adiciona gradualmente ruído aos dados na direção oposta da amostragem até que o sinal original seja destruído. Quando o processo de difusão consiste em pequenas quantidades de ruído gaussiano, as transições da cadeia de amostragem são definidas para gaussianas condicionais, permitindo uma parametrização simplificada da rede neural, já que a rede não precisa aprender uma função extremamente complexa para mapear entre estados arbitrários, em vez disso, ela aprende a prever os parâmetros de uma distribuição gaussiana bem definida.

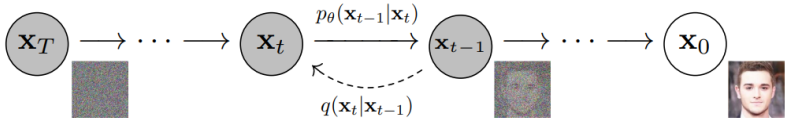


Figura 1.5: *Ilustração da reversão do processo de difusão. Imagem retirada de Ho et al., 2020*

O que distingue os modelos de difusão de outros modelos de variáveis latentes é o processo anterior aproximado, chamado de processo de difusão ou processo progressivo. Este processo é fixado em uma cadeia de Markov que adiciona ruído gaussiano aos dados de acordo com um escalonamento de variância especificado. Matematicamente, isso é representado por $q(x_{1:T}|x_0)$, onde $x_{1:T}$ são as variáveis latentes, x_0 são os dados originais e $q(x_{1:T}|x_0)$ descreve a probabilidade de obter as sequências de variáveis latentes $x_{1:T}$ a partir dos dados originais x_0 [Ho *et al.*, 2020]. A adição de ruído é modelada como uma distribuição normal, onde cada passo na cadeia adiciona uma quantidade de ruído definida pela variância β_t, segundo a equação:

$$x_t = \sqrt{1-\beta_t} \cdot x_{t-1} + \sqrt{\beta_t} \cdot \epsilon$$

Onde x_t é o estado dos dados no passo t, x_{t-1} é o estado dos dados no passo anterior $(t-1)$, β_t é a variância do ruído adicionado no passo t e ϵ é uma amostra aleatória de uma distribuição normal padrão (com média 0 e variância 1).

1.4.1 Cadeias de Markov

As Cadeias de Markov são um conceito fundamental em probabilidade e estatística, nomeadas em homenagem ao matemático russo Andrey Markov. Elas são usadas para modelar sistemas que se movem através de diferentes estados, com a propriedade chave de que a probabilidade de transição para o próximo estado depende apenas do estado atual, e não do histórico de estados anteriores. Esta característica é conhecida como a "propriedade de Markov"ou "ausência de memória".

No contexto dos Modelos de Difusão, particularmente os Denoising Diffusion Probabilistic Models (DDPMs), as Cadeias de Markov têm certas propriedades que as tornam

particularmente relevantes:

- **Transições Probabilísticas**: Em uma Cadeia de Markov, a mudança de um estado para outro é governada por probabilidades. Nos DDPMs, isso se traduz em como o ruído é adicionado ou removido em cada etapa do processo.
- **Estados Discretos e Contínuos**: As Cadeias de Markov podem ter estados discretos ou contínuos. Nos DDPMs, esses estados correspondem a diferentes níveis de ruído na imagem ou dados.
- **Processo Estocástico**: A Cadeia de Markov é um exemplo de processo estocástico, onde o resultado é intrinsecamente aleatório. Nos DDPMs, isso se reflete na forma aleatória como o ruído é adicionado e posteriormente removido.
- **Estacionariedade**: Alguns processos de Markov têm a propriedade de estacionariedade, onde as distribuições de probabilidade estabilizam-se ao longo do tempo. Em DDPMs, busca-se alcançar uma distribuição estacionária de ruído no processo de difusão.
- **Reversibilidade**: Em alguns casos, as Cadeias de Markov são reversíveis, o que significa que o processo pode ser revertido de maneira confiável. Nos DDPMs, isso é crucial, pois o processo de denoising busca reverter o processo de difusão, removendo o ruído para reconstruir a imagem ou dados originais.
- **Independência Condicional**: A propriedade de que o próximo estado depende apenas do estado atual (e não da sequência completa de eventos anteriores) é central nos DDPMs, pois cada passo do processo de difusão ou de denoising considera apenas o estado imediato anterior.

1.4.2 Autoencoders

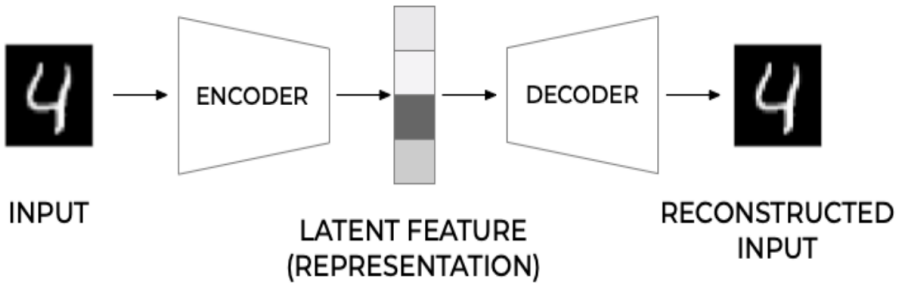


Figura 1.6: *Estrutura geral de um autoencoder. Imagem retirada de Michelucci, 2022*

Autoencoders (AE) são uma classe de redes neurais artificiais utilizadas para aprendizagem não supervisionada, com o objetivo de reduzir a dimensionalidade e descobrir representações latentes dos dados. Um autoencoder aprende a comprimir (codificar) a entrada em uma representação de menor dimensão e, em seguida, reconstruir (decodificar) a entrada a partir desta representação da forma mais precisa possível.

Um AE é definido por duas funções de mapeamento:

1. Codificador: $f_\theta : \mathbb{R}^n \to \mathbb{R}^m$, onde θ são os parâmetros da função de codificação e $m < n$ para um autoencoder compressivo.
2. 2. Decodificador: $g_\phi : \mathbb{R}^m \to \mathbb{R}^n$, onde ϕ são os parâmetros da função de decodificação.

O objetivo é minimizar a perda de reconstrução:

$$L(x, g_\phi(f_\theta(x)))$$

Onde x é a entrada e L é uma função de perda, como o erro quadrático médio (MSE), medindo a diferença entre a entrada original e a reconstruída.

Denoising Autoencoders são uma componente crucial em modelos de difusão, desempenhando um papel vital na etapa de reversão do processo de difusão. Em um modelo de difusão, a introdução progressiva de ruído em dados originais cria uma sequência de representações cada vez mais degradadas, culminando em uma distribuição que é essencialmente ruído puro. O papel do denoising autoencoder é o de reverter esse processo, reconstruindo os dados originais a partir dessas representações ruidosas. Funcionando como uma rede neural, o denoising autoencoder é treinado para prever o ruído que foi adicionado em cada etapa do processo de difusão, permitindo assim que ele seja removido de forma eficaz. Esse processo é iterativo, onde em cada passo, o autoencoder utiliza informações contextuais para reduzir o ruído e recuperar gradualmente a estrutura e as características originais dos dados. Essa abordagem contrasta com os métodos tradicionais de geração de dados, oferecendo uma maneira mais robusta e eficiente de reconstruir dados complexos, como imagens, a partir de informações altamente corrompidas. A capacidade do denoising autoencoder de recuperar informações precisas a partir de dados extremamente ruidosos o torna uma ferramenta valiosa em uma ampla gama de aplicações de aprendizado de máquina, particularmente em tarefas de geração e restauração de imagens.

1.4.3 Treinamento de Funcionamento de Modelos de Difusão

O treinamento de modelos de difusão é um processo que envolve duas fases distintas: a fase de difusão (forward) e a fase de reversão (reverse). Durante a fase de difusão, o modelo introduz progressivamente ruído nos dados originais, transformando-os gradualmente até que se assemelhem a ruído puro. Este processo é modelado como uma cadeia de Markov, onde cada passo adiciona uma pequena quantidade de ruído gaussiano. A fase de reversão, por outro lado, envolve o treinamento de uma rede neural para prever o ruído adicionado em cada etapa da difusão e, em seguida, subtraí-lo para restaurar a versão menos ruidosa dos dados.

O Processo de Difusão

O processo de difusão começa com dados originais e, em cada etapa, adiciona ruído gaussiano com base em um escalonamento de variância definido. Essa adição de ruído é realizada de maneira controlada e sistemática, seguindo o padrão definido pelo escalonador, que permite que o modelo aprenda a distribuição de dados ruidosos de maneira eficiente. Essa fase prepara o terreno para a etapa crucial de reversão, onde a principal tarefa

do modelo é reconstruir os dados originais a partir de suas versões corrompidas pelo ruído.

A fase de Reversão

Durante a fase de reversão, o desafio é desfazer o processo de difusão, removendo o ruído adicionado para recuperar os dados originais. Isto é realizado por um denoising autoencoder, que é treinado para estimar o ruído que foi adicionado em cada passo da difusão. Este treinamento é geralmente realizado utilizando técnicas como a inferência variacional, onde o objetivo é ajustar os parâmetros da rede neural de maneira que a distribuição gerada por ela se aproxime o máximo possível da distribuição real dos dados.

Otimização e Inferência Variacional

A inferência variacional é uma técnica em aprendizado de máquina e estatística para aproximar distribuições de probabilidade complexas. Ela é usada quando o cálculo direto de uma distribuição de probabilidade é complicado ou ineficiente, o que é comum em muitos problemas de aprendizado de máquina, como na modelagem de redes neurais profundas.

A otimização nos modelos de difusão é feita minimizando o limite variacional do logaritmo negativo da verossimilhança. Em inferência variacional, muitas vezes é difícil ou impossível calcular a verossimilhança diretamente, portanto, ao invés de trabalhar com a verossimilhança propriamente dita, trabalha-se com uma aproximação dela, conhecida como *limite variacional*. Isso envolve ajustar os parâmetros do modelo para que a probabilidade dos dados reconstruídos sob o modelo se aproxime da probabilidade sob a distribuição de dados verdadeira. Esse processo de otimização é fundamental para garantir que o modelo aprenda efetivamente a reverter o processo de difusão e recupere os dados originais com alta fidelidade.

Em estatística, a verossimilhança (ou *likelihood*) é uma medida de quão bem um modelo estatístico representa os dados observados. O logaritmo negativo da verossimilhança é frequentemente usado como uma função de custo em aprendizado de máquina, pois transforma a multiplicação de probabilidades (característica da verossimilhança) em uma soma de logaritmos, tornando o cálculo mais gerenciável. Minimizar o logaritmo negativo da verossimilhança é equivalente a maximizar a verossimilhança. O limite variacional do logaritmo negativo da verossimilhança é dado por:

$$\mathbb{E}_q\left[-\log p_\theta(x_0)\right] \leq \mathbb{E}_q\left[-\log \frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)}\right] = \mathbb{E}_q\left[-\log p(x_T) - \sum_{t\geq 1} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})}\right] := L$$

Onde $\mathbb{E}_q$ é a esperança sob a distribuição q, $-\log p_\theta(x_0)$ é o logaritmo negativo da verossimilhança dos dados originais x_0 sob o modelo com parâmetros θ, $p_\theta(x_{0:T})$ é a probabilidade conjunta dos dados originais x_0 e da trajetória ruidosa $x_{1:T}$ sob o modelo, $q(x_{1:T}|x_0)$ a distribuição de probabilidade do processo de difusão que gera as variáveis latentes $x_{1:T}$ a partir dos dados originais x_0, $p(x_T)$ a probabilidade dos dados no final do

processo de difusão, $p_\theta(x_{t-1}|x_t)$ a probabilidade condicional de reverter de x_t para x_{t-1} sob o modelo, $q(x_t|x_{t-1})$ a probabilidade condicional de transição no processo de difusão de x_{t-1} para x_t e L é o limite variacional, também conhecido como ELBO (Evidence Lower Bound), que é usado como função de custo para treinamento do modelo. [Ho *et al.*, 2020]

1.4.4 Difusão vs GANs

Modelos de difusão e GANs (Redes Generativas Adversárias) representam abordagens distintas e inovadoras na geração de imagens, cada uma com características únicas. Enquanto GANs foram, durante um tempo, líderes na geração de imagens devido à sua capacidade de produzir amostras visualmente impressionantes, eles enfrentaram desafios significativos como a dificuldade de treinamento, a necessidade de hiperparâmetros cuidadosamente escolhidos, e uma tendência a capturar menos diversidade. Os modelos de difusão, por outro lado, emergiram como uma abordagem poderosa, oferecendo benefícios como a cobertura eficiente da distribuição de dados e um processo de treinamento mais estável e escalável.

Tecnicamente, GANs e modelos de difusão divergem significativamente em suas abordagens de implementação. GANs operam através de uma arquitetura adversária composta por dois componentes principais: o gerador, que cria imagens, e o discriminador, que avalia sua autenticidade. O treinamento de um GAN envolve um jogo adversário onde o gerador tenta produzir imagens cada vez mais realistas, enquanto o discriminador aprende a distinguir entre imagens reais e geradas. Este processo pode ser desafiador, frequentemente resultando em problemas de convergência e no fenômeno do colapso de modo, onde o gerador produz uma variedade limitada de saídas. Por outro lado, modelos de difusão funcionam em um paradigma completamente diferente. Eles gradualmente introduzem ruído nos dados e depois aprendem a revertê-lo, em um processo que se assemelha à restauração de uma imagem. Esta abordagem baseia-se em cadeias de Markov e usa inferência variacional para treinar a rede, resultando em um processo de geração mais controlado e previsível, e evitando muitas das armadilhas associadas ao treinamento de GANs. A diferença fundamental reside na maneira como cada método aborda a geração de imagens: enquanto os GANs aprendem a gerar novas imagens a partir de um espaço latente, os modelos de difusão concentram-se em remover sistematicamente o ruído para revelar uma imagem.

Recentes avanços nos modelos de difusão mostraram que eles são capazes de alcançar e até superar a qualidade das imagens geradas pelos GANs. Essa melhoria foi possível devido a aprimoramentos na arquitetura do modelo e no desenvolvimento de métodos para equilibrar a fidelidade e a diversidade das imagens geradas. Esse progresso estabeleceu os modelos de difusão como uma alternativa competitiva aos GANs, destacando-se em várias métricas e conjuntos de dados. Essa evolução reflete o dinamismo contínuo no campo da geração de imagens, onde novas técnicas e abordagens estão constantemente deslocando o estado da arte.

1.5 Difusão Latente

A difusão latente representa uma evolução significativa nos modelos de difusão, superando desafios associados à geração de imagens em alta resolução e à complexidade computacional. Os modelos tradicionais de difusão operam diretamente no espaço de pixels, o que resulta em um consumo elevado de recursos computacionais, tanto para o treinamento quanto para a inferência. Para contornar essas limitações, a difusão latente aplica os modelos de difusão no espaço latente de autoencoders pré-treinados.

O espaço latente de autoencoders pré-treinados refere-se à representação compacta de dados que um autoencoder aprende a codificar, esse espaço é caracterizado por uma dimensão menor do que o espaço original dos dados (como os pixels de uma imagem), capturando as características essenciais. A eficiência da difusão latente é derivada da redução na dimensionalidade dos dados que o modelo de difusão precisa manipular, se o espaço original tem dimensão n e o espaço latente tem dimensão m, com $m << n$, então a complexidade computacional do modelo de difusão pode ser significativamente menor no espaço latente.

Essa abordagem permite alcançar um equilíbrio quase ótimo entre a redução da complexidade e a preservação de detalhes, elevando significativamente a fidelidade visual das imagens geradas.

1.5.1 Treinamento em espaço latente

A transição para o espaço latente começa com a análise de modelos de difusão já treinados no espaço de pixels. A aprendizagem é dividida em duas fases: uma fase de compressão perceptual, que elimina detalhes de alta frequência mas aprende pouca variação semântica, e uma fase de modelagem generativa, que aprende a composição semântica e conceitual dos dados. O treinamento é iniciado com um autoencoder, que fornece um espaço representacional de menor dimensão, mas perceptualmente equivalente ao espaço de dados. Importante destacar que, diferentemente de trabalhos anteriores, não é necessário depender de uma compressão espacial excessiva, pois os Modelos de Difusão Latente (LDMs) são treinados neste espaço latente aprendido, que apresenta melhores propriedades de escala em relação à dimensionalidade espacial. A complexidade reduzida permite uma geração de imagens eficiente a partir do espaço latente com uma única passagem de rede. Além disso, o estágio de autoencoding universal requer treinamento apenas uma vez e pode ser reutilizado para múltiplos treinamentos de DMs (Diffusion Models) ou para explorar tarefas completamente diferentes.

1.5.2 Aplicações da difusão latente

Os LDMs são modelos probabilísticos projetados para aprender uma distribuição de dados $p(x)$ por meio da denoising gradativa de uma variável distribuída normalmente. Isso corresponde a aprender o processo reverso de uma Cadeia de Markov fixa de comprimento T, isso é, dado um processo sequencial de T passos onde os dados são corrompidos incrementalmente com ruído até um estado altamente ruidoso, o modelo deve aprender a reverter essa sequência, passo a passo, removendo o ruído para reconstruir os dados

originais. Esses modelos podem ser interpretados como uma sequência igualmente ponderada de autoencoders de denoising, treinados para prever uma variante denoised da entrada ruidosa. Assim, os LDMs representam uma abordagem inovadora para a geração de imagens, combinando a eficiência do treinamento em espaços latentes com a flexibilidade e qualidade dos modelos de difusão para uma ampla gama de tarefas, incluindo síntese de imagens a partir de texto, geração de imagens incondicional e super-resolução.

Capítulo 2

Materiais e Metodologia

Este capítulo apresenta a metodologia adotada para explorar e avaliar as capacidades e eficácia dos modelos de reconhecimento de imagem treinados em imagens geradas por modelos de difusão. Serão delineadas as abordagens técnicas, os procedimentos experimentais e as ferramentas empregadas para conduzir essa análise. O objetivo é fornecer uma visão clara e detalhada dos métodos utilizados, garantindo a replicabilidade e a validação dos resultados obtidos.

Além disso, este capítulo detalha os conjuntos de dados utilizados, os critérios de seleção de dados e as métricas de avaliação utilizadas para medir a eficácia dos modelos. Também serão apresentados os modelos de difusão escolhidos e a razão por trás dessa escolha, incluindo os critérios que foram considerados. A ideia é analisar se os modelos de difusão são capazes de gerar um conjunto de dados bom o suficiente para auxiliar no treinamento dos classificadores ou possivelmente substituir um conjunto de dados reais.

2.1 Modelos Utilizados

Foram utilizados 4 modelos diferentes de difusão, sendo 3 deles derivados do Stable Diffusion. Todos os modelos utilizados possuem licenças permissivas e estão disponíveis abertamente ao público sem cobranças. Todos os modelos foram retirados do website huggingface.co, uma plataforma e comunidade online *open source* para trabalhos relacionados à aprendizado de máquina.

2.1.1 Stable Diffusion

O Stable Diffusion é um modelo de aprendizado profundo, baseado em técnicas de difusão e desenvolvido para a geração de imagens detalhadas condicionadas a descrições textuais. Lançado em 2022, este modelo é uma colaboração entre a CompVis Group da Ludwig Maximilian University of Munich e a Runway, com o suporte da Stability AI e dados de organizações sem fins lucrativos. O Stable Diffusion é classificado como um modelo de difusão latente (LDM) e inclui três componentes principais: um autoencoder variacional (VAE), U-Net e um codificador de texto opcional. O VAE compacta a imagem

para um espaço latente de menor dimensão, enquanto o bloco U-Net, com uma estrutura ResNet, desfaz a difusão para obter uma representação latente. A etapa de denoising pode ser condicionada a textos, imagens ou outras modalidades, utilizando um mecanismo de atenção cruzada. A eficiência computacional aumentada para treinamento e geração é uma das vantagens dos LDMs.

Um dos aspectos mais notáveis do "Stable Diffusion"é o seu status como um modelo de código aberto, com os pesos do modelo e o código liberados publicamente. Esta abordagem de código aberto, com uma licença permissiva, marca uma diferença significativa em relação a modelos anteriores de texto para imagem, como DALL-E e Midjourney, que eram acessíveis apenas através de serviços em nuvem. A disponibilidade do Stable Diffusion para o público em geral democratiza o acesso a tecnologias de geração de imagem avançadas e incentiva uma ampla gama de aplicações e inovações.

Em termos de capacidades, o Stable Diffusion se destaca na geração de imagens condicionadas a partir de descrições textuais, oferecendo aplicações em inpainting, outpainting e traduções de imagem para imagem guiadas por prompts de texto. Para o treinamento deste modelo, foram utilizados dados provenientes de organizações sem fins lucrativos, com a intenção de criar um modelo robusto e diversificado, capaz de compreender e interpretar uma ampla gama de entradas textuais. A eficiência computacional no treinamento e na geração de imagens é outra vantagem dos LDMs, tornando o Stable Diffusion uma ferramenta valiosa tanto para pesquisadores quanto para criadores de conteúdo.

Limitações

O modelo "Stable Diffusion"enfrenta limitações, incluindo a degradação na qualidade de imagem quando desviando da resolução de treinamento original de 512x512, embora versões posteriores tenham ampliado isso para 768x768 e 1024x1024. Há dificuldades na geração precisa de membros humanos devido à qualidade dos dados na base LAION. Adaptações específicas requerem dados adicionais e treinamento, sendo sensíveis à qualidade desses novos dados. O modelo também exibe um viés derivado dos dados nos quais foi treinado, esses que são predominantemente imagens com descrições em inglês, resultando em representações mais precisas e tendenciosas para culturas ocidentais ou brancas.

Stable Diffusion v2.1

O Stable Diffusion v2.1 é uma evolução do Stable Diffusion v2.0, tendo sido refinado a partir dele. Esse refinamento envolveu um processo adicional de 55 mil passos no mesmo conjunto de dados, com uma configuração de `punsafe=0.1`, seguido por mais 155 mil passos com `punsafe=0.98`. Aqui, "punsafe"refere-se a uma pontuação atribuída pelo filtro NSFW (Not safe for work) do conjunto de dados LAION-5B, indicando a probabilidade de uma imagem ser considerada imprópria. O modelo foi treinado no dataset LAION-5B e outros subconjuntos do mesmo. O modelo está disponível em https://huggingface.co/stabilityai/stable-diffusion-2-1.

Openjourney v4

O modelo Openjouney v4 foi treinado para gerar imagens similares ao modelo proprietário Midjourney, esse que é amplamente considerado um dos melhores modelos de geração de imagens disponíveis atualmente. O Openjourney é baseado no Stable Diffusion v1.5 e foi treinado com 124 mil imagens adicionais geradas pela quarta versão do Midjoruney. O modelo está disponível em https://huggingface.co/prompthero/openjourney-v4.

Realistic Vision v1.4

Outro modelo derivado do Stable Diffusion, o Realistic Vision foi refinado especificamente de modo a gerar imagens o mais realistas possíveis, diferente de outros modelos derivados do Stable Diffusion e do próprio Stable Diffusion, que podem gerar imagens cartoonizadas. O modelo está disponível em https://huggingface.co/SG161222/Realistic_Vision_V1.4.

2.1.2 Kandinsky v2.2

O modelo Kandinsky v2.2, uma evolução do Kandinsky 2, incorpora as melhores práticas do Dall-E 2 e da difusão latente. Utilizando o modelo CLIP (Contrastive Language-Image Pre-Training) como codificador de texto e imagem, o Kandinsky v2.2 realiza um mapeamento de difusão entre os espaços latentes das modalidades CLIP, o que melhora o desempenho visual do modelo e expande as possibilidades na manipulação de imagens guiadas por texto. Para o mapeamento de difusão, o modelo emprega um transformador com 20 camadas, 32 cabeças de atenção e um tamanho oculto de 2048. O Kandinsky 2.1 foi treinado em um amplo conjunto de dados de imagem-texto LAION HighRes e ajustado em conjuntos de dados internos, demonstrando a capacidade de adaptar-se a uma variedade de contextos e estilos de imagem. O modelo está disponível em https://huggingface.co/kandinsky-community/kandinsky-2-2-prior.

2.2 Datasets de verificação utilizados

Foram selecionados dois conjuntos de dados diferentes para verificar a performance dos modelos treinados nas imagens geradas. O primeiro deles, Animals-10, contém 28 mil imagens de animais de 10 classes diferentes, das quais apenas 10 mil foram utilizadas para comparação direta com o conjunto artificial gerado. O segundo conjunto, Fruit Classification, possui 22 mil imagens de frutas e vegetais de 33 classes diferentes, das quais foram escolhidas 5000 imagens de 10 classes diferentes para comparação direta com o conjunto artificial gerado. Ambos os datasets estão disponíveis gratuitamente no website kaggle.com, uma plataforma extremamente popular de ciência de dados.

Esses datasets foram escolhidos por representarem desafios distintos e realistas de classificação, permitindo-nos avaliar de forma abrangente a precisão, a robustez e a versatilidade dos modelos de reconhecimento de imagem. A análise dos resultados obtidos com esses datasets proporcionará insights valiosos sobre o desempenho dos modelos em tarefas de classificação complexas e variadas.

2.2.1 Animals-10

O conjunto de dados Animals-10 possui 28 mil imagens das seguintes classes:

- Dog (Cachorro)
- Cat (Gato)
- Horse (Cavalo)
- Spider (Aranha)
- Butterfly (Borboleta)
- Chicken (Galinha)
- Sheep (Ovelha)
- Cow (Vaca)
- Squirrel (Esquilo)
- Elephant (Elefante)

Todas as imagens foram coletadas do Google Imagens e checadas manualmente por um humano. O dataset contém dados incorretos para simular condições reais. O dataset está disponível em https://www.kaggle.com/datasets/alessiocorrado99/animals10.

2.2.2 Fruits Classification

O conjunto de dados Fruit Classification possui 10000 imagens das seguintes classes:

- Apple (Maçã)
- Banana (Banana)

- Grape (Uva)
- Mango (Manga)
- Strawberry (Morango)

As imagens no conjunto de dados possuem diversas proporções, tamanhos e cores, e foram capturadas sob diferentes condições de iluminação. O conjunto de dados original é dividido em uma proporção de 97-2-1 para treinamento, validação e teste, respectivamente, porém essa proporção não foi seguida durante os experimentos. O conjunto está disponível em https://www.kaggle.com/datasets/utkarshsaxenadn/fruits-classification.

2.3 A biblioteca utilizada

A biblioteca utilizada neste trabalho é o Pytorch. O PyTorch é reconhecido por sua flexibilidade e facilidade de uso, especialmente no desenvolvimento rápido de protótipos e experimentos de aprendizado profundo, o que é essencial para a pesquisa e desenvolvimento iterativo. Sua interface intuitiva e dinâmica facilita o acompanhamento do processo de modelos e a depuração, enquanto o suporte para computação em GPU acelera significativamente a fase de treinamento dos modelos. A combinação desses fatores faz do PyTorch uma escolha robusta e eficiente para projetos de aprendizado de máquina e geração de modelos.

A escolha do PyTorch como framework para este trabalho também foi fortemente influenciada por sua integração com os modelos pré-treinados disponíveis na plataforma Hugging Face, além do conhecimento prévio que possuo sobre a ferramenta.

2.4 Metódo de avaliação dos modelos

Neste estudo, foi empregada uma metodologia sistemática para avaliar o desempenho dos modelos de reconhecimento de imagem. A avaliação foi realizada em dois temas distintos: animais e frutas. Para cada tema, foram treinados três modelos distintos: um utilizando exclusivamente imagens geradas por IA, outro com uma mistura de 50% de imagens reais e 50% de imagens geradas, e um terceiro apenas com imagens reais, servindo como controle de referência. A acurácia de cada modelo foi meticulosamente medida tanto em um conjunto de imagens artificiais quanto em imagens reais para estabelecer uma comparação direta.

Este procedimento de avaliação foi replicado para cada modelo de geração de imagem empregado, permitindo não apenas uma análise comparativa entre os conjuntos de treinamento real e artificial dentro de um mesmo tema, mas também uma comparação cruzada entre os diferentes modelos de geração de imagem. Tal abordagem proporciona uma visão ampla sobre a eficácia dos modelos gerativos e seus impactos na capacidade de reconhecimento dos modelos subsequentes.

Capítulo 3

Experimentos e Resultados

3.1 Configuração dos Experimentos

Essa seção tratará dos experimentos conduzidos e fara uma discussão breve dos resultados obtidos. O código fonte do projeto está disponível em https://github.com/lucabarcelos/tcc e também será referenciado.

As principais ferramentas utilizadas foram os quatro modelos de difusão citados na seção 3.1, disponíveis no website HuggingFace, e os dois conjuntos de dados de refência citados na seção 2.2 e disponíveis no website Kaggle.

Os principais arquivos utilizados foram `datasetgen.py`, `traintest.py` e `analyze.ipynb`. Esses arquivos são responsáveis respectivamente por gerar os conjuntos de dados artificiais, treinar os diferentes modelos classificadores e medir suas acurácias, e por fim, analisar, comparar e gerar visualizações dos resultados obtidos.

Começando pelo arquivo `datasetgen.py`, o programa espera uma quantidade mínima de três argumentos em sua execução, esses sendo qual(is) modelo(s) de difusão utilizar, quantas imagens devem ser geradas, e qual o tema das imagens (animais ou frutas). O programa então carrega os modelos de difusão escolhidos, e gera um conjunto de dados totalmente artificial no tema escolhido com a quantidade de imagens especificada. Os *prompts* utilizados para gerar as imagens também especificam de forma aleatória a distância e angulo em que o elemento principal deve se encontrar, dentre uma seleção pré-definida de distâncias e ângulos, com o intuito de gerar imagens com uma variedade maior dentre cada classe.

Em seguida o arquivo `traintest.py` é utilizado para gerar os diferentes modelos classificadores e medir o desempenho dos mesmos.

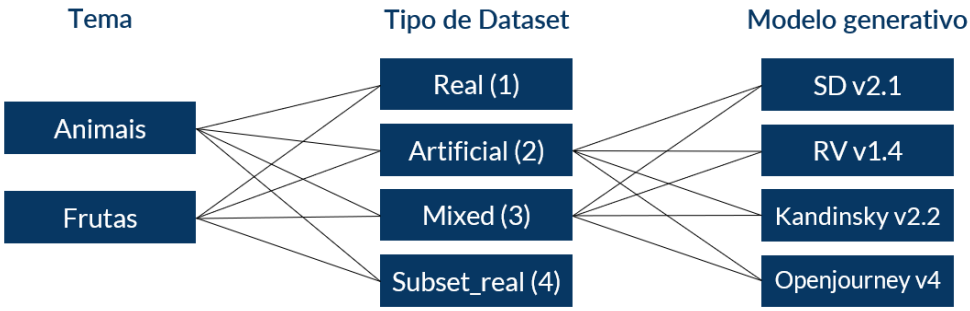


Figura 3.1: *Combinações dos modelos classificadores.*

Foram treinados vinte modelos classificadores diferentes, cada um utilizando uma combinação diferente entre tema, tipo de *dataset*, e modelo generativo. Primeiro é escolhido o tema, podendo ser **Animais** ou **Frutas**, seguindo os temas dos conjuntos de dados reais que foram escolhidos para verificação. Em seguida é escolhido o tipo do conjunto de dados entre quatro tipos diferentes que serão utilizados para comparação: (1) Real - Conjunto de dados contendo apenas imagens reais; (2) Artificial - Conjunto de dados contendo apenas imagens artificiais geradas pelos modelos de difusão; (3) *Mixed* - Conjunto de dados contendo uma proporção 50/50 entre imagens reais e imagens artificiais; (4) Subset_real - Conjunto de dados contendo apenas imagens reais mas com 50% menos imagens em relação aos outros tipos de conjunto. Por último, é selecionado o modelo generativo que foi utilizado no caso dos conjuntos de dados (2) e (3).

Cada um dos conjuntos de dados utilizados no treinamento dos modelos classificadores continham um total de 1000 imagens, com exceção dos conjuntos do tipo (4), que continham um total de 500 imagens. Todos os modelos classificadores foram treinados utilizando a mesma arquitetura e hiperparâmetros. A arquitetura utilizada nos modelos foi a implementação do **Pytorch** da resnet34, essa que é baseada em He *et al.*, 2015. A função erro utilizada foi a *Cross-Entropy Loss*, com o algoritmo de otimização *Adam*. Os modelos foram treinados por 30 épocas, com *batch_size* igual a 32 e *learning_rate* igual 0.001. Todas as imagens utilizadas no treinamento e verificação possuem dimensão 256x256 pixels e foram normalizadas.

3.2 Resultados experimentais

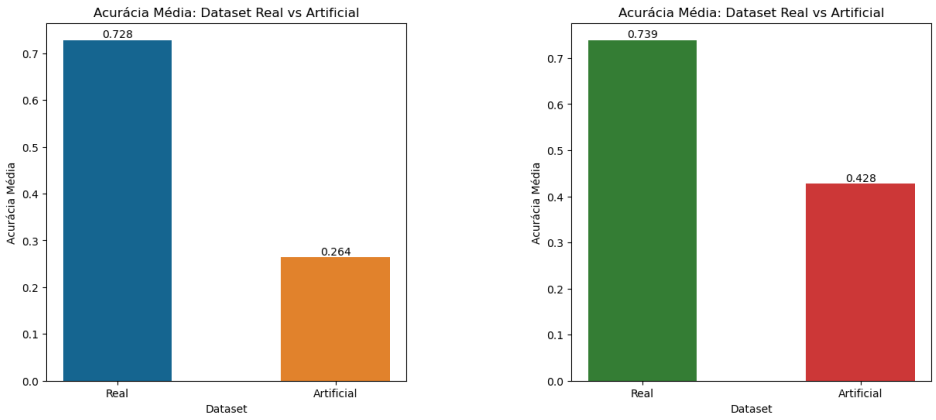
Animais				
	Stable Diffusion v2.1	Realistic Vision v1.4	Openjourney v4	Kandinsky v2.2
(1) Real	0.7281	0.7281	0.7281	0.7281
(2) Artificial	0.3363	0.2518	0.2668	0.1994
(3) Mixed	0.6952	0.6962	0.7020	0.6669
(4) Subset_real	0.6274	0.6274	0.6274	0.6274

Tabela 3.1: *Acurácia dos modelos sobre o tema Animais.*

Frutas				
	Stable Diffusion v2.1	Realistic Vision v1.4	Openjourney v4	Kandinsky v2.2
(1) Real	0.7390	0.7390	0.7390	0.7390
(2) Artificial	0.4443	0.4213	0.4246	0.4210
(3) Mixed	0.6946	0.7163	0.7061	0.6878
(4) Subset_real	0.6910	0.6910	0.6910	0.6910

Tabela 3.2: *Acurácia dos modelos sobre o tema Frutas.*

As tabelas 3.1 e 3.2 contém as medidas de acurácia para os os modelos classificadores de tema Animais e Frutas respectivamente. Como podemos observar, há uma diferença grande entre a acurácia de modelos treinados no conjunto de dados real e nos conjuntos de dados artificiais, com a medida de acurácia do modelo treinado em dados reais sendo em média 2.75 vezes maior em comparação aos treinados em dados artificiais no caso do tema de Animais, e 1.72 vezes maior no caso do tema de Frutas, diferença essa que possivelmente pode ser atribuída à quantidade menor de classes nos conjuntos de dados de frutas, como podemos ver na figura 3.2.

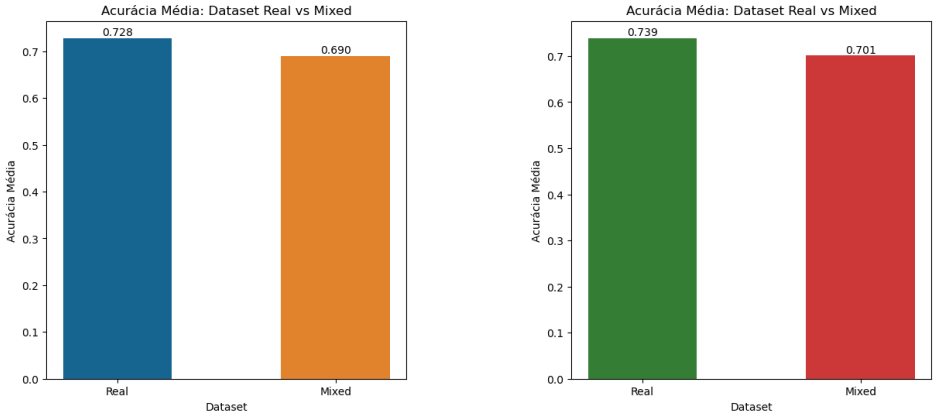


(a) *Real vs Artificial - Animais.* **(b)** *Real vs Artificial - Frutas.*

Figura 3.2: *Comparação entre modelos treinados em conjunto de dados reais e artificiais.*

3.3 Comparações adicionais

3.3.1 Real vs Mixed

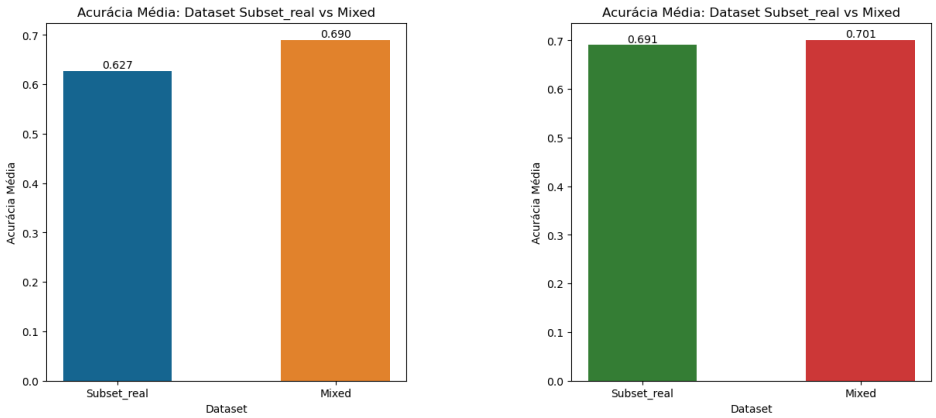


(a) *Real vs Mixed - Animais.* **(b)** *Real vs Mixed - Frutas.*

Figura 3.3: *Comparação entre modelos treinados em conjunto de dados reais e mixed.*

Como podemos observar na figura 3.3, a acurácia média dos modelos treinados nos conjuntos de dados **(3) Mixed** é muito próxima à dos modelos treinados nos conjuntos de dados reais, com menos de 4 pontos percentuais separando-os em ambos os temas. Porém, essa comparação não é suficiente para determinarmos se as imagens artificias do conjunto *Mixed* estão contribuindo de forma relevante para essa proximidade, para isso precisamos da próxima comparação.

3.3.2 Mixed vs Subset_real



(a) *Mixed vs Subset_real - Animais.* **(b)** *Mixed vs Subset_real - Frutas.*

Figura 3.4: *Comparação entre modelos treinados em conjunto de dados mixed e subset_real.*

A figura 3.4 mostra a comparação entre os modelos classificadores treinados nos conjuntos **(3) Mixed** e **(4) Subset_real**. O conjunto (4) foi incluído neste trabalho com a

intenção de servir de base de comparação para o desempenho dos modelos treinados nos conjuntos (3). Essa análise nos permite avaliar o quão relevantes são as imagens artificiais para o conjunto *Mixed*, já que estamos comparando modelos treinados em 500 imagens reais e 500 imagens artificiais contra modelos treinados apenas em 500 imagens reais.

O experimento mostra que, em ambos os casos (Animais e Frutas), as imagens artificiais tiveram um impacto positivo no desempenho dos modelos classificadores. Esse impacto, no entanto, teve uma magnitude muito maior quando tratamos do tema de Animais, se comparado ao tema de Frutas. No caso do tema de Animais, o modelo treinado no conjunto de dados *Mixed* teve um desempenho mais próximo à aquele treinado no conjunto de dados **(1) Real** do que no conjunto **(4) Subset_real**, ou seja, as imagens artificiais foram responsáveis por diminuir em mais da metade a diferença entre um modelo treinado em 500 imagens reais, e um modelo treinado em 1000 imagens reais. Contudo, no caso das Frutas, os modelos treinados nos conjuntos de dados (3) e (4) demonstraram praticamente a mesma performance, com apenas 1 ponto percentual separando-os.

Esse resultado nos leva a crer que imagens artificiais podem auxiliar no treinamento dos classificadores nos casos onde não temos imagens o suficiente para completar a tarefa em questão, já que provavelmente 500 imagens não foram o suficiente para treinar o modelo classificador quando temos 10 classes diferentes (no caso do tema de Animais), mas foram o suficiente para treinar o modelo para reconhecer 5 classes diferentes (no caso do tema de Frutas).

3.3.3 Comparação entre os modelos generativos

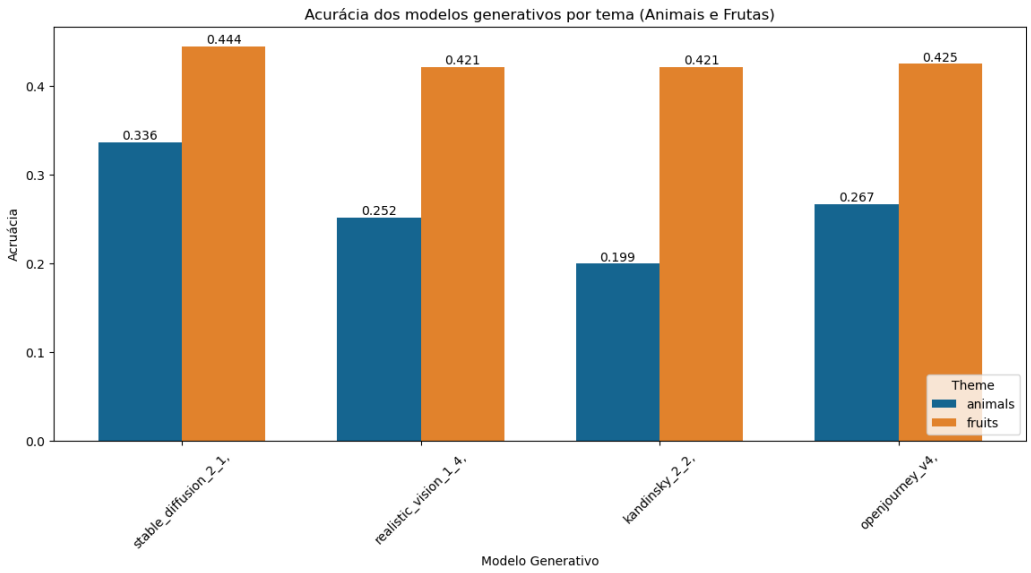


Figura 3.5: *Comparação da acurácia entre os modelos generativos.*

Os resultados sugerem que o modelo generativo mais recente, o Stable Diffusion v2.1, apresenta desempenho superior na tarefa de gerar um conjunto de dados de treinamento, especialmente quando abordamos o tema de Animais, o que pode ser atribuído à capacidade geração de imagens mais complexas do modelo. Curiosamente, a diferença de desempenho

entre os modelos é menos acentuada no tema das frutas, indicando que a complexidade reduzida das imagens de frutas talvez não beneficie tanto das inovações mais recentes quanto as imagens de animais. Isso nos leva a acreditar que a evolução dos modelos generativos impacta de forma variada conforme a complexidade do tema abordado.

Capítulo 4

Conclusão

Este trabalho buscou avaliar se imagens geradas artificialmente podem ser uma alternativa válida à imagens reais no treinamento de modelos classificadores, analisando como os modelos classificadores se comportam quando as imagens artificiais são usadas como substitutas à imagens reais e como se comportam quando são usadas como complemento à imagens reais.

Para isso foram treinados diversos modelos classificadores abrangendo dois temas diferentes, quatro proporções entre imagens reais e imagens artificiais diferentes, e quatro modelos generativos diferentes para gerar os conjuntos de dados artificiais. O desempenho dos modelos foi medido usando a acurácia das predições em conjuntos de dados reais de referência.

Este trabalho concluiu que, embora imagens geradas artificialmente não possam substituir completamente as imagens reais no treinamento de modelos classificadores, elas têm um papel valioso como complemento, especialmente em situações onde há escassez de dados reais. No conjunto de dados que tratava de animais, a insuficiência de exemplos reais limitou a eficácia do modelo classificador, mas a inclusão de imagens artificiais ampliou significativamente a capacidade do modelo de generalizar. Isso indica que, embora os dados reais sejam insubstituíveis para alcançar a máxima precisão, as imagens artificiais servem como um recurso auxiliar efetivo para melhorar a performance em cenários de dados limitados.

A complexidade inerente às imagens de animais exigiu modelos generativos mais complexos para produzir dados artificiais úteis. A influência da qualidade dos modelos generativos foi mais pronunciada neste tema, ressaltando que a fidelidade na geração de imagens complexas é fundamental para o apoio eficaz aos dados reais. Essa observação aponta para a necessidade de desenvolvimento contínuo e especialização de modelos generativos capazes de capturar a riqueza e a complexidade de dados do mundo real, caso esses venham a ser utilizados como uma forma de complementar conjuntos de dados reais.

Referências

[Dhariwal e Nichol 2021] Prafulla Dhariwal e Alex Nichol. *Diffusion Models Beat GANs on Image Synthesis*. 2021. arXiv: 2105.05233 [cs.LG] (citado na pg. 1).

[Grossi e Buscema 2008] Enzo Grossi e Massimo Buscema. "Introduction to artificial neural networks". *European journal of gastroenterology and hepatology* 19 (jan. de 2008), pp. 1046–1054. doi: 10.1097/MEG.0b013e3282f198a0 (citado na pg. 4).

[He *et al.* 2015] Kaiming He, Xiangyu Zhang, Shaoqing Ren e Jian Sun. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV] (citado nas pgs. 10, 11, 30).

[Ho *et al.* 2020] Jonathan Ho, Ajay Jain e Pieter Abbeel. *Denoising Diffusion Probabilistic Models*. 2020. arXiv: 2006.11239 [cs.LG] (citado nas pgs. 13, 17).

[Michelucci 2022] Umberto Michelucci. *An Introduction to Autoencoders*. 2022. arXiv: 2201.03898 [cs.LG] (citado na pg. 14).

[Ronneberger *et al.* 2015] Olaf Ronneberger, Philipp Fischer e Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015. arXiv: 1505.04597 [cs.CV] (citado na pg. 12).

[Vaswani *et al.* 2023] Ashish Vaswani *et al. Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL] (citado nas pgs. 6, 7).

[Zhou *et al.* 2023] Yongchao Zhou, Hshmat Sahak e Jimmy Ba. *Training on Thin Air: Improve Image Classification with Generated Data*. 2023. arXiv: 2305.15316 [cs.CV] (citado na pg. 1).