Universidade de Brasília
Faculdade de Tecnologia
Departamento de Engenharia Elétrica

Imagens Sintéticas para Aperfeiçoar a Geração de Legendas Automáticas

Ricardo Lima Rodrigues

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Engenharia Elétrica

Orientador
Prof. Dr. Díbio Leandro Borges

Brasília
2021

Universidade de Brasília
Faculdade de Tecnologia
Departamento de Engenharia Elétrica

Imagens Sintéticas para Aperfeiçoar a Geração de Legendas Automáticas

Ricardo Lima Rodrigues

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Engenharia Elétrica

Prof. Dr. Díbio Leandro Borges (Orientador)
Universidade de Brasília

Prof.a Dr.a Roberta Barbosa Oliveira
CIC-UnB

Prof. Dr. Wilson Henrique Veneziano
CIC-UnB

Prof. Dr. João Paulo Leite
Coordenador do Bacharelado em Engenharia Elétrica

Brasília, 10 de Novembro de 2021

Dedicatória

Dedico esse TCC a minha mãe, que sempre me apoiou em minhas vontades e decisões

Agradecimentos

Agradeço à todos os professores que me acompanharam e incentivaram nessa jornada, principalmente àqueles que compartilharam seu amor pela física e matemática, me colocando nessa jornada. Também agradeço aos meus pais, que tornaram tudo isso possível.

Resumo

Gerar descrições textuais de imagens tem sido um tópico importante em visão computacional e *Natural Language Processing (NLP)*. Grande parte do problema em se trabalhar com qualquer modelo de *machine learning*, ou *deep learning*, vem da base de dados utilizada, de como esses dados são tratados e de quão bem esses dados são capazes de fazer uma representação geral do problema. Assim, esse trabalho teve como objetivo geral o estudo sobre técnicas para aperfeiçoar a geração de legendas para imagens, utilizando modelos de *deep learning*, pela diversificação do *dataset* utilizado no treinamento do modelo. Foram usados diferentes tipos de *Generative Adversarial Networks* (GANs) para gerar imagens sintéticas, como o AttnGAN, o DM-GAN e suas respectivas versões melhoradas com *contrastive learning*, as quais substituíram imagens reais na base de dados utilizada, sem modificar o tamanho original do *dataset*. Foram então utilizadas diversas métricas como o IS, o FID e o *R-Precision* para analisar o desempenho das diferentes GANs. Com ajuda da pontuação BLEU, foi analisada a performance do gerador de legendas sob o treinamento de diferentes *datasets* com imagens reais e imagens sintéticas geradas pelas diferentes GANs. Assim, pode-se demonstrar o aumento de performance e acurácia do gerador de legendas para imagens reais, unicamente por consequência da introdução de imagens sintéticas na base de dados original. Grande destaque é dado para o DM-GAN ao superar as pontuações de candidatos anteriores e, consequentemente, abrir caminho para o desenvolvimento de um novo modelo de geração de legendas capaz de superar o estado da arte atual.

Palavras-chave: NLP, visão computacional, *deep learning*, GANs, *contrastive learning*, gerador de legendas, imagens sintéticas.

Abstract

Generating textual descriptions of images has been an important topic in computer vision and natural language processing (NLP). The main issue in working with any machine or deep learning model comes from the data acquired, how it's processed and how well this data is able to generalize the problem. Thus, this article's main goal is to study the necessary techniques required to improve image captioning, utilizing deep learning models, through the diversification of data used. Different types of Generative Adversarial Networks (GANs) are utilized to generate synthetic images such as AttnGAN, DM-GAN, and their respective versions improved with contrastive learning. The synthetic images replace real ones in the dataset without modifying its original size. Then, different metrics are utilized to estimate the GANs final performance, such as IS, FID and R-Precision. With BLEU score's help, the efficiency of the image captioning module is analyzed by testing different datasets mixed of real and synthetic images generated by GANs. The goal is to demonstrate increased performance and accuracy of the captioning generator for real images due exclusively to the introduction of synthetic images into the original dataset. Great emphasis is given to the DM-GAN, as it surpasses the scores of previous candidates and, therefore, paves the way of development for a new caption generator capable of surpassing current state of the art approaches.

Keywords: NLP, computer vision, deep learning, GANs, contrastive learning, image captioning, synthetic images.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

AttnGAN Attentional Generative Adversarial Networks.

AttnGAN+CL Attentional Generative Adversarial Networks with Constrastive Learning.

CA Conditioning Augmentation.

CNN Convolutional Neural Network.

DAMSM Deep Attentional Multimodal Similarity Model.

DM-GAN Dynamic Memory Generative Adversarial Networks.

DM-GAN+CL Dynamic Memory Generative Adversarial Networks with Constrastive Learning.

FID Fréchet Inception Distance.

GAN Generative Adversarial Network.

GPU Graphics Processing Unit.

IS Inception Score.

LSTM Long Short Term Memory.

NLP Natural Language Processing.

NT-Xent Normalized Temperature-scaled Cross Entropy Loss.

NWGM Normalized Weighted Geometric Mean.

RNN Recurrent Neural Network.

SNN Siamese Neural Network.

StackGAN Stacked Generative Adversarial Networks.

1 Introdução

Neste capítulo será feita uma breve introdução sobre o surgimento das imagens sintéticas e como isso impactou no desenvolvimento das primeiras GANs. Além disso, será feita uma breve descrição do que é uma GAN e de como as imagens sintéticas podem auxiliar no treinamento e performance de um modelo gerador de imagens.

1.1 A Necessidade de Imagens Sintéticas

A principal tarefa de um cientista de dados, ou engenheiro de dados, é a coleta/criação e tratamento do maior número de dados possível para o treinamento de um modelo visando, consequentemente, atingir uma maior acurácia para tal modelo. Resultados de baixa acurácia e uso ineficiente de determinado modelo podem ser a consequência direta de dados insuficientes ou incompletos. Em quase a totalidade dos casos, o número de imagens, por exemplo, necessário para se treinar um modelo é diretamente proporcional à performance do modelo [3]. É comum considerar que, dependendo do problema em que se está trabalhando, pode-se conseguir um bom dataset pesquisando no Kaggle ou em qualquer outro site de dados, quando na realidade, esse não é o caso. Não se pode esperar resolver problemas do mundo real mediante a obtenção de dados pré-coletados em todas as situações, sendo necessário em alguns casos criar o próprio dataset do zero [4].

A tarefa tem um custo ainda maior quando percebemos a necessidade de dados rotulados na maioria dos processos. A coleta manual de dados é tediosa, mas no caso de uma base de dados textual esse processo pode ser automatizado com bons resultados. A necessidade de uma grande base de dados para se realizar o treinamento de um modelo somada à necessidade da base ter uma boa qualidade e ainda ser rotulada (por exemplo no treinamento de um modelo de geração de legendas), mostra a enorme tarefa que é a preparação de uma boa base de dados de imagens. A tarefa se torna ainda mais difícil quando precisamos não apenas rotular e separar as imagens em grupos, mas também necessitamos de descrições textuais fiéis ao que está acontecendo em cada imagem, com uma certa consistência.

É fácil então perceber a necessidade e demanda que se tem para as imagens sintéticas em um contexto de formação da base de dados. A geração de imagens a partir de uma descrição textual resolve dois problemas de uma vez: primeiramente a questão de um maior número de imagens para o treinamento de um modelo, e também soluciona o problema da necessidade de imagens rotuladas, uma vez que a descrição textual usada para gerá-las já funciona como o próprio rótulo das imagens. É nesse contexto que surgem as GANs, e começam a ser utilizadas imagens sintéticas para auxiliar no treinamento de diferentes modelos geradores de legendas [5][6].

1.2 GANs

Durante muitos anos, o maior sucesso do *deep learning* era relacionado a modelos discriminadores, geralmente os que mapeavam uma entrada sensorial rica e de alta dimensão em um rótulo de classe. Esse sucesso se baseou, primeiramente, em algoritmos de *back-propagation* e *dropout* com um gradiente comportado. Os modelos de geração profunda tinham um impacto menor, devido à dificuldade de se aproximar cálculos probabilísticos intratáveis, modelos que surgiam na estimativa de máxima verossimilhança e estratégias relacionadas. O método mais atual e promissor para a geração de imagens a partir de texto é conhecido como GAN, desenvolvido por Iam Goodfellow [4]. O método é baseado na geração profunda, em um processo onde diferentes redes neurais se desafiam mutualmente, num processo recursivo, para aprender a criar e melhorar conteúdo.

Na estrutura das *adversarial nets* proposta, o modelo gerativo é colocado contra um adversário: um modelo discriminativo que aprende a determinar se uma amostra é da distribuição do modelo ou da distribuição de dados. O modelo gerativo pode ser pensado como análogo a uma equipe de falsificadores tentando produzir uma moeda falsa e usá-la sem serem detectados, enquanto o modelo discriminativo é análogo à polícia, tentando detectar a moeda falsificada [4]. A competição nesse jogo leva ambas as equipes a melhorar seus métodos até que as falsificações sejam indistinguíveis das moedas reais. Essa estrutura pode produzir algoritmos de treinamento específicos para muitos tipos de modelo e algoritmos de otimização. Nas GANs, o modelo gerativo gera amostras ao passar um ruído aleatório por um perceptron multicamadas, sendo o modelo discriminativo também um perceptron multicamadas. Esse caso especial é chamado de *adversarial Networks*. Nesse caso, ambos os modelos podem ser treinados usando apenas os algoritmos de *back-propagation* e *dropout*, altamente bem-sucedidos, e colhem amostras do modelo gerativo utilizando apenas a *forward propagation*. Nenhuma inferência aproximada ou cadeias de Markov são necessárias.

1.3 O desafio da Legenda de Imagens

Hoje, no desenvolvimento e aperfeiçoamento do *deep learning* baseia-se em dois ramos que são considerados os principais nos quesitos pesquisa, desenvolvimento e interesse público: o ramo dos classificadores de imagens, e os classificadores de texto baseados em NLP. A tarefa de legendar imagens agrega então os dois principais ramos de pesquisa em inteligência computacional. A tarefa de gerar descrições textuais de imagens exige um amplo conhecimento, tanto no tratamento e classificação de imagens, quanto no entendimento textual dos classificadores de linguagem natural. Além disso, necessita do entendimento conjunto do que uma legenda representa na imagem, e vice-versa.

Gerar automaticamente legendas para uma imagem é uma tarefa muito perto do entendimento central de uma cena, um dos principais objetivos da *computer vision* (visão computacional). Não apenas os modelos de geração de legendas devem ser poderosos o suficiente para resolver os desafios de *computer vision* de determinar quais objetos estão numa imagem, mas também devem ser capazes de capturar e expressar sua relação em linguagem natural [7]. É um desafio extremamente importante para os algoritmos de *machine learning*, que seria equivalente a imitar a notável habilidade humana de compactar grandes quantidades de informações visuais em linguagem descritiva.

Como já dito anteriormente, a validade e performance de qualquer modelo depende, antes de tudo, da base de dados utilizada para treinar e testar o modelo. Nesse trabalho sera demonstrado como uma base de dados composta de imagens reais e imagens sintéticas, geradas por GANs, pode aumentar a performance de um gerador de legendas para imagens reais, gerando descrições melhores e mais detalhadas do que um gerador de legendas treinado apenas por imagens reais.

Assim, serão testadas o AttnGAN [8], o DM-GAN [9], e suas respectivas versões melhoradas com *contrastive learning* [1], no treinamento do gerador de legendas para imagens. Hossain *et al.* [3] fez algo similar com o AttnGAN, tema que será melhor tratado no próximo capítulo, onde é feita uma breve revisão dos artigos que inspiraram o desenvolvimento e realização deste trabalho.

2 Revisão de Literatura

Recentemente, um grande número de estudos [4][8][9][10] apresentaram resultados promissores na tarefa de síntese de imagens a partir de texto, a maioria dos quais usa GANs como o modelo de partida. Zhang *et al.* [10] propõem a arquitetura da *Stacked Generative Adversarial Networks (StackGAN)*, que produz imagens de baixa a alta resolução. O *Attentional Generative Adversarial Networks (AttnGAN)* [8] apresenta um mecanismo de atenção e, devido ao uso do *Deep Attentional Multimodal Similarity Model (DAMSM)*, é capaz de computar a similaridade entre a imagem gerada e a legenda, utilizando-se tanto da informação de nível global de frase, quanto da informação de nível de palavra de baixa granularidade. O *Dynamic Memory Generative Adversarial Networks (DM-GAN)* [9] introduz uma *adversarial network* com memória dinâmica para gerar imagens de alta qualidade. Ele utiliza um módulo de memória dinâmica para refinar a imagem gerada inicialmente, uma porta de gravação de memória para destacar as informações de texto relevantes e uma porta de resposta para atualizar as representações das imagens. Em todos esses modelos, pode-se ver uma clara melhora na abordagem das *adversarial networks* para a geração de imagens sintéticas, em cada instância com uma performance melhor que a anterior. Primeiramente, a introdução das *adversarial networks* por Goodfellow *et al.* [4], seguida da melhora do modelo pela utilização do método de atenção por Xu *et al.* [8] e seguida pela introdução da memória dinâmica por Zhu *et al.* [9].

A abordagem de Ye *et al.* [1] é derivada do desenvolvimento recente do *contrastive learning*, e portanto, tem a vantagem de melhor desempenho e menor custo computacional. Além disso, a abordagem é generalizada para que possa ser aplicada aos modelos existentes baseados em GANs para síntese de imagens baseada em texto. Ye *et al.* [1] estuda o AttnGAN [8] e o DM-GAN [9] como os modelos de base para avaliar a abordagem do *contrastive learning*, os mesmos modelos escolhidos nesta monografia para se estudar o impacto de imagens falsas na geração de legendas para imagens. O uso do *contrastive learning* reflete a gradual melhora das imagens sintéticas ao longo dos anos, e hoje é o estado da arte para geração de imagens falsas, mais especificamente, o *Dynamic Memory Generative Adversarial Networks with Constrastive Learning (DM-GAN+CL)* [1].

Com a melhora dos modelos de geração de imagens sintéticas, também houve melhora

nos modelos de geração de legendas. Hossain *et al.* [11] apresenta uma abrangente pesquisa e estudo sobre o tema, onde os métodos são agrupados em várias categorias como: legenda de imagens baseada em templates, legenda de imagem baseada em recuperação e nova geração de legendas. As novas legendas podem ser geradas tanto do espaço visual [12] quanto do espaço multimodal [7]. Um método típico dessa categoria analisa primeiramente o conteúdo visual da imagem e, em seguida, gera as legendas da imagem usando um modelo de linguagem. A maioria dos métodos dessa categoria usa uma arquitetura de codificador-decodificador para gerar legendas de imagens [12]. Nesses métodos, uma *Convolutional Neural Network (CNN)* padrão é utilizada como módulo codificador na extração das representações da imagem e uma *Long Short Term Memory (LSTM)* é utilizada como decodificador para gerar legendas usando essas representações. No entanto, esses métodos têm problemas na identificação de objetos proeminentes da imagem.

Métodos baseados em atenção [3][7] podem representar os objetos proeminentes das legendas pois focam seletivamente nos objetos relevantes de uma imagem. Assim, neste trabalho, foi utilizado um método baseado em atenção para gerar as legendas descritivas de uma imagem.

Esses métodos de legenda de imagens baseados em *deep learning* usam, popularmente, três conjuntos comuns de dados, disponíveis publicamente: o MS COCO [2] (o mesmo utilizado neste trabalho), o Flickr30k e o Flickr8k para treinar e testar as redes. Esses conjuntos de dados foram coletados, e as descrições textuais (legendas) foram feitas por humanos. No entanto segundo Hossain *et al.* [3], os métodos baseados em *deep learning* têm alguns problemas para trabalhar com esses dados:

- Esses métodos necessitam de um conjunto de dados grande e diverso para aprenderem as representações visuais;
- Ocorre *overfitting* nos modelos existentes para objetos comuns que co-ocorrem em um mesmo contexto. Por exemplo, se um modelo é treinado para uma cena que contém uma cama e um quarto, mas é testado em contextos ainda não vistos, como por exemplo, uma cama e uma floresta, o modelo terá dificuldade em generalizar para essas cenas;
- Rotular manualmente um grande volume de dados é caro, tendencioso e demorado, como já mencionado anteriormente.

Dados sintéticos podem ser uma ótima opção para contornar esses problemas. Assim como neste artigo, os trabalhos de Hossain *et al.* [3], Dai *et al.* [5] e Chen *et al.* [6] também utilizam GANs na legenda de imagens. Dai *et al.* [5] utilizou GANs condicionais (CGAN) para avaliar as legendas geradas, onde visam melhorar a naturalidade e diversidade das

legendas. Do mesmo modo, Chen *et al.* [6] usou a mesma CGAN com *reinforced learning* para lidar com a avaliação inconsistente. Hossain *et al.* [3] se utilizou da AttnGAN [8], em um *dataset* misto de imagens falsas e reais, com um mecanismo de atenção para chegar no estado da arte da legenda de imagens.

A proposta desse trabalho é lidar com as inconsistências do *dataset* pelo uso de imagens falsas e reais, assim como Hossain *et al.* [3], mas dando ênfase à conhecida divergência entre descrições textuais gerada pelas anotações humanas no *dataset*, fato melhor explicado na Seção 3.4.2. Além da AttnGAN [8] utilizada por Hossain *et al.* [3], será também utilizada a DM-GAN de Zhu *et al.* [9] e as abordagens com *contrastive learning* AttnGAN+CL e DM-GAN+CL de Ye *et al.* [1]. Dessa maneira, espera-se que o DM-GAN supere a abordagem da AttnGAN na utilização de *datasets* com imagens reais e sintéticas, e que o uso do *contrastive learning* aumente ainda mais a performance desses modelos.

No próximo capítulo serão explicados os métodos utilizados para desenvolver esse trabalho.

3 Materiais e Métodos

Esta seção descreve a metodologia e a teoria que sustentam os modelos e métodos implementados, e será dividida em duas partes:

1. A primeira relacionada à geração de imagens sintéticas utilizando GANs [1][8][9];
2. A segunda relacionada à arquitetura do modelo de *deep learning* utilizado para se gerar legendas automáticas para imagens [3][7][13].

3.1 Generative Adversarial Networks

A tarefa de gerar imagens automaticamente estando em acordo com descrições em NLP é um problema fundamental em muitas aplicações, como a geração de arte, mostrado na Figura 3.1, e design auxiliado por computador. Também promove a pesquisa em aprendizado multimodal e une duas das áreas mais pesquisadas atualmente em inteligência computacional: imagens e linguagem.

Assim, foram utilizadas quatro tipos de GAN para desenvolver este trabalho:

1. AttnGAN [8], Seção 3.2;

Figura 3.1: O Deep Learning cria imagens sintéticas bizarras e fascinantes (Fonte: `https://deepdreamgenerator.com/`).

2. DM-GAN [9], Seção 3.3;
3. AttnGAN+CL [1], Seção 3.4.2;
4. DM-GAN+CL [1], Seção 3.4.2.

3.2 Attentional Generative Adversarial Networks

Xu *et al.* [8] ressignificou em 2018 a síntese de imagens para um dos modelos mais bem sucedidos e reconhecidos atualmente, baseado em métodos de atenção. A AttnGAN consegue sintetizar, em diferentes subregiões da imagem, detalhes refinados (baixa granularidade), prestando atenção nas palavras relevantes presentes nas descrições em NLP. É ainda proposto um DAMSM para calcular a perda de correspondência entre texto e imagem para treinar o gerador.

O AttnGAN proposto[1] supera significativamente o estado da arte anterior, aumentando o melhor *inception score* em 14,14% no *dataset* CUB e 170,25% no *dataset* COCO [2], o mesmo utilizado neste trabalho, e também o mais desafiador. A Figura 3.2 traz uma representação da arquitetura do modelo do AttnGAN.

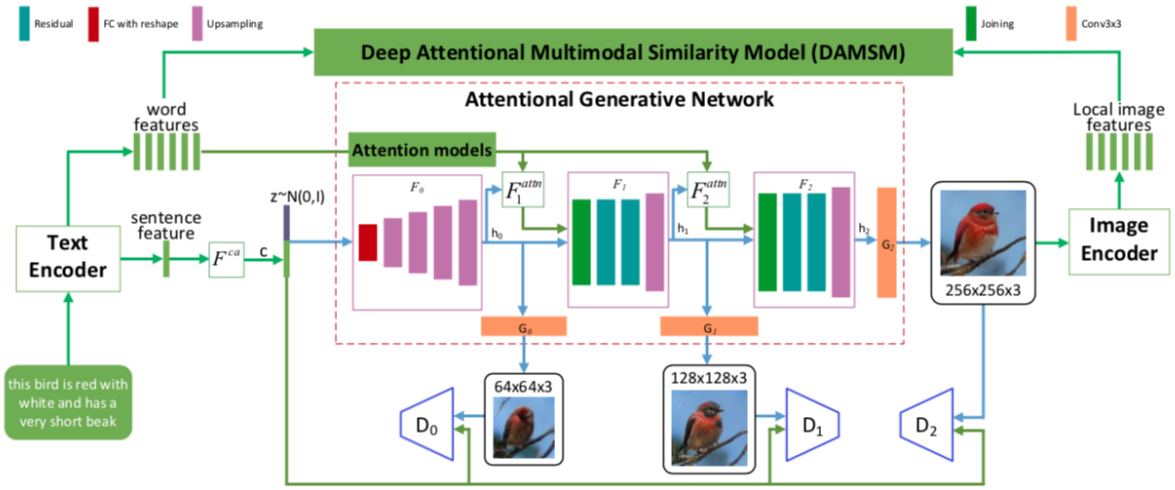


Figura 3.2: A arquitetura do AttnGAN proposto. Cada modelo de atenção recupera automaticamente as condições (ou seja, os vetores de palavras mais relevantes) para gerar diferentes sub-regiões da imagem. O DAMSM fornece o *matching loss* de imagem-texto para a rede gerativa (Fonte: [8]).

[1]O código completo da AttnGAN está disponível em: `https://github.com/taoxugit/AttnGAN`

3.2.1 Attentional Generative Network

Os modelos de GANs atuais para geração de imagens a partir de texto [4][10] geralmente codificam a descrição do texto da frase completa em um único vetor como a condição para a geração da imagem, mas carecem de informações dependentes de palavras relevantes. Com esse modelo baseado em atenção, é possível permitir à rede desenhar diferentes sub-regiões da imagem, condicionadas às palavras mais relevantes destas sub-regiões.

Como podemos ver na Figura 3.2, a GAN proposta possui m geradores $(G_0, G_1, ..., G_{m-1})$, que recebem como entrada os estados ocultos (*hidden states*) $(h_0, h_1, ..., h_{m-1})$, e geram imagens de escala pequena para grande $(\hat{x}_0, \hat{x}_1, ..., \hat{x}_{m-1})$, especificamente:

$$\begin{aligned} h_0 &= F_0(z, F^{ca}(\bar{e}) \\ h_i &= F_i(h_{i-1}, F_i^{attn}(e, h_{i-1})), \text{ para } i = 1, 2, ..., m-1 \\ \hat{x}_i &= G_i(h_i) \end{aligned} \tag{3.1}$$

Aqui, z é um vetor de ruído, geralmente amostrado de uma distribuição normal padrão. A variável $\bar{e}$ é um vetor de frases global, e *"e"* é a matriz de vetores de palavras. F^{ca} representa o Conditioning Augmentation (CA) [10] que converte o vetor de frases $\bar{e}$ para o vetor condicionante. F_i^{attn} é o modelo de atenção proposto no i-ésimo estágio da AttnGAN. F^{ca}, F_i^{attn}, F_i e G_i são modelados como redes neurais.

O modelo de atenção $F^{attn}(e, h)$ possui duas entradas: as características (*features*) de palavras $e \in \mathbb{R}^{D \times T}$ e as características de imagens da última camada oculta $h \in \mathbb{R}^{\hat{\mathrm{D}} \times \mathrm{N}}$. As características de palavras são primeiramente convertidas dentro do espaço semântico comum das características de imagens por meio da adição de uma nova camada perceptron, i.e., $e' = Ue$, onde $U \in \mathbb{R}^{\hat{\mathrm{D}} \times \mathrm{D}}$. Em seguida, um vetor contexto de palavras é calculado para cada sub-região da imagem, baseado em suas características ocultas h. Cada coluna de h é um vetor característica de uma sub-região da imagem. Para a j-ésima sub-região, o seu vetor contexto de palavra é uma representação dinâmica de vetores de palavra relevantes para h_j, que é calculado por:

$$c_j = \sum_{i=0}^{T-1} \beta_{j,i} e'_i, \text{ onde } \beta_{j,i} = \frac{exp(s'_{j,i})}{\sum_{k=0}^{T-1} exp(s'_{j,k})} \tag{3.2}$$

$s'_{j,i} = h_j^T e'_i$, e $\beta_{j,i}$ indica o peso que o modelo impõe à i-ésima palavra ao gerar a j-ésima sub-região da imagem. Em seguida, a matriz de contexto de palavra para o conjunto de características de imagem h é inserida em $F^{attn}(e, h) = (c_0, c_1, ..., c_{N-1}) \in \mathbb{R}^{\hat{\mathrm{D}} \times \mathrm{N}}$. Finalmente, as características de imagem e as características de contexto-palavra correspondentes são combinadas para a geração de imagens no próximo estágio.

Para gerar imagens realistas com múltiplos níveis (i.e., nível de frase e nível de palavra) de condições, a função objetiva final da rede geradora de atenção é definida como:

$$\mathcal{L} = \mathcal{L}_G + \lambda \mathcal{L}_{DAMSM}, \text{ onde } \mathcal{L}_G = \sum_{i=0}^{m-1} \mathcal{L}_{G_i} \tag{3.3}$$

Aqui, λ é um hiper-parâmetro para balancear os termos da Equação 3.3. O primeiro termo é a perda da GAN (GAN *loss*) que, em conjunto, se aproxima das distribuições condicionais e incondicionais. No i-ésimo estágio da AttnGAN, o gerador G_i tem um discriminador correspondente D_i. A perda adversária (*adversarial loss*) para G_i é definida como:

$$\mathcal{L}_{G_i} = \underbrace{-\frac{1}{2}E_{\hat{x}_i \sim p_{G_i}}[log(D_i(\hat{x}_i))]}_{\text{perda incondicional}} \underbrace{-\frac{1}{2}E_{\hat{x}_i \sim p_{G_i}}[log(D_i(\hat{x}_i, \bar{e}))]}_{\text{perda condicional}}, \tag{3.4}$$

onde a perda incondicional determina se a imagem é real ou falsa, enquanto a perda condicional determina se a imagem e frase são correspondentes ou não.

Alternadamente ao treinamento de G_i, cada discriminador D_i é treinado para classificar a entrada entre as classes real ou falsa pela minimização da perda de entropia cruzada (*cross-entropy loss*), definida por:

$$\begin{aligned} \mathcal{L}_{D_i} = & \underbrace{-\frac{1}{2}E_{x_i \sim p_{data_i}}[\log D_i(x_i)] - \frac{1}{2}E_{\hat{x}_i \sim p_{G_i}}[\log(1 - D_i(\hat{x}_i))]+}_{\text{perda incondicional}} \\ & \underbrace{-\frac{1}{2}E_{x_i \sim p_{data_i}}[\log D_i(x_i, \bar{e})] - \frac{1}{2}E_{\hat{x}_i \sim p_{G_i}}[\log(1 - D_i(\hat{x}_i, \bar{e}))]}_{\text{perda condicional}}, \end{aligned} \tag{3.5}$$

onde x_i vem da verdadeira distribuição de imagem p_{data_i} na i-ésima escala, e $\hat{x}_i$ vem da distribuição do modelo p_{G_i}, na mesma escala. Os discriminadores da AttnGAN são estruturalmente separados e, portanto, podem ser treinados em paralelo, e cada um deles concentra-se em uma única escala de imagem.

O segundo termo da Equação 3.3 $\mathcal{L}_{DAMSM}$ é a perda de correspondência (*matching loss*) do nível de palavra relevante da imagem-texto calculada pelo DAMSM, que será elaborado na Subseção 3.2.2.

3.2.2 Deep Attentional Multimodal Similarity Model

O DAMSM é responsável pelo aprendizado de duas redes neurais que mapeiam sub-regiões da imagem e palavras da frase para um espaço semântico comum, medindo assim

a similaridade imagem-texto no nível da palavra, com o intuito de calcular a *fine-grained loss* [8] para a geração da imagem.

O codificador de texto é uma LSTM bidirecional [8] que extrai vetores semânticos da descrição do texto. No LSTM bidirecional, cada palavra corresponde a dois estados ocultos, um para cada direção. Então, os dois estados ocultos são concatenados para representar o significado semântico de uma palavra. A matriz de características de todas as palavras é indicada por $e \in \mathbb{R}^{\hat{\mathrm{D}} \times \mathrm{T}}$. A sua i-ésima coluna e_i é o vetor de características para a i-ésima palavra. D é a dimensão do vetor palavra e T é o número de palavras. Enquanto isso, os últimos estados ocultos do LSTM bidirecional são concatenados como o vetor de frases global, denotado por $\bar{e} \in \mathbb{R}^{\mathrm{D}}$.

O codificador de imagem é uma CNN que mapeia as imagens para vetores semânticos. As camadas intermediárias da CNN aprendem características locais de diferentes sub-regiões da imagem, enquanto as camadas posteriores aprendem características globais da imagem. Mais especificamente, o codificador de imagem é construído em cima do modelo Inception-v3 pré treinando no ImageNet, referenciado por Xu *et al.* [8], e extremamente similar ao modelo utilizado para extrair os vetores característica da imagem por Hossain *et al.* [3]. Primeiramente, a imagem de entrada é redimensionada para 299×299 *pixels*, depois é extraída a matriz local de características $f \in \mathbb{R}^{768 \times 289}$ (redimensionada de $768 \times 17 \times 17$) da camada *"mixed_6e"* do Inception-v3. Cada coluna de f é o vetor característica de uma sub-região da imagem, 768 é a dimensão do vetor característica local e 289 é o número de sub-regiões dentro da imagem. Enquanto isso, o vetor característica global $\hat{f} \in \mathbb{R}^{2048}$ é extraído da última camada da média de *pooling* do Inception-v3. Finalmente, convertemos as características da imagem para um espaço semântico comum de características de texto pela adição de uma camada perceptron:

$$v = Wf, \quad \overline{v} = \overline{W}\bar{f}, \tag{3.6}$$

onde $v \in \mathbb{R}^{\mathrm{D} \times 289}$, e sua i-ésima coluna v_i é o vetor característica visual para a i-ésima sub-região da imagem, onde $\overline{v} \in \mathbb{R}^{\mathrm{D}}$ é o vetor global para a imagem completa. D é a dimensão do espaço de características multimodal (i.e., modalidades de texto e imagem). Para eficiência, todos os parâmetros pertencentes às camadas construídas pelo modelo Inception-v3 são fixadas, e os parâmetros pertencentes às camadas recentemente adicionadas são aprendidos em conjunto com o resto da rede.

O *matching score* imagem-texto orientado por atenção é desenvolvido para medir a correspondência entre um par frase-imagem baseado em um modelo de atenção entre imagem e texto.

Primeiramente é calculada a matriz de similaridade para todos os possíveis pares de palavras na frase e em sub-regiões dentro da imagem por:

$$s = e^T v, \tag{3.7}$$

onde $s \in \mathbb{R}^{\mathrm{T}\times 289}$ e $s_{i,j}$ é o produto escalar de similaridade entre a i-ésima palavra da frase e a j-ésima sub-região da imagem. Segundo [8], é benéfica a normalização da matriz de similaridade como:

$$\overline{s}_{i,j} = \frac{exp(s_{i,j})}{\sum_{k=0}^{T-1} exp(s_{k,j})} \tag{3.8}$$

Então, é construído um modelo de atenção para calcular um vetor contexto-região para cada palavra. O vetor contexto-região c_i é uma representação dinâmica das sub-regiões da imagem, relacionado com a i-ésima palavra da frase. É calculado como a soma ponderada sobre todos vetores regionais visuais:

$$c_i = \sum_{j=0}^{288} \alpha_j v_j, \text{ onde } \alpha_j = \frac{exp(\gamma_1 \overline{s}_{i,j})}{\sum_{k=0}^{288} exp(\gamma_1 \overline{s}_{i,k})} \tag{3.9}$$

Aqui, γ_1 é um fator que determina quanta atenção é dada para as características das sub-regiões relevantes quando se calcula o vetor contexto-região para uma palavra.

Finalmente, é definida a relevância entre a i-ésima palavra e a imagem usando a semelhança entre os cossenos de c_i e e_i, i.e., $R(c_i, e_i) = (c_i^T e_i)/(\|c_i\|\|e_i\|)$. Inspirado na formulação de erro mínimo de classificação em reconhecimento de fala, Xu *et al.* [8] define a pontuação de correspondência imagem-texto orientado por atenção (*attention-driven image-text matching score*) entre a imagem completa *(Q)* e a descrição completa de texto *(D)* como:

$$R(Q, D) = \log(\sum_{i=1}^{T-1} exp(\gamma_2 R(c_i, e_i)))^{\frac{1}{\gamma_2}}, \tag{3.10}$$

onde γ_2 é um fator que determina o quanto se deve aumentar a importância do par contexto-palavra para região mais relevante. Quando $\gamma_2 \to \infty$, $R(Q, D)$ se aproxima de $max_{i=1}^{T-1} R(c_i, e_i)$.

A perda DAMSM (DAMSM *loss*) é projetada para aprender o modelo de atenção de uma forma semi-supervisionada, onde a única supervisão é a correspondência entre imagens completas e frases inteiras (uma sequência de palavras). Muito similar com o que foi apresentado por Fang *et al.* [14], para um *batch* de pares frase-imagem $\{(Q_i, D_i)\}_{i=1}^{M}$, a probabilidade condicional da frase D_i ser correspondente à imagem Q_i é calculada como:

$$P(D_i|Q_i) = \frac{exp(\gamma_3 R(Q_i, D_i))}{\sum_{j=1}^{M} exp(\gamma_3 R(Q_i, D_j))}, \tag{3.11}$$

onde γ_3 é um fator experimental de suavização. Nesse *batch* de palavras, apenas D_i corresponde à imagem Q_i, e todas as outras *M-1* frases são tratadas como descrições não-correspondentes. Seguindo Fang *et al.* [14] a função de perda (*loss function*) é definida como o negativo do logaritmo da probabilidade condicional das imagens combinarem com suas descrições de texto correspondentes:

$$\mathcal{L}_1^w = -\sum_{i=1}^{M} \log P(D_i|Q_i), \tag{3.12}$$

onde *"w"* representa a palavra (*word*).

Simetricamente, minimizamos:

$$\mathcal{L}_2^w = -\sum_{i=1}^{M} \log P(Q_i|D_i), \tag{3.13}$$

onde $P(Q_i|D_i) = \frac{exp(\gamma_3 R(Q_i,D_i))}{\sum_{j=1}^{M} exp(\gamma_3 R(Q_j,D_i))}$ é a probabilidade condicional de que a frase D_i é combinada com sua imagem correspondente Q_i. Se redefinirmos a Equação 3.10 como $R(Q,D) = (\overline{v}^T\overline{e})/(\|\overline{v}\|\|\overline{e}\|)$ e a substituirmos nas Equações 3.11,3.12 e 3.13, pode-se obter as funções de perda (*loss functions*) $\mathcal{L}_1^s$ e $\mathcal{L}_2^s$ (onde *"s"* representa a frase) usando o vetor frases $\overline{e}$ e o vetor global de imagens $\overline{v}$.

Finalmente, o DAMSM *loss* é definido como:

$$\mathcal{L}_{DAMSM} = \mathcal{L}_1^w + \mathcal{L}_2^w + \mathcal{L}_1^s + \mathcal{L}_2^s \tag{3.14}$$

Empiricamente baseado num *dataset* de validação deixado de fora do treinamento principal, os hiper-parâmetros dessa seção foram definidos como: '$\gamma_1 = 5,\ \gamma_2 = 5,\ \gamma_3 = 10$ e $M = 50$. O DAMSM é pré-treinado pela minimização de $\mathcal{L}_{DAMSM}$ usando pares reais texto-imagem. Como o tamanho das imagens para o pré treinamento da DAMSM não é limitado pelo tamanho das imagens que podem ser geradas, imagens reais de tamanho 299×299 foram utilizadas. Ainda, o codificador pré-treinado do DAMSM fornece vetores palavra visualmente discriminativos aprendidos a partir de dados de pares imagem-texto para a GAN.

Assim, foram propostos dois modelos que atuam de formas distintas na AttnGAN. (i) O mecanismo de atenção na rede gerativa (Equação 3.2) permite que a AttnGAN selecione automaticamente a condição de nível da palavra para a geração de diferentes sub-regiões da imagem, e (ii) com um mecanismo de atenção (Equação 3.9), onde o DAMSM é capaz de calcular o *matching loss* imagem-texto refinado $\mathcal{L}_{DAMSM}$. É importante mencionar que $\mathcal{L}_{DAMSM}$ é aplicado apenas na saída do último gerador G_{m-1}, já que a eventual meta do AttnGAN é gerar imagens grandes até o último gerador.

3.3 Dynamic Memory Generative Adversarial Networks

Como vimos anteriormente, o AttnGAN [8] pode ser considerado um método de múltiplos estágios para a geração de imagens, assim como o StackGAN [10]: primeiramente geram imagens iniciais de baixa resolução e depois refinam essas imagens para alta resolução.

Apesar dos métodos de estágios múltiplos terem alcançado progresso extraordinário no entendimento da relação entre conteúdo visual e linguagem natural, como inicialmente proposto por Goodfellow *et al.* [4], ainda existem dois problemas:

1. O resultado da geração é altamente dependente da qualidade inicial das imagens. O processo de refinamento das imagens não consegue produzir alta resolução se as imagens iniciais forem ruins;

2. Cada palavra numa dada frase possui um nível diferente de informação retratando o conteúdo da imagem. Os modelos atuais utilizam a mesma representação de palavras em diferentes processos de refinamento de imagem, o que torna o processo de refinamento inefetivo. A informação da imagem deve ser levada em consideração para determinar a importância de cada palavra para o refinamento.

Assim, a contribuição da DM-GAN[2] em comparação com modelos anteriores [8][10] é:

- Propor um novo modelo de GAN combinado com um componente de memoria dinâmica para gerar imagens de alta qualidade, mesmo se as imagens inicias não forem geradas propriamente;

- Propor um *gate* de escrita de memória capaz de selecionar palavras relevantes de acordo com a imagem inicial;

- Um *gate* de resposta é proposto para adaptativamente fundir informação da imagem e memória;

- Os resultados experimentais mostram que a DM-GAN supera o estado da arte de outras abordagens anteriores [8][10].

Como mostrado na Figura 3.3, a arquitetura do modelo DM-GAN é composta de dois estágios: geração da imagem inicial e refinamento de imagem baseado em memoria dinâmica.

No estágio de geração da imagem inicial, primeiramente, o texto de entrada é transformado numa representação interna (uma característica de frase s e várias características de palavra W) por um codificador de texto. Então, um *deep conventional generator* prevê uma imagem inicial x_0 áspera e pouco detalhada, de acordo com a característica da frase

[2]Todo o código do modelo DM-GAN está disponível em `https://github.com/MinfengZhu/DM-GAN`

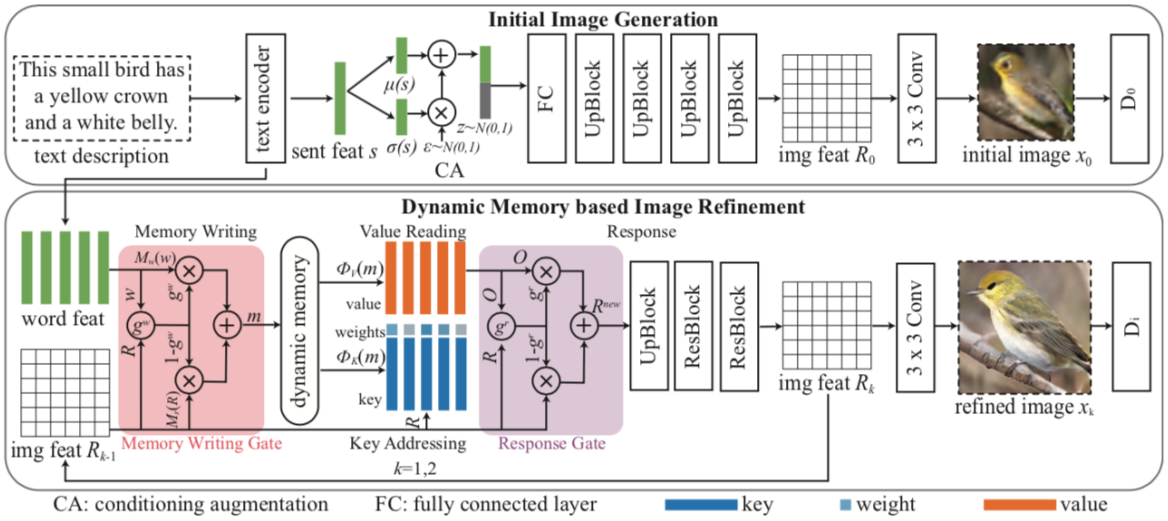


Figura 3.3: A arquitetura do DM-GAN para síntese de imagens a partir de texto. O modelo primeiramente gera uma imagem inicial, e depois refina a imagem para gerar uma de alta qualidade. (Fonte: [9]).

e um vetor de ruído aleatório $z : x_0, R_0 = G_0(z, s)$, onde R_0 é a característica da imagem. O vetor de ruído é amostrado de uma distribuição normal.

No estágio de refinamento de imagem baseado em memoria dinâmica, mais refinamento é adicionado ao conteúdo visual das imagens inicias difusas para gerar uma imagem foto-realista $x_i : x_i = G_i(R_{i-1}, W)$, onde R_{i-1} é a característica da imagem do último estágio. O estágio de refinamento pode ser repetido diversas vezes para recuperar informações mais pertinentes e gerar uma imagem de alta resolução com detalhes mais refinados.

O estágio de refinamento da imagem baseado em memoria dinâmica é dividido em quatro componentes: Escrita de Memória, Endereçamento de chave, Leitura de Valor e Resposta. A operação de Escrita de Memória guarda a informação do texto numa estrutura de memória de valor chave para recuperação posterior. Então, as operações de Endereçamento de chave e Leitura de Valor são utilizadas para refinar as características visuais de imagens de baixa qualidade pela leitura de características do módulo de memória. Finalmente, a operação de Resposta é adotada para controlar a fusão entre características da imagem e a leitura da memória. É proposto um *gate* de escrita na memória para destacar informações de palavras importantes, de acordo com o conteúdo da imagem na etapa de escrita da memória.

3.3.1 Memória Dinâmica

Começando com a entrada de representação de palavras W, imagem x e características da imagem R_i:

$$W = \{w_1, w_2, ..., w_T\}, w_i \in \mathbb{R}^{\mathrm{nw}} \tag{3.15}$$

$$R_i = \{r_1, r_2, ..., r_N\}, r_i \in \mathbb{R}^{\mathrm{N_r}}, \tag{3.16}$$

onde T é o número de palavras, N_w é a dimensão das características de palavra, N é o número de *pixels* na imagem e a característica de *pixels* na imagem é um vetor com dimensão N_r. A intenção é o aprendizado de um modelo de refinamento de imagem utilizando uma maneira mais eficaz de unir texto e informação: por meio de transformações não triviais entre as memórias de chave e valor. O estágio de refinamento inclui as quatro etapas a seguir.

Escrita de Memória: codificar o conhecimento prévio é uma parte importante da memória dinâmica, que permite a recuperação de imagens de alta qualidade a partir do texto. Uma maneira de se escrever a memória é considerando apenas informações parciais do texto:

$$m_i = M(w_i), m_i \in \mathbb{R}^{\mathrm{N_m}}, \tag{3.17}$$

onde $M(\cdot)$ denota a convolução 1×1 que incorpora características de palavra no espaço de características de memória com dimensões N_m.

Endereçamento de Chave: nessa etapa, memórias relevantes são recuperadas utilizando a memória de chave. Calculamos um peso para cada slot de memória, como a probabilidade de similaridade entre o slot de memória m_i e a característica de imagem r_j:

$$\alpha_{i,j} = \frac{exp(\phi_K(m_i)^T r_j)}{\sum_{l=1}^{T} exp(\phi_K(m_l)^T r_j)}, \tag{3.18}$$

onde $\alpha_{i,j}$ é a probabilidade de similaridade entre a i-ésima memória e a j-ésima característica de imagem, e $\phi_K()$ é o processo de acesso à memória chave que mapeia as características da memória na dimensão N_r. $\phi_K()$ é implementado como uma convolução 1×1.

Leitura de Valor: a representação da memória de saída é definida como a soma ponderada de memórias de valor de acordo com a probabilidade de similaridade:

$$o_j = \sum_{i=1}^{T} \alpha_{i,j} \phi_V(m_i), \tag{3.19}$$

onde $\phi_V()$ é o processo de acesso à memória de valor que mapeia as características da memória na dimensão N_r. $\phi_V()$ é implementado como uma convolução 1×1.

Resposta: Depois de receber a memória de saída, a imagem atual e a representação de saída são combinadas para fornecer uma nova característica de imagem. Uma nova abordagem será simplesmente concatenar as características da imagem e a representação de saída. As novas características de imagem são obtidas por:

$$r_i^{new} = [o_i, r_i], \tag{3.20}$$

onde $[\cdot, \cdot]$ representa a operação de concatenação. Então, podemos utilizar um bloco de *upsampling* e vários blocos residuais para aumentar a escala das novas características de imagem em uma imagem de alta resolução. O bloco de *upsampling* consiste em uma camada de *nearest neighbour upsampling* e uma convolução 3×3. Por fim, a imagem refinada x é obtida a partir das novas características de imagem utilizando uma convolução 3×3.

3.3.2 Gated Memory Writing

Ao invés de considerar apenas a informação de texto parcial utilizando a Equação 3.17, o *gate* de escrita de memória permite ao modelo DM-GAN selecionar a palavra relevante para refinar as imagens iniciais. O *gate* de escrita de memória g_i^w combina as características de imagem R_i do último estágio com as características de palavra W para calcular a importância da palavra:

$$g_i^w(R, w_i) = \sigma(A * w_i + B * \frac{1}{N}\sum_{i=1}^{N} r_i), \tag{3.21}$$

onde σ é a função sigmoide, A é uma matriz $1 \times N_w$, e B é uma matriz $1 \times N_r$. Então, o *slot* de memória $m_i \in \mathbb{R}^{\mathrm{N_m}}$ é escrito combinando as características de imagem e de palavra:

$$m_i = M_w(w_i) * g_i^w + M_r(\frac{1}{N}\sum_{i=1}^{N} r_i) * (1 - g_i^w), \tag{3.22}$$

onde $M_w(\cdot)$ e $M_r(\cdot)$ representam a operação de convolução 1×1. $M_w(\cdot)$ e $M_r(\cdot)$ incorporam características de imagem e palavra no mesmo espaço de características com dimensão N_m.

3.3.3 Gated Response

O mecanismo de *gating* adaptativo é utilizado para controlar dinamicamente o fluxo de informação e atualizar características da imagem:

$$
\begin{aligned}
g_i^r &= \sigma(W_{[o_i, r_i]+b}) \\
r_i^{new} &= o * g_i^r + r_i * (1 - g_i^r),
\end{aligned}
\tag{3.23}
$$

onde g_i^r é o *gate* de resposta para fusão de informação, σ é a função sigmoide, W e b são respectivamente a matriz de parâmetros e o termo de *bias*.

3.3.4 Função Objetiva

A função objetiva da rede geradora é definida como:

$$
L = \sum_i L_{G_i} + \lambda_1 L_{CA} + \lambda_2 L_{DAMSM}, \tag{3.24}
$$

onde λ_1 e λ_2 são respectivamente os pesos do CA *loss* e do DAMSM *loss*. G_0 representa o gerador do estágio de geração inicial. G_i representa o gerador da i-ésima iteração do estágio de refinamento da imagem.

Adversarial Loss: a *adversarial loss* de G_i é definida como:

$$
L_{G_i} = -\frac{1}{2}[E_{x \sim p_{G_i}} \log D_i(x) + E_{x \sim p_{G_i}} \log D_i(x, s)], \tag{3.25}
$$

onde o primeiro termo é a perda incondicional, que faz a imagem gerada o mais real possível, e o segundo termo é a perda condicional, que faz a imagem corresponder à frase de entrada. Alternativamente, a *adversarial loss* para cada discriminador D_i é definida como:

$$
\begin{aligned}
L_{D_i} = & \underbrace{-\frac{1}{2}[E_{x \sim p_{data}} \log D_i(x) + E_{x \sim p_{G_i}} \log(1 - D_i(x))}_{\text{perda incondicional}} \\
& \underbrace{+E_{x \sim p_{data}} \log D_i(x, s) + E_{x \sim p_{G_i}} \log(1 - D_i(x, s))]}_{\text{perda condicional}},
\end{aligned}
\tag{3.26}
$$

onde a perda incondicional é projetada para distinguir a imagem gerada de imagens reais, e a perda condicional determina se a imagem e a frase de entrada correspondem, assim como foi visto na Seção 3.2.1.

Conditioning Augmentation Loss: a técnica de CA [10] é proposta para aumentar os dados de treinamento e evitar *overfitting*, reamostrando o vetor da frase de entrada a partir de uma distribuição gaussiana independente. Assim, o CA *loss* é definido como a divergência de Kullback-Leibler entre a distribuição gaussiana padrão e a distribuição gaussiana dos dados de treinamento:

$$L_{CA} = D_{KL}(\mathcal{N}(\mu(s), \Sigma(s)) \| \mathcal{N}(0, I)), \tag{3.27}$$

onde $\mu(s)$ e $\Sigma(s)$ são respectivamente as matrizes de covariâncias média e diagonal da característica da frase. $\mu(s)$ e $\Sigma(s)$ são calculados por camadas totalmente conectadas.

DAMSM Loss: é utilizado o DAMSM *loss* de [8] para medir o grau de correspondência entre imagens e descrições de texto. O DAMSM *loss* torna as imagens geradas mais bem condicionadas às descrições de texto.

3.3.5 Detalhes de Implementação

Muitos dos detalhes de implementação são incorporados da AttnGAN [8], mostrando como o DM-GAN é uma implementação mais moderna do AttnGAN. Para a incorporação de texto, foi empregado um codificador de texto LSTM bidirecional pré-treinado por Xu *et al.* [8] e seus parâmetros foram concertados durante o treinamento. Cada característica de palavra corresponde aos estados ocultos de duas direções. A característica de frase é gerada pela concatenação dos últimos estados ocultos de duas direções. O estágio de geração da imagem inicial primeiro sintetiza imagens com resolução de 64×64 e, em seguida, o estágio de refinamento de imagem baseado em memória dinâmica refina as imagens para resolução de 128×128 e 256×256.

O processo de refinamento com o módulo de memória dinâmica só é repetido duas vezes devido a limitações da GPU. Ao se introduzir a memória dinâmica às imagens de baixa resolução, (ou seja, 16×16, 32×32) não se pode melhorar ainda mais o desempenho, já que imagens de baixa resolução não têm uma boa geração e suas características se comportam como vetores aleatórios. Para todas as redes discriminadoras é aplicada a normalização espectral [9] após cada convolução, com o intuito de evitar gradientes incomuns, melhorando assim o desempenho da síntese de imagens baseada em texto. Por padrão, foram utilizados $N_w = 256$, $N_r = 64$ e $N_m = 128$ como as dimensões dos vetores característica de texto, imagem e memória respectivamente. Foram definidos $\lambda_1 = 1$ e $\lambda_2 = 50$ como hiper-parâmetros para o *dataset* COCO [2]. Todas as redes foram treinadas utilizando o otimizador ADAM [15] com *batch size*=10, $\beta_1 = 0.5$ e $\beta_2 = 0.999$ [9]. A *learning rate* utilizada foi de 0,0002, e o modelo DM-GAN foi treinado com 120 épocas no *dataset* COCO [2].

3.4 Melhorando a Síntese de Imagens Baseada em Texto Utilizando Contrastive Learning

Após conhecer os principais modelos utilizados na síntese de imagens a partir de texto, baseados em GAN [4], ainda existe um ponto em comum entre eles que precisa ser discutido: a consistência do *dataset* utilizado (nesse caso COCO [2]). Se todas as descrições textuais fossem perfeitas, essa seção seria desnecessária. Na prática, todas as descrições de imagens (o *dataset* COCO possui cinco descrições diferentes para cada imagem, com um exemplo na Figura 3.10) são escritas por seres humanos, e para cada imagem as descrições obtidas têm variações gigantescas em termos de conteúdo e escolha de palavras. A discrepância linguística entre as descrições de uma mesma imagem levam a um desvio de realidade no que está sendo retratado pelas imagens sintéticas. Assim, foi proposto um modelo de *contrastive learning* [1][3] para melhorar a qualidade e aumentar consistência semântica das imagens sintéticas.

Na tarefa de correspondência entre imagem e texto, é pré treinado um codificador de imagem e um codificador de texto no intuito de se aprender quais representações visuais e textuais são semanticamente consistentes com o par imagem-texto. Enquanto isso, é desenvolvido o aprendizado das representações textuais consistentes, onde são agregadas as legendas que pertencem à mesma imagem e separadas as legendas de imagens distintas, por meio de *contrastive learning*. Os codificadores de imagem e texto pré-treinados são aproveitados para extrair características visuais e textuais consistentes no seguinte estágio de treinamento da GAN. Em seguida, é utilizado o *contrastive loss* para minimizar a distância entre imagens falsas geradas a partir de descrições de texto relacionadas a um mesmo tema geral de uma imagem, enquanto são maximizadas aquelas relacionadas a diferentes temas gerais de uma imagem. Os modelos de conversão de texto para imagem existentes são generalizados para uma estrutura unificada, no intuito de que a abordagem possa ser integrada e seu desempenho, consequentemente, melhorado.

3.4.1 Pré-Treinamento com Contrastive Learning

Na tarefa de síntese de imagens a partir de texto, o objetivo da correspondência entre texto e imagem é aprender quais representações de texto são semanticamente consistentes com suas imagens. Como as representações de texto vão ser aproveitadas como as condições para se guiar a geração das imagens, é interessante desenvolver uma abordagem de pré-treino mais eficaz, com o intuito de se melhorar a qualidade das imagens sintéticas.

[3]Todo o código do AttnGAN+CL e do DM-GAN+CL está disponível em `https://github.com/huiyegit/T2I_CL`

Inspirado por recentes algoritmos de *contrastive learning* [16], Ye *et al.*[1] propõe uma nova abordagem para o pré-treinamento da correspondência entre texto e imagem. O aprendizado das representações textuais na correspondência às representações visuais é obtido por meio do DAMSM *loss*. Além disso, as representações textuais são treinadas pela junção das legendas correspondentes a mesma imagem e pela separação das legendas correspondentes a imagens diferentes, por meio do *contrastive loss*. Como podemos ver na Figura 3.4b, a arquitetura é constituída pelos seguintes três componentes principais:

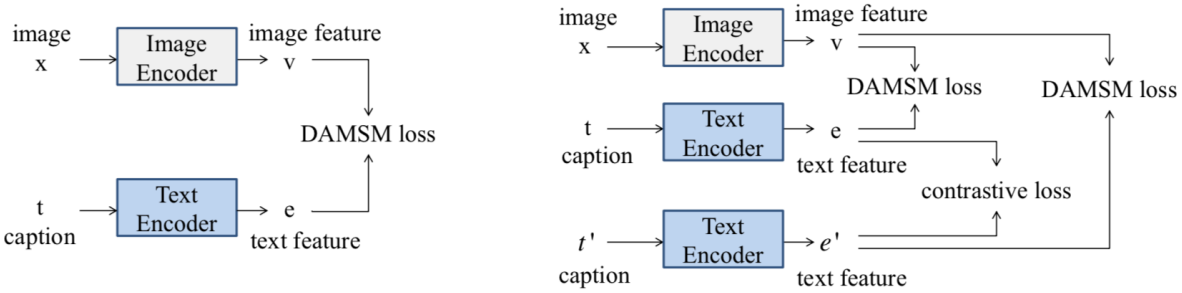


Figura 3.4: Comparação de arquitetura entre modelos antigos e a abordagem de Ye *et al.* [1] para correspondência entre texto e imagem (Fonte: [1]).

1. **Amostragem de Dados:** em cada etapa de treinamento, um *minibatch* de imagens x, legendas t e legendas t' é amostrado, onde as duas legendas t e t' correspondem às imagens x. Para a correspondência imagem-texto são considerados dois pares imagem-legenda positivos (x_i, t_i) e (x_i, t'_i) para cada imagem x_i, para o cálculo da perda DAMSM. Além disso, pode-se considerar o par legenda-legenda (t_i, t'_i) como o par positivo para realizar o cálculo do *contrastive loss*.

2. **Codificador de imagem f e codificador de texto g**: é adotado um codificador de imagem f para extrair as representações do vetor visual e as características de sub-regiões das imagens amostradas. Além disso, é utilizado um codificador de texto g para extrair as representações do vetor textual e as características de palavra de amostras de texto. O codificador de texto g é compartilhado na estrutura. A arquitetura tem a flexibilidade de permitir diversas escolhas de *deep neural networks*. Para se ter uma comparação justa com os modelos de partida, foram adotados os mesmos Inception-v3 e LSTM bidirecional usados pela AttnGAN [8] para instanciar o codificador de imagem f e o codificador de texto g.

3. **Loss Function:** similar aos modelos de partida, também foi adotado o DAMSM *loss* como *matching loss* da imagem-texto. Além disso, o *contrastive loss* é definido

em pares de dois ramos de legendas de entrada. O *contrastive loss* é calculado para minimizar a distância entre representações textuais relacionadas a mesma imagem, enquanto são maximizadas aquelas relacionadas a imagens distintas. É utilizado o *loss* de entropia cruzada com escala de temperatura normalizada (*normalized temperature-scaled cross entropy loss*) NT-Xent [1] como *contrastive loss*. Dado um par tal que $sim(a, b) = a^T b/\|a\|\|b\|$ representa o produto escalar entre a normalizado e b. Assim, a *loss function* para a i-ésima amostra é definida como:

$$L(i) = -\log \frac{exp(sim(u_i, u_j)/\tau)}{\sum_{k=1}^{2N} 1_{k \neq i} exp(sim(u_i, u_k)/\tau)}, \tag{3.28}$$

onde a i-ésima e j-ésima amostras fazem o par positivo, $1_{k \neq i}$ é uma função indicadora cujo valor é 1 se $k \neq i$, τ representa um parâmetro de temperatura e N é o *batch size* (e.g., N imagens e $2N$ legendas). O *contrastive loss* geral é calculado passando por todos os pares positivos em um *minibatch*, que pode ser definido como:

$$L_c = \frac{1}{2N} \sum_{i=1}^{2N} L(i) \tag{3.29}$$

O Algoritmo 1 da Figura 3.5 resume o método proposto:

Algoritmo 1 *Constrastive Learning* para correspondência imagem texto

Entrada: *Batch size N*, temperatura τ, codificador de imagem *f*, codificador de texto *g*

Saída: Codificador de imagem otimizado *f* e codificador de texto *g*

for{1,...,# de iterações de treinamento}do
  Amostrar um *minibatch* de imagens **x**
  Amostrar um *minibatch* de legendas **t** associadas à **x**
  Amostrar outro *minibatch* de legendas **t'** associadas à **x**
  $v = f(x)$ //representação da imagem
  $e = g(t)$ //representação do texto
  $e' = g(t')$ //representação do texto
  $\mathcal{L}_1 = DAMSM(v, e)$ //*matching loss* texto-imagem
  $\mathcal{L}_2 = DAMSM(v, e')$ //*matching loss* texto-imagem
  $\mathcal{L}_c =$ NT-Xent(e, e') //Equação 3.29
  $\mathcal{L} = \mathcal{L}_1 + \mathcal{L}_2 + \mathcal{L}_c$
  Atualiza redes f e g para minimizar $\mathcal{L}$
end for

Figura 3.5: Algoritmo 1 (Fonte: [1])

3.4.2 Contrastive Learning para o Treinamento da GAN

Na prática, as descrições escritas por humanos para uma mesma imagem possuem uma grande variação em termos de conteúdo e escolha de palavras, especialmente quando as cenas retratadas são complexas. A discrepância linguística entre as diferentes descrições da mesma imagem leva ao condicionamento das imagens sintéticas à legendas que se distanciam de uma verdade comum. O método de *contrastive learning* é aplicado para aumentar a consistência entre imagens geradas a partir de legendas relacionadas a uma mesma imagem, motivando-as a ficarem mais perto de uma verdade comum. Mais uma vez, a abordagem consiste nos seguintes três componentes:

Amostragem de Dados: a abordagem de amostragem de dados é similar àquela do estágio de pré-treinamento. Em cada etapa de treinamento é amostrado um *minibatch* de imagens **x**, legendas **t** e legendas **t'**, assim como na Seção 3.4.1. O *deep generative model* tem como saída as imagens falsas **x** e **x'** condicionadas às legendas **t** e **t'** respectivamente. Então, consideramos o par imagem-imagem (x_i, x_i') como o par positivo no *contrastive learning*.

Arquitetura do Modelo: nessa seção, a estrutura de Ye *et al.* [1] é derivada das GANs básicas [4]. Formalmente, o gerador G e o discriminador D seguem o objetivo do mínimo-máximo:

$$\min_G \max_D \mathop{E}_{x\sim P_r}[\log(D(x)] + \mathop{E}_{z\sim P_z}[\log(1 - D(G(z)))] \tag{3.30}$$

onde x é uma amostra real da distribuição de dados P_r, e a entrada z é amostrada de uma distribuição anterior P_z, como uma distribuição uniforme ou gaussiana.

Ye *et al.* [1] generaliza as abordagens de geração de imagens a partir de texto utilizando GANs recentes [8][9][10] para uma estrutura unificada, como mostrado na Figura 3.6. A estrutura das GANs com informação auxiliar e, que é codificada a partir da legenda de entrada t por um codificador de texto pré-treinado g é mantida. Então, ambos o gerador G e o discriminador D são condicionados nessa condição textual e. O processo de treinamento é similar às GANs padrão, com o seguinte objetivo:

$$\min_G \max_D \mathop{E}_{x\sim P_r}[\log(D(x, e)] + \mathop{E}_{z\sim P_z}[\log(1 - D(G(z, e), e))] \tag{3.31}$$

Ye *et al.* [1] estende ainda mais a estrutura generalizada de texto para imagem para uma *Siamese Neural Network (SNN)* e integra a abordagem de *contrastive learning*. Como podemos ver na Figura 3.7, o codificador de imagem f pega as imagens falsas x e x' como entrada, e extrai as representações visuais v e v' para calcular o *contrastive loss*. Os dois ramos da arquitetura compartilham do mesmo gerador G, discriminador D, codificador de imagem f e codificador de texto g. O codificador de imagem f e o codificador de texto

g são pré-treinados na tarefa de correspondência entre texto e imagem e trabalham no modo de *evaluation* (avaliação) na fase de treinamento GAN.

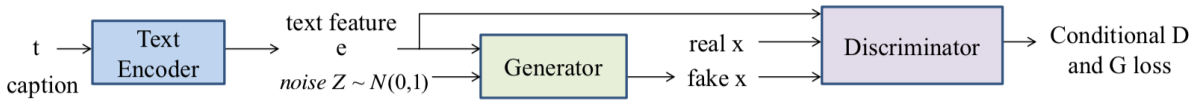


Figura 3.6: A arquitetura generalizada de Ye *et al.* [1] para a síntese de imagens a partir de texto (Fonte: [1]).

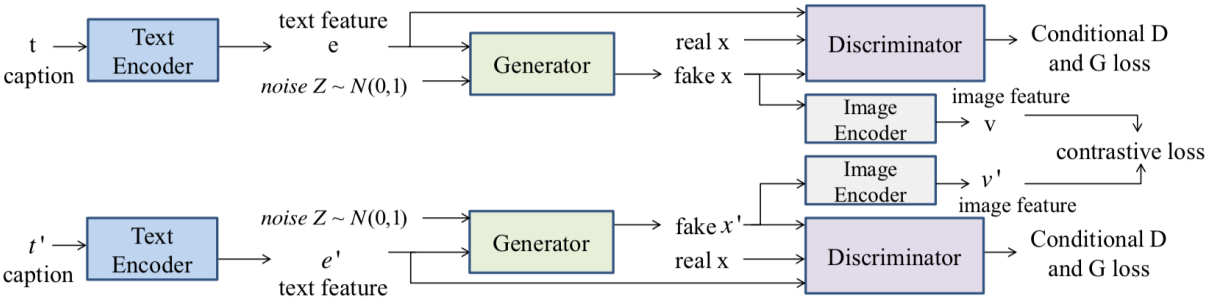


Figura 3.7: A arquitetura generalizada de Ye *et al.* [1] com *contrastive learning* para a síntese de imagens a partir de texto (Fonte: [1]).

Loss Function: além das *adversarial losses* da Equação 3.31, é definido o *contrastive loss* em pares de imagens falsas geradas de dois ramos de legendas de entrada. O *contrastive loss* é utilizado para minimizar a distância entre as imagens falsas geradas de duas descrições de texto relacionadas a mesma imagem, enquanto maximizam aquelas relacionadas a imagens distintas. É aplicado o mesmo *NT-Xent loss* da Seção 3.4.1. O Algoritmo 2 da Figura 3.8 resume o método proposto.

3.5 Legendando Imagens

Trabalhos recentes [3][7][11] melhoraram significativamente a qualidade de geração de legendas usando uma combinação de CNNs para se obter uma representação vetorial das imagens e *Recurrent Neural Network (RNN)* para decodificar essas representações em frases de linguagem natural, exatamente a estratégia replicada nesse trabalho.

Pequenas alterações foram feitas no código original de geração de legendas[4] para se obter os resultados demonstrados neste trabalho. O código foi altamente inspirado no

[4]Código disponível em: `https://github.com/MakarovArtyom/Image-Captioning-with-Attention`

Algoritmo 2 *Constrastive Learning* para treinamento GAN

Entrada: *Batch size N*, temperatura τ,coeficiente λ_c, gerador G, discriminador D, codificador de imagem pré treinado f,codificador de texto pré-treinado g

Saída: G e D otimizados.

1:**for**{1,...,# de iterações de treinamento}do
2: Amostrar um *minibatch* de imagens $\mathbf{x}\sim \mathrm{P}_r$
3: Amostrar um *minibatch* de variáveis latentes $\mathbf{z}\sim \mathrm{P}_z$
4: Amostrar um *minibatch* de legendas **t** associadas à **x**
5: Amostrar um *minibatch* de legendas **t'** associadas à **x**
6: $e = g(t)$
7: $e' = g(t')$
8: $\mathcal{L}_{D_1} = \frac{1}{N}\sum_{i=1}^{N}[\log D(x_i, e_i) + \log(1 - D(G(z_i, e_i), e_i))]$
9: $\mathcal{L}_{D_2} = \frac{1}{N}\sum_{i=1}^{N}[\log D(x_i, e'_i) + \log(1 - D(G(z_i, e'_i), e'_i))]$
10: $\mathcal{L}_D = \mathcal{L}_{D_1} + \mathcal{L}_{D_2}$
11: Atualizar D para minimizar $\mathcal{L}_D$
12: Amostra ruído **z**, legendas **t** e **t'** como nos passos 3, 4 e 5
13: Calcula **e**,**e'** como nos passos 6 e 7
14: $\mathcal{L}_{G_1} = \frac{1}{N}\sum_{i=1}^{N}\log(1 - D(G(z_i, e_i), e_i))$
15: $\mathcal{L}_{G_2} = \frac{1}{N}\sum_{i=1}^{N}\log(1 - D(G(z_i, e'_i), e'_i))$
16: $v = f(G(z, e))$ //representação da imagem
17: $v' = f(G(z, e'))$ //representação da imagem
18: $\mathcal{L}_c = \text{NT-Xent}(v, v')$ //Equação 3.29
19: $\mathcal{L}_G = \mathcal{L}_{G_1} + \mathcal{L}_{G_2} + \lambda_c\mathcal{L}_c$
20: Atualiza G para minimizar $\mathcal{L}_G$
21:**end for**

Figura 3.8: Algoritmo 2 (Fonte: [1])

trabalho de Xu *et al.* [7][5] e nas melhores práticas para legendas de imagens no Pytorch[6], também inspirado em Xu *et al.* [7].

Nesta seção serão descritas as duas variantes do modelo baseado em atenção, primeiro descrevendo sua estrutura comum. A principal diferença é a definição da função ϕ que será descrita em detalhes na Seção 3.5.3. Vetores serão representados com fonte em negrito e matrizes com letras maiúsculas. Também é importante deixar claro qual é a diferença entre os métodos de atenção descritos na Seção 3.5.3 para justificar o uso do método "Soft" neste trabalho.

3.5.1 Codificador CNN

O codificador utilizado é um modelo pré-treinado *Resnet-152* usado como extrator de características, que consiste de blocos convolucionais incorporados de conexões de atalho,

[5]Código disponível em: `https://github.com/yunjey/show-attend-and-tell`
[6]Disponíveis em: `https://github.com/sgrvinod/a-PyTorch-Tutorial-to-Image-Captioning`

como mostrado na Figura 3.9. O modelo pega uma única imagem bruta e gera uma legenda $\mathbf{y}$ codificada como uma sequência de 1 em K palavras codificadas.

$$y = \{\mathbf{y}_1, ..., \mathbf{y}_c\}, \mathbf{y}_i \in \mathbb{R}^{\mathrm{K}}, \tag{3.32}$$

onde K é o tamanho do vocabulário e C é o tamanho da legenda.

É utilizada uma CNN com o objetivo de se extrair um conjunto de vetores de características, aos quais vamos nos referir como vetores de anotação. O extrator produz L vetores, cada qual sendo uma representação D-dimensional correspondente a uma parte da imagem.

$$a = \{\mathbf{a}_1, ..., \mathbf{a}_L\}, \mathbf{a}_i \in \mathbb{R}^{\mathrm{D}} \tag{3.33}$$

Com o objetivo de se obter uma correspondência entre os vetores características e as partições 2-D da imagem, são extraídas características de uma camada convolucional mais baixa. Isso permite que o decodificador tenha foco seletivo em certas partes da imagem ao selecionar um subconjunto de todos os vetores característica.

3.5.2 Decodificador com LSTM

É utilizada uma LSTM que produz as legendas pela geração de uma palavra a cada divisão de tempo, condicionada a um vetor contexto, ao último estado oculto e às palavras geradas

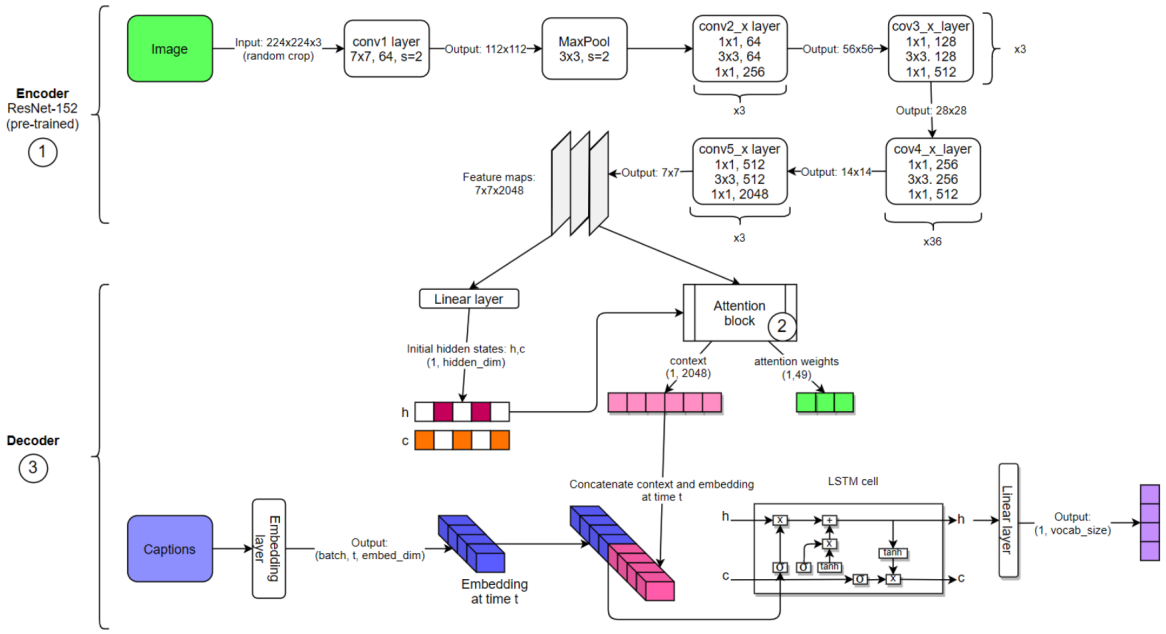


Figura 3.9: Modelo de arquitetura codificador-decodificador (Fonte: [7]).

anteriormente. A implementação do LSTM utiliza $T_{s,t} : \mathbb{R}^{s} \rightarrow \mathbb{R}^{t}$ para representar uma transformação simples com parâmetros que são aprendidos,

$$\begin{pmatrix} \mathbf{i}_t \\ \mathbf{f}_t \\ \mathbf{o}_t \\ \mathbf{g}_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} T_{D+m+n,n} \begin{pmatrix} \mathbf{E}\mathbf{y}_{t-1} \\ \mathbf{h}_{t-1} \\ \hat{z}_t \end{pmatrix} \tag{3.34}$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t \tag{3.35}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t) \tag{3.36}$$

Aqui, $\mathbf{i}_t, \mathbf{f}_t, \mathbf{c}_t, \mathbf{o}_t, \mathbf{h}_t$ são respectivamente a entrada, o *forget*, a memória, a saída e o estado oculto do LSTM. O vetor $\hat{z} \in \mathbb{R}^{D}$ é o vetor contexto, que captura a informação visual associada a um local de entrada particular, como explicado abaixo. $\mathbf{E} \in \mathbb{R}^{m \times k}$ é uma matriz de incorporação, m e n representam a incorporação e a dimensionalidade do LSTM respectivamente, e sejam σ e $\odot$ a ativação sigmoide logística e a multiplicação por elemento (produto de Hadamard), respectivamente.

O vetor contexto $\hat{z}_t$ (Equações 3.34, 3.35 e 3.36) é uma representação dinâmica da parte relevante da imagem de entrada no tempo t. É definido um mecanismo ϕ que calcula $\hat{z}_t$ a partir dos vetores de anotação $\mathbf{a}_i,\ i = 1, ..., L$, correspondendo às características extraídas em diferentes locais da imagem. Para cada local i, o mecanismo gera um peso positivo α_i, que pode ser interpretado ou como a probabilidade de que o local i é o lugar certo para se concentrar na produção da próxima palavra (o mecanismo de atenção estocástico "hard"), ou como a importância relativa que se dá ao local i na junção de todos os a_i's. O peso α_i de cada vetor anotação a_i é calculado por um modelo de atenção f_{att}, onde é utilizado um perceptron multicamadas [8] condicionado no último estado oculto h_{t-1}. Podemos perceber que o estado oculto varia conforme a saída RNN avança na sua sequência de saídas, ou seja, o local onde a rede procura depois depende da sequência de palavras que já foi gerada.

$$e_{ti} = f_{att}(\mathbf{a}_i, \mathbf{h}_{t-1}) \tag{3.37}$$

$$\alpha_{ti} = \frac{exp(e_{ti})}{\sum_{k=1}^{L} exp(e_{tk})} \tag{3.38}$$

Assim que os pesos (que se somam a um) são calculados, o vetor contexto $\hat{z}_t$ é calculado por:

$$\hat{z}_t = \phi(\{\mathbf{a}_i\}, \{\alpha_i\}), \tag{3.39}$$

onde ϕ é a função que retorna um único vetor, dado o conjunto de vetores anotação e seus respectivos pesos. Os detalhes da função ϕ são discutidos na Seção 3.5.3.

O estado inicial de memória e o estado oculto do LSTM são previstos pela média dos vetores anotação, alimentada por dois MLPs separados (*init,c* e *init,h*):

$$\begin{aligned} \mathbf{c}_0 &= f_{init,c}(\frac{1}{L}\sum_i^L \mathbf{a}_i) \\ \mathbf{h}_0 &= f_{init,h}(\frac{1}{L}\sum_i^L \mathbf{a}_i) \end{aligned} \tag{3.40}$$

Xu *et al.* [7] utiliza o *deep output layer* para calcular a probabilidade da palavra de saída dado o estado LSTM, o vetor contexto e a última palavra:

$$p(\mathbf{y}_t|\mathbf{a}, \mathbf{y}_1^{t-1}) \propto exp(\mathbf{L}_0(\mathbf{E}\mathbf{y}_{t-1} + \mathbf{L}_h\mathbf{h}_t + \mathbf{L}_z\hat{z}_t)), \tag{3.41}$$

onde $\mathbf{L}_0 \in \mathbb{R}^{\mathrm{K}\times\mathrm{m}}, \mathbf{L}_\mathrm{h} \in \mathbb{R}^{\mathrm{m}\times\mathrm{n}}, \mathbf{L}_\mathrm{z} \in \mathbb{R}^{\mathrm{m}\times\mathrm{D}}$, e $\mathbf{E}$ são parâmetros aprendidos, inicializados aleatoriamente.

3.5.3 O Método de Atenção Estocástica "Hard" vs Determinística "Soft"

Nesta seção serão discutidos dois mecanismos alternativos para o modelo de atenção f_{att}: atenção estocástica e atenção determinística [7]. Na Figura 3.10 podemos ver uma representação visual do funcionamento do modelo de atenção.

Atenção Estocástica "Hard"

A variável de localização é representada por s_t: onde o modelo decide focar a atenção quando gera a t-ésima palavra. $s_{t,i}$ é uma variável indicadora *"one-hot"* que é definida como 1 se a i-ésima localização (fora de L) é usada para extrair características visuais. Ao tratarmos os locais de atenção como variáveis intermediárias latentes, pode-se atribuir uma distribuição categórica (distribuição multinoulli), parametrizada por $\{\alpha_i\}$, e $\hat{z}_t$ como uma variável aleatória:

$$p(s_{t,i} = 1 \mid s_{j<t}, \mathbf{a}) = \alpha_{t,i} \tag{3.42}$$

$$\hat{z}_t = \sum_i s_{t,i}\mathbf{a}_i \tag{3.43}$$

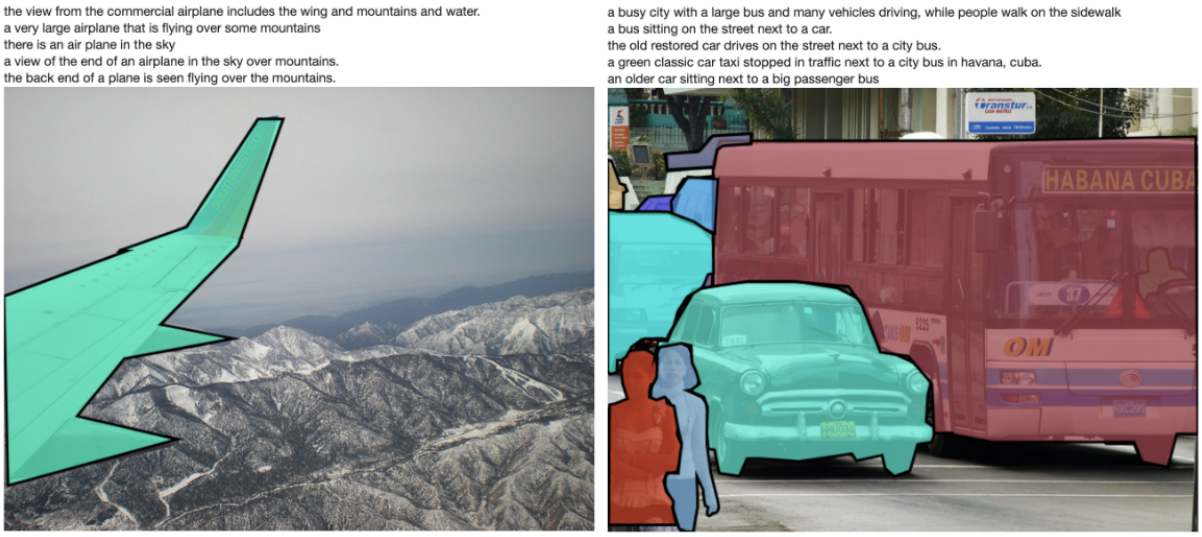


Figura 3.10: Exemplo do funcionamento do modelo de atenção no *dataset* MS COCO [2] (Fonte: [2]).

É definida uma nova função objetiva L_s, que é um limite inferior variável da verossimilhança logarítmica marginal $\log p(\mathbf{y} \mid \mathbf{a})$ da observação da sequência de palavras $\mathbf{y}$, dadas as características de imagem $\mathbf{a}$. O algoritmo de aprendizado para os parâmetros W dos modelos pode ser derivado pela otimização direta de L_s:

$$\begin{aligned} L_s &= \sum_s p(s \mid \mathbf{a}) \log p(\mathbf{y} \mid s, \mathbf{a}) \\ &\leq \log \sum_s p(s \mid \mathbf{a}) p(\mathbf{y} \mid s, \mathbf{a}) \\ &= \log p(\mathbf{y} \mid \mathbf{a}), \end{aligned} \tag{3.44}$$

$$\frac{\partial L_s}{\partial W} = \sum_s p(s \mid \mathbf{a}) \left[\frac{\partial \log p(\mathbf{y} \mid s, \mathbf{a})}{\partial W} + \log p(\mathbf{y} \mid s, \mathbf{a}) \frac{\partial \log p(\mathbf{s} \mid \mathbf{a})}{\partial W} \right] \tag{3.45}$$

A Equação 3.45 sugere uma aproximação de amostragem do gradiente referente aos parâmetros do modelo baseada em Monte Carlo. Isso pode ser feito pela amostragem da localização s_t de uma distribuição categórica (Multinoulli) definida pela Equação 3.42.

$$\tilde{s}_t \sim \text{Multinoulli}_L(\{\alpha_i\})$$

$$\frac{\partial L_s}{\partial W} \approx \frac{1}{N} \sum_{n=1}^{N} \left[\frac{\partial \log p(\mathbf{y} \mid \tilde{s}^n, \mathbf{a})}{\partial W} + \log p(\mathbf{y} \mid \tilde{s}^n, \mathbf{a}) \frac{\partial \log p(\tilde{s}^n \mid \mathbf{a})}{\partial W} \right] \tag{3.46}$$

A média móvel de uma linha de base é usada para reduzir a variância no estimador de gradiente de Monte Carlo. Ao se observar o k-ésimo *minibatch*, a média móvel da linha

de base é estimada como a soma acumulada das probabilidades logarítmicas anteriores, com decaimento exponencial:

$$b_k = 0.9 \times b_{k-1} + 0.1 \times \log p(\mathbf{y} \mid \tilde{s}_k, \mathbf{a})$$

Para reduzir ainda mais a variância do estimador, um termo de entropia na distribuição de multinoulli $H[s]$ é adicionado. Ainda, com probabilidade de 0.5 para uma dada imagem, a localização de atenção amostrada $\tilde{s}$ é ajustada para seu valor esperado α. Ambas as técnicas melhoram a robustez do algoritmo de aprendizagem de atenção estocástica. A regra final de aprendizado para o modelo é a seguinte:

$$\frac{\partial L_s}{\partial W} \approx \frac{1}{N} \sum_{n=1}^{N} \left[\frac{\partial \log p(\mathbf{y} \mid \tilde{s}^n, \mathbf{a})}{\partial W} + \lambda_r (\log p(\mathbf{y} \mid \tilde{s}^n, \mathbf{a}) - b) \frac{\partial \log p(\tilde{s}^n \mid \mathbf{a})}{\partial W} + \lambda_e \frac{\partial H[\tilde{s}^n]}{\partial W} \right], \tag{3.47}$$

onde λ_r e λ_e são dois hiper-parâmetros ajustados por *cross-validation*. Como apontado por Xu *et al.* [7], essa formulação é equivalente à regra de aprendizado por reforço, onde a recompensa pela atenção escolher uma sequência de ações é um valor real proporcional à probabilidade logarítmica da sentença alvo considerando a trajetória de atenção amostrada.

Ao se fazer uma *"hard choice"* em cada ponto, $\phi(\{\mathbf{a}_i\}, \{\alpha_i\})$, da Equação 3.39, é uma função que retorna um $\mathbf{a}_i$ amostrado em cada ponto no tempo baseado em uma distribuição multinoulli parametrizada por α.

Atenção Determinística "Soft"

Para se aprender a atenção estocástica é necessário que se faça a amostragem da localização de atenção s_t todas as vezes. Alternativamente, pode-se obter a esperança do vetor contexto $\hat{z}_t$ diretamente por:

$$E_{p(s_t|a)}[\hat{z}_t] = \sum_{i=1}^{L} \alpha_{t,i} \mathbf{a}_i, \tag{3.48}$$

e formular um modelo de atenção determinístico, calculando um vetor anotação ponderada de atenção suave ("soft") $\phi(\{\mathbf{a}_i\}, \{\alpha_i\}) = \sum_i^L \alpha_i \mathbf{a}_i$. Isso corresponde à injeção de um contexto ponderado α "soft"no sistema. O modelo completo é suave e diferenciável sob a atenção determinística, então, o aprendizado de ponta a ponta é trivial ao se utilizar o *back-propagation* padrão.

Aprender a atenção determinística também pode ser entendido como: a otimização aproximada da verossimilhança marginal da Equação 3.44 sob a variável aleatória de

localização de atenção s_t da última seção. A ativação oculta do LSTM $\mathbf{h}_t$ é uma projeção linear do vetor de contexto estocástico $\hat{z}_t$, seguido pela não linearidade de tanh. Pela aproximação de Taylor de primeira ordem, o valor esperado $E_{p(s_t|a)}[\hat{z}_t]$ é equivalente ao cálculo de $\mathbf{h}_t$ utilizando uma única *"forward propagation"*, como o vetor de contexto esperado $E_{p(s_t|a)}[\hat{z}_t]$. Considerando a Equação 3.41, $\mathbf{n}_t = \mathbf{L}_0(\mathbf{E}\mathbf{y}_{t-1} + \mathbf{L}_h\mathbf{h}_t + \mathbf{L}_z\hat{z}_t)$, $\mathbf{n}_{t,i}$ representa n_t calculado ao se definir o valor da variável aleatória $\hat{z}$ como a_i. A média geométrica ponderada normalizada (*normalized weighted geometric mean*) (NWGM) é definida para a previsão da k-ésima palavra softmax:

$$\begin{aligned} NWGM[p(y_t = k \mid \mathbf{a})] &= \frac{\prod_i exp(n_{t,k,i})^{p(s_{t,i}=1|a)}}{\sum_j \prod_i exp(n_{t,j,i})^{p(s_{t,i}=1|a)}} \\ &= \frac{exp(E_{p(s_t|a)}[n_{t,k}])}{\sum_j exp(E_{p(s_t|a)}[n_{t,j}])} \end{aligned} \tag{3.49}$$

A equação acima mostra que a média geométrica ponderada normalizada da previsão da legenda pode ser bem aproximada pelo uso do vetor contexto esperado, onde $E[\mathbf{n}_t] = \mathbf{L}_0(\mathbf{E}\mathbf{y}_{t-1} + \mathbf{L}_h E[\mathbf{h}_t] + \mathbf{L}_z E[\hat{z}_t])$. Isso mostra que a NWGM de uma unidade softmax é obtida aplicando o softmax às esperanças das projeções lineares subjacentes. Ainda segundo Xu *et al.* [7], $NWGM[p(y_t = k \mid \mathbf{a})] \approx E[p(y_t = k \mid \mathbf{a})]$ sob a ativação softmax. Isso significa que a esperança das saídas sobre todas as localizações de atenção possíveis induzidas pela variável aleatória s_t é calculada por simples *feedforward propagation* com vetor de contexto esperado $E[\hat{z}_t]$. Em outras palavras, o modelo de atenção determinística é uma aproximação da probabilidade marginal sobre os locais de atenção.

Atenção Estocástica Dupla

Ao fazer a construção obtemos que $\sum_i \alpha_{t,i} = 1$, já que são a saída de um softmax. Ao se treinar a versão determinística do modelo, Xu *et al.* [7] introduz uma forma de regularização estocástica dupla, onde é encorajado que $\sum_t \alpha_{t,i} \approx 1$. Isso pode ser interpretado como encorajar o modelo a prestar atenção equivalente a todas as partes da imagem ao longo da geração. Experimentalmente, é observado que essa penalidade foi importante quantitativamente na melhora da pontuação BLEU, e que qualitativamente isso leva a legendas mais ricas e descritivas. Além disso, o modelo de atenção *"soft"* prevê um escalar *"gating"* β do último estado oculto $\mathbf{h}_{t-1}$ em cada espaço de tempo t, tal qual $\phi(\{\mathbf{a}_i\}, \{\alpha_i\}) = \beta \sum_i^L \alpha_i \mathbf{a}_i$, onde $\beta_t = \sigma(f_\beta(\mathbf{h}_{t-1}))$. Foi percebido que os pesos de atenção colocam mais ênfase nos objetos das imagens ao incluir o escalar β.

Concretamente, o modelo é treinado ponta a ponta pela minimização da seguinte probabilidade logarítmica negativa penalizada:

$$L_d = -\log(P(\mathbf{y}|\mathbf{x})) + \lambda \sum_{i}^{L}(1 - \sum_{t}^{C} \alpha_{t,i})^2 \tag{3.50}$$

Com os métodos teóricos finalizados passamos para o Capítulo 4, onde serão introduzidos os resultados qualitativos e quantitativos respectivos às diferentes GANs testadas e ao gerador de legendas utilizado nesse trabalho.

4 Experimentos e Resultados

Todos os modelos foram avaliados no *dataset* COCO [2]. O *dataset* possui 82783 imagens de treinamento, 40504 imagens de validação e 40504 imagens de teste. Cada imagem possui 5 legendas descritivas.

4.1 Comparativo das Imagens Geradas por GANs

Método de Avaliação: foram escolhidas as mesmas métricas utilizadas por Ye *et al.* [1]: o *Inception Score (IS)*, o *Fréchet Inception Distance (FID)* e a *R-Precision* como métricas quantitativas para avaliação da performance. O IS calcula a divergência KL entre as distribuições de probabilidade condicional e marginal. No geral, um IS maior indica que o modelo consegue sintetizar imagens falsas com melhor qualidade e diversidade. O FID calcula a distância de Fréchet entre imagens sintéticas e reais no espaço de características extraído do modelo pré-treinado Inception-v3. Um FID menor indica que os dados sintéticos são mais realistas e similares aos dados reais. O *R-Precision* calcula a precisão da tarefa de recuperação imagem-texto para avaliar em que medida as imagens sintéticas correspondem às legendas de entrada. Um *R-Precision* maior indica que as imagens geradas têm maior consistência com as descrições textuais. O código fonte para cálculo das três métricas é público e está disponível no git da DM-GAN[1][9].

Detalhes de Implementação: foi utilizado o modelo AttnGAN [8] como linha de base para a geração de imagens sintéticas. Como já foi descrito anteriormente, sabe-se que o DM-GAN [9] possui métricas de avaliação melhores que o AttnGAN [8], como mostrado na Tabela 4.1. Todas as métricas estarão disponíveis mais à frente, mas no treinamento do modelo de geração de legendas foram utilizadas imagens sintéticas do AttnGAN, do DM-GAN [9], do AttnGAN+CL e do DM-GAN+CL [1].

Foram seguidas as mesmas implementações de treinamento registradas em [1], com 120 épocas no treinamento do AttnGAN+CL e com 200 épocas no DM-GAN+CL. As métricas são avaliadas a cada 10 épocas.

[1] `https://github.com/MinfengZhu/DM-GAN`

4.1.1 Estudo Qualitativo

Podemos comparar os diferentes métodos, onde visualizamos as imagens sinéticas geradas pelas legendas de exemplo. Como podemos perceber, na Figura 4.1 as imagens geradas pela AttnGAN são as menos realistas, muitas vezes com objetos não identificáveis. A abordagem integrada com *contrastive learning* (AttnGAN+CL) produz imagens mais realistas e que, na maioria dos casos, têm melhor correspondência às descrições de texto. Na 4^a coluna da Figura 4.1, o barco da abordagem com *contrastive learning* possui cores vermelhas e brancas, conforme a descrição, ao passo que a imagem da AttnGAN não mostra o barco vermelho. Se compararmos o modelo de base (AttnGAN) com o DM-GAN, já é possível perceber como a abordagem de memória dinâmica possui suas vantagens, com objetos mais bem definidos, melhor separação de elementos e melhor correspondência textual, em muitos casos superior ao AttnGAN+CL. Na 2^a coluna da Figura 4.1, podemos perceber como a DM-GAN possui um contorno melhor de objetos, representando melhor uma bancada com diferentes frutas do que a AttnGAN e a AttnGAN+CL. Como mostram os dados quantitativos da Tabela 4.1, o modelo DM-GAN+CL é o que possui os melhores resultados. Nas diferentes imagens é possível perceber os diferentes objetos e uma boa correspondência textual, se comparado com os outros modelos.

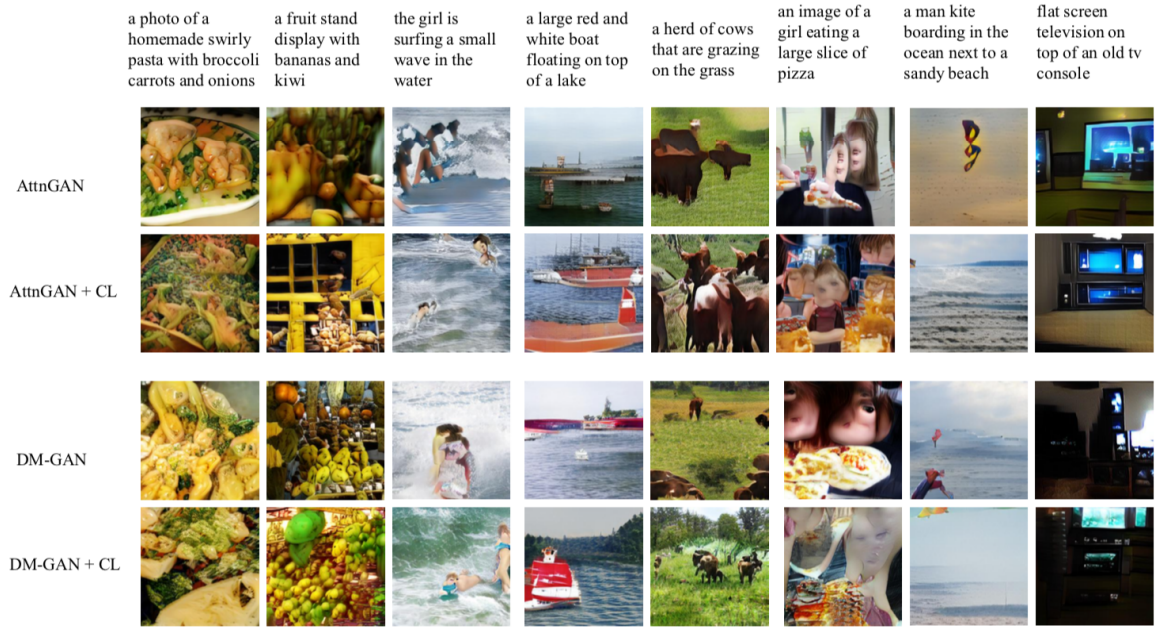


Figura 4.1: Comparação entre exemplos de imagens geradas pelos diferentes modelos de GAN no *dataset* COCO [2] (Fonte: [1]).

4.1.2 Estudo Quantitativo

Conforme Ye *et al.* [1], a abordagem com *contrastive learning* é aplicada em duas partes: correspondência texto-imagem e o treinamento GAN. É combinada a abordagem de *contrastive learning* com os modelos de base AttnGAN e DM-GAN, e são conduzidos experimentos para determinar sua eficácia. Os resultados experimentais são mostrados na Tabela 4.1. A abordagem com *contrastive learning* para a tarefa de correspondência entre texto e imagem ajuda a melhorar a performance em termos do IS, FID e *R-Precision*, com uma única exceção, onde o IS de +CL1 é .55 menor do que o DM-GAN. Quando é adicionada a abordagem com *contrastive learning* para o treinamento GAN somada à abordagem anterior, o processo completo com *contrastive learning* mostra uma melhora ainda maior em termos do IS, FID e *R-Precision*, com exceção do IS do AttnGAN da abordagem completa, se comparada com a anterior (+CL1) que é 0.11 menor.

Tabela 4.1: Comparativo entre AttGAN e DM-GAN sobre o dataset COCO [2]. CL1 E CL2 representam a abordagem de contrastive learning no pré-treinamento e no treinamento GAN, respectivamente (Fonte: [1]).

Modelo	IS ↑	FID ↓	R-Precision ↑
AttnGAN	$25.05 \pm .64$	30.67	$84.24 \pm .58$
+CL1	$\mathbf{25.87 \pm .41}$	26.89	$85.93 \pm .63$
+CL1+CL2	$25.70 \pm .62$	**23.93**	$\mathbf{86.55 \pm .51}$
DM-GAN	$31.53 \pm .39$	27.04	$91.82 \pm .49$
+CL1	$30.98 \pm .69$	25.29	$92.10 \pm .56$
+CL1+CL2	$\mathbf{33.34 \pm .51}$	**20.79**	$\mathbf{93.40 \pm .39}$

4.2 Legendando Imagens

Dataset: o objetivo deste Trabalho é a demonstração de que o uso de imagens sintéticas melhora a performance de um gerador de legendas conforme proposto por Hossain *et al.* [3]. Para realizar esses experimentos e checar a mudança de performance nos modelos foi utilizado o *dataset* COCO [2] com 70% de imagens reais e 30% de imagens sintéticas. As imagens reais são as originais do *dataset* COCO, e as sintéticas são geradas pelas GANs. Cada imagem possui cinco legendas diferentes para uma verdade em comum. Essas legendas foram utilizadas para gerar imagens sintéticas rotuladas. Assim como Hossain *et al.* [3], foi explicitamente mantida a divisão de treinamento e validação, conforme marcado no *dataset*. Assim, se uma imagem sintética foi gerada a partir da legenda de uma determinada imagem do treinamento, essa imagem foi usada apenas no treinamento e a recíproca é verdadeira para o conjunto de validação.

Se uma imagem sintética foi gerada a partir da descrição textual de uma determinada imagem real, a imagem sintética vai substituir essa imagem real no *dataset*, assim teremos um conjunto de treinamento e validação de tamanho igual ao da base original. Dessa forma é possível manter o modelo sem um viés de tamanho do *dataset* utilizado, e assim pode-se dizer que, com certeza, a diferença encontrada durante os experimentos foi um resultado direto do tipo de imagem (real ou sintética) com o qual o modelo foi treinado e testado. Assim, o modelo de base tem 82783 imagens de treinamento e 40504 imagens de validação. Dessa forma, os modelos testados também terão 82783 imagens de treinamento, das quais 24835 serão sintéticas e 57948 serão reais. Do mesmo modo, terão 40504 imagens de validação, das quais 12151 serão sintéticas e 28353 serão reais.

Detalhes de Implementação: foi utilizado o modelo pré-treinado *Resnet-152* para extração das características de imagens, que consiste de blocos convolucionais e de construção, incorporados com conexões de atalho. Ele recebe uma amostra de imagem 224×224 cortada aleatoriamente com transformadas aplicadas e extrai 2048 mapas de características com um tamanho de 7×7 cada. Os vetores de recursos foram obtidos do último bloco de convolução, sem aplicação da camada totalmente conectada. Isso permite que a rede de atenção seja seletiva e se concentre em vários recursos da imagem durante a decodificação. O modelo em questão se utiliza de um módulo de atenção determinística *"soft"*, conforme descrito na Seção 3.5.3. Também foi utilizado o otimizador ADAM [15] para treinar o modelo.

O *batch size* utilizado foi de 64. Um *batch size* menor aumenta o efeito de regularização e facilita o alocamento de 1 *batch* na memória. A dimensionalidade dos *embeddings* de imagem e palavra utilizada foi de 256. O número de características no estado oculto do decodificador foi 512. O número de épocas escolhidas foi 14 e o *learning rate* utilizado foi de 0.0001. O modelo foi treinado no Google Colab pro+, que levou em média 23 horas numa GPU NVTV100-SXM2. O código utilizado foi construído no Pytorch e adaptado de um modelo existente[2].

4.2.1 Estudo Qualitativo

Foram usadas imagens reais para fazer o estudo qualitativo das legendas. Um comparativo de performances testado com imagens sintéticas é apresentado na próxima seção. Foram geradas legendas utilizando os modelos de treinamento sem imagens sintéticas (normal) e com imagens sintéticas geradas pelas AttnGAN, AttnGAN+CL, DM-GAN e DM-GAN+CL.

Cada um dos modelos possui suas vantagens, mas pode-se dizer que, no geral, os modelos treinados com imagens reais e sintéticas tiveram uma melhor performance na

[2]Código disponível em: `https://github.com/MakarovArtyom/Image-Captioning-with-Attention`

geração de legendas para imagens reais. Como podemos ver nas imagens das Figuras 4.2 e 4.3, o modelo normal (treinado apenas com imagens reais) teve uma performance aceitável, mas na maioria das vezes foi superado pelos modelos treinados por imagens reais e sintéticas.

Na primeira imagem da Figura 4.2, o modelo normal gera uma boa legenda mencionando a presença de um relógio e uma torre de relógio. O DM-GAN gera uma legenda melhor ao mencionar com precisão a localização da torre de relógio na rua de uma cidade. Ainda podemos perceber como o *contrastive learning* melhora a descrição para o modelo AttnGAN, gerando uma legenda idêntica ao da DM-GAN. Na segunda imagem da Figura 4.2 vemos uma boa representação da piora de performance do uso do *contrastive learning* no AttnGAN. Tanto o AttnGAN quanto o DM-GAN geram legendas melhores do que o modelo normal, dando um destaque para os modelos DM-GAN e DM-GAN+CL que colocaram a palavra *"game"* no contexto da frase. Na terceira imagem da Figura 4.2, podemos ver novamente como os modelos AttnGAN e DM-GAN melhoram as legendas se

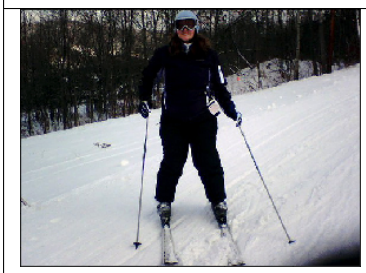
	Normal: A clock tower with a clock on the side of a street. AttnGAN: A street sign with a a in the background. AttnGAN+CL: A clock tower with a clock on a city street. DM-GAN: A clock tower with a clock on a city street. DM-GAN+CL: a large clock tower with a clock on it.
	Normal: A group of people playing a frisbee in a field. AttnGAN: A group of people playing frisbee in a field. AttnGAN+CL: A group of people playing a field with a frisbee. DM-GAN: A group of people playing a game of frisbee. DM-GAN+CL: A group of people playing a game of frisbee.
	Normal: A man is skiing down a snowy slope. AttnGAN: A woman is skiing down a snowy slope. AttnGAN+CL: A man in a snow skiing down a snowy slope. DM-GAN: A woman is skiing down a snowy slope. DM-GAN+CL: A woman on skis on a snowy slope

Figura 4.2: Exemplos do gerador de legendas treinado com diferentes tipos de GAN.

comparados com o modelo normal. Vemos novamente como o *contrastive learning* piorou a performance do AttnGAN, e como o modelo só conseguiu identificar que a pessoa na imagem era uma mulher após a introdução dos modelos treinados por imagens reais e sintéticas. O modelo DM-GAN+CL gera uma legenda um pouco diferente de DM-GAN, porém os dois fazem uma boa descrição da imagem.

Na primeira imagem da Figura 4.3, o modelo DM-GAN foi o único que descreveu as pessoas em cima dos elefantes, além de ser a melhor legenda para a imagem. Todos os outros modelos detectaram os elefantes, mas não conseguiram descrever uma boa representação da imagem. Na segunda imagem da Figura 4.3 podemos ver novamente como os modelos de imagens sintéticas têm resultados melhores do que o modelo normal, dando um destaque especial para o modelo DM-GAN, que gerou a melhor legenda, colocando o gato dentro da pia e a pia dentro de um banheiro. Os outros modelos geraram legendas aceitáveis. Na última imagem da Figura 4.3 acontece algo interessante: vemos novamente o aumento de performance dos modelos treinados com imagens sintéticas, onde o modelo DM-GAN é o único que descreve a cor do trem, e temos uma piora de performance no DM-GAN+CL.

	Normal: A group of elephants are standing in a. AttnGAN: A group of elephants standing in a a. AttnGAN+CL: A group of elephants are standing in a. DM-GAN: A group of people riding elephants. DM-GAN+CL: A group of elephants are standing in a dirt.
	Normal: A cat is sitting on a sink sink. AttnGAN: A cat is sitting on a toilet sink. AttnGAN+CL: A cat is sitting on a toilet sink. DM-GAN: A cat sitting on a sink in a bathroom. DM-GAN+CL: A cat is sitting on a toilet bowl.
	Normal: A train train on a train station. AttnGAN: A train is traveling down the tracks near a building. AttnGAN+CL: A train is on a train station with a train. DM-GAN: A red train is on a track with a train station. DM-GAN+CL: A train train a train station with a train station.

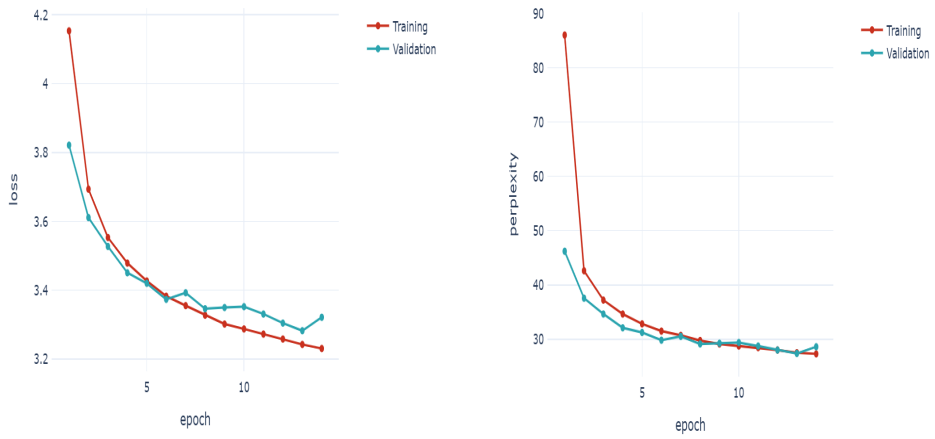
Figura 4.3: Exemplos do gerador de legendas treinado com diferentes tipos de GAN.

4.2.2 Estudo Quantitativo

No estudo quantitativo foram analisadas as métricas da função *loss* de treinamento comparada com a de validação, a *perplexity* do treinamento comparada com a de validação e a pontuação BLEU de cada um dos modelos. A métrica de *perplexity* indica o quão bem um modelo de probabilidade prediz uma amostra, onde nosso modelo de linguagem é uma distribuição de probabilidade sobre frases. No geral, quanto menor a *perplexity*, melhor o resultado, indicando uma menor medida de aleatoriedade no sistema. A métrica BLEU funciona contando os *n-grams* correspondentes nas legendas geradas com os *n-grams* das legendas verdadeiras, ou seja, o BLEU-1 conta quantos *1-grams* tem de equivalência entre a legenda real e a gerada, o BLEU-2 conta quantos *2-grams* tem de equivalência entre a legenda real e a gerada, e assim por diante. Quanto maior o BLEU, mais perto as legendas geradas estão das reais.

Na tabela 4.2 o "R" significa real e o "S" sintético. Assim, se tivermos um modelo de treino R+S e teste R+S isso significa que o modelo foi treinado por imagens reais e sintéticas e testado por imagens reais e sintéticas.

Inicialmente, observando a Figura 4.4 podemos ver a comparação treinamento × validação da *loss function* na Figura 4.4a, e da *perplexity* na Figura 4.4b, do modelo treinado apenas com imagens reais. Vemos na última época (14) a *loss* do treinamento chegar em 3.27 e a da validação chegar em 3.37. As curvas da Figura 4.4a estão próximas, mas após a 7^a época começam a divergir, com a curva de validação começando a subir e se distanciar da curva de treinamento, o que pode indicar o início de um possível *overfitting*. Isso indica que o modelo tem uma boa performance no treinamento, mas talvez não tão boa com novos dados, podendo perder a capacidade de generalizar. O *overfitting* poderia ser confirmado com o treinamento de mais épocas, mas é difícil dizer com uma variação

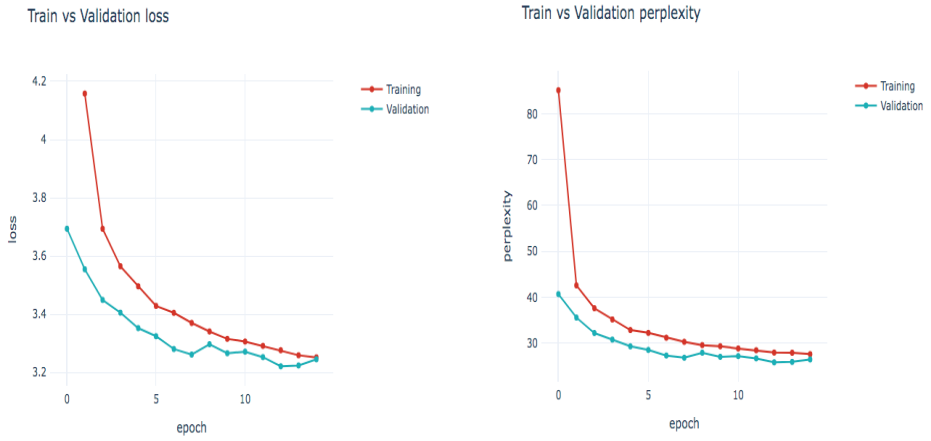


(a) Loss (b) Perplexity

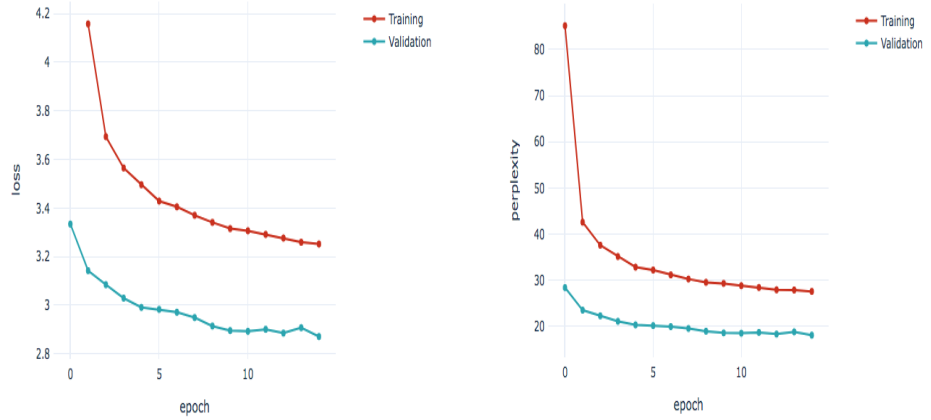
Figura 4.4: *Loss* e *Perplexity* do modelo de base, apenas com imagens reais.

de apenas 0.1 entre validação e treinamento. No geral, é um bom modelo que performa aproximadamente bem em novos dados. Podemos ver na Figura 4.4b as curvas atingirem 29.0 para o *perplexity* no treinamento e 29.7 para o *perplexity* na validação.

Ao observarmos a Figura 4.5, podemos ver as curvas do treinamento × validação da *loss function* para validação com imagens sintéticas e reais na Figura 4.5a, e para validação apenas com imagens reais na Figura 4.5c, todas correspondendo ao modelo de geração de imagens sintéticas AttnGAN. As Figuras 4.5b e 4.5d representam a *perplexity* para validação com imagens reais e sintéticas e apenas para imagens reais, respectivamente. Podemos ver na Figura 4.5a as curvas de treinamento e validação terminando em 3.24 para a *loss*, uma boa melhora se comparada com a Figura 4.4a. Podemos ver a diferença a partir da 7^a época, onde as duas curvas ficam mais próximas ao invés de se afastarem, indicando uma boa aproximação do modelo. Na Figura 4.5c podemos ver a curva de



(a) Loss (S+R) (b) Perplexity (S+R)

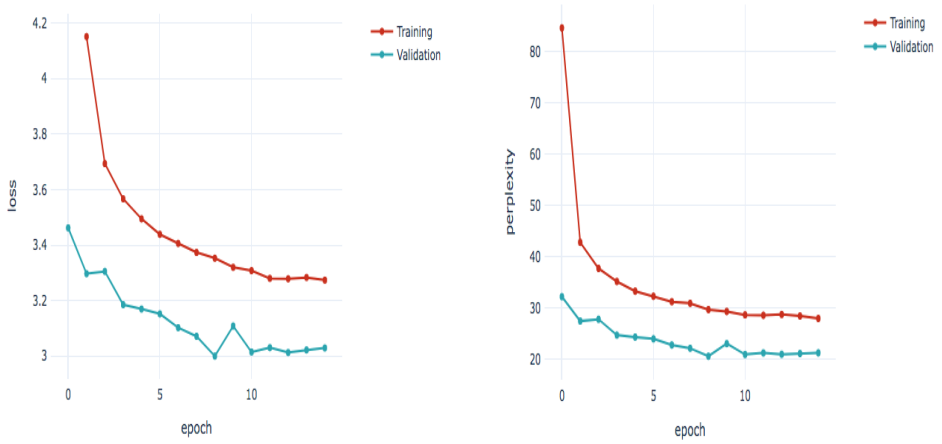


(c) Loss (R) (d) Perplexity (R)

Figura 4.5: *Loss* e *Perplexity* do treinamento com AttnGAN, com imagens reais e sintéticas. (a) e (b) mostram a validação com imagens sintéticas e reais. (c) e (d) representam a validação apenas com imagens reais.

validação terminando em 2.83 para a *loss*, mostrando que o modelo treinado com imagens reais e sintéticas geradas pela AttnGAN tem uma performance melhor se testado em imagens reais, conforme sugerido por Hossain *et al.* [3]. O gráfico da Figura 4.5c sugere exatamente isso: uma *loss* menor em toda validação sugere que o modelo performa melhor em imagens reais. O mesmo pode ser estendido para a *perplexity*: vemos o treinamento terminando em 28.8, a validação da Figura 4.5b terminando em 28.6 e a validação da Figura 4.5d terminando em 19.8.

A Figura 4.6 nos mostra o treinamento × validação para a *loss* e o *perplexity* do modelo treinado com imagens reais e sintéticas do AttnGAN+CL, onde a *loss* do treinamento termina em 3.28. Ao compararmos a *loss* da validação com imagens reais e sintéticas na Figura 4.6a, que termina em 3.06, com a *loss* da validação apenas com imagens reais da Figura 4.6c, que termina em 3.18, percebemos algo inédito: o modelo performa melhor



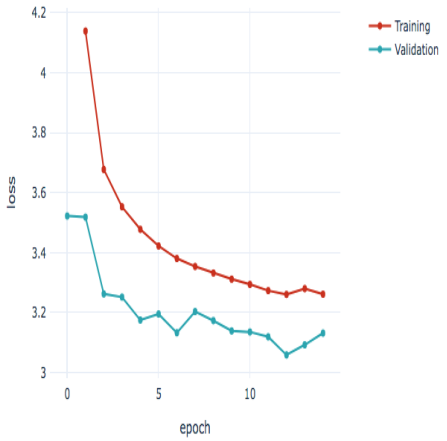
(a) Loss (S+R) (b) Perplexity (S+R)

(c) Loss (R) (d) Perplexity (R)

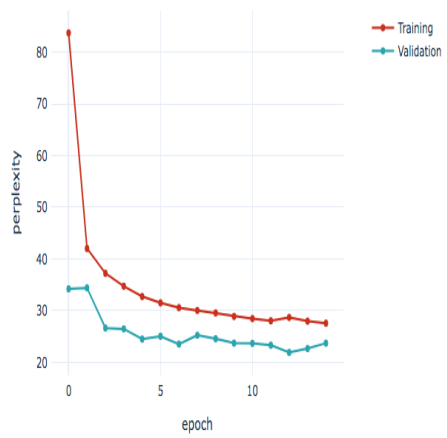
Figura 4.6: *Loss* e *Perplexity* do treinamento com AttnGAN+CL, com imagens reais e sintéticas. (a) e (b) mostram a validação com imagens sintéticas e reais. (c) e (d) representam a validação apenas com imagens reais.

no *dataset* com imagens reais e sintéticas do que apenas com imagens reais. Isso nos indica que talvez o modelo AttnGAN+CL esteja mais apto a legendar imagens sintéticas do que imagens reais. Percebemos o mesmo comportamento nas curvas de *perplexity*, que no treinamento com imagens reais e sintéticas da Figura 4.6b termina com validação de 21.3, e no treinamento, apenas com imagens reais, termina com a validação de 24.7. O treinamento do *perplexity* terminou em 28.8.

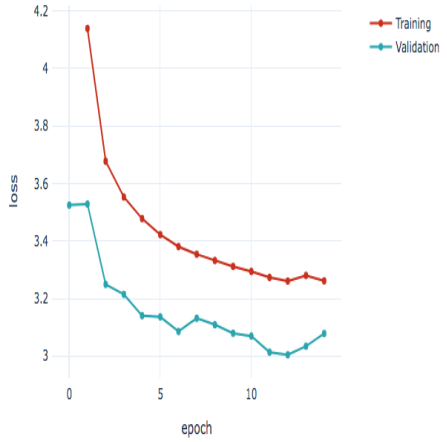
A Figura 4.7 nos mostra o treinamento × validação para a *loss* e o *perplexity* do modelo treinado com imagens reais e sintéticas do DM-GAN. A curva de treinamento para a *loss* termina em 3.23, e percebemos que as curvas da *loss* para validação com imagens reais e sintéticas da Figura 4.7a, que termina em 3.17, e da *loss* para validação com imagens reais da Figura 4.7c, que termina em 3.14, são extremamente similares. As curvas de validação abaixo das de treinamento nos indicam que o modelo performa melhor



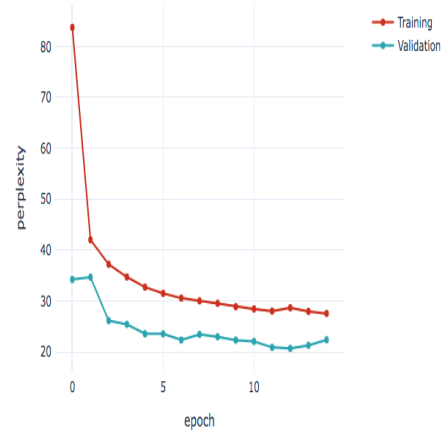
(a) Loss (S+R)



(b) Perplexity (S+R)



(c) Loss (R)

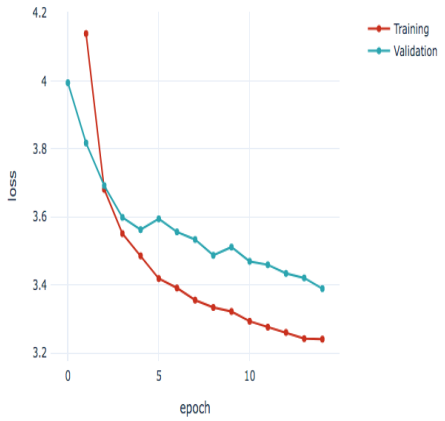


(d) Perplexity (R)

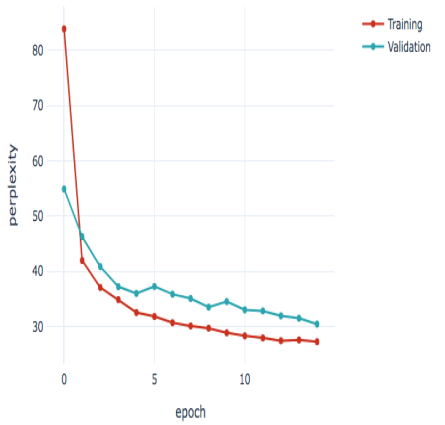
Figura 4.7: *Loss* e *Perplexity* do treinamento com DM-GAN, com imagens reais e sintéticas. (a) e (b) mostram a validação com imagens sintéticas e reais. (c) e (d) representam a validação apenas com imagens reais.

na validação do que no treinamento, além de informar que o modelo performa bem tanto no *dataset* com imagens reais e sintéticas, quanto no *dataset* apenas de imagens reais, apesar de ter uma melhor performance apenas com imagens reais. As curvas de *perplexity* nos indicam exatamente essa situação, onde a curva de treinamento termina em 28.7, a curva de validação para imagens reais e sintéticas da Figura 4.8b termina em 24.3, e a curva de validação apenas para imagens reais da Figura 4.7d termina em 22.7.

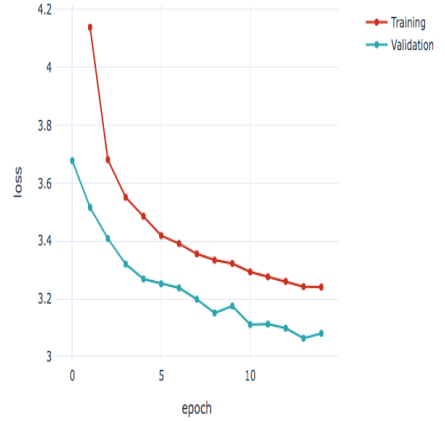
Ao observarmos a Figura 4.8, podemos ver as curvas do treinamento × validação da *loss function* para validação com imagens sintéticas e reais na Figura 4.8a e para validação apenas com imagens reais na Figura 4.8c, correspondente ao modelo de imagens geração de imagens sintéticas DM-GAN+CL. As Figuras 4.8b e 4.8d representam a *perplexity* para validação com imagens reais e sintéticas e apenas imagens reais, respectivamente. Na Figura 4.8a vemos a menor *loss* para a curva de treinamento, terminando em 3.22,



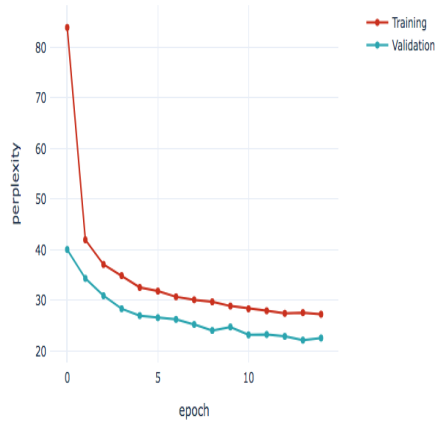
(a) Loss (S+R)



(b) Perplexity (S+R)



(c) Loss (R)



(d) Perplexity (R)

Figura 4.8: *Loss* e *Perplexity* do treinamento com DM-GAN+CL, com imagens reais e sintéticas. (a) e (b) mostram a validação com imagens sintéticas e reais. (c) e (d) representam a validação apenas com imagens reais.

e uma curva de validação que também decai ao logo das épocas, mas que termina em 3.39. Isso nos indica que o modelo tem uma performance melhor no treinamento do que com imagens novas, mas diferente da Figura 4.4a, não vemos a curva voltar a subir. A curva de validação da Figura 4.8c termina em 3.09, e nos mostra algo similar ao que vimos na Figura 4.5c: o modelo tem uma melhor performance ao ser testado apenas com imagens reais. Podemos ver o mesmo acontecer nas Figuras 4.8b e 4.8d, onde a curva de treinamento termina em 28.6, e as curvas de validação terminam em 30.3 e 23.4, respectivamente.

Ao analisarmos a Tabela 4.2, confirmamos o que pode ser observado anteriormente: os modelos testados apenas em imagens reais têm performances melhores se comparados aos testados em imagens sintéticas e reais, com exceção do modelo AttnGAN+CL, que performou melhor quando testado com imagens reais e sintéticas. Se levarmos em conta a pontuação BLEU, as pontuações BLEU mais importantes são a BLEU-3 e a BLEU-4, e nessas pontuações vemos uma melhora significativa dos modelos AttnGAN, DM-GAN e DM-GAN+CL comparados ao modelo de base. Como sugerido por Hossain *et al.* [3], o modelo AttnGAN teve uma ótima performance no teste de imagens reais, porém o modelo DM-GAN teve a melhor performance, tanto no teste com imagens reais e sintéticas quanto no teste com imagens reais, com exceção do BLEU-3 no teste de imagens reais, onde o AttnGAN performou melhor.

Tabela 4.2: Comparativo da pontuação BLEU entre o modelo de base, o modelo treinado com imagens sintéticas do AttnGAN, do AttnGAN+CL,do DM-GAN, do DM-GAN+CL e do modelo de Hossain [3].

Modelo	BLEU-1	BLEU-2	BLEU-3	BLEU-4
Modelo de Base (Treino R;Teste R)	67.30	46.84	28.31	14.82
AttnGAN (Treino R+S;Teste R+S)	66.52	44.46	27.92	16.99
AttnGAN (Treino R+S;Teste R)	68.79	46.77	30.33	18.57
AttnGAN+CL (Treino R+S;Teste R+S)	66.35	45.12	26.93	16.76
AttnGAN+CL (Treino R+S;Teste R)	67.88	44.99	26.87	16.00
DM-GAN (Treino R+S;Teste R+S)	69.40	**48.92**	**31.12**	**20.01**
DM-GAN (Treino R+S;Teste R)	**70.42**	47.21	29.98	18.79
DM-GAN+CL (Treino R+S;Teste R+S)	64.90	43.78	26.0	15.65
DM-GAN+CL (Treino R+S;Teste R)	65.38	44.76	28.67	17.06
Hossain [3] (Treino R+S;Teste R)	73.6	54.7	44.2	33.6

No Capítulo 5 serão discutidos os diferentes resultados obtidos nesse capítulo, bem como considerações gerais sobre o teor e objetivo deste trabalho e as principais diferenças entre o modelo utilizado e a performance obtida por Hossain *et al.* [3].

5 Discussão

O objetivo deste trabalho foi fazer uma comparação justa, e testar as hipóteses sobre aumento de performance de um gerador de legendas pela introdução de imagens sintéticas no treinamento do modelo que surgiram do trabalho de Hossain *et al.* [3]. É importante destacar as diferenças obtidas por Hossain *et al.* [3], e os motivos pela melhor performance, como retratado na Tabela 4.2. O modelo de Hossain tem um custo computacional consideravelmente maior, onde foi adaptado o modelo de Xu *et al.* [7], e cujo tempo médio de treinamento durou 28 dias, utilizando uma TITAN Xp GPU. Dentre as principais diferenças de implementação estão:

- O uso da DenseNet121 com camadas completamente conectadas como CNN codificadora;
- O uso do modelo de Xu *et al.* [7], com um custo computacional elevado;
- O treinamento do modelo por 100 épocas, totalizando uma média de 28 dias.

Dito isso, o modelo utilizado neste trabalho teve uma performance extremamente satisfatória. Também serviu como resposta para alguns métodos de tratamento e construção do *dataset* por Hossain *et al.* [3] que não ficaram claros. Ao utilizar um *dataset* misto de treinamento, com imagens sintéticas e reais, Hossain não explicitou em seu artigo como foi o ordenamento do *dataset*. Não foi explicado se foram substituídas as imagens reais pelas sintéticas, a proporção de substituição ou se as imagens sintéticas foram utilizadas apenas para expandir o *dataset* inicial. Como explicado anteriormente, uma maior quantidade de dados para treinamento de um modelo tende a gerar resultados melhores, o que nos leva a considerar que talvez o modelo de Hossain tenha performado melhor devido a um maior número de imagens de treinamento, e não necessariamente por causa do uso de imagens sintéticas por si só.

Grande parte do que se foi feito neste trabalho se deu no intuito de provar se apenas a introdução de imagens sintéticas no *dataset*, sem aumentar o seu tamanho, se provaria suficiente para aumentar a performance de um modelo gerador de legendas. Foi nesse sentido que se manteve o tamanho do *dataset* de treinamento em 82783 imagens, exatamente o mesmo número de imagens utilizado para treinar o modelo de base. A escolha

de uma proporção de 70% de imagens reais e 30% de imagens sintéticas também foi arbitrária, complementando essa hipótese. Assim, é importante perceber que as comparações para fins de estudo desta monografia devem ser feitas com relação ao modelo de base utilizado, treinado apenas com imagens reais, já que esse funciona como a real métrica de comparação para os testes finais.

Deste modo, já se pode pensar em diversas maneiras de se aumentar ainda mais a performance do modelo:

- Trocar a CNN codificadora usada por uma com melhor performance nesse tipo de modelo. Nesse caso, trocar a Resnet152 pela DenseNet121;
- Encontrar a proporção de imagens reais e sintéticas otimizada para esse modelo;
- Utilização da atenção estocástica dupla no módulo de atenção, conforme sugerido por Xu *et al.* [7];
- Realizar um treinamento com maior custo computacional e consequentemente mais épocas;
- Utilizar o modelo de Xu *et al.* [7] com imagens sintéticas do DM-GAN.

Também é importante discutir como os modelos com *contrastive learning* não tiveram necessariamente um resultado melhor nas métricas utilizadas, conforme era esperado. Como foi demonstrado na Tabela 4.1, as imagens geradas pela DM-GAN+CL têm os melhores resultados, além de serem imagens com uma qualidade maior e serem mais próximas da realidade. Nesse sentido, também é importante levar em consideração a performance qualitativa dos modelos de geração de legendas. Ficou demonstrado que o DM-GAN, em muitas ocasiões, gerou legendas que descreviam melhor as imagens do que o DM-GAN+CL, muitas vezes descrevendo objetos que foram "esquecidos". Nesse sentido, talvez o uso de outras métricas, além das utilizadas, fossem necessárias para um melhor julgamento e separação da performance dos modelos, como por exemplo, o METEOR, o ROUGH-L e o CIDEr-D. Também é importante salientar a possibilidade da existência de situações específicas onde talvez um modelo tenha uma performance melhor do que o outro.

Conforme foi observado, o *contrastive learning* é capaz tanto de melhorar quanto piorar a performance do modelo, a depender da ocasião. É possível que o *contrastive learning* tenha uma melhora limitada e específica na legenda de imagens, conforme sugerido pelo modelo AttnGAN+CL, que aparenta ter uma melhor performance na legenda de imagens sintéticas, inclusive melhor do que qualquer outro modelo. O próprio DM-GAN+CL teve uma boa performance, porém o DM-GAN foi o melhor modelo, comprovando que o *contrastive learning* não funciona para situações mais generalizadas

A partir dos resultados obtidos, podemos afirmar que a legenda de imagens utilizando imagens sintéticas do DM-GAN é superior ao método de AttnGAN proposto por Hossain *et al.* [3], se levarmos em conta o modelo de base utilizado. Levando em consideração os resultados deste trabalho, é possível que se utilizado o modelo de Xu *et al.* [7] combinado com o DM-GAN, resultados superiores aos de Hossain *et al.* [3] são não apenas possíveis, como esperados.

No Capítulo 6 serão tratadas as conclusões e considerações finais do trabalho.

6 Conclusões

O objetivo inicial deste trabalho era comprovar que a introdução de imagens sintéticas no treinamento de um modelo gerador de legendas para imagens reais aumentaria sua performance, conforme sugerido por Hossain *et al.*, mas não totalmente esclarecido. Foram necessários estudos para manter o tamanho original do *dataset* do modelo de base com a preocupação de um viés tendencioso relacionado ao tamanho da base de dados usada. Quatro tipos diferentes de GANs foram utilizadas para se testar o gerador de legendas, e era esperado que o estado da arte para geração de imagens [1] também produziria os melhores resultados na geração de descrições textuais. Quando o DM-GAN [9] obteve os melhores resultados, uma nova discussão é necessária sobre a eficiência do *contrastive learning* na geração de legendas.

Como foi discutido, é uma possibilidade real que o *contrastive learning* tenha um papel nichado na geração de legendas, conforme sugerido pelos resultados obtidos com o AttnGAN+CL, o qual aparenta ter mais eficiência na legenda de imagens sintéticas dos que reais. Futuros estudos são necessários sobre a validade e utilização de GANs diferentes para situações diferentes.

Mesmo utilizando um modelo mais simples, menos robusto e com menor custo computacional do que Hossain [3], o modelo utilizado teve uma ótima performance, melhor do que muitos modelos anteriores citados em [3], e que há alguns anos eram considerados o estado da arte na geração de legendas para imagens. Mais do que isso, se comprovou que apenas com a introdução de imagens sintéticas, sem mudar o tamanho do *dataset* inicial, foi possível aumentar substancialmente a performance de um gerador de legendas se comparados ao modelo de base utilizado. Foi confirmada a hipótese de Hossain *et al.* [3] sobre o poder das imagens sintéticas geradas por GANs no treinamento de um gerador de legendas e sua capacidade de aumentar a performance na geração de descrições para imagens reais.

Foi levantada a hipótese de melhor performance do modelo DM-GAN sobre o modelo AttnGAN na geração de legendas, e é necessário um estudo mais abrangente com o modelo utilizado por Hossain *et al.* utilizando a DM-GAN, com o intuito de comprovar o que foi demonstrado nesse trabalho.

Referências

[1] Ye, Hui, Xiulong Yang, Martin Takac, Rajshekhar Sunderraman e Shihao Ji: *Improving text-to-image synthesis using contrastive learning.* arXiv preprint arXiv:2107.02423, 2021. ix, xi, 3, 4, 6, 7, 8, 20, 21, 22, 23, 24, 25, 33, 34, 35, 48

[2] Lin, Tsung Yi, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár e C Lawrence Zitnick: *Microsoft coco: Common objects in context.* Em *European conference on computer vision*, páginas 740–755. Springer, 2014. ix, xi, 5, 8, 19, 20, 29, 33, 34, 35

[3] Hossain, Md. Zakir, Ferdous Sohel, Mohd Fairuz Shiratuddin, Hamid Laga e Mohammed Bennamoun: *Text to image synthesis for improved image captioning.* IEEE Access, 9:64918–64928, 2021. xi, 1, 3, 5, 6, 7, 11, 24, 35, 41, 44, 45, 47, 48

[4] Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville e Yoshua Bengio: *Generative adversarial nets.* Advances in neural information processing systems, 27, 2014. 1, 2, 4, 9, 14, 20, 23

[5] Dai, Bo, Sanja Fidler, Raquel Urtasun e Dahua Lin: *Towards diverse and natural image descriptions via a conditional gan.* Em *Proceedings of the IEEE International Conference on Computer Vision*, páginas 2970–2979, 2017. 2, 5

[6] Chen, Chen, Shuai Mu, Wanpeng Xiao, Zexiong Ye, Liesi Wu e Qi Ju: *Improving image captioning with conditional generative adversarial nets.* Em *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, páginas 8142–8150, 2019. 2, 5, 6

[7] Xu, Kelvin, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel e Yoshua Bengio: *Show, attend and tell: Neural image caption generation with visual attention.* Em *International conference on machine learning*, páginas 2048–2057. PMLR, 2015. 3, 5, 7, 24, 25, 26, 28, 30, 31, 45, 46, 47

[8] Tao Xu, Pengchuan Zhang, Qiuyuan Huang Han Zhang Zhe Gan Xiaolei Huang Xiaodong He: *Attngan: Fine-grained text to image generation with attentional generative adversarial networks.* 2018. 3, 4, 6, 7, 8, 11, 12, 14, 19, 21, 23, 27, 33

[9] Zhu, Minfeng, Pingbo Pan, Wei Chen e Yi Yang: *Dm-gan: Dynamic memory generative adversarial networks for text-to-image synthesis.* Em *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, páginas 5802–5810, 2019. 3, 4, 6, 7, 8, 15, 19, 23, 33, 48

[10] Zhang, Han, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang e Dimitris N Metaxas: *Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks*. Em *Proceedings of the IEEE international conference on computer vision*, páginas 5907–5915, 2017. 4, 9, 14, 18, 23

[11] Hossain, MD Zakir, Ferdous Sohel, Mohd Fairuz Shiratuddin e Hamid Laga: *A comprehensive survey of deep learning for image captioning*. ACM Computing Surveys (CsUR), 51(6):1–36, 2019. 5, 24

[12] Vinyals, Oriol, Alexander Toshev, Samy Bengio e Dumitru Erhan: *Show and tell: A neural image caption generator*. Em *Proceedings of the IEEE conference on computer vision and pattern recognition*, páginas 3156–3164, 2015. 5

[13] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser e Illia Polosukhin: *Attention is all you need*. Em *Advances in neural information processing systems*, páginas 5998–6008, 2017. 7

[14] Fang, Hao, Saurabh Gupta, Forrest Iandola, Rupesh K Srivastava, Li Deng, Piotr Dollár, Jianfeng Gao, Xiaodong He, Margaret Mitchell, John C Platt *et al.*: *From captions to visual concepts and back*. Em *Proceedings of the IEEE conference on computer vision and pattern recognition*, páginas 1473–1482, 2015. 12, 13

[15] Kingma, Diederik P e Jimmy Ba: *Adam: A method for stochastic optimization*. arXiv preprint arXiv:1412.6980, 2014. 19, 36

[16] Chen, Ting, Simon Kornblith, Mohammad Norouzi e Geoffrey Hinton: *A simple framework for contrastive learning of visual representations*. Em *International conference on machine learning*, páginas 1597–1607. PMLR, 2020. 21