



CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM MODELAGEM MATEMÁTICA E COMPUTACIONAL

Visual Question Answering (VQA):
NOVAS ABORDAGENS UTILIZANDO REDES
NEURAIS DE APRENDIZADO PROFUNDO

ARACELI MARCIA SALDANHA

Orientador: Prof. Alisson Marques da Silva
CEFET-MG

BELO HORIZONTE
JANEIRO DE 2025.

ARACELI MARCIA SALDANHA

***Visual Question Answering (VQA):* NOVAS
ABORDAGENS UTILIZANDO REDES NEURAIS DE
APRENDIZADO PROFUNDO**

Dissertação apresentada ao Programa de Pós-Graduação em Modelagem Matemática e Computacional do Centro Federal de Educação Tecnológica de Minas Gerais, como requisito parcial para a obtenção do título de Mestre em Modelagem Matemática e Computacional.

Área de concentração: Modelagem Matemática e Computacional

Linha de pesquisa: Sistemas Inteligentes

Orientador: Prof. Alisson Marques da Silva
CEFET-MG

CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM MODELAGEM MATEMÁTICA E COMPUTACIONAL
BELO HORIZONTE
JANEIRO DE 2025.

S162v Saldanha, Araceli Marcia
Visual Question Answering (VQA): novas abordagens utilizando redes neurais de
aprendizado profundo / Araceli Marcia Saldanha. – 2025.
97 f.

Dissertação de mestrado apresentada ao Programa de Pós-Graduação em
Modelagem Matemática e Computacional.

Orientador: Alisson Marques da Silva.

Dissertação (mestrado) – Centro Federal de Educação Tecnológica de Minas
Gerais.

1. Recuperação de informações – Teses. 2. Linguística – Processamento de dados
– Teses. 3. Conjunto de dados – Teses. 4. Teoria dos grafos – Processamento de dados
– Teses. 5. Atenção – Modelos matemáticos – Teses. 6. Redes neurais (Computação) –
Teses. 7. Perceptron – Teses. I. Silva, Alisson Marques da. III. Centro Federal de
Educação Tecnológica de Minas Gerais. IV. Título.

CDD 004

Agradecimentos

À minha família, por todo o esforço, carinho, dedicação em cada momento e pelo incentivo que sempre me impulsionou a seguir em frente.

Aos meus amigos, pela parceria constante, pelas palavras de apoio e pelos momentos de descontração que tornaram essa caminhada mais leve e especial.

Ao meu companheiro Antônio, pelo apoio, paciência e companheirismo, que estiveram ao meu lado em cada etapa desta jornada.

Aos professores Hani Camille Yehia e Thiago Magela Rodrigues Dias, pela generosidade em dedicarem seu tempo e conhecimento para compor minha banca de avaliação e enriquecer este trabalho com suas contribuições valiosas.

Ao Prof. Alisson, pela atenção, pelo compromisso em compartilhar seu conhecimento e pela dedicação em orientar-me com paciência e sabedoria.

Aos profissionais do bandeirão, pela dedicação e carinho ao preparar refeições que não apenas sustentaram o corpo, mas também aqueceram a alma durante longos dias de estudo.

A todos os professores que, em diferentes momentos da minha trajetória, contribuíram para minha formação acadêmica e pessoal, deixando marcas que me inspiram e motivam.

Resumo

VQA (*Visual Question Answering*) é um problema cujo objetivo é projetar sistemas capazes de compreender o conteúdo de uma imagem de forma semelhante aos humanos e se comunicar efetivamente sobre essa imagem em linguagem natural. Embora o VQA tenha recebido ampla atenção nos últimos anos, ainda permanece um problema em aberto devido a diversos desafios, como vieses, limitações dos *datasets* e dificuldades na interpretabilidade dos resultados, o que faz com que essa área esteja em constante estudo. Diante desse contexto, o presente trabalho busca aprimorar a acurácia dos resultados por meio do desenvolvimento de dois métodos e quatorze abordagens, utilizando o *dataset* VizWiz. O primeiro método emprega grafos para a representação conjunta dos dados de entrada e utiliza uma rede neural para grafos, *Graph Attention Networks* (GAT), para a solução do problema. Já o segundo método realiza a fusão dos dados de entrada por meio de concatenação e emprega uma rede *Multilayer Perceptron* (MLP) para gerar as respostas. As abordagens propostas implementam os métodos desenvolvidos com diferentes configurações, variando aspectos como modelos para a representação dos dados de entrada, funções de perda, entre outros aspectos. A maioria das abordagens desenvolvidas alcançou resultados superiores aos apresentados na literatura analisada, com destaque para a abordagem 14, que obteve uma acurácia geral de 62.07, superando o melhor resultado reportado na literatura, que era 60.15. Além disso, mesmo as abordagens mais simples baseadas no método 1 demonstraram desempenho notável nas categorias *unanswerable* e *number*, atingindo acurácias de 100 e 23.58, respectivamente.

Palavras-chave: *Visual Question Answering*, VizWiz, grafos, *Graph Attention Networks*, *Multilayer Perceptron*.

Abstract

VQA (Visual Question Answering) is a problem whose goal is to design systems capable of understanding the content of an image in a human-like manner and effectively communicating about that image in natural language. Although VQA has received widespread attention in recent years, it remains an open problem due to several challenges, such as biases, limitations of datasets, and difficulties in interpreting results, which puts this area under constant study. Given this context, this work seeks to improve the accuracy of the results by developing two methods and fourteen approaches using the dataset VizWiz. The first method employs graphs for the joint representation of the input data and uses a graph neural network, Graph Attention Network (GAT), to solve the problem. The second method fuses the input data through concatenation and employs a Multilayer Perceptron (MLP) network to generate the answers. The proposed approaches implement the developed methods with different configurations, varying aspects such as models for representing the input data and loss functions, among other aspects. Most of the developed approaches achieved results superior to those presented in the analyzed literature, emphasizing approach 14, which obtained an overall accuracy of 62.07, surpassing the best result reported in the literature, which was 60.15. Furthermore, even the most straightforward approaches based on method 1 demonstrated remarkable performance in the unanswerable and number categories, reaching accuracies of 100 and 23.58, respectively.

Keywords: Visual Question Answering, VizWiz, graphs, Graph Attention Networks, Multilayer Perceptron.

Lista de Figuras

Figura 1 – Exemplo de entrada e saída em um sistema de VQA, no qual a entrada consiste em uma imagem e uma pergunta, enquanto a saída corresponde à resposta gerada para a pergunta referente a imagem.	2
Figura 2 – Exemplo para a configuração aberta (AGRAWAL et al., 2017).	7
Figura 3 – Exemplo para a configuração fechada (AGRAWAL et al., 2017).	8
Figura 4 – Publicações e citações sobre o VQA no período 2014-2024.	8
Figura 5 – Exemplos de imagem real e abstrata presentes no VQA 1.0.	10
Figura 6 – Exemplos do <i>dataset</i> VQA 2.0 ilustrando que para a mesma pergunta têm se respostas diferentes conforme a imagem associada.	10
Figura 7 – Exemplo do <i>dataset</i> VQA-CP v2 representando a diferença da distribuição das respostas na construção dos conjuntos de treino e teste.	11
Figura 8 – Exemplos do VizWiz ilustrando os desafios apresentados por este <i>dataset</i>	12
Figura 9 – Exemplo do <i>dataset</i> GQA em que as perguntas exploram a cena contida na imagem.	13
Figura 10 – Exemplos de imagens radiológicas presentes no VQA-RAD.	13
Figura 11 – Exemplos do <i>dataset</i> CLEVR cujas perguntas exploram a relação entre os objetos na imagem.	14
Figura 12 – Exemplos do <i>dataset</i> TextVQA, cujas respostas estão contidas em forma de texto nas imagens.	15
Figura 13 – Exemplos do <i>dataset</i> COCO-QA ilustrando que uma imagem pode estar associada a diferentes perguntas.	15
Figura 14 – Exemplos do DAQUAR <i>dataset</i> explorando cenas internas de um imóvel.	16
Figura 15 – Exemplos do <i>dataset</i> Visual7W, em que apenas uma das quatro respostas apresentadas é verdadeira.	17
Figura 16 – Exemplos do <i>dataset</i> OK-VQA ilustrando a diversidade de assuntos que o <i>dataset</i> possui.	18
Figura 17 – Exemplo de imagem/pergunta com um fato de suporte do <i>dataset</i> FVQA.	19
Figura 18 – Exemplo de QA do <i>dataset</i> VizWiz.	23
Figura 19 – Ilustração do processo de obtenção de atributos representativos para a imagem e pergunta.	25
Figura 20 – Exemplos de fusão de atributos.	26
Figura 21 – Exemplos de obtenção de resposta.	27
Figura 22 – Exemplo de arquitetura da MLP.	29
Figura 23 – Transformação da entrada do BERT em <i>embedding</i>	30
Figura 24 – Estrutura do CLIP. Adaptado de Radford et al. (2021).	32
Figura 25 – Módulos do BLIP2. Adaptado de Li et al. (2023).	33

Figura 26 – Estrutura do <i>Autoencoder</i> . Adaptado de Bank, Koenigstein e Giryas (2023).	34
Figura 27 – Exemplo de grafo (WEST et al., 2001).	34
Figura 28 – Exemplo de grafo cujos vértices possuem atributos.	36
Figura 29 – Arquitetura do ResNet-152 (PUSTOKHIN et al., 2023).	37
Figura 30 – Arquitetura do viT.	38
Figura 31 – <i>Pipeline</i> do Método 1 exemplificado com 4 <i>patches</i> .	52
Figura 32 – Exemplo de obtenção de <i>embeddings</i> para a pergunta e as palavras que a compõem.	54
Figura 33 – Exemplo de obtenção dos <i>embeddings</i> para a imagem e seus <i>patches</i> , sendo considerados quatro neste caso.	55
Figura 34 – As conexões entre os <i>patches</i> são demonstradas no exemplo, que utiliza apenas 4 <i>patches</i> .	56
Figura 35 – Exemplo das conexões de um <i>patch</i> da imagem.	56
Figura 36 – Exemplo das conexões da imagem completa.	57
Figura 37 – Exemplo das conexões da pergunta inteira.	57
Figura 38 – Exemplo de análise de <i>dependency parser</i> .	58
Figura 39 – Exemplo das conexões para o vértice representando a palavra <i>type</i> .	58
Figura 40 – Exemplo de <i>pipeline</i> do Método 2 (DEUSER et al., 2022).	59
Figura 41 – Exemplo da ação conjunta das tarefas (i) e (ii).	61
Figura 42 – Exemplo de anotação para um par imagem/pergunta do VizWiz-std.	68
Figura 43 – Imagem <i>VizWiz_train_00000000.jpg</i> da base VizWiz-std.	69
Figura 44 – Rede do <i>Autoencoder</i> .	88

Lista de Tabelas

Tabela 1 – Tamanhos dos <i>datasets</i> presentes na literatura do VQA. (*) Valores aproximados. (**) 0.1 é utilizado no conjunto <i>challenge</i>	19
Tabela 2 – Resultados de acurácia obtidos em diferentes trabalhos da literatura VQA, destacando as melhores performances.	21
Tabela 3 – Composição dos elementos utilizados nas abordagens.	67
Tabela 4 – Comparação dos resultados das abordagens com a literatura.	71
Tabela 5 – Parâmetros utilizados no treinamento da rede <i>Autoencoder</i>	89

Lista de Quadros

Quadro 1 – Subtarefas de visão computacional requeridas para resolver o VQA. Adaptado de Manmadhan e Kooor (2020a).	2
Quadro 2 – Desafios referentes a alguns <i>datasets</i> VQA.	9
Quadro 3 – Métricas comumente usadas nos <i>datasets</i>	25
Quadro 4 – <i>Datasets</i> , modelos e métricas usados nos trabalhos da literatura VQA avaliada.	50
Quadro 5 – Diferenças entre os métodos 1 e 2.	62

Lista de Abreviaturas e Siglas

ABC-CNN - *An Attention Based Convolutional Neural Network*

AHR - *Answer Hit Rate*

ANR - *Answer Noise Rate*

BERT - *Bidirectional Encoder Representations from Transformers*

BC - *Base de Conhecimento*

BLIP - *Bootstrapping Language-Image Pre-training*

CE-GAT - *Conditional Enhanced Graph Attention Network*

CLIP - *Contrastive Language-Image Pre-Training*

CNN - *Convolutional Neural Networks*

COMET - *Commonsense Transformer*

FSWAP - *Feature Swapping*

GAT - *Graph Attention Networks*

GAT2R - *Graph Attention Network Relational Reasoning*

GloVe - *Global Vectors for Word Representation*

GNN - *Graph Neural Network*

GPT-3 - *Generative Pre-trained Transformer 3*

GradCAM - *Gradient-weighted Class Activation Mapping*

GRU - *Gated Recurrent Unit*

GQA - *Graph Question Answering*

GVQA - *Grounded Visual Question Answering*

IA - *Inteligência Artificial*

LIME - *Local Interpretable Model-Agnostic Explanations*

LLMs - *Large Language Models*

LoRRA - *Look, Read, Reason & Answer*

LSTM - *Long Short Term Memory*

MAGP - Mapa de atenção guiada por perguntas

MDETR - *DEtection TRansformer model*

MLP - *Multilayer Perceptron*

MS COCO - *Microsoft Common Objects in Context*

NMN - *Neural Module Network*

OCR - *Optical Character Recognition*

OPT - *Open pre-trained transformer language models*

OWL - *Open-World Localization*

PNP-VQA - *Plug-and-Play VQA*

PLN - Processamento de Linguagem Natural

RCR - Representação de Conhecimento e Raciocínio

ReGAT - *Relation-aware Graph Attention Network*

ROI - *Regions of Interest*

RNN - *Recurrent Neural Networks*

T5-XXL - *Text-To-Text Transfer Transformer-XXL*

VGGNet - *Visual Geometry Group Net*

VLC-BERT - *Vision-Language Commonsense BERT*

VQA - *Visual Question Answering*

VC - Visão Computacional

WUPS - *Wu-Palmer Similarity measure set*

Sumário

1 – Introdução	1
1.1 Preliminares	1
1.2 Motivação e Desafios	3
1.3 Objetivos	5
1.3.1 Objetivo Geral	5
1.3.2 Objetivos Específicos	5
1.4 Contribuições da Dissertação	5
1.5 Organização do Trabalho	6
2 – Fundamentação Teórica	7
2.1 VQA	7
2.1.1 <i>Datasets</i>	9
2.1.2 Métricas	22
2.1.3 Relação de métricas e <i>datasets</i>	25
2.2 <i>Pipeline</i> Geral do VQA	25
2.2.1 Extração de atributos	25
2.2.2 Fusão dos atributos	26
2.2.3 Obtenção de respostas	27
2.3 Inteligência Artificial	28
2.3.1 MLP	28
2.3.2 BERT	29
2.3.3 CLIP	31
2.3.4 BLIP2	32
2.3.5 <i>Autoencoder</i>	33
2.3.6 Grafos	34
2.3.7 GNNs e GAT	35
2.3.8 ResNet-152	36
2.3.9 ViT	37
3 – Revisão da Literatura	40
3.1 Trabalhos Relacionados	40
3.2 Resumo da Revisão da Literatura	49
4 – Abordagem Proposta	52
4.1 Método 1	52
4.1.1 Pré-processamento	53

4.1.2	Extração de atributos	54
4.1.3	Fusão dos atributos	55
4.1.4	Obtenção das respostas	57
4.2	Método 2	59
4.2.1	Extração de atributos	59
4.2.2	Fusão dos atributos	60
4.2.3	Obtenção das respostas	60
4.3	Comparação dos métodos	61
4.4	Abordagens	62
5	Resultados	68
5.1	Metodologia dos Experimentos	68
5.2	Resultados Experimentais e Comparações	70
6	Considerações Finais	73
	Referências	75
	 Apêndices	 87
	APÊNDICE A – Autoencoder	88
	APÊNDICE B – Arquitetura das Abordagens	90
	APÊNDICE C – Bibliotecas	97

1 Introdução

Neste capítulo, apresenta-se uma visão inicial do problema abordado, destacando os desafios e as motivações que guiam esta pesquisa. A Seção 1.1 fornece uma contextualização do problema, enquanto a Seção 1.2 discute as motivações e os desafios relacionados ao tema. A Seção 1.3 detalha os objetivos a serem alcançados ao longo da pesquisa. As contribuições específicas da pesquisa são descritas na Seção 1.4. Finalmente, o capítulo é encerrado com a Seção 1.5, que resume o conteúdo de cada um dos capítulos subsequentes.

1.1 Preliminares

A capacidade cognitiva humana permite identificar objetos em uma imagem, compreender suas posições espaciais, inferir atributos e relações entre eles, e raciocinar sobre o propósito de cada elemento em um contexto específico. Além disso, a capacidade cognitiva possibilita formular perguntas sobre as imagens e comunicar as informações obtidas a partir desta análise. Entretanto, como fazer com que um computador realize essa tarefa com eficiência similar à humana? O desenvolvimento de sistemas de visão computacional capazes de responder a perguntas em linguagem natural sobre imagens tem sido um desafio ambicioso e, por muito tempo, considerado intratável. No entanto, desde 2014, observou-se um progresso significativo nesse campo, resultando no surgimento do VQA (abreviação de *Visual Question Answering*). O objetivo do VQA é projetar sistemas que compreendam o conteúdo de uma imagem de forma semelhante à dos humanos e se comuniquem sobre ela em linguagem natural. Isso significa que o sistema recebe uma imagem e uma pergunta em linguagem natural como entrada, e produz uma resposta também em linguagem natural (ZHANG et al., 2016; KAFLE; KANAN, 2017), conforme ilustrado na Figura 1. Essa tarefa é desafiadora, pois exige que modelos de visão computacional e modelos de linguagem natural interajam e se complementem. O Quadro 1¹ apresenta alguns exemplos de tarefas relacionadas à área de visão computacional que estão envolvidas no VQA.

¹ As perguntas e respostas apresentadas como exemplo ao longo deste trabalho estão escritas em inglês, uma vez que a maioria dos *datasets* de VQA disponíveis está nessa língua.



Figura 1 – Exemplo de entrada e saída em um sistema de VQA, no qual a entrada consiste em uma imagem e uma pergunta, enquanto a saída corresponde à resposta gerada para a pergunta referente a imagem.

Tarefas de Visão Computacional	Perguntas representantes do VQA
Reconhecimento de objetos	<i>What is in the image?</i>
Deteção de objetos	<i>Are there any dogs in the picture?</i>
Classificação de atributos	<i>What color is the umbrella?</i>
Classificação de cena	<i>Is it raining?</i>
Contagem	<i>How many people are there in the image?</i>
Reconhecimento de atividades	<i>Is the child crying?</i>
Relacionamento espacial entre objetos	<i>What is between cat and sofa?</i>
Raciocínio de senso comum	<i>Does this person have 20/20 vision?</i>
Raciocínio da base de conhecimento	<i>Is this a vegetarian pizza?</i>

Quadro 1 – Subtarefas de visão computacional requeridas para resolver o VQA. Adaptado de [Manmadhan e Kooor \(2020a\)](#).

As pesquisas em VQA abrangem diversas áreas da Inteligência Artificial (IA), incluindo Visão Computacional (VC), Processamento de Linguagem Natural (PLN) e Representação de Conhecimento e Raciocínio (RCR). A RCR tem a capacidade de raciocinar e extrair semântica das mídias processadas, enquanto o PLN atua na compreensão das perguntas e na geração de respostas com base nos resultados obtidos pela VC, que é responsável por adquirir e processar as imagens.

Para obter bons resultados nas tarefas de VQA, é essencial que o modelo vá além de apenas processar imagens e textos separadamente. O modelo computacional deve aprender a raciocinar conjuntamente sobre ambas as representações para responder de forma eficaz às perguntas. Por exemplo, ao analisar a foto de um gato, a pergunta *Is the cat black?* pode ser respondida com base nas informações visuais, desde que o modelo entenda que a pergunta se refere à cor do gato. A orientação sobre o que deve ser analisado vem da parte textual. Dessa forma, o modelo VQA precisa integrar informações de diferentes modalidades e raciocinar sobre essa representação combinada ([PANDHRE; SODHANI, 2017](#)).

1.2 Motivação e Desafios

Um dos fatores que desperta o interesse dos pesquisadores no estudo do VQA é a ampla gama de potenciais aplicações. Entre elas, destaca-se uma aplicação de grande relevância social e impacto direto, como evidenciado por [Kafle e Kanan \(2017\)](#) e [Manmadhan e Kovoov \(2020a\)](#), que consiste em auxiliar pessoas cegas ou com deficiência visual, permitindo que obtenham informações sobre imagens tanto no ambiente virtual quanto na vida real. Por exemplo, ao navegar em suas redes sociais, um usuário com deficiência visual pode se beneficiar de um sistema de legendas descritivas vinculadas a uma imagem, recorrendo ao VQA para explorar a imagem de forma mais aprofundada.

Além dessa aplicação específica, o VQA pode ser empregado para aprimorar a interação entre humanos e computadores, proporcionando um meio mais natural de consulta de conteúdo visual. Suas implicações abrangem, por exemplo, o aprimoramento de mecanismos de busca por imagens, algo de grande utilidade comercial para plataformas de compras *online* ([MANMADHAN; KOVOOR, 2020a](#)), já que a precisão nos resultados pode atrair clientes ao atender suas consultas de forma mais eficiente. Adicionalmente, o VQA pode facilitar a extração de informações em grandes *datasets*. Por exemplo, em uma busca por imagens capturadas em condições chuvosas, bastaria perguntar *Is it raining?* em relação às imagens do *dataset*.

A inclusão do VQA também pode aumentar a eficácia de serviços educacionais *online* ([KOSHTI et al., 2024](#)), permitindo que os alunos interajam com elementos visuais. Nesse contexto, o VQA pode ser utilizado para instruir crianças em jogos ou atividades educativas. Outro campo de aplicação do VQA surge na análise de dados de sistemas de vigilância, onde pode auxiliar analistas a resumir as informações visuais disponíveis.

Em uma evolução futura, o conceito de *Visual Dialogue* pode até mesmo orientar ações em linguagem natural para robôs ([KAFLE; KANAN, 2017](#); [MANMADHAN; KOVOOR, 2020a](#)). Na área médica, como discutido por [Hu et al. \(2024\)](#), o VQA pode oferecer uma segunda opinião na interpretação de imagens médicas, diminuindo a chance de diagnósticos incorretos. Além disso, o VQA tem o potencial de aliviar a carga de trabalho dos radiologistas, assumindo parte do papel de consultor especializado ao responder perguntas tanto de médicos quanto de pacientes ([HU et al., 2024](#)). Isso evitaria interrupções no fluxo de trabalho, aumentando a eficiência dos sistemas de saúde, especialmente em países com recursos limitados.

Embora o VQA tenha recebido ampla atenção nos últimos anos, ainda é uma tarefa aberta sem solução definitiva ([MIAO et al., 2022](#)). Diversos estudos têm abordado os desafios que impactam a evolução das soluções para o VQA, incluindo os vieses presentes nos *datasets*. Esses vieses podem derivar de várias fontes, são difíceis de rastrear e podem levar a avaliações de modelos severamente enganosas ([KAFLE; SHRESTHA; KANAN,](#)

2019). Um exemplo comum é o viés do fotógrafo em *benchmarks* de visão e linguagem, onde as imagens encontradas nos repositórios frequentemente compartilham semelhanças em postura e composição devido às preferências humanas por pontos de vista específicos. Quando esses mesmos vieses estão presentes nos *datasets* de teste, os algoritmos podem simplesmente memorizar padrões superficiais (por exemplo, responder *yes* para perguntas do tipo *Is there an object in the image?*) em vez de aprender a resolver a tarefa de forma efetiva (responder *yes* apenas se o objeto estiver realmente presente) (KAFLE; SHRESTHA; KANAN, 2019).

Além disso, estudos têm mostrado o impacto dos vieses de linguagem. Por exemplo, em Agrawal et al. (2017), foi observado que o modelo de linguagem pode explorar priorizações estatísticas sutis sobre tipos de perguntas; assim, a pergunta *What color is the banana?* pode ser respondida como *yellow* sem necessidade de analisar a imagem. Em Hirota, Nakashima e Garcia (2022), constatou-se que no *dataset* GQA, esportes ou atividades ao ar livre como patinação e surfe predominam nas 20 principais respostas para perguntas masculinas, enquanto não há respostas relacionadas a essas atividades para perguntas femininas. Em Zhang et al. (2016), os resultados indicam que quase metade das perguntas no *dataset* VQA 1.0 podem ser respondidas corretamente por uma rede neural que ignora completamente a imagem e utiliza apenas a pergunta.

Outro desafio importante é a limitação dos *datasets* disponíveis para a avaliação dos modelos. Alguns *datasets*, como o DAQUAR (MALINOWSKI; FRITZ, 2014), são muito pequenos para treinar e avaliar modelos mais complexos com sucesso. Além disso, o DAQUAR contém exclusivamente cenas internas, limitando a variedade de perguntas possíveis. As imagens frequentemente apresentam desordem significativa e, em alguns casos, condições extremas de iluminação, tornando as perguntas difíceis de responder até mesmo para humanos.

O *dataset* COCO-QA (REN; KIROS; ZEMEL, 2015), enfrenta problemas devido ao algoritmo de processamento de linguagem natural usado para formular as perguntas, resultando em perguntas mal formuladas, muitas com erros gramaticais e outras completamente ininteligíveis (KAFLE; KANAN, 2017). Em Kafle, Shrestha e Kanan (2019), observa-se uma tentativa de balancear a base de dados para combater vieses, utilizando cenas visuais geradas sinteticamente. No entanto, isso não é representativo da realidade e tem utilidade limitada no mundo real.

Como destacado em Manmadhan e Koor (2020a), uma das limitações da pesquisa de VQA é a falta de *datasets* orientados a objetivos, ou seja, *datasets* cuidadosamente projetados para garantir que as informações necessárias para atingir um objetivo ou resolver um problema específico estejam incluídas. Exemplos de *datasets* disponíveis publicamente para a pesquisa de VQA orientada a objetivos são o VizWiz (GURARI et al., 2018) e VQA-Med (ABACHA et al., 2019).

Outro fator de impacto para o VQA é a falta de interpretação e confiança nos resultados. Para os seres humanos, é simples fornecer explicações, apontar evidências, transmitir confiança nas previsões e admitir *I don't know* quando as informações são insuficientes. No entanto, quase nenhum dos algoritmos de visão e linguagem existentes possui essas habilidades, tornando os modelos altamente não interpretáveis e não confiáveis (KAFLE; KANAN, 2017; KAFLE; SHRESTHA; KANAN, 2019). Ainda não está claro quais fatores contribuem para a melhoria do desempenho, sendo difícil desacoplar as contribuições de dados de texto e imagem isoladamente, o que torna complicado afirmar se os algoritmos estão certos pelas razões certas.

Uma prática comum é a visualização de mapas de atenção para métodos baseados em atenção, como o Grad-CAM (SELVARAJU et al., 2017). No entanto, essas visualizações pouco esclarecem se os modelos “prestaram atenção” às regiões corretas da imagem. Primeiro, a maioria dos *datasets* de visão e linguagem não inclui mapas de atenção que possam ser comparados com os mapas de atenção previstos, dificultando a avaliação da qualidade da previsão. Em segundo lugar, mesmo que esses dados estivessem disponíveis, não fica claro quais regiões da imagem o modelo deveria observar (KAFLE; SHRESTHA; KANAN, 2019).

1.3 Objetivos

1.3.1 Objetivo Geral

O objetivo geral deste trabalho é propor uma metodologia eficaz para a resolução do problema de VQA, com o intuito de melhorar a acurácia no *dataset* VizWiz.

1.3.2 Objetivos Específicos

Os objetivos específicos deste trabalho são:

- Propor dois métodos para resolver o problema do VQA;
- Propor múltiplas abordagens a partir de dois métodos-base, variando os modelos utilizados na representação de imagens e perguntas, como CLIP, BLIP2, BERT, entre outros parâmetros.
- Avaliar e comparar as abordagens propostas com os resultados encontrados na literatura para verificar efetividade das mesmas.

1.4 Contribuições da Dissertação

Diferentemente dos estudos presentes na literatura, este trabalho se destaca por propor, implementar e avaliar diversas abordagens voltadas especificamente para a melhoria

do desempenho no *dataset* VizWiz. A escolha desse *dataset* justifica-se pelos desafios únicos que apresenta, como a qualidade das imagens e as perguntas complexas geradas por usuários com deficiência visual, características que ainda não foram suficientemente exploradas na literatura e que exigem soluções mais robustas e adaptadas. Assim, parte-se da hipótese de que uma estrutura bem definida e otimizada para esse *dataset* desafiador tende a apresentar um desempenho igualmente satisfatório em outros contextos. As principais contribuições desta pesquisa são:

- Uma análise detalhada dos problemas e desafios associados ao VQA, destacando os principais obstáculos enfrentados pelas abordagens atuais e discutindo algumas de suas possíveis aplicações.
- Uma revisão abrangente dos principais *datasets* e métricas utilizadas na avaliação das soluções propostas para o VQA, destacando resultados do estado da arte para os *datasets* analisados, além de estabelecer a relação entre as métricas comumente aplicadas a cada *dataset*.
- Uma extensa revisão da literatura, abrangendo publicações de 2014 a 2024, explorando diversas metodologias aplicadas ao VQA.
- A proposição de novas abordagens visando a melhoria da métrica de acurácia na base VizWiz.

1.5 Organização do Trabalho

Este trabalho está organizado em seis capítulos. O Capítulo 2 apresenta um aprofundamento sobre VQA, incluindo a evolução da área, bem como os principais *datasets* e métricas utilizados. Além disso, o Capítulo 2 revisita as técnicas de Inteligência Artificial utilizadas nesta pesquisa. O Capítulo 3 revisa os artigos mais relevantes da literatura sobre VQA. No Capítulo 4, são descritos os métodos e as abordagens propostas, enquanto o Capítulo 5 discute os resultados obtidos. Finalmente, o Capítulo 6 traz as conclusões do estudo realizado e os direcionamentos para pesquisas futuras.

2 Fundamentação Teórica

Neste capítulo, são apresentados os fundamentos teóricos necessários para a compreensão deste trabalho. Na Seção 2.1, abordam-se os conceitos fundamentais relacionados ao VQA, destacando os principais *datasets* utilizados na área, como o VQA 2.0, VizWiz e GQA, entre outros, com ênfase em suas características, desafios e nas métricas de avaliação mais comuns. Em seguida, na Seção 2.2, discutem-se as principais estratégias descritas na literatura para a resolução do problema de VQA. Por fim, a Seção 2.3 explora os fundamentos de inteligência artificial, com foco nos modelos empregados neste trabalho, detalhando suas arquiteturas e características mais relevantes para as análises realizadas.

2.1 VQA

No contexto do VQA, a tarefa da máquina consiste em identificar a resposta correta para uma pergunta relacionada a uma imagem, sendo que, nos *datasets* disponíveis atualmente, essa resposta é geralmente apresentada em forma de palavras ou frases curtas. O problema pode ser abordado por meio de duas configurações principais: uma de resposta aberta e outra baseada em múltipla escolha.

Na configuração aberta, as perguntas feitas sobre uma imagem podem ter uma ampla variedade de respostas possíveis. As respostas não são limitadas a um conjunto específico de opções e podem variar em formato e conteúdo. Por exemplo, uma pergunta aberta pode ser *Why is the woman eating a salad rather than pizza?* como ilustrado na Figura 2. A resposta poderia ser *she likes salad* ou *she is on a diet*, entre outras. Nessa configuração, geralmente, utiliza-se um modelo de geração de texto para produzir a resposta.



Why is the woman eating a salad rather than pizza?

Figura 2 – Exemplo para a configuração aberta (AGRAWAL et al., 2017).

Por outro lado, na configuração de múltipla escolha as perguntas têm um conjunto limitado de respostas possíveis. Normalmente, essas respostas são simples, como *yes*

ou *no*, ou uma escolha entre algumas opções predeterminadas. Por exemplo, na Figura 3, apresenta-se a pergunta *Is the TV on?*, acompanhada de um conjunto limitado de opções de resposta, sendo a alternativa (a) a mais adequada. Neste cenário o problema geralmente é resolvido com um modelo de classificação. Isso torna a avaliação de uma resposta gerada mais fácil do que no ambiente aberto, onde a comparação entre a saída da máquina e uma resposta verdadeira (ou seja, fornecida por humanos) enfrenta problemas com sinônimos e paráfrases (TENNEY; WU; HENGEL, 2017; WU et al., 2017).

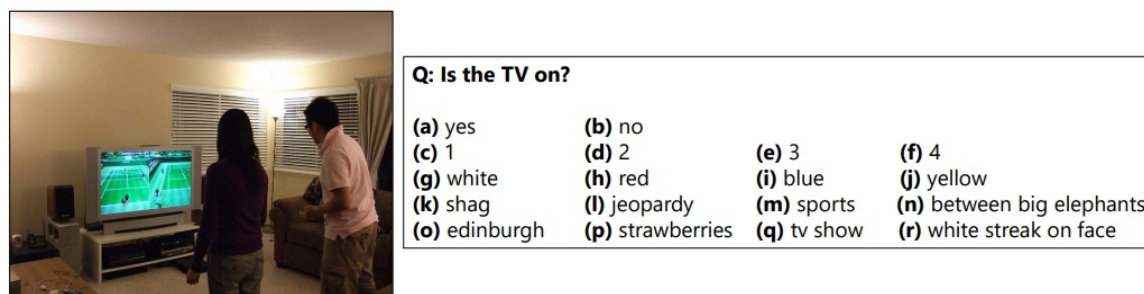


Figura 3 – Exemplo para a configuração fechada (AGRAWAL et al., 2017).

O VQA é uma área em crescente expansão, como evidenciado na Figura 4. Esta figura, extraída da plataforma *Web of Science*, apresenta o resultado de uma busca pela palavra-chave “*vqa visual question answering*” no período de janeiro de 2014 à dezembro de 2024, mostrando o aumento no número de publicações e citações relacionadas ao tema, com um total de 1.070 publicações encontradas. A busca foi refinada para evitar o uso apenas da sigla VQA, que também se refere à área de *Video Quality Assessment*. Praticamente todos os anos, desenvolvedores de *datasets* promovem desafios de VQA para incentivar o desenvolvimento de novas soluções, alguns dos quais estão listados no Quadro 2.

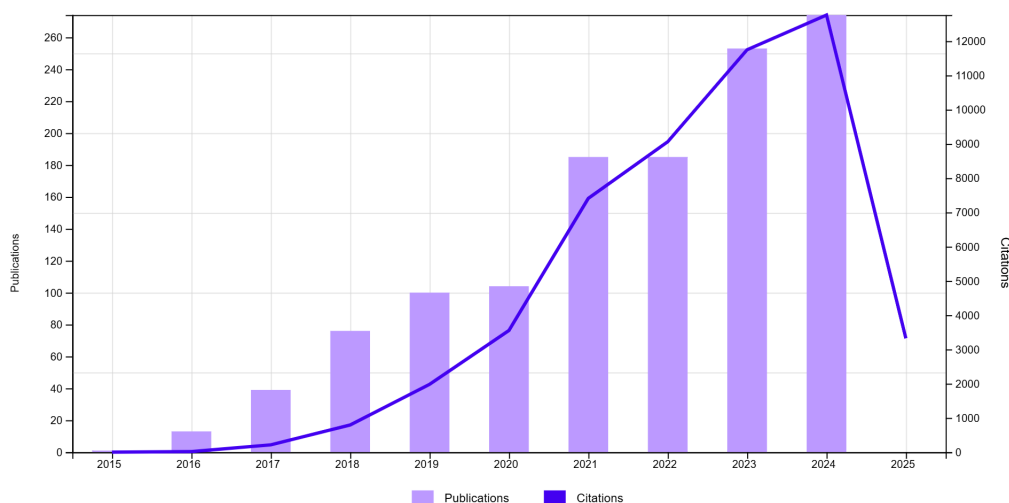


Figura 4 – Publicações e citações sobre o VQA no período 2014-2024.

Dataset	Desafio
VQA 2.0	https://eval.ai/web/challenges/challenge-page/830/overview
TextVQA	https://textvqa.org/challenge/
VQA-Med	https://www.imageclef.org/2020/medical/vqa/
CVQA	https://eval.ai/web/challenges/challenge-page/2305/overview
VizWiz	https://eval.ai/web/challenges/challenge-page/2185/overview
MMMU	https://eval.ai/web/challenges/challenge-page/2179/overview

Quadro 2 – Desafios referentes a alguns *datasets* VQA.

A literatura sobre VQA oferece uma variedade de perspectivas e abordagens. O *survey* apresentado em Wu et al. (2017) realiza uma análise detalhada dos pares de perguntas e respostas do *dataset Visual Genome*, explorando o potencial uso de grafos de cena e sua integração com uma base de conhecimento (do inglês *Knowledge Base*). Em contraste, o trabalho de Kafle e Kanan (2017) compara o VQA com outras tarefas que também envolvem visão e linguagem, como *image captioning*, discutindo como os vieses presentes em alguns *datasets* limitam a capacidade desses *benchmarks* de avaliar de forma justa os algoritmos propostos.

Além disso, os *surveys* apresentados em Manmadhan e Koor (2020b) e Ishmam, Shovon e Dey (2024) se concentram nas principais etapas envolvidas na solução do VQA, destacando os modelos mais utilizados em cada uma dessas etapas. Em contraste, o trabalho de Zou e Xie (2020) aborda de forma mais ampla algumas das soluções propostas na literatura para o VQA.

2.1.1 Datasets

Nesta seção, realiza-se uma revisão dos principais *datasets* empregados na literatura de VQA. Ressalta-se que foi adotado o termo *dataset* para se referir aos conjuntos ou bases de dados ao longo deste trabalho.

VQA 1.0: O VQA 1.0 (AGRAWAL et al., 2017) é a primeira versão do *dataset* para VQA, composto por imagens reais e abstratas, como exemplificado na Figura 5. As imagens reais foram extraídas do *Microsoft Common Objects in Context* (MS COCO), e o conjunto de treinamento contém 82.783 imagens, 248.349 perguntas e 2.483.490 respostas. O conjunto de validação inclui 40.504 imagens, 121.512 perguntas e 1.215.120 respostas, enquanto o conjunto de teste possui 81.434 imagens e 244.302 perguntas. O subconjunto de imagens abstratas é composto por cenas de desenhos animados e foi introduzido para atrair pesquisadores interessados em explorar o raciocínio complexo necessário para a tarefa de VQA. Este também é dividido em três conjuntos: o conjunto de treinamento contém 20.000 imagens, 60.000 perguntas e 600.000 respostas; o conjunto de validação com 10.000 imagens, 30.000 perguntas e 300.000 respostas; e o conjunto de teste possui 20.000 imagens e 60.000 perguntas.

Esses conjuntos apresentam vieses de linguagem, pois uma grande proporção das perguntas pode ser respondida sem a necessidade de observar as imagens correspondentes. Além disso, algumas perguntas são subjetivas, o que pode influenciar a análise da capacidade de um modelo resolver o problema do VQA (KUMARI et al., 2023).

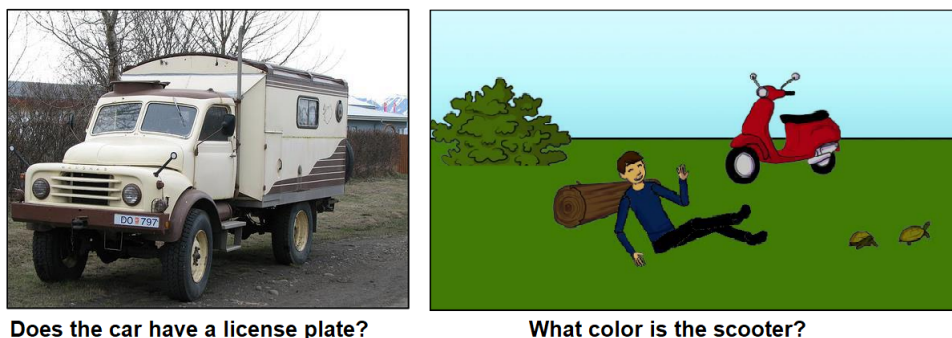


Figura 5 – Exemplos de imagem real e abstrata presentes no VQA 1.0.

VQA 2.0: O *dataset* VQA 2.0 (GOYAL et al., 2017) foi desenvolvido com base na hipótese de que um *dataset* balanceado forçaria o modelo a focar na informação visual. Para alcançar esse objetivo, foram introduzidas imagens semelhantes às do VQA 1.0, associadas à mesma pergunta, mas com respostas diferentes. Isso força o modelo a compreender as informações contidas na imagem, já que a mesma pergunta pode ter duas respostas diferentes para duas imagens distintas, conforme ilustrado na Figura 6. O VQA 2.0 consiste em aproximadamente 1.1 milhão de pares de imagem/pergunta com 13 milhões de respostas associadas. Este *dataset* possui maior abrangência, pois, ao contrário de outros que se concentram exclusivamente em cenas abstratas, foi construído de maneira balanceada a partir de imagens do mundo real provenientes do MS COCO (ZOU; XIE, 2020).



Figura 6 – Exemplos do *dataset* VQA 2.0 ilustrando que para a mesma pergunta têm-se respostas diferentes conforme a imagem associada.

VQA-CP: O *dataset* VQA-CP (*Visual Question Answering under Changing Priors*) proposto por Agrawal et al. (2018) é composto por dois subconjuntos: VQA-CP v1 e VQA-CP v2. Ambos foram criados a partir da reorganização dos conjuntos de treinamento e

validação dos *datasets* VQA 1.0 e VQA 2.0, respectivamente. A reorganização foi feita de forma que a distribuição das respostas por tipo de pergunta (como *how many*, *what is the color*, etc.) é diferente por *design* na divisão de teste em comparação com a divisão de treino. A Figura 7 ilustra um exemplo, no qual a resposta mais comum para a pergunta do tipo *What color is* seja *white* no conjunto de treinamento e a resposta pode ser *black* para a pergunta do mesmo tipo no conjunto de teste.

O *dataset* VQA-CP v1 é composto por 118 mil imagens, 245 mil perguntas e 2.5 milhões de respostas no conjunto de treinamento, além de 87 mil imagens, 125 mil perguntas e 1.3 milhão de respostas no conjunto de validação. O VQA-CP v2 abrange 121 mil imagens, 438 mil perguntas e 4.4 milhões de respostas no conjunto de treinamento, e 98 mil imagens, 220 mil perguntas e 2.2 milhões de respostas no conjunto de validação (ZOU; XIE, 2020).

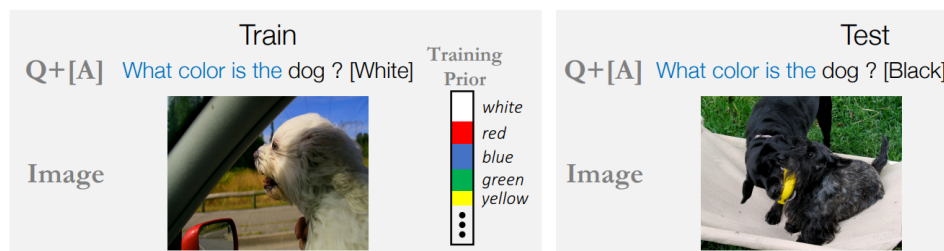


Figura 7 – Exemplo do *dataset* VQA-CP v2 representando a diferença da distribuição das respostas na construção dos conjuntos de treino e teste.

VizWiz: O *dataset* VizWiz (GURARI et al., 2018) é originado a partir de fotos tiradas por pessoas cegas usando um telefone celular, acompanhadas de perguntas faladas sobre o conteúdo das imagens. Para a construção deste *dataset*, os áudios das perguntas foram transcritos para texto e, posteriormente, o processo de *crowdsourcing* foi utilizado para fornecer 10 respostas para cada par imagem/pergunta, com uma *tag* de confiança associada a cada resposta, indicando o grau de certeza, como *yes*, *no* ou *maybe*. O conjunto de treinamento inclui 20.523 pares de imagens/perguntas e 205.230 pares de respostas/confiança, enquanto o conjunto de validação contém 4.319 pares de imagem/pergunta e 43.190 pares de confiança de resposta/resposta. O conjunto de testes compreende 8.000 pares de imagem/pergunta, distribuídos em 4 tipos distintos de respostas: *yes/no*, *number*, *others* e *unanswerable*.

Em contraste com outros *datasets*, as imagens do VizWiz são frequentemente de baixa qualidade, muitas vezes devido a problemas como iluminação insuficiente, falta de foco e enquadramento inadequado, como pode ser visto nos exemplos da Figura 8. As perguntas no VizWiz podem ser mais informais ou apresentar imperfeições na gravação de áudio, como cortes no início ou captura de áudio de fundo. Adicionalmente, não há garantia de que todas as perguntas possam ser respondidas corretamente, uma vez que as pessoas cegas não podem verificar o conteúdo visual das imagens,

que podem estar desfocadas, mal iluminadas ou até mesmo cobertas por dedos, entre outros problemas.

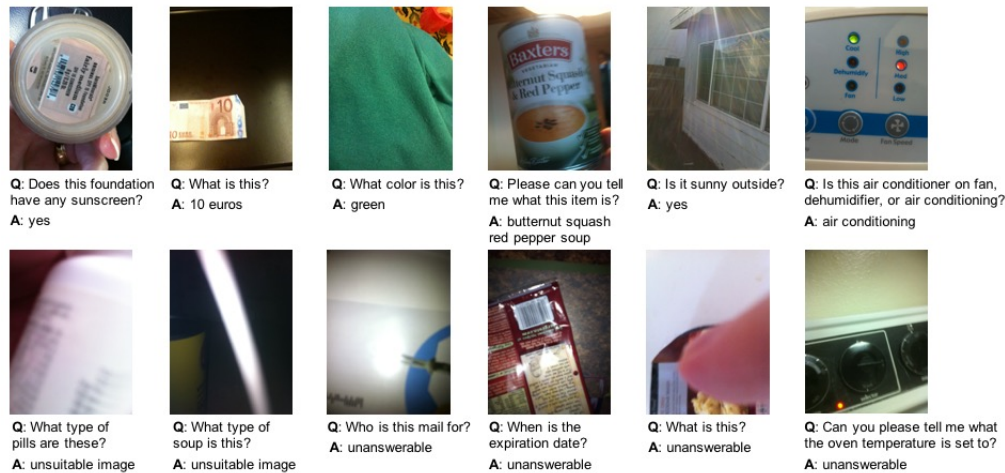
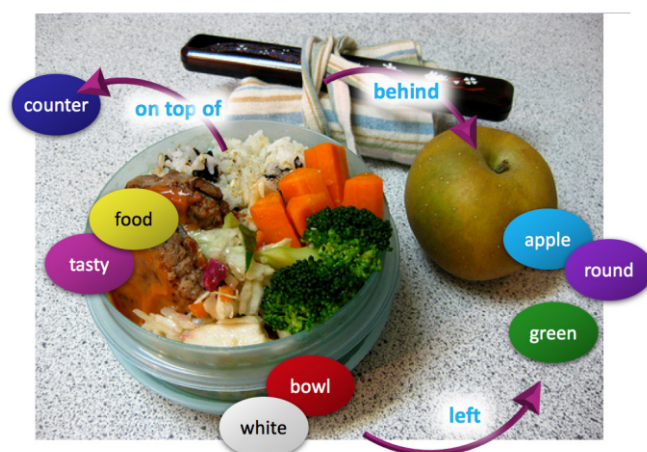


Figura 8 – Exemplos do VizWiz ilustrando os desafios apresentados por este *dataset*.

GQA: O *dataset* GQA (HUDSON; MANNING, 2019b) é composto por 22.669.678 perguntas e 113.018 imagens do cotidiano. Cada imagem pode ter várias perguntas associadas, como ilustrado na Figura 9. O GQA está vinculado a um grafo de cena que descreve objetos, atributos e suas relações na imagem, oferecendo uma versão mais refinada baseada no *Visual Genome*. Além das imagens e dos pares de perguntas e respostas, o GQA disponibiliza dois tipos de atributos visuais pré-extraídas para cada imagem: (i) atributos de grade convolucional com dimensões $7 \times 7 \times 2048$, obtidas por meio de uma rede ResNet-101 treinada no ImageNet; e (ii) atributos de detecção de objetos com dimensões $N_{\text{det}} \times 2048$, onde N_{det} representa o número de objetos detectados em cada imagem, limitado a um máximo de 100 por imagem, extraídas a partir de um detector Faster R-CNN. Adicionalmente, cada resposta é acompanhada por uma frase explicativa, fornecendo suporte para a identificação de possíveis erros. O GQA é notoriamente desafiador, pois muitas das suas perguntas requerem habilidades avançadas de raciocínio e compreensão espacial. Durante sua construção, foi feito um controle rigoroso sobre a distribuição das respostas para diferentes tipos de perguntas, visando minimizar possíveis vieses e garantir a robustez do *dataset*.



Is the **bowl** to the right of the **green apple**?
 What type of **fruit** in the image is **round**?
 What color is the **fruit** on the right side, red or **green**?
 Is there any **milk** in the **bowl** to the left of the **apple**?

Figura 9 – Exemplo do *dataset* GQA em que as perguntas exploram a cena contida na imagem.

VQA-RAD: VQA-RAD (LAU et al., 2018) é um *dataset* desenvolvido especificamente para a tarefa de VQA no contexto médico. Ele é composto por 315 imagens e 3.515 perguntas, abrangendo um total de 486 respostas distintas. O *dataset* é dividido em conjuntos de treinamento e teste, com cada imagem associada a múltiplas perguntas. As perguntas no VQA-RAD podem ser classificadas em dois tipos principais: perguntas fechadas, cujas respostas são restritas a opções como *yes/no* e outras alternativas limitadas, e perguntas abertas, que não têm uma estrutura rígida e podem ter múltiplas respostas corretas. Estas perguntas são divididas em 11 categorias: anormalidade, atributo, cor, contagem, modalidade, órgão, outro, plano, raciocínio posicional, objeto/condição presença e tamanho. Alguns exemplos de imagens radiológicas de diferentes regiões anatômicas presentes neste *dataset* são apresentados na Figura 10.

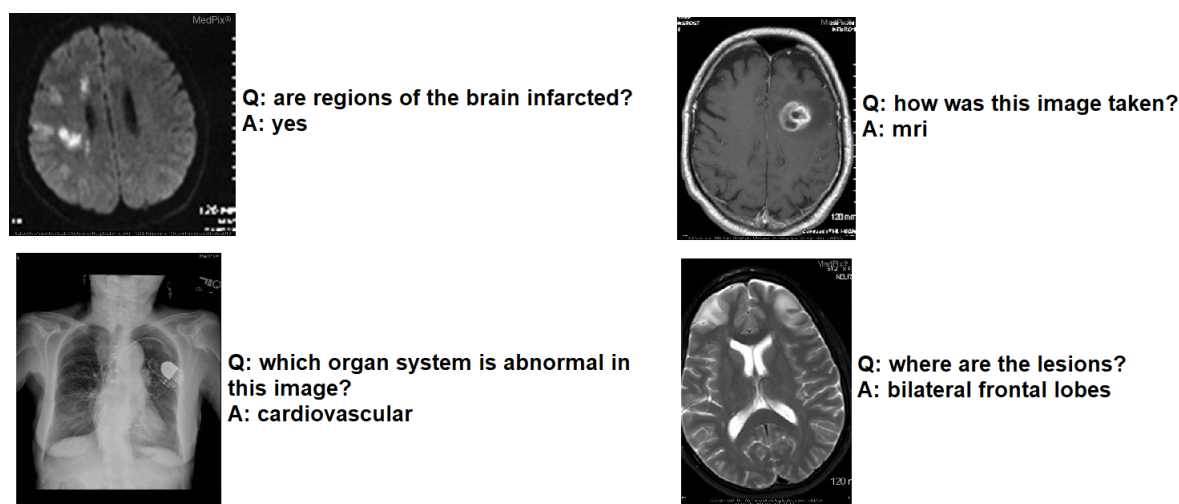
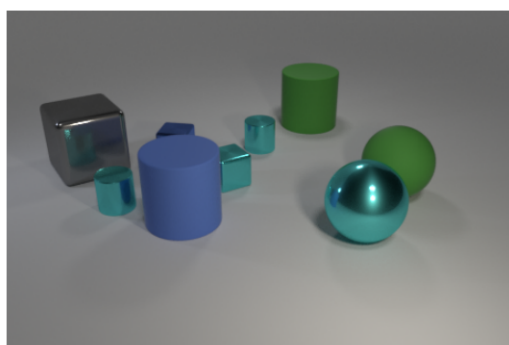


Figura 10 – Exemplos de imagens radiológicas presentes no VQA-RAD.

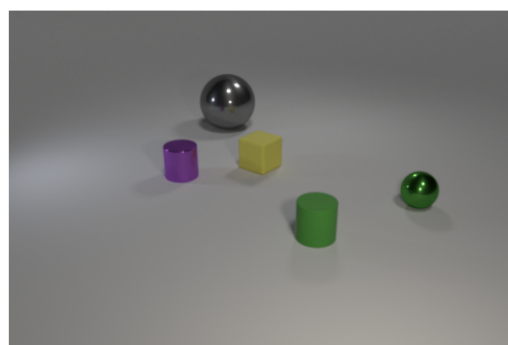
CLEVR: CLEVR (JOHNSON et al., 2017) é um *dataset* sintético criado para avaliar as capacidades de raciocínio visual abstrato de modelos, especialmente no que diz respeito a relações espaciais, cores, entre outras características. As imagens do CLEVR são geradas por renderizações fotorrealistas e apresentam objetos com diversas formas, cores, tamanhos e materiais, incluindo possíveis oclusões, conforme ilustrado na Figura 11.

No CLEVR o conjunto de treinamento possui 70.000 imagens e 699.989 perguntas, enquanto o conjunto de validação é constituído por 15.000 imagens e 149.991 perguntas. Já o conjunto de teste é composto por 15.000 imagens e 14.988 perguntas. As perguntas no CLEVR são caracterizadas por possuírem comprimento significativo, exigindo o tratamento de cadeias de inferências longas e complexas para se obter uma resposta. Isso torna o CLEVR um *benchmark* desafiador para a avaliação de habilidades de raciocínio visual em modelos de VQA.



Q: Are there fewer metallic objects that are on the left side of the large cube than cylinders to the left of the cyan shiny block?
A: yes

Q: There is a green rubber thing that is left of the rubber thing that is right of the rubber cylinder behind the gray shiny block; what is its size?
A: large



Q: Is the number of tiny metal objects left of the purple metallic cylinder the same as the number of small metal objects to the left of the small metal sphere?
A: no

Q: Do the large shiny object and the thing to the left of the big gray object have the same shape?
A: no

Figura 11 – Exemplos do *dataset* CLEVR cujas perguntas exploram a relação entre os objetos na imagem.

TextVQA: O *dataset* TextVQA (SINGH et al., 2019) é composto por imagens amostradas do *Open Image Dataset* (KUZNETSOVA et al., 2020), com foco em categorias propensas a conter texto, como *outdoor*, sinal de trânsito e quadro branco. Isso resulta em perguntas que demandam leitura e raciocínio sobre o texto presente na imagem, conforme pode ser visto na Figura 12.

No TextVQA versão v0.5.1, encontram-se 21.953 imagens e 34.602 perguntas no conjunto de treinamento, 3.166 imagens e 5.000 perguntas no conjunto de validação e 3.289 imagens e 5.734 perguntas no conjunto de teste. Cada pergunta possui 10 respostas anotadas por seres humanos, oferecendo uma ampla gama de respostas para uma avaliação mais robusta.

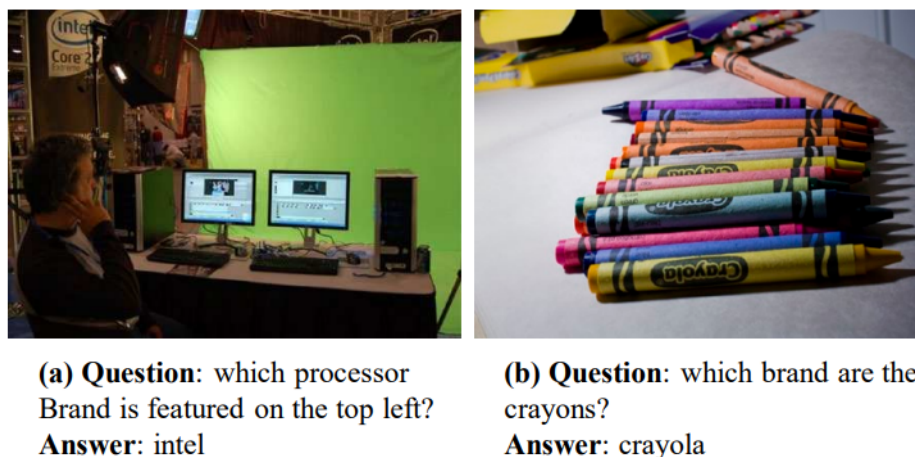


Figura 12 – Exemplos do *dataset* TextVQA, cujas respostas estão contidas em forma de texto nas imagens.

COCO-QA: O *dataset* COCO-QA (REN; KIROS; ZEMEL, 2015) foi gerado automaticamente a partir de legendas do *dataset* Microsoft COCO. Devido ao método automático de geração de perguntas, muitas apresentam formulações estranhas ou são gramaticalmente incorretas. COCO-QA abrange um total de 123.287 imagens, com 78.736 perguntas para treinamento e 38.948 perguntas para teste. As perguntas são categorizadas em uma das quatro classes: objeto (69.84%), número (7.47%), cor (16.59%) e localização (6.10%). Além disso, todas as respostas neste *dataset* são constituídas por uma única palavra, como pode ser visto no exemplo da Figura 13.

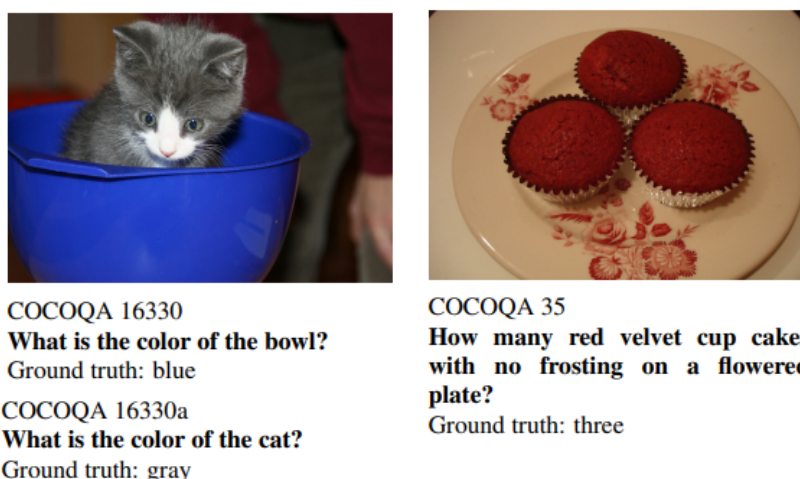


Figura 13 – Exemplos do *dataset* COCO-QA ilustrando que uma imagem pode estar associada a diferentes perguntas.

DAQUAR: O DAQUAR (*Dataset for Question Answering on Real World Images*), proposto por Malinowski e Fritz (2014), foi construído com imagens de cenas internas do *dataset* NYU-Depth v2 (COUPRIE et al., 2013), conforme ilustrado na Figura 14. Este *dataset* contém um total de 1.449 imagens, divididas em 795 para treinamento e 654 para

teste. Foram coletados 12.468 pares de perguntas/respostas, sendo 6.794 destinados ao treinamento e 5.674 aos testes.

Uma das principais limitações do DAQUAR é a restrição das respostas a um conjunto predefinido de 16 cores e 894 categorias de objetos. Além disso, o *dataset* apresenta fortes vieses, com tendência dos humanos a focar em objetos proeminentes, como mesas e cadeiras.



Q: How many white objects in this picture ?

A: 9



Q: What color is the chair in front of the wall on the left side of the stacked chairs ?

A: blue



Q: What is the largest white object on the left side of the picture ?

A: printer

Figura 14 – Exemplos do DAQUAR *dataset* explorando cenas internas de um imóvel.

Visual7W: O *dataset* Visual7W (ZHU et al., 2016) lida principalmente com sete tipos de perguntas, que originou seu nome: *What, Where, When, Who, Why, How, Which*. Este *dataset* é composto por 47.300 imagens provenientes do MS COCO e 327.939 perguntas. Cada pergunta é acompanhada por quatro respostas candidatas geradas por humanos, sendo apenas uma delas correta. Isso pode ser observado nos exemplos apresentados na Figura 15, em que a alternativa destacada em verde representa a resposta correta. Além disso, todos os objetos mencionados nas perguntas são visualmente fundamentados, com suas *bounding box* ¹ na imagem. O número de objetos anotados é 561.459 e o de categorias de objetos é 36.579.

¹ Uma *bounding box* (ou caixa delimitadora) é um retângulo que envolve um objeto em uma imagem. Ela é usada em visão computacional para identificar, localizar e delimitar visualmente objetos de interesse em tarefas como detecção de objetos ou segmentação. Seu formato é definido por coordenadas (geralmente dois pontos opostos, como os cantos superior esquerdo e inferior direito) ou pela posição do centro, largura e altura.

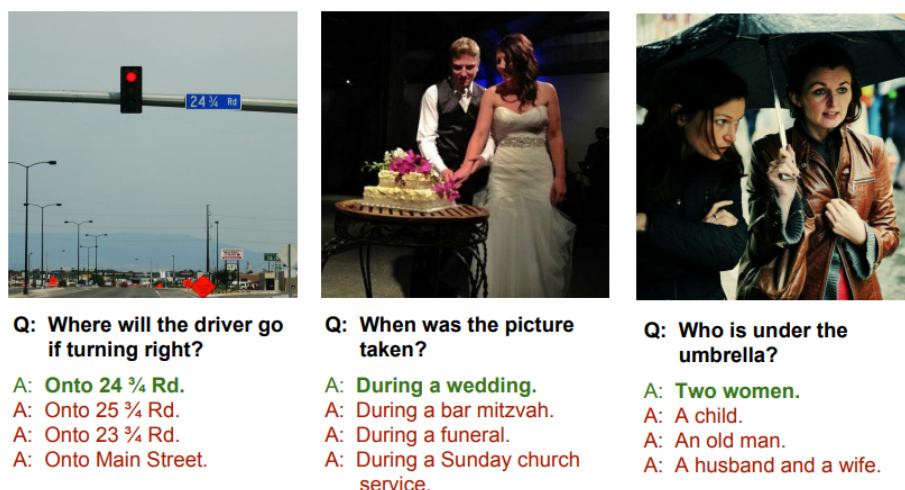


Figura 15 – Exemplos do *dataset* Visual7W, em que apenas uma das quatro respostas apresentadas é verdadeira.

Outside Knowledge Visual Question Answering (OK-VQA): O *dataset* OK-VQA (MARINO et al., 2019) foi projetado para tarefas de VQA que demandam a incorporação de conhecimento externo para gerar respostas. O *dataset* é composto por 14.055 perguntas abertas, abrangendo uma ampla variedade de categorias de conhecimento, conforme ilustrado na Figura 16. Cada pergunta está associada a cinco respostas corretas. Além disso, o *dataset* contém um total de 14.031 imagens. Durante a anotação, as perguntas foram criteriosamente filtradas para assegurar que todas exigissem conhecimento externo, como informações provenientes de fontes como a *Wikipedia*. Esforços adicionais foram realizados para minimizar o viés no *dataset*, reduzindo a quantidade de perguntas cujas respostas ocorrem com baixa frequência.

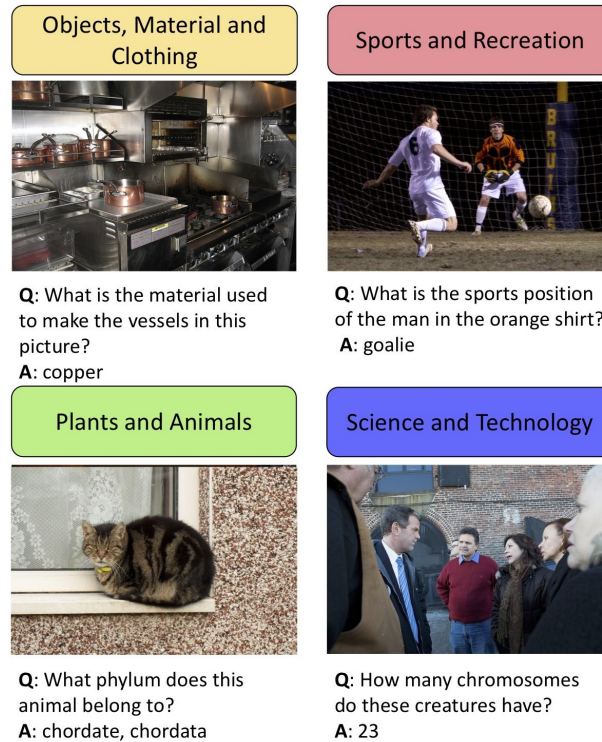


Figura 16 – Exemplos do *dataset* OK-VQA ilustrando a diversidade de assuntos que o *dataset* possui.

FVQA: O FVQA (WANG et al., 2017) é um *dataset* criado para lidar com perguntas que exigem informações externas (não visuais). Ele foi projetado para incluir anotações adicionais de forma a facilitar o treinamento supervisionado. Em contraste com a maioria dos *datasets* VQA que fornecem apenas pares de perguntas e respostas, o FVQA inclui, para cada pergunta e resposta, um fato relacionado a conceitos visuais, facilitando a integração de conhecimento externo no processo de resposta, como ilustrado na Figura 17. Esses fatos foram extraídos de bases de conhecimento como *DBpedia* (AUER et al., 2007), *ConceptNet* (SPEER; CHIN; HAVASI, 2017) e *Webchild* (TANDON; MELO; WEIKUM, 2017). O FVQA é composto por 2.190 imagens e 5.826 perguntas.



Question: What can the red object on the ground be used for ?

Answer: Firefighting

Support Fact: Fire hydrant can be used for fighting fires.

Figura 17 – Exemplo de imagem/pergunta com um fato de suporte do *dataset* FVQA.

A Tabela 1 resume o número de imagens, perguntas e a proporção entre as perguntas e respostas para cada um dos *datasets* apresentados nas seções anteriores. O VQA 2.0 se destaca como o maior *dataset* em termos de volume de imagens, enquanto o VQA-RAD é o menor. Já em relação à quantidade de perguntas, o GQA apresenta o maior número, contrastando com o VQA-RAD, que também possui o menor número de perguntas.

<i>Dataset</i>	Número de			Proporção perguntas/respostas		
	imagens (*)	perguntas (*)	respostas/pergunta	treinamento	validação	teste
VQA 1.0	254 K	764 K	10	0.4	0.2	0.4
VQA 2.0	1.1 M	1.1 M	10	-	-	-
VQA-CP v1	205 K	370 K	10	0.4	0.2	0.4
VQA-CP v2	219 K	658 K	10	-	-	-
VizWiz	32.8 K	32.8 K	10	0.6	0.1	0.3
GQA	113 K	22.6 M	1	0.7	0.1	0.1 (**)
VQA-RAD	315	3.5 K	-	-	-	-
CLEVR	100 K	864.9K	-	-	-	-
TextVQA	28.4 K	45.3 K	-	-	-	-
COCO-QA	123.2 K	117.6 K	-	-	-	-
DAQUAR	1.4 K	12.4 K	-	-	-	-
Visual7W	47.3 K	327.9 K	-	-	-	-
OK-VQA	14 K	14 K	-	-	-	-
FVQA	2.1 K	5.8 K	-	-	-	-

Tabela 1 – Tamanhos dos *datasets* presentes na literatura do VQA. (*) Valores aproximados. (**) 0.1 é utilizado no conjunto *challenge*.

A Tabela 2 apresenta a acurácia obtida em trabalhos da literatura com os principais *datasets* do VQA. Os resultados mostram que o CLEVR e o VQA 2.0 alcançam as maiores acurácias, destacando o bom desempenho dos modelos nesses *datasets*. Em contrapartida,

o DAQUAR e o VQA-CP v2 apresentam os menores valores de acurácia, sugerindo que esses *datasets* apresentam maiores desafios ou limitações para os modelos analisados.

Datasets	Ref/Acc	1º	2º	3º	4º	5º
VQA 1.0 (test-std)	Ref. Acc.	Ben-Younes et al. (2017) 67,36	Kim et al. (2016) 66,89	Fukui et al. (2016) 66,50	Lu et al. (2018) 66,09	Gao et al. (2018) 65,90
VQA 2.0 (test-std)	Ref. Acc.	Wang et al. (2023) 84,03	Li et al. (2022) 83,62	Wang et al. (2023) 82,52	Zeng et al. (2023) 81,80	Bao et al. (2022) 81,30
VQA-CP v2 (test)	Ref. Acc.	Chen et al. (2020) 58,95	Han et al. (2021) 57,32	Ensemble Gat et al. (2020) 56,74	Gat et al. (2020) 54,55	Clark, Yatskar e Zettlemoyer (2019) 52,05
VizWiz (test-std)	Ref. Acc.	Deuser et al. (2022) 60,15	Singh et al. (2019) 54,72	Huang et al. (2021) 53,40	Kim et al. (2018) 52,00	Gurari et al. (2018) 47,50
GQA (test-std)	Ref. Acc.	Zhao et al. (2021) 65,14	Hudson e Manning (2019a) 63,17	Kamath et al. (2021) 62,45	Tan e Bansal (2019) 60,30	Hu et al. (2019) 56,10
VQA-RAD	Ref. Acc.	He et al. (2024) 81,90	Zhang et al. (2023) 81,60	Li et al. (2023a) 79,20	Lin et al. (2023) 77,60	Li et al. (2023b) 76,80
CLEVR	Ref. Acc.	Yi et al. (2018) 99,80	Kamath et al. (2021) 99,70	Wang et al. (2021) 99,40	Mascharka et al. (2018) 99,10	Mao et al. (2018) 98,90
TextVQA (test-std)	Ref. Acc.	Chen et al. (2022) 73,10	Yang et al. (2021) 54,71	Wang et al. (2022) 53,69	-	-
COCO-QA	Ref. Acc.	Wu et al. (2016) 70,98	Wu et al. (2016) 69,73	Lu et al. (2016) 65,40	Kafle e Kanan (2016) 63,18	Yang et al. (2016) 61,60
DAQUAR	Ref. Acc.	Wu et al. (2016) 46,13	Wu et al. (2016) 45,79	Xi et al. (2020) 45,77	Yang et al. (2016) 45,50	Kafle e Kanan (2016) 45,17
Visual7W	Ref. Acc.	Hu et al. (2017) 72,53	Do et al. (2019) 72,30	Nguyen et al. (2022) 71,90	Fukui et al. (2016) 62,20	-
OK-VQA	Ref. Acc.	Hu et al. (2024) 66,80	Driess et al. (2023) 66,10	Chen et al. (2023) 66,10	Chen et al. (2022) 64,50	Shao et al. (2023) 62,50
FVQA (top-3)	Ref. Acc.	Yu et al. (2020) 91,20	Narasimhan, Lazebnik e Schwing (2018) 80,25	Narasimhan e Schwing (2018) 75,60	Li et al. (2019) 70,08	Wang et al. (2017) 64,65

Tabela 2 – Resultados de acurácia obtidos em diferentes trabalhos da literatura VQA, destacando as melhores performances.

2.1.2 Métricas

Nesta seção, são apresentadas algumas das métricas comumente utilizadas na literatura de VQA. A avaliação dos modelos baseiam-se geralmente nas anotações fornecidas pelos *datasets*, e pode variar desde métricas simples, como acurácia, até métricas mais sofisticadas, como BLEU, que consideram a qualidade da resposta em termos de similaridade com respostas humanas.

Acurácia: A métrica de acurácia simples é amplamente empregada na avaliação do desempenho dos modelos em cenários de múltipla escolha e perguntas abertas. Ela pode ser obtida por:

$$Acc = \frac{\text{N.º previsões corretas}}{\text{Total de previsões}} \quad (1)$$

No entanto, a acurácia simples pode ser excessivamente rigorosa, uma vez que trata todos os erros de maneira uniforme, independentemente de sua gravidade. Por exemplo, ao responder à pergunta *What animals are in the picture?*, um modelo indicar *dog* em vez do rótulo correto *dogs*, ele é penalizado da mesma maneira que se tivesse indicado *zebra*. Em perguntas com múltiplas respostas corretas, como *What's in the tree?*, onde *bald eagle* seria uma resposta aceitável, um modelo que responda *eagle* ou *bird* é penalizado da mesma forma que se tivesse respondido *yes*. Devido a essas limitações, diversas variações da métrica de acurácia foram propostas para a avaliação de sistemas de VQA (KAFLE; KANAN, 2017). Uma variação comumente utilizada é a acurácia por imagem (AGRAWAL et al., 2017), calculada conforme:

$$Acc(img) = \min \left(\frac{\# \text{ humans that said ans}}{3}, 1 \right) \quad (2)$$

na qual uma resposta é considerada correta (acurácia igual a 1) se a mesma resposta exata foi dada por pelo menos três anotadores humanos. Zero correspondências resultam em acurácia zero e, a partir daí, cada correspondência adiciona 0.33 pontos à acurácia, até um máximo de 1. A Figura 18 ilustra um exemplo do uso da acurácia por imagem, no qual, se a resposta predita pelo modelo foi *green apple*, a acurácia para esse exemplo seria 1, pois 7 anotadores forneceram exatamente *green apple* como resposta à pergunta apresentada, conforme demonstrado na Equação 3.

Q: 'What flavor of juice is this? Thank you.'

A:

green apple
gushing green apple
green apple
green apple
green apple
gushing green apple
green apple
gushing green apple
green apple
green apple

Figura 18 – Exemplo de QA do *dataset* VizWiz.

$$Acc(img) = \min\left(\frac{7}{3}, 1\right) = 1. \quad (3)$$

A acurácia geral (*overall accuracy*) do *dataset* é calculada como a média da acurácia por imagem para todas as amostras. No entanto, essa métrica apresenta alguns problemas. Ela pode, por exemplo, permitir múltiplas respostas corretas para uma mesma pergunta, mesmo que sejam respostas diretamente opostas. No COCO-VQA, mais de 13% das respostas de *yes/no* incluem tanto *yes* quanto *no* fornecidos por mais de três anotadores. Assim, tanto responder *yes* quanto *no* resultaria na pontuação máxima. Além disso, a coleta de respostas verdadeiras para cada pergunta é um processo caro e demorado.

WUPS: A métrica WUPS (*Wu-Palmer similarity*) (WU; PALMER, 1994) calcula a semelhança semântica entre duas palavras, variando de 0 a 1, onde 1 é a correspondência perfeita (KAFLE; KANAN, 2016). Embora essa métrica seja útil em muitos contextos, ela apresenta duas limitações significativas que dificultam seu uso. Primeiro, alguns pares de palavras podem ser lexicalmente muito semelhantes, mas carregam significados completamente distintos. Isso é especialmente problemático em perguntas sobre atributos, como cor. Por exemplo, se a resposta correta for *white* e a resposta fornecida for *black*, o WUPS ainda atribuiria uma alta pontuação de 0.91, apesar das respostas serem semanticamente opostas. A segunda limitação do WUPS é que ele não pode ser aplicado a respostas em formato de frases ou sentenças, que são frequentemente encontradas em alguns *datasets*. Isso restringe sua aplicabilidade em contextos em que as respostas precisam ser mais elaboradas ou contextualmente dependentes.

BLEU: A métrica *BiLingual Evaluation Understudy* (BLEU) é amplamente utilizada no contexto de tradução automática e se adapta bem a outros cenários de processamento

de linguagem natural, como o VQA. Seu objetivo é medir a qualidade das respostas geradas por modelos, avaliando a fluência, adequação e extensão em comparação a uma resposta de referência fornecida por humanos. O BLEU funciona através da comparação de *n-grams* (sequências de palavras) da resposta gerada com os *n-grams* da resposta de referência, contando as correspondências entre as duas. Por isso, a métrica é mais adequada para cenários em que a resposta tem mais de uma palavra, proporcionando uma avaliação detalhada da similaridade entre as respostas do modelo e as de referência.

Consistency: A métrica *Consistency* avalia a capacidade de um modelo produzir respostas que não sejam contraditórias. Em outras palavras, ao responder a múltiplas perguntas sobre uma mesma imagem, as respostas do modelo não devem ser conflitantes. Por exemplo, se o modelo responde *yes* para a pergunta *Is it the middle of summer?* e *winter* para *What season are we in?*, ele estaria se contradizendo (TASCON-MORALES; MÁRQUEZ-NEILA; SZNITMAN, 2023). Essa métrica assegura que o modelo mantenha coerência lógica em suas respostas.

Validity: A métrica *Validity* verifica se a resposta gerada pelo modelo é teoricamente válida. Para cada pergunta q , define-se um escopo de respostas válidas V , que inclui todas as respostas que poderiam estar corretas em algum contexto. Por exemplo, para perguntas relacionadas a cores, o conjunto V incluiria apenas nomes de cores; para perguntas do tipo *What animal ...*, apenas animais seriam respostas válidas, e para perguntas binárias, apenas *yes* ou *no* seriam aceitos. Se a previsão do modelo p pertencer ao conjunto V , atribui-se 1 ponto; caso contrário, 0. A pontuação é calculada para todas as perguntas no *dataset* (HUDSON; MANNING, 2019b).

Plausibility: A métrica *Plausibility* avalia se as respostas do modelo são plausíveis no mundo real. Para cada pergunta q sobre um atributo a de um objeto o , a resposta gerada p é verificada em relação aos dados do mundo real. Se a resposta ocorre pelo menos uma vez com o objeto o nos grafos de cena do *dataset*, atribui-se 1 ponto. Por exemplo, em uma pergunta sobre a cor de uma maçã, *green* e *red* seriam consideradas respostas plausíveis, enquanto *blue* ou *purple* seriam implausíveis, pois não ocorrem naturalmente com maçãs (HUDSON; MANNING, 2019b). Embora essa métrica tenha limitações com objetos raros, ela fornece uma boa ideia do nível de conhecimento de mundo que o modelo adquiriu.

Além das métricas mencionadas, também são utilizadas as métricas BLEU-4, como pode ser visto em Patro, Patel e Namboodiri (2020), Wu, Chen e Mooney (2020), Vedd et al. (2022); METEOR, usada em Jiang et al. (2020), Sharma e Jalal (2021a), Hu et al. (2023); ROUGE-L, utilizada por Ayyubi et al. (2020), Vaideeswaran et al. (2022), Chen et al. (2023); CIDEr, vista em Li et al. (2018), Shah et al. (2019), Hu et al. (2023); e SPICE, encontrada em Ganju, Russakovsky e Gupta (2017), Jin et al. (2022), Suo et al. (2023), dentre outras.

2.1.3 Relação de métricas e *datasets*

O Quadro 3 relaciona os *datasets* encontrados na literatura com as métricas que comumente são utilizadas para avaliar e comparar a performance das abordagens, sendo que a acurácia é a mais usada.

Métrica	Datasets suportados
Acurácia	DAQUAR, COCO-QA, FVQA, Visual7W, CLEVR, VizWiz, TextVQA, GQA, VQA 1.0, VQA 2.0, VQA-CP v1, VQA-CP v2, VQA-RAD, OK-VQA
WUPS	COCO-QA, DAQUAR
BLEU	VizWiz
Consistency	GQA
Validity	GQA
Plausibility	GQA
BLEU-4	VQA-X, VQA 2.0, MIMIC-Diff-VQA, VCR, A-OKVQA
METEOR	VQA-X, VQA 2.0, MIMIC-Diff-VQA, A-OKVQA
CIDEr	VQA-X, VQA 2.0, MIMIC-Diff-VQA, A-OKVQA
SPICE	VQA-X, A-OKVQA
ROUGE-L	VQA-X, VQA 2.0, MIMIC-Diff-VQA, VQA-E, VCR, A-OKVQA

Quadro 3 – Métricas comumente usadas nos *datasets*.

2.2 Pipeline Geral do VQA

Nesta seção, são apresentadas as principais estratégias utilizadas na literatura para a solução do problema de VQA. Geralmente, a resolução do problema é estruturada em três etapas principais: Extração de atributos (Seção 2.2.1), Fusão dos atributos (Seção 2.2.2) e Obtenção de Respostas (Seção 2.2.3).

2.2.1 Extração de atributos

Os atributos, ou *embeddings*, são representações numéricas, geralmente na forma de vetores, que descrevem tanto a imagem quanto a pergunta, como ilustrado na Figura 19. Para obtê-las, os dados são processados por um modelo apropriado ao tipo de entrada. Ao atingir a camada intermediária da rede, as saídas geradas nessa etapa correspondem aos atributos extraídos, representando as características mais relevantes da imagem e da pergunta.

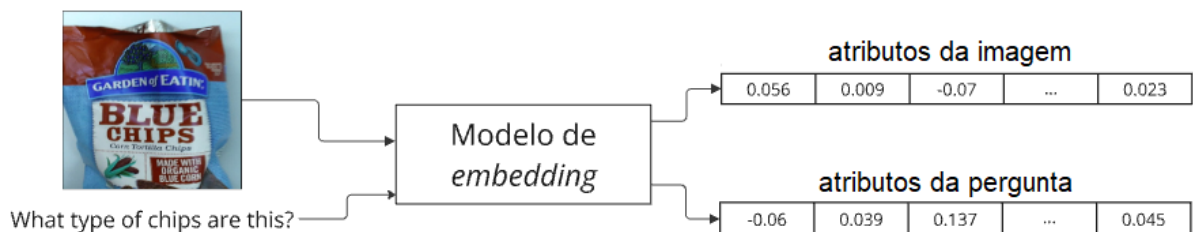


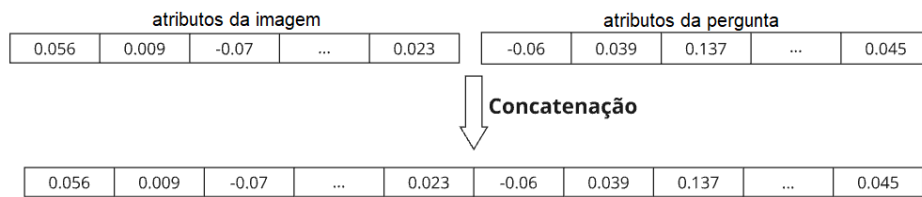
Figura 19 – Ilustração do processo de obtenção de atributos representativos para a imagem e pergunta.

No exemplo apresentado, os atributos obtidos para a imagem foram $[0.056, 0.009, \dots, 0.023]$, enquanto os atributos para a pergunta foram $[-0.06, 0.039, \dots, 0.045]$. A extração de atributos visuais para a solução de VQA é comumente realizada por CNNs (WANG; JI, 2018; JHA et al., 2019; MOHOLKAR, 2021). Além disso, modelos mais recentes, como ViT (BAZI et al., 2023) e CLIP (SONG et al., 2022), têm se destacado pela eficiência na captura de atributos visuais.

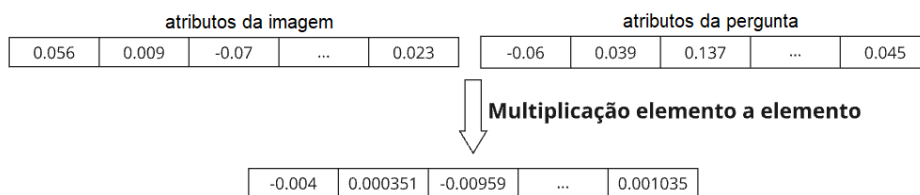
Por outro lado, para a extração de atributos das perguntas, diversos modelos de processamento de linguagem são utilizados. Exemplos incluem redes do tipo LSTM (SHI et al., 2018; TALAFHA; AL-AYYOUB, 2018; JHA et al., 2019), BERT (BAZI et al., 2023), GloVe (TALAFHA; AL-AYYOUB, 2018; HUANG et al., 2019; MA et al., 2021), Word2Vec (SHI et al., 2018; MA et al., 2021), entre outros.

2.2.2 Fusão dos atributos

Essa etapa é responsável pela integração dos atributos da imagem e da pergunta. Essa integração geralmente ocorre por meio de operações como concatenação ou multiplicação elemento a elemento, conforme representado na Figura 20.



(a) Concatenação de atributos.



(b) Multiplicação elemento a elemento de atributos.

Figura 20 – Exemplos de fusão de atributos.

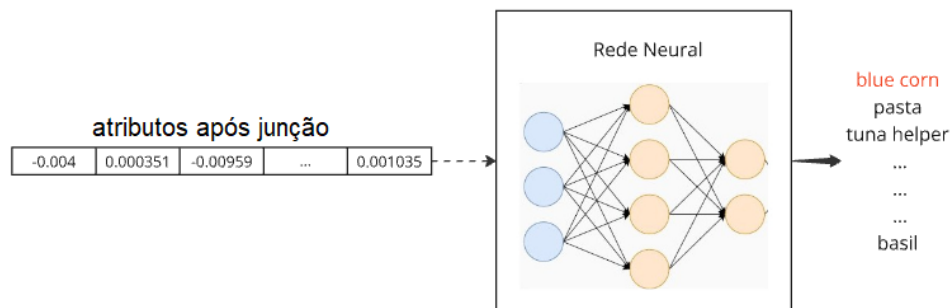
A concatenação consiste em combinar os atributos lado a lado, sem modificar seus valores individuais, como visto na Figura 20a. Nesse processo, ao concatenar um vetor de tamanho n com outro de tamanho m , obtém-se um vetor de tamanho $n + m$, sem a necessidade de que ambos tenham o mesmo comprimento.

Já na multiplicação elemento a elemento, exemplo da Figura 20b, cada valor de um conjunto de atributos é multiplicado diretamente pelo valor correspondente em outro conjunto, preservando suas posições. Nesse caso, é imprescindível que os atributos tenham

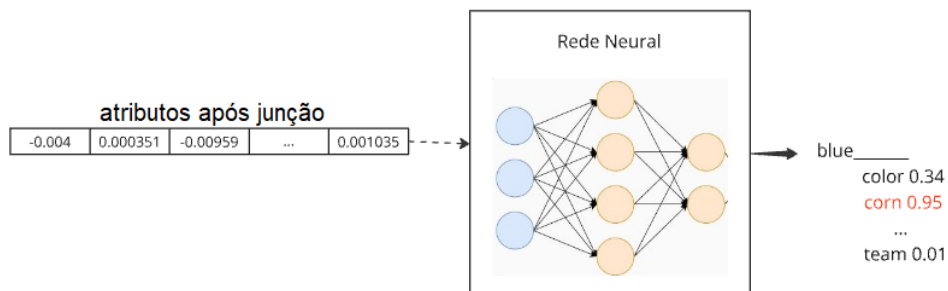
o mesmo tamanho. Além dessas técnicas, abordagens mais avançadas, como o uso de grafos e *bilinear pooling*, também são aplicadas.

2.2.3 Obtenção de respostas

A partir da combinação dos atributos extraídas na etapa anterior, o próximo passo consiste em definir uma arquitetura de rede neural capaz de fornecer a resposta à pergunta formulada. Esse problema pode ser abordado de duas formas principais: geração de texto e classificação, como ilustrado na Figura 21.



(a) Classificação.



(b) Geração de texto.

Figura 21 – Exemplos de obtenção de resposta.

Na abordagem de classificação, mostrada na Figura 21a, o modelo é treinado com dados rotulados e, durante a inferência, atribui um rótulo ao novo dado com base no conhecimento adquirido. Sua saída é um valor discreto, podendo ser binário, como *yes* ou *no*, ou composto por múltiplas classes.

Por outro lado, a abordagem de geração de texto é exemplificada na Figura 21b. Nesse caso, o modelo prevê a próxima palavra ou sequência de palavras com base nas anteriores e no contexto, gerando texto de maneira fluida. A saída é uma sequência contínua de palavras, gerada palavra por palavra, conforme as probabilidades aprendidas. Modelos como LLMs, BERT e LSTM são frequentemente utilizados para realizar essa tarefa.

A Figura 19 apresenta os dados de entrada e a pergunta formulada, *What type of chips are these?*, utilizada no exemplo de obtenção da resposta. Na abordagem de

classificação, representada na Figura 21a, as classes possíveis incluem opções como *pasta*, *tuna helper*, ..., *basil* e *blue corn*, sendo esta última identificada como a resposta correta. Já na abordagem de geração de texto, detalhada na Figura 21b, observa-se que a sequência *blue corn* é gerada como a resposta mais adequada, uma vez que a combinação de *corn* com a primeira palavra gerada, *blue*, apresentou o maior valor de probabilidade.

2.3 Inteligência Artificial

Nesta seção, são abordados os principais conceitos de modelos de Inteligência Artificial (IA) relevantes para esta dissertação. Inicialmente, é explorada a rede neural *Multi-layer Perceptron*. Em seguida, discute-se o BERT e sua contribuição para o processamento de linguagem natural, além do CLIP e BLIP2, modelos que integram visão e linguagem. Também são apresentados os *Autoencoders* e sua aplicabilidade na redução de dimensionalidade e representação de dados. Métodos baseados em grafos, como GNNs e GATs, são analisados por sua capacidade de modelar relações complexas. Por fim, são descritas as contribuições das arquiteturas ResNet-152 e *Vision Transformers* no campo da visão computacional.

2.3.1 MLP

Segundo Haykin (2001), o Perceptron Multicamadas (MLP, do inglês *Multilayer Perceptron*) é uma importante classe de redes neurais. A arquitetura de uma rede MLP consiste de uma camada de entrada, uma ou mais camadas ocultas e uma camada de saída, sendo que as camadas são formadas por um conjunto de neurônios, como pode ser visto na Figura 22. O exemplo ilustrado na Figura 22 é de uma rede totalmente conectada, ou seja, um neurônio em qualquer camada da rede está conectado a todos os neurônios da camada anterior. O fluxo de sinal através da rede progride para frente, da esquerda para a direita e de camada em camada.

A MLP tem sido aplicada com sucesso para resolver diversos problemas difíceis, utilizando um algoritmo de retropropagação de erro (*error back-propagation*) para seu treinamento supervisionado (HAYKIN, 2001). Basicamente, a aprendizagem por retropropagação de erro consiste de dois passos através das diferentes camadas da rede: um passo para frente, a propagação, e um passo para trás, a retropropagação. No passo para frente, um vetor de entrada é aplicado aos neurônios da rede e seu efeito se propaga através da rede, camada por camada. Finalmente, um conjunto de saídas é produzido como a resposta real da rede. Durante o passo de propagação, os pesos da rede são todos fixos. Durante o passo para trás, por outro lado, os pesos são todos ajustados de acordo com uma regra de correção de erro. Especificamente, a resposta real da rede é subtraída de uma resposta desejada (alvo) para produzir um sinal de erro. Este sinal de erro é então

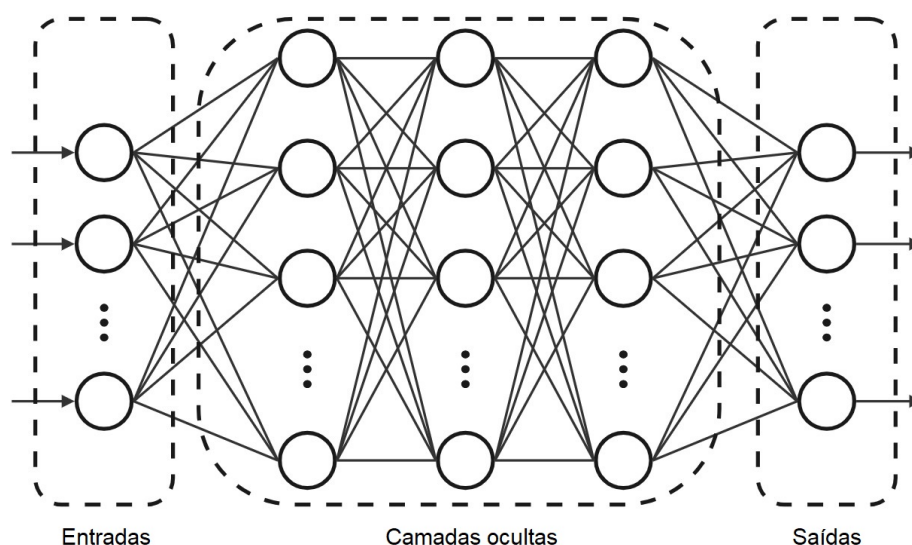


Figura 22 – Exemplo de arquitetura da MLP.

propagado para trás através da rede, contra a direção das conexões sinápticas, vindo daí o nome de retropropagação de erro. Os pesos são ajustados para fazer com que a resposta real da rede se mova para mais perto da resposta desejada, em um sentido estatístico.

Uma rede MLP tem três características distintivas ([HAYKIN, 2001](#)):

1. O modelo de cada neurônio da rede inclui uma função de ativação não-linear. Essa não-linearidade é suave, isto é, diferenciável em qualquer ponto. Algumas funções de ativação comumente utilizadas são: tangente hiperbólica, sigmóide e *Rectified Linear Unit* (ReLU).
2. A rede contém uma ou mais camadas de neurônios ocultos, que não são parte da entrada ou da saída da rede. Estes neurônios ocultos capacitam a rede a aprender tarefas complexas extraindo progressivamente as características mais significativas dos padrões (vetores) de entrada.
3. A rede exibe um alto grau de conectividade, determinado pelas sinapses da rede. Uma modificação na conectividade da rede requer uma mudança em seus pesos.

2.3.2 BERT

BERT, *Bidirectional Encoder Representations from Transformers*, é um modelo de linguagem baseado no *encoder* dos *Transformers*² ([VASWANI et al., 2017](#)). Sua principal capacidade está em aprender representações de contexto bidirecional, considerando tanto o contexto à esquerda quanto à direita de uma palavra como essenciais para a compreensão do seu significado. Assim, o modelo gera representações diferentes para a mesma palavra

² O *Transformer* é uma arquitetura de redes neurais baseada no mecanismo de atenção, permitindo que o modelo foque em diferentes partes da entrada simultaneamente. Isso melhora a eficiência e permite treinamento paralelo, sendo altamente eficaz em tarefas de processamento de linguagem natural.

dependendo do contexto, como no caso de banco (lugar para sentar) e banco (instituição financeira) (KENTON; TOUTANOVA, 2019).

O BERT foi treinado em um grande corpus, *Wikipedia (2.5B) + BooksCorpus (0.8B)*, utilizando duas tarefas simultâneas: *Masked Language Modeling* (MLM) e *Next Sentence Prediction* (NSP) (KENTON; TOUTANOVA, 2019). A tarefa de *Masked Language Modeling* (MLM) consiste em mascarar aleatoriamente 15% das palavras de entrada e treinar o modelo para prever essas palavras ocultas. Por exemplo, na frase *my dog is hairy*, se o quarto token (correspondente a *hairy*) for selecionado, o mascaramento ocorre de três formas: 80% das vezes, a palavra é substituída pelo token [MASK], resultando em *my dog is [MASK]*; 10% das vezes, é substituída por uma palavra aleatória, como *my dog is apple*; e, nos 10% restantes, a palavra é mantida inalterada, como *my dog is hairy*. Esse processo força o modelo a aprender representações contextualizadas para melhorar a precisão nas predições.

Já a tarefa de *Next Sentence Prediction* (NSP) foi introduzida para capturar a relação entre sentenças consecutivas, uma habilidade essencial em tarefas como *Question Answering* (QA) e *Natural Language Inference* (NLI). Durante o pré-treinamento, o modelo recebe duas sentenças A e B. Em 50% dos casos, B é a próxima sentença real que segue A no corpus, sendo rotulada como *IsNext*. Nos outros 50%, B é uma sentença aleatória extraída do corpus, recebendo o rótulo *NotNext*. Essa abordagem permite ao BERT compreender conexões semânticas e dependências entre sentenças, indo além da análise isolada de palavras ou frases.

O modelo é composto por uma pilha de 12 *encoders*, que, embora idênticos em estrutura, não compartilham pesos entre si (KENTON; TOUTANOVA, 2019). A primeira etapa em cada *encoder* é a representação das palavras por meio de *embeddings*, conforme ilustrado na Figura 23. O *embedding* ocorre apenas no *encoder* inferior. Cada *encoder* recebe uma lista de vetores, cada um com tamanho máximo 512. No *encoder* inferior, essa lista consiste dos *embeddings* das palavras, enquanto nos demais *encoders*, consiste das saídas do *encoder* imediatamente inferior. O tamanho dessa lista é um hiperparâmetro que pode ser ajustado, representando o comprimento da sentença mais longa no conjunto de treinamento.

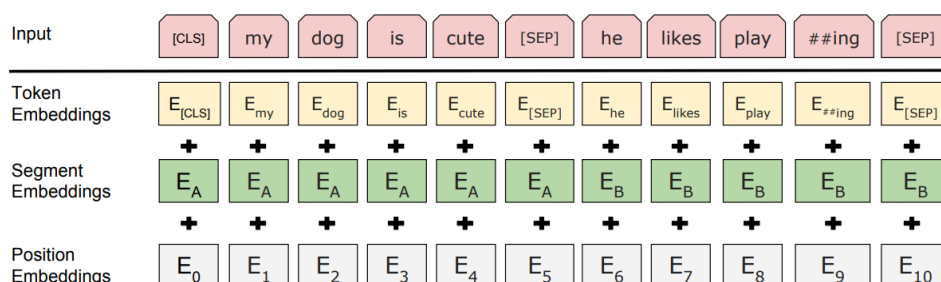


Figura 23 – Transformação da entrada do BERT em *embedding*.

Para contabilizar a ordem das palavras na sequência de entrada, o *Transformer* adiciona um vetor posicional a cada *embedding* de entrada. Esses vetores seguem um padrão específico aprendido pelo modelo, auxiliando-o a determinar a posição de cada palavra ou a distância entre palavras na sequência (VASWANI et al., 2017). Após gerar os *embeddings* das palavras, cada vetor passa pelas duas subcamadas do *encoder*: *self-attention* e uma rede neural *feed-forward*. A camada de *self-attention* permite que o *encoder* considere outras palavras na frase de entrada ao codificar uma palavra específica. As saídas da camada de *self-attention* são então processadas por uma rede neural *feed-forward*, aplicada de forma independente a cada posição (KENTON; TOUTANOVA, 2019).

2.3.3 CLIP

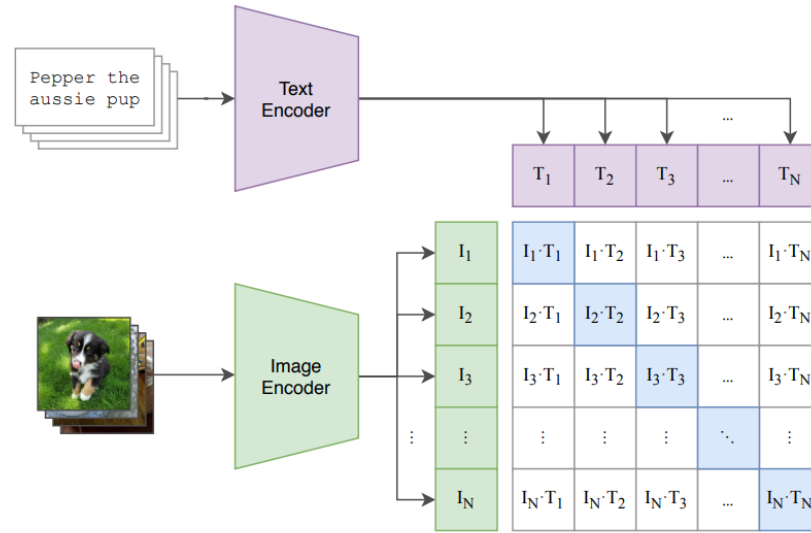
O CLIP (*Contrastive Language-Image Pretraining*) é um modelo de aprendizado profundo desenvolvido pela OpenAI em 2021, projetado para lidar com tarefas de aprendizado multimodal (RADFORD et al., 2021). Treinado de forma auto supervisionada com 400 milhões de pares de imagens e textos, o CLIP tem como principal objetivo alinhar representações visuais e textuais em um espaço de *embeddings* compartilhado. O modelo é composto por dois codificadores principais: um codificador de texto, que utiliza a arquitetura *Transformer*, e um codificador de imagens, no qual foram experimentados as arquiteturas ResNet-50 e *Vision Transformer* (ViT) (RADFORD et al., 2021).

A Figura 24 ilustra as principais etapas do funcionamento do CLIP, que serão detalhadas a seguir:

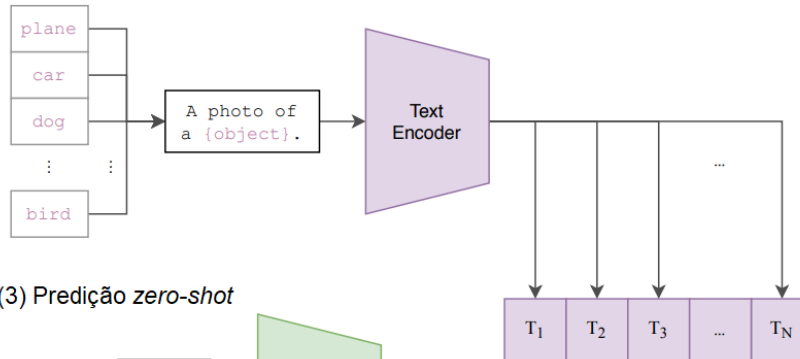
1. Pré-treinamento contrastivo: O modelo CLIP utiliza dois *encoders*, um para texto e outro para imagens, que aprendem a mapear representações textuais e visuais para um espaço compartilhado. Durante o pré-treinamento, pares de texto e imagem, como *Pepper the aussie pup* e a imagem correspondente de um filhote de cachorro, são processados pelos *encoders* de texto e imagem, respectivamente. Os vetores resultantes $T_1, T_2, \dots$ para texto e $I_1, I_2, \dots$ para imagens são combinados para calcular similaridades entre todas as combinações possíveis de pares texto-imagem. A otimização é realizada de forma a maximizar a similaridade dos pares corretos enquanto minimiza a similaridade de combinações incorretas (RADFORD et al., 2021).
2. Criação de classificadores a partir de descrições textuais: Para uma tarefa de classificação, o modelo gera descrições textuais padronizadas para cada classe, como *A photo of an object*, sendo *object* substituído pelo nome da classe (por exemplo, *dog*, *plane*, etc.). Essas descrições são processadas pelo *encoder* do texto, gerando vetores $T_1, T_2, \dots, T_n$ que representam cada classe no espaço multimodal (RADFORD et al., 2021).
3. Predição *zero-shot*: Na etapa de inferência, uma imagem de entrada é processada pelo *encoder* da imagem, gerando um vetor de características I_1 . Esse vetor é

comparado aos vetores das descrições textuais gerados na etapa anterior $I_1 \cdot T_1, I_1 \cdot T_2, \dots, I_1 \cdot T_n$. A classe com maior similaridade é selecionada como a predição do modelo. Esse processo permite que o CLIP realize tarefas de classificação sem necessidade de ajuste fino específico para um *dataset* (RADFORD et al., 2021).

(1) Pré-treinamento contrastivo



(2) Criação de classificadores a partir de descrições textuais



(3) Predição zero-shot

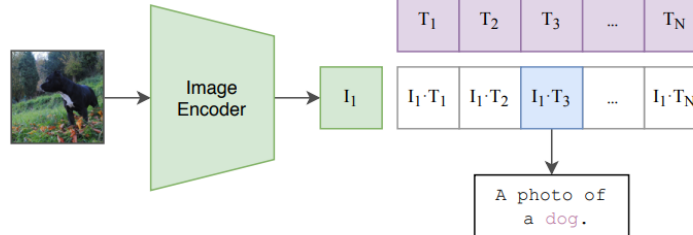


Figura 24 – Estrutura do CLIP. Adaptado de Radford et al. (2021).

2.3.4 BLIP2

BLIP2 (*Bootstrapping Language-Image Pre-training 2*) é uma arquitetura composta por três componentes principais: um codificador de imagem congelado, uma *Large Language Model* (LLM) congelada e um *Querying Transformer* (Q-Former), conforme descrito por Li et al. (2023) e ilustrado na Figura 25.

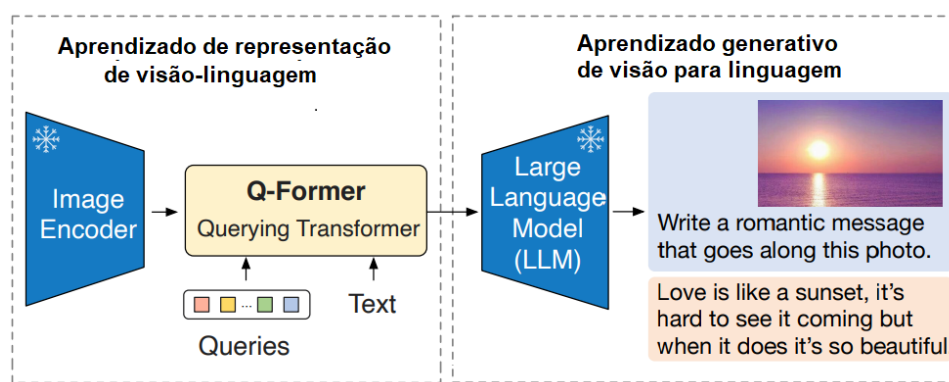


Figura 25 – Módulos do BLIP2. Adaptado de Li et al. (2023).

O codificador de imagem congelado é responsável por processar a entrada visual e extrair uma representação numérica correspondente. Este componente, pré-treinado previamente, mantém seus pesos inalterados durante todo o treinamento do modelo BLIP2 (LI et al., 2023). Da mesma forma, a LLM é utilizada sem ajuste de parâmetros, garantindo estabilidade e aproveitando seu pré-treinamento robusto.

Entre esses dois módulos, encontra-se o Q-Former, descrito como um *transformer* leve que conecta o codificador de imagem à LLM. Este componente utiliza vetores de consulta, que são aprendidos ao longo do treinamento, para selecionar as representações visuais mais relevantes para o contexto textual fornecido (LI et al., 2023).

O treinamento do BLIP2 ocorre em dois estágios principais. No primeiro estágio, denominado aprendizado de representação de visão-linguagem, o Q-Former é ajustado para alinhar as representações visuais do codificador de imagem com as instruções textuais. Esse processo visa maximizar a informação mútua entre as modalidades de imagem e texto. Por exemplo, se uma imagem retrata um cachorro brincando em um parque e a instrução textual é *Describe the action in the image*, o modelo ajusta suas representações para que a saída textual reflita com precisão o conteúdo visual (LI et al., 2023).

No segundo estágio, denominado aprendizado generativo de visão para linguagem, o foco está em conectar a saída do Q-Former à LLM. Nesse processo, o Q-Former é treinado para produzir representações visuais que a LLM possa interpretar e transformar em respostas textuais adequadas. Seguindo o exemplo anterior, o modelo seria ajustado para gerar uma descrição como *A dog is playing in the park*, demonstrando sua capacidade de transformar informações visuais em texto (LI et al., 2023).

2.3.5 Autoencoder

Um *Autoencoder* é uma rede neural totalmente conectada, porém com características específicas em sua arquitetura. Como descrito por Bank, Koenigstein e Giryas (2023) e visto na Figura 44, sua estrutura é composta por três partes principais: *Encoder*

(Codificador), *Latent Space* (Espaço Latente) e *Decoder* (Decodificador).

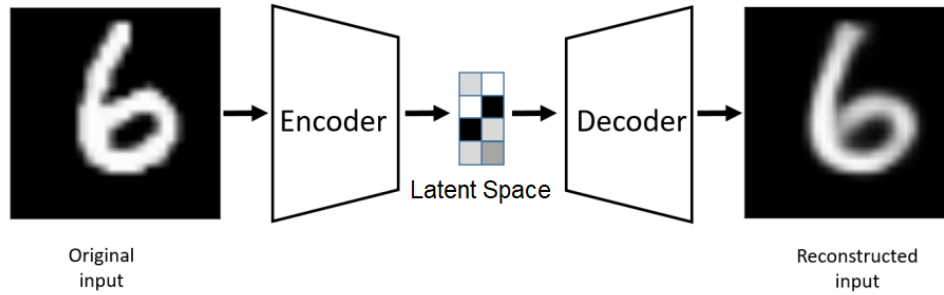


Figura 26 – Estrutura do *Autoencoder*. Adaptado de [Bank, Koenigstein e Giryas \(2023\)](#).

O *Encoder* é a parte inicial do *Autoencoder*, responsável por receber os dados de entrada e sintetizar as informações principais. Essa etapa transforma os dados em um espaço latente de menor dimensionalidade ([BANK; KOENIGSTEIN; GIRYES, 2023](#)). O *Latent Space*, por sua vez, representa uma versão compacta e essencial dos dados originais, reduzidos a uma forma que preserva as características mais relevantes.

Por fim, o *Decoder* reverte esse processo, reconstruindo os dados originais a partir do espaço latente. Esse fluxo de redução e reconstrução é fundamental para o funcionamento do *Autoencoder* ([BANK; KOENIGSTEIN; GIRYES, 2023](#)).

2.3.6 Grafos

Um grafo é uma estrutura $G = (V, E)$, onde V é um conjunto finito de vértices e $E \subseteq V \times V$, um conjunto finito de arcos, caso o grafo seja orientado, ou arestas, caso contrário. De acordo com [Ventura \(2009\)](#), os grafos podem ser utilizados para representar graficamente as relações entre elementos de dados. Cada aresta $e \in E$ é denotada pelo par de vértices $e = \{v, w\}$ que a formam. Os vértices v e w , chamados de extremos da aresta, são considerados adjacentes ou vizinhos. Além disso, a aresta e é dita incidente aos vértices v e w . A Figura 27 exemplifica um grafo com $V = \{u, v, w, x, y\}$ e $E = \{uv, uw, ux, vx, vw, wx, xy\}$.

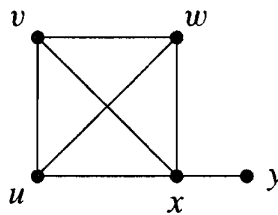


Figura 27 – Exemplo de grafo ([WEST et al., 2001](#)).

Vértices e arestas podem conter informações associadas. Quando essas informações consistem em rótulos ou pesos simples, como números ou letras, o grafo é chamado

de grafo rotulado (VENTURA, 2009). Por outro lado, se vértices e arestas apresentam informações adicionais, o grafo é denominado grafo de atributos, e esses atributos podem ser associados tanto aos vértices quanto às arestas, ou ainda a ambos.

Os grafos também podem ser representados computacionalmente por uma matriz de adjacência. Para um grafo G com $|V|$ vértices, utiliza-se uma matriz M de dimensão $|V| \times |V|$. Como destacado por Ventura (2009), em um grafo não direcionado, simples e sem pesos, a entrada (i, j) da matriz contém o valor 1 se v_i e v_j forem adjacentes, e 0 caso contrário. Já em grafos com arestas ponderadas, a entrada (i, j) contém o peso da aresta $\{i, j\}$. Em grafos simples e não direcionados, as matrizes de adjacência apresentam propriedades específicas: são simétricas e possuem diagonal nula. Por fim, os grafos desempenham um papel fundamental na representação de conhecimento e na solução de problemas do mundo real em diversas áreas (VENTURA, 2009).

2.3.7 GNNs e GAT

Redes neurais para grafos (GNNs, do inglês *Graph Neural Networks*) formam uma classe poderosa de redes neurais capazes de operar em dados estruturados em grafos, aprendendo representações de vértices (*embeddings*) por meio da agregação de informações da vizinhança local de cada vértice. Esse conceito é amplamente referenciado como *message passing* na literatura de aprendizado de representação de grafos (SCARSELLI et al., 2008; GILMER et al., 2017).

Segundo Hamilton (2020), a intuição fundamental por trás do *message passing* é bastante simples: a cada iteração, cada vértice acumula informações de sua vizinhança local. À medida que as iterações avançam, o *embedding* de cada vértice passa a conter informações provenientes de outros vértices do grafo, integrando informações cada vez mais profundas e estruturais sobre o grafo. Para ser mais preciso, após a primeira iteração ($k = 1$), o *embedding* de cada vértice contém informações sobre sua vizinhança imediata, ou seja, os atributos de seus vizinhos de grafo que podem ser alcançados por um caminho de comprimento 1. Após a segunda iteração ($k = 2$), cada vértice contém informações de sua vizinhança a partir de um caminho de comprimento 2, e assim sucessivamente.

As funções de agregação usadas no *message passing* podem ser de 3 tipos (RAHMAN, 2020): MAX, AVERAGE e SUM. A MAX agrega o valor máximo dos vizinhos, a AVERAGE soma as contribuições dos vizinhos e agrega esse valor dividindo pelo número de vizinhos, enquanto que a SUM soma as contribuições dos vizinhos.

No exemplo ilustrado na Figura 28, tem-se um grafo com quatro vértices, A, B, C, D , onde cada vértice possui um conjunto próprio de atributos. O vértice A é o único com um *loop*, significando que ele está conectado a si mesmo. Aplicando uma iteração de *message passing* e fazendo a agregação por média, obtemos para o vértice A o *embedding*

$[2.66, 1, 2, 2]$, ou seja, sua nova representação é resultado da média dos atributos de A , B e C .

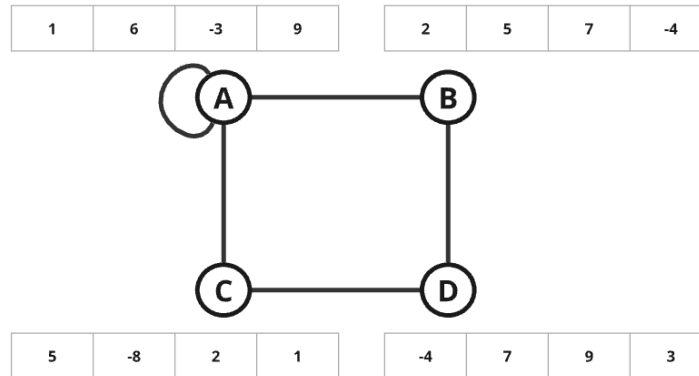


Figura 28 – Exemplo de grafo cujos vértices possuem atributos.

Algumas variantes importantes de GNNs incluem *GraphSAGE* (HAMILTON; YING; LESKOVEC, 2017), *Graph Convolutional Networks* (GCN) (KIPF; WELLING, 2016) e *Graph Attention Networks* (GAT) (VELICKOVIC et al., 2018). A rede GAT, utilizada no desenvolvimento deste trabalho, representa uma variante específica de GNNs projetada para aprimorar o *message passing* através da introdução de um mecanismo de atenção treinável. Esse mecanismo permite que um vértice identifique os vizinhos mais relevantes ao agregar mensagens, atribuindo pesos diferentes para cada par vértice-vizinho (VELICKOVIC et al., 2018). Assim, os coeficientes de atenção indicam a importância dos atributos dos vizinhos para a atualização dos atributos daquele vértice, conferindo um certo nível de interpretabilidade.

2.3.8 ResNet-152

A ResNet-152, um dos modelos da família *Residual Networks* (ResNets), é uma rede neural com 152 camadas, desenvolvida para solucionar o problema de degradação do desempenho em redes muito profundas. Esse problema ocorre devido à dificuldade de aprendizado e à tendência de redes mais profundas sofrerem com gradientes que desaparecem ou explodem, dificultando a convergência dos modelos. A introdução dos blocos residuais, proposta por He et al. (2016), foi a inovação central dessa arquitetura, permitindo o treinamento eficiente de redes significativamente profundas.

Diferentemente das redes tradicionais, onde as camadas sucessivas aprendem diretamente o mapeamento desejado, a ResNet utiliza um mapeamento residual. Em vez de aprender a função $H(x)$, que corresponde ao mapeamento esperado, a rede aprende uma função residual $F(x) = H(x) - x$, onde x é a entrada. Essa abordagem parte do princípio de que é mais fácil aprender a diferença (resíduo) do que o mapeamento completo, especialmente em redes profundas (HE et al., 2016).

Os blocos residuais introduzem conexões de atalho (*skip connections*), que essencialmente ignoram uma ou mais camadas intermediárias, adicionando diretamente a entrada x à saída do bloco. Essas conexões preservam a identidade da entrada ao longo das camadas, facilitando o fluxo de gradientes e reduzindo o problema do desaparecimento de gradientes durante o treinamento (HE et al., 2016).

Com suas 152 camadas treináveis, a ResNet-152 é estruturada em blocos residuais compostos por camadas convolucionais, funções de ativação como ReLU, e operações de normalização e *pooling*, conforme ilustrado na Figura 29. A profundidade adicional aumenta a capacidade de aprendizado da rede, permitindo a captura de padrões complexos e representações mais ricas. Além disso, essa arquitetura evita que a precisão sature ou decaia à medida que a rede se aprofunda, um desafio comum em redes profundas.

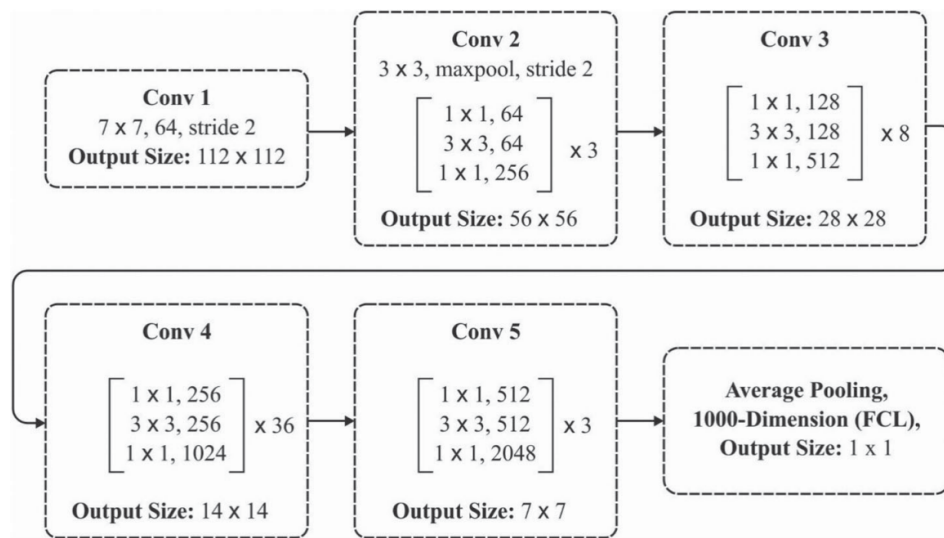


Figura 29 – Arquitetura do ResNet-152 (PUSTOKHIN et al., 2023).

A ResNet-152 rapidamente se destacou em tarefas de classificação de imagens, vencendo o *ImageNet Large Scale Visual Recognition Challenge* (ILSVRC) em 2015 com uma taxa de erro significativamente menor do que arquiteturas anteriores (HE et al., 2016). Desde então, sua arquitetura tornou-se uma referência em *benchmarks* de visão computacional e tem sido amplamente utilizada em aplicações diversas. Exemplos incluem a extração de atributos em tarefas como o VQA, onde a ResNet-152 é empregada para capturar características visuais detalhadas das imagens.

2.3.9 ViT

O *Vision Transformer* (ViT) é uma arquitetura de rede neural projetada para o processamento de imagens, inspirada nos modelos *Transformer*. Introduzido por Dosovitskiy (2020), o ViT revolucionou a análise de imagens ao substituir convoluções tradicionais por blocos de *Transformer*, que utilizam o mecanismo de atenção para explorar relações entre

diferentes regiões de uma imagem. Essa abordagem mostrou-se altamente competitiva, especialmente em *datasets* de grande escala, como o *ImageNet-21k*, superando ou igualando o desempenho das redes convolucionais tradicionais.

A arquitetura do ViT começa dividindo a imagem de entrada em pequenos blocos, chamados *patches*, como ilustrado na Figura 30. Cada *patch* é linearizado em um vetor e projetado em uma dimensão especificada por um *embedding* linear. Esse processo transforma a imagem original em uma sequência de *embeddings*, que serve como entrada para o bloco de *Transformer* (DOSOVITSKIY, 2020). Diferentemente das CNNs, que aprendem representações locais por meio de convoluções, o ViT aplica o mecanismo de atenção em toda a sequência de *patches*, permitindo capturar dependências de longo alcance entre diferentes partes da imagem.

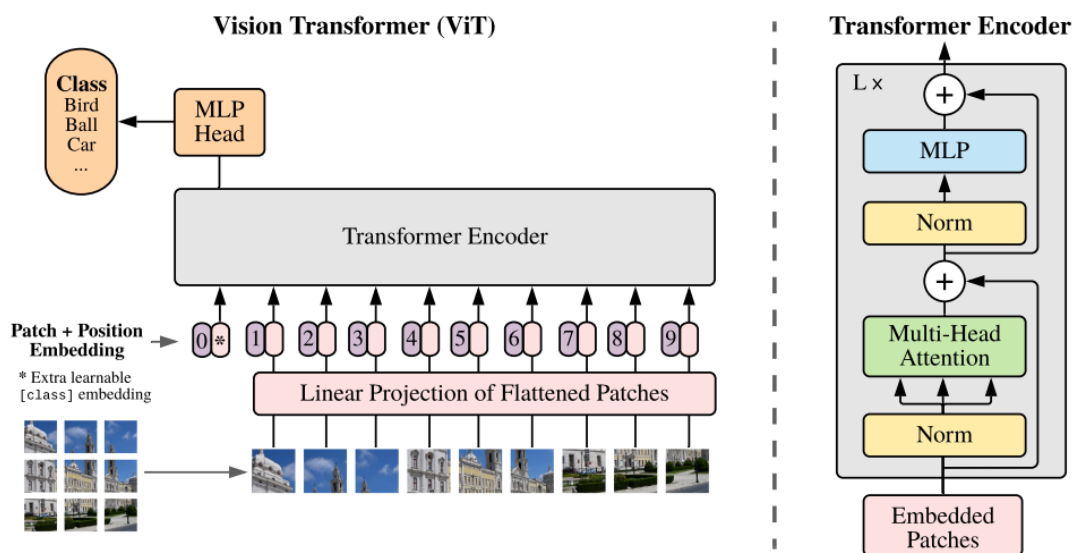


Figura 30 – Arquitetura do ViT.

O ViT também possui um *token* de classificação, inspirado no BERT, que aprende uma representação global da imagem. Esse *token* é utilizado para realizar previsões, como a classificação da imagem, ao final do modelo. Além disso, *embeddings* de posição são adicionados aos *embeddings* dos *patches*, preservando a ordem espacial das informações, essencial para a interpretação de estruturas visuais.

A principal inovação do ViT está na utilização do mecanismo de atenção auto-regressiva, ou *self-attention*, que permite capturar relações não locais entre diferentes regiões da imagem (DOSOVITSKIY, 2020). Ao contrário das CNNs, onde o campo receptivo cresce de maneira fixa, o ViT tem a capacidade de considerar toda a imagem desde o início, alocando recursos computacionais de forma dinâmica para áreas de maior interesse. Essa flexibilidade torna o ViT altamente eficiente em tarefas que envolvem imagens de alta complexidade e padrões de longa distância.

Quando aplicado a *datasets* grandes, o ViT frequentemente supera as CNNs em

tarefas de classificação, devido à sua capacidade de modelar relações globais de forma eficaz. No entanto, em *datasets* menores, sua performance é limitada, pois o modelo carece de um viés indutivo local e depende de grandes volumes de dados para capturar relações globais entre os *patches*. Como apontado por [Dosovitskiy \(2020\)](#), essa limitação faz com que as CNNs permaneçam superiores em cenários com dados mais escassos.

Além disso, o ViT apresenta desafios relacionados ao alto poder computacional necessário para o treinamento, especialmente em arquiteturas mais profundas ou com maior número de *patches*. A ausência de convoluções fixas exige que o modelo aprenda a estrutura espacial das imagens a partir do zero, o que aumenta a complexidade computacional e a demanda por *hardware* robusto.

Desde sua introdução, o ViT trouxe uma nova perspectiva para a visão computacional e abriu espaço para modelos baseados em *Transformers* que ultrapassam as barreiras das CNNs em termos de precisão e escalabilidade. Sua aplicação em tarefas como detecção de objetos, segmentação e análise de vídeos demonstra a versatilidade e o potencial do mecanismo de atenção em diferentes domínios.

3 Revisão da Literatura

Neste capítulo, apresenta-se uma visão geral sobre a literatura existente sobre o VQA, organizada cronologicamente. Aborda-se desde os estudos iniciais até as pesquisas mais recentes, destacando as principais contribuições ao longo do tempo.

3.1 Trabalhos Relacionados

Desde o lançamento do primeiro *dataset* de VQA em 2014, essa área de pesquisa tem experimentado avanços significativos, impulsionados tanto pela criação de novos *data-sets* quanto pelo desenvolvimento de novos algoritmos (KAFLE; KANAN, 2017). Inicialmente, destaca-se o trabalho de Agrawal et al. (2017), que introduziu o *dataset* VQA 1.0. Para resolver o problema no *dataset* VQA 1.0, Agrawal et al. (2017) empregaram uma rede LSTM (*Long Short Term Memory*) de duas camadas para representar as perguntas e a camada oculta final da rede VGGNet (*Visual Geometry Group Net*) para capturar os atributos das imagens. A fusão das representações ocorreu por meio de uma multiplicação elemento a elemento, seguida por uma camada totalmente conectada culminando na obtenção das respostas.

Com a introdução do mecanismo de atenção (VASWANI et al., 2017), que permite aos modelos focarem em partes específicas das entradas e saídas, o conceito foi amplamente explorado, tanto no processamento de linguagem natural quanto na visão computacional. Um exemplo disso é o trabalho de Chen et al. (2015), que foi inspirado na tendência humana de direcionar a atenção para regiões informativas relevantes antes de formular respostas. Eles propuseram a arquitetura ABC-CNN (*An Attention-Based Convolutional Neural Network*), uma rede neural convolucional configurável baseada em um mapa de atenção guiado por perguntas (QAM, do inglês *question-guided attention map*). Essa abordagem revolucionou a maneira como as redes neurais interpretam informações visuais relevantes para responder a perguntas, buscando características visuais semanticamente alinhadas entre as perguntas e as imagens. Esse alinhamento é realizado por meio de um processo de convolução com um *kernel* convolucional configurável, gerado pela transformação dos *embeddings* semânticos das perguntas para o espaço visual. Assim, o QAM se adapta, atribuindo diferentes níveis de importância às regiões da imagem na resposta a uma pergunta. No entanto, vale destacar que essa abordagem é mais eficaz para perguntas cujas respostas sejam monossilábicas.

Uma evolução importante no campo foi a abordagem de Li e Jia (2016), que introduziu um método de raciocínio baseado em redes neurais. Esse modelo iterativo atualiza a representação da pergunta, incorporando informações da imagem. Na camada de compre-

ensão da imagem, o modelo pré-treinado VGGNet é utilizado para transformar os atributos de detecção de objetos e as informações globais da imagem em vetores semânticos. A codificação da pergunta em linguagem natural é realizada por uma GRU (*Gated Recurrent Unit*), seguida por uma etapa de *pooling* que combina os componentes da pergunta com os atributos da imagem, gerando uma representação atualizada. Em seguida, o problema de classificação é resolvido para obter a resposta final. Além disso, a visualização das máscaras de atenção revelou a capacidade do modelo de focar em regiões altamente relevantes da imagem, identificando, inclusive, oportunidades de melhoria, especialmente em perguntas relacionadas à contagem de objetos.

Para combater vieses de linguagem na base VQA 1.0, [Agrawal et al. \(2017\)](#) e [Goyal et al. \(2017\)](#) propuseram uma abordagem inovadora, introduzindo a ideia de imagens complementares. Essa abordagem associa pares de imagens semelhantes a cada pergunta, gerando respostas distintas para a mesma pergunta. Ao avaliar modelos propostos na literatura usando essa nova base, foi observada uma significativa queda de desempenho, evidenciando os efeitos dos vieses nas estratégias anteriores. Posteriormente, [Agrawal et al. \(2018\)](#) ajustaram a partição dos *datasets* VQA 1.0 e VQA 2.0, incorporando o modelo GVQA (*Grounded Visual Question Answering*) com vieses induzidos e restrições arquiteturais para evitar “trapaças”, com base em dados de treinamento pré-existentes.

Outro avanço importante no campo dos algoritmos de VQA foi a introdução do *dataset* VizWiz, desenvolvido para apoiar pessoas com deficiência visual ([GURARI et al., 2018](#)). As análises mostraram uma baixa sobreposição de respostas entre o VizWiz e outros *datasets* existentes, como o VQA 2.0, com apenas 824 das 3.000 respostas principais do VizWiz coincidindo com as do VQA 2.0. Ao avaliar diversos algoritmos propostos, os pesquisadores refinaram alguns e constataram que o VizWiz oferece novas perspectivas para a resolução do problema de VQA.

Adicionalmente, [Singh et al. \(2019\)](#) desenvolveram o *dataset* TextVQA, motivados pela descoberta de que perguntas feitas por deficientes visuais frequentemente buscam extrair informações de texto presente em imagens do ambiente. Este *dataset* visa impulsionar avanços na resolução desse desafio. Eles também apresentaram o modelo LoRRA (*Look, Read, Reason & Answer*), que integra habilidades de observação, leitura, raciocínio e resposta. O modelo é composto por três componentes principais: (i) um módulo VQA que realiza raciocínio e inferência com base na imagem e na pergunta; (ii) um componente de leitura que interpreta o texto presente na imagem; e (iii) um módulo de resposta que faz previsões a partir de um espaço definido de respostas. A representação da imagem é feita por meio da extração de regiões com Faster-RCNN e da geração de *embeddings* usando ResNet-152 (*Residual Network-152*). A identificação de texto é realizada com o sistema Rosetta OCR (*Optical Character Recognition*) e a representação textual com *FastText*. As perguntas são representadas com uma combinação de GloVe (*Global Vectors for Word*

Representation) e uma rede LSTM, e a fusão dessas características é realizada por uma rede MLP, mostrando resultados promissores, apesar de atualmente se limitar à tarefa de OCR.

Enfatizando a interpretabilidade de modelos, [Wu e Mooney \(2019\)](#) abordaram a geração de explicações textuais para as previsões. Eles adotaram uma abordagem de múltiplos passos, começando pela segmentação da imagem e utilizando um módulo de atenção a objetos em conjunto com um modelo VQA pré-treinado. A etapa subsequente envolveu o treinamento de um módulo de explicação, projetado para gerar explicações textuais condicionadas aos atributos da pergunta, resposta e das áreas segmentadas da imagem. Esse método estabelece uma relação direta entre os termos na explicação textual e os elementos segmentados na imagem, conferindo maior interpretabilidade ao modelo. Ainda, a utilização do modelo LIME (*Local Interpretable Model-Agnostic Explanations*) ajudou a identificar quais objetos segmentados influenciam mais nas respostas e avaliar a correspondência entre os objetos mencionados na explicação e as regiões relevantes.

Também focados em sistemas de raciocínio interpretáveis, [Vedantam et al. \(2019\)](#) formularam um modelo de variável latente sequencial baseado em probabilidades. Este modelo tem a capacidade de conjuntamente analisar perguntas em forma de programas, raciocinar acerca de programas abstratos e executá-los sobre imagens utilizando redes neurais modulares. Por meio dessa abordagem, os pesquisadores conseguiram conferir às estruturas neuro-simbólicas propriedades desejáveis para sistemas de raciocínio transparentes e interpretáveis. O estudo revelou que os programas gerados pela abordagem proposta são mais compreensíveis, demandando um menor número de exemplos de treinamento. Além disso, demonstrou-se a viabilidade de explorar cenários contrafactuais, os quais permitem testar as crenças do modelo sobre os programas que levariam a respostas específicas com base nas imagens.

Em uma abordagem no ramo da medicina, [Abacha et al. \(2019\)](#) compartilharam resultados da competição ImageCLEF 2019 ao apresentarem o VQA-Med, um *dataset* derivado de imagens médicas de radiologia do MedPix5. Esse *dataset* contém quatro categorias principais de perguntas: modalidade, plano, sistema de órgãos e anormalidade. Notavelmente, as perguntas médicas foram formuladas para obter respostas diretamente do conteúdo das imagens, eliminando a necessidade de inferência adicional. Com 4.200 imagens radiológicas e 15.292 pares de perguntas e respostas, o *dataset* VQA-Med impulsionou a competição, com destaque para o uso de técnicas de aprendizado profundo, incluindo arquiteturas baseadas em CNNs, como VGGNet ou ResNet, e arquiteturas transformer como BERT, combinando mecanismos de atenção para otimizar o desempenho.

A abordagem *Relation-aware Graph Attention Network* (ReGAT), apresentada por [Li et al. \(2019\)](#), inova ao representar cada imagem como um grafo e modelar as relações inter-objetos de diferentes categorias através de um mecanismo de atenção em grafos. Essa

técnica visa aprender representações de relações adaptativas às perguntas apresentadas. A metodologia explora duas modalidades de relações visuais: (i) relações explícitas, que capturam as interações semânticas e posições geométricas entre os objetos, e (ii) relações implícitas, que revelam as complexidades subjacentes entre as regiões da imagem.

Assim como [Wu e Mooney \(2019\)](#) e [Vedantam et al. \(2019\)](#) buscaram por modelos interpretáveis, [Riquelme et al. \(2020\)](#) propuseram uma metodologia que fornece tanto explicações visuais quanto textuais para as respostas do VQA. A sua abordagem envolve um mapa de atenção supervisionado, treinado em paralelo com a tarefa principal de VQA. Esse mapa é aplicado a uma base de conhecimento (KB, do inglês *Knowledge Base*), permitindo a seleção de relações relevantes para cada par de pergunta e imagem. A KB é composta por relações do mundo real entre objetos, enriquecida com rótulos derivados de descrições de regiões e anotações de objetos do *dataset Visual Genome*. A incorporação da KB resultou em um impacto significativo nas previsões e interpretabilidade do modelo, embora desafios como o *overfitting* e viés relacionado ao conjunto de treinamento também tenham sido destacados.

Outra contribuição relevante é a arquitetura proposta por [Rahman et al. \(2021\)](#), que incorpora um módulo de *Attention of Attention* (AoA) no contexto de um codificador-decodificador. Esse módulo AoA é projetado para estabelecer uma relação entre os resultados de atenção e as perguntas. No processo, ele gera um vetor de informação e um *attention gate* usando os resultados de atenção e o contexto atual. Esses elementos são multiplicados por meio de outra atenção, resultando em informações assistidas finais. Além disso, os autores também apresentam um mecanismo de fusão baseado em atenção multimodal. Esse mecanismo visa incorporar características da imagem e da pergunta, com a rede de fusão ajustando dinamicamente os pesos de cada modalidade para obter a representação final das características usadas para fazer a predição final das respostas.

Ainda explorando a utilização do mecanismo de atenção, a abordagem apresentada por [Sharma e Jalal \(2021b\)](#) é capaz de capturar informações visuais previamente vistas. A representação da imagem é realizada usando uma ResNet101 pré-treinada na ImageNet, enquanto as representações das perguntas são obtidas por meio de um modelo GloVe pré-treinado e uma GRU. Uma GNN é empregada para capturar as relações semânticas visuais entre objetos ou regiões de interesse, e as representações da imagem e da pergunta são inseridas em um modelo LSTM com atenção. Esse modelo aborda a geração de respostas como uma tarefa de geração de sentenças, permitindo a produção de respostas mais abrangentes e detalhadas.

Outros trabalhos que também fizeram a utilização de variantes da GNN foram os realizados por [Liang, Jiang e Liu \(2021\)](#) e [Nuthalapati et al. \(2021\)](#). No campo das abordagens baseadas em grafos de cena, [Liang, Jiang e Liu \(2021\)](#) propuseram uma estrutura chamada GraphVQA. Essa estrutura utiliza uma rede neural de grafos guiada

por linguagem, que executa uma pergunta em linguagem natural através de iterações de *message passing* entre os nós do grafo. Enquanto [Nuthalapati et al. \(2021\)](#) introduziram um modelo denominado *Conditional Enhanced Graph Attention Network* (CE-GAT) para representar grafos de cena. O CE-GAT integra atributos nos vértices e arestas do grafo, conectando-se a um codificador de perguntas para gerar respostas precisas.

Focados na identificação de vieses e avaliação de atalhos multimodais presentes em conjuntos de treinamento, [Dancette et al. \(2021\)](#) introduziram o protocolo de avaliação *VQA-CounterExamples*. Ao explorar esses atalhos, os autores evidenciaram como certas palavras-chave nas perguntas, juntamente com objetos específicos nas imagens, podiam sistematicamente levar a respostas incorretas. Para solucionar esse problema, eles propuseram uma abordagem em duas etapas. Primeiramente, os objetos nas imagens foram detectados e rotulados utilizando um modelo Faster R-CNN. Em seguida, cada exemplo foi resumido através de indicadores binários, representando palavras-chave da pergunta, respostas e rótulos dos objetos identificados. Através de um algoritmo de mineração de regras, identificaram co-ocorrências frequentes, resultando em um conjunto de regras preditivas simples capazes de explicar os atalhos multimodais identificados.

Dando sequência à evolução do VQA no ramo da medicina assim como feito por [Abacha et al. \(2019\)](#), a estratégia delineada por [Liu et al. \(2022\)](#) utiliza-se de dois módulos centrais. O primeiro módulo constrói um amplo conjunto de imagens radiológicas de órgãos variados e modalidades, empregando uma rede neural profunda para extrair características visuais. O segundo módulo representa perguntas por meio da combinação entre o GloVe e uma GRU, com características fundidas por meio de redes de atenção bilineares e em sequência a resposta final é gerada por um classificador MLP.

Explorando a utilização de dados sintéticos gerados por computação gráfica, [Cascante-Bonilla et al. \(2022\)](#) propuseram uma abordagem para VQA que controla completamente o espaço visual e de linguagem, proporcionando uma gama mais diversificada de cenários sem a preocupação de expor dados confidenciais ou pessoais. Seu processo de geração de perguntas e respostas é conduzido por uma gramática baseada em modelo, vinculada a um grafo de cena predefinido e à imagem correspondente. O *embedding* de texto empregado foi o GloVe pré-treinado juntamente com uma LSTM. Introduziram também a técnica de *Feature Swapping* (FSWAP), que consiste em alternar aleatoriamente atributos de nível de objeto durante o treinamento. Isso tornou o modelo VQA mais invariante de domínio, permitindo a substituição de atributos de dados reais por aquelas obtidas em dados sintéticos e vice-versa. Apesar de resultados promissores, a abordagem apresentou limitações na geração de objetos animados, devido às restrições das ferramentas utilizadas na geração dos dados.

A problemática da geração de dados também foi abordada por [Changpinyo et al. \(2022\)](#), que investigaram a possibilidade de gerar automaticamente anotações VQA a

partir de recursos de imagem-texto existentes, como legendas de imagem. Nesse contexto, um modelo neural T5-XXL (*Text-To-Text Transfer Transformer-XXL*) foi utilizado para gerar perguntas a partir de respostas candidatas obtidas automaticamente a partir das legendas das imagens. No entanto, esse processo revelou algumas limitações, incluindo saídas errôneas e o risco de introdução de preconceitos e estereótipos presentes nos dados. O estudo enfatizou que, apesar das vantagens da geração automática de dados, é necessário lidar com os desafios associados a alucinações de modelos generativos e amostragens inadequadas.

Por sua vez, o modelo *Graph Attention Network Relational Reasoning* (GAT2R), proposto por Miao et al. (2022), aborda principalmente a geração de grafos de cena e a subsequente previsão de respostas para esses grafos. O módulo de geração do grafo de cena extrai características regionais e espaciais dos objetos por meio de um modelo de detecção de objetos. Ele utiliza um decodificador de relação para prever conexões entre pares de objetos, formando assim um grafo de cena. A previsão da resposta, baseada no grafo de cena, dinamicamente atualiza as representações dos nós utilizando uma rede de atenção guiada pela pergunta. Em seguida, realiza a fusão multimodal entre essas representações e as características da pergunta, gerando a resposta final.

Adicionalmente, Koshti, Gupta e Kalla (2022) exploraram as capacidades de representação do BERT como um extrator robusto de atributos textuais para o VQA. Para a representação da imagem, adotaram o Faster R-CNN com ResNet-101 para localizar e extrair atributos dos objetos presentes. A concatenação dos atributos de diferentes camadas do codificador BERT alimentou uma Bi-LSTM, responsável pela representação das perguntas. A fim de aprimorar a representação conjunta dos domínios visual e textual, uma co-atenção alternada com multiplicação elemento a elemento foi empregada, seguida de classificação por meio de uma MLP.

A adaptação de sistemas de aprendizado de máquina para a vida real representa um desafio complexo, dado que as situações reais e os usuários frequentemente diferem dos cenários presentes em *datasets* sintéticos. Disparidades nas distribuições de imagens, perguntas e respostas são frequentes. Embora grande parte das pesquisas em VQA se concentre na perspectiva de compreensão de máquina, é incerto se as arquiteturas modeladas e avaliadas nesse contexto podem ser prontamente adaptadas para situações de acessibilidade. Diante dessa incerteza, Cao et al. (2022) investigaram a lacuna entre esses domínios, avaliando sete modelos propostos na literatura. Tais modelos, originalmente treinados no *dataset* VQA 2.0, foram submetidos ao *dataset* VizWiz, um *dataset* direcionado a acessibilidade. Os resultados indicaram que avanços nas arquiteturas voltadas à compreensão de máquina também impulsionaram o desempenho na esfera de acessibilidade. Contudo, constatou-se que a disparidade entre as performances dos modelos nos dois contextos ainda é grande e tende a aumentar. Cabe ressaltar que as categorias mais desafiadoras

para as gerações atuais de modelos abrangem o reconhecimento de objetos, textos e cores, bem como a resolução de perguntas ambíguas.

Inspirados pelo conceito de *Neural Module Network* (NMN), [Tiong et al. \(2022\)](#) conceberam uma abordagem VQA modular e *zero-shot*¹ chamada PNP-VQA (*Plug-and-Play VQA*). A premissa central do PNP-VQA reside na criação de uma interface entre modelos pré-treinados de linguagem e visão, sem a necessidade de treinamento adicional. O método se desdobra em três fases principais: primeiro, identifica as regiões relevantes da imagem para a pergunta, alcançada por meio de uma variação do GradCAM (*Gradient-weighted Class Activation Mapping*), uma técnica de interpretabilidade; em seguida, gera um conjunto diversificado de legendas usando o modelo BLIP (*Bootstrapping Language-Image Pre-training*), a partir das regiões da imagem identificadas; por fim, obtém a resposta através de um modelo codificador-decodificador, que incorpora as perguntas e as legendas geradas anteriormente como entradas. Este último modelo, alimentado pelas perguntas e legendas geradas na fase anterior, gera respostas de forma aberta e é avaliado com base em correspondência exata. Embora o PNP-VQA seja adaptável às evoluções dos modelos pré-treinados, ele herda seus vieses e carrega um custo adicional de inferência devido às várias etapas envolvidas.

Outro trabalho que se diferencia pela ausência de necessidade de ajustes finos nos extratores de atributos é o do [Deuser et al. \(2022\)](#) que apresentaram uma arquitetura fundamentada no modelo CLIP. Para representar imagens, o modelo emprega o *Vision Transformer*, enquanto a representação textual é baseada no *Transformer*. O processo consiste na extração dos atributos correspondentes a imagens e perguntas, as quais são posteriormente concatenadas e direcionadas por camadas lineares para o processo de classificação. Adicionalmente, essa abordagem introduz um método para a seleção das respostas utilizadas durante o treinamento, optando pela resposta mais frequente que obtém a pontuação mais alta para cada par imagem/pergunta. Uma característica notável dessa proposta é a rapidez e eficiência do treinamento, viabilizando um processo robusto sem a necessidade de recursos computacionais extensivos.

[Cao e Jiang \(2023\)](#) apresentaram um enfoque semelhante. Eles propuseram uma rede modular *zero-shot* para o VQA, decompondo perguntas em subtarefas e alocando essas subtarefas para modelos pré-treinados. A abordagem consiste em analisar as perguntas em etapas de raciocínio, redirecionando cada subtarefa para um modelo pré-treinado específico, sem a necessidade de ajuste fino. Essa metodologia emprega três modelos distintos: OWL (*Open-World Localization*) para detecção de objetos, MDETR (*DEtection TRansformer model*) para localização de expressões de referência e CLIP para gerar respostas abertas.

¹ *Zero-shot* é uma abordagem em aprendizado de máquina onde um modelo consegue realizar tarefas ou fazer previsões em domínios ou classes que ele não foi especificamente treinado. Em outras palavras, o modelo pode generalizar e aplicar seu conhecimento a novas situações sem exemplos anteriores específicos.

Dado que os modelos de linguagem e visão atuais têm limitações na compreensão espacial, [Cao e Jiang \(2023\)](#) também definiram heurísticas para auxiliar no raciocínio durante o processo.

Enquanto isso, [Hu et al. \(2024\)](#) concentraram-se em criar uma abordagem interpretável, usando grafos como ferramenta essencial. Eles construíram um *dataset* amplo de dados médicos, extraídos de relatórios do *dataset KeyInfo*, para formar pares de perguntas e respostas, focados em imagens de raios-x do tórax. Seu modelo utiliza um grafo que explora relações visuais, espaciais e semânticas. As relações semânticas baseiam-se em um grafo de conhecimento com informações sobre estruturas anatômicas e doenças. A representação dos vértices é construída a partir das regiões de interesse (ROI, do inglês *Regions Of Interest*), obtidas com um modelo pré-treinado Faster R-CNN. Três categorias de relações foram consideradas para formar as arestas: (i) relações espaciais, baseadas na ROI; (ii) relações semânticas, fundamentadas em conhecimento especializado; e (iii) relações visuais, revelando conexões subjacentes. Na modelagem das perguntas, a técnica GloVe é combinada com uma camada GRU, e as respostas são obtidas com uma rede MLP.

No contexto de ampliar a capacidade de compreensão, [Ravi et al. \(2023\)](#) centraram sua atenção na incorporação de raciocínio de senso comum no VQA. Essa abordagem incorpora o *Commonsense Transformer* (COMET), um modelo treinado em bases de conhecimento curadas, que gera inferências de senso comum a partir das imagens de entrada para responder a perguntas. O modelo resultante, denominado VLC-BERT (*Vision-Language Commonsense BERT*), integra essas inferências com regiões relevantes da imagem em um *Transformer* VL-BERT, gerando respostas embasadas em conhecimento contextual. Apesar de suas perspectivas promissoras, essa abordagem enfrenta desafios devido às limitações do COMET e das bases de conhecimento utilizadas, o que a impede de superar modelos de grande escala como o GPT-3 (*Generative Pre-trained Transformer 3*), por exemplo.

Um enfoque alternativo foi adotado por [Guo et al. \(2023\)](#), que exploraram o uso de LLMs (*Large Language Models*) para abordar o desafio do VQA. A abordagem central do método denominado Img2LLM é a integração do BLIP e do GradCam para identificar as regiões relevantes das imagens em relação às perguntas formuladas. Isso é seguido pela geração de legendas para essas regiões identificadas. Utilizando essas legendas, um modelo de geração de perguntas é empregado para criar pares sintéticos de perguntas e respostas. Esses pares, juntamente com as legendas das imagens, são utilizados como entrada para um modelo LLM, o OPT (*Open pre-trained transformer language models*). Esse método proporciona a execução do VQA sem a necessidade de treinamento de ponta a ponta ou de redes de controle de qualidade textual especializadas. Isso viabiliza modelos mais flexíveis e de menor custo, embora a geração de legendas e pares de perguntas e respostas possa introduzir uma carga adicional de inferência.

Também motivados pela crescente adoção de modelos generativos e pela limitação da métrica de acurácia, que é rigorosa e subestima o desempenho desses sistemas, visto que, em tais configurações, as respostas geradas pelos modelos podem não coincidir com nenhuma das respostas de referência, mesmo sendo a resposta correta para a pergunta, [Mañas, Krojer e Agrawal \(2024\)](#) propuseram uma nova métrica automática de avaliação, LAVE (*LLM-Assisted VQA Evaluation*). A construção do *prompt* inclui os componentes típicos: descrição da tarefa, alguns exemplos de entrada/saída e a entrada para um exemplo de teste. Esse *prompt* é então enviado para uma LLM para gerar uma classificação de qualidade. Algumas das LLMs testadas foram Vicuna-v1.3, GPT-3.5-Turbo (também conhecido como ChatGPT) e Flan-T5. Os testes demonstraram que o LAVE é capaz de recuperar uma fração considerável de respostas corretas que a acurácia considerava erradas. No entanto, uma métrica VQA baseada em LLM sofre dos problemas inerentes às LLMs, como alucinações, onde os modelos fazem afirmações incorretas. Além disso, o uso de LLMs para produzir justificativas junto com classificações torna a avaliação lenta devido à natureza sequencial da geração autoregressiva; e a variante de melhor desempenho foi o LAVE GPT-3.5, que requer acesso a uma API paga, o que pode impedir sua adoção generalizada.

O estudo realizado por [Kunlamai et al. \(2024\)](#) apresentou uma nova aplicação para o VQA, visando agilizar as tarefas administrativas relacionadas às inspeções de pontes. Por exemplo, ao automatizar o processo de geração de relatórios de inspeção, a predefinição de perguntas para as imagens, como parte de um fluxo de trabalho automatizado, permite que o modelo VQA reduza significativamente a carga de trabalho humana, fornecendo respostas a várias perguntas simultaneamente. Além disso, a integração de algoritmos VQA em robôs autônomos, como veículos aéreos não tripulados (VANT), promete viabilizar inspeções autônomas. Caso seja detectado algum dano na ponte, o VANT pode interagir com o modelo VQA para obter *feedback*, facilitando a identificação da causa do dano. Esse *feedback* capacita o VANT a ajustar dinamicamente sua trajetória de voo e plano de inspeção. Considerando os desafios decorrentes do alto custo associado à criação de *datasets* de treinamento que exigem conhecimento especializado, o estudo utilizou um conjunto de relatórios de inspeção gerados entre 2004 e 2018 para 3.118 pontes no Japão, mantidos pelo Departamento de Desenvolvimento Regional de Kanto, para a construção do *dataset* do VQA. A partir disso, diversos modelos pré-existentes, como MMBERT e ALBEF, foram treinados, permitindo-lhes aprender representações semânticas específicas do domínio e extrair informações valiosas para a manutenção e diagnóstico de pontes a partir de imagens e textos de entrada. Posteriormente, esses modelos pré-treinados foram ajustados para a tarefa do VQA.

[Qian et al. \(2024\)](#) apresentaram um novo *dataset*, NuScenes-QA, voltado para o contexto de condução autônoma. O VQA aplicado a este cenário permite que os passageiros compreendam o ambiente ao seu redor por meio da linguagem, aumentando sua experiência e confiança na utilização de sistemas de condução inteligentes. Em comparação com tarefas

tradicionais de VQA, esta aplicação é mais desafiadora, uma vez que os dados visuais brutos são multimodais, incluindo imagens e nuvens de pontos capturadas por câmeras. Além disso, os dados são *multiframe* devido à aquisição contínua em tempo real, e as cenas externas apresentam tanto elementos em movimento no primeiro plano quanto um fundo estático. O NuScenes-QA contém 34 mil cenas visuais e 460 mil pares de perguntas e respostas. Os experimentos realizados com este *dataset* consistem basicamente em três componentes: extração de atributos das imagens e nuvens de pontos, extração de atributos dos objetos, e fusão desses atributos seguida pela previsão de respostas.

Enfrentando a falta de *datasets* específicos de domínio assim como Qian et al. (2024) e Koshti et al. (2024) desenvolveram o EDUVQA, um *dataset* voltado para a educação. Este *dataset* consiste em quatro categorias de imagens: animais, plantas, vegetais e frutas. O EDUVQA contém 1.500 imagens, 12.000 perguntas, 750 fatos únicos e 1.200 respostas únicas, com os fatos sendo compostos por um trio SVO (sujeito, verbo, objeto). Os experimentos demonstraram que a inclusão do SVO permite que o modelo integre conhecimento externo e responda a perguntas mais complexas, resultando em melhorias de performance.

3.2 Resumo da Revisão da Literatura

Em síntese, as áreas de Visão Computacional e Processamento de Linguagem Natural têm apresentado avanços contínuos, abrangendo desde aprimoramentos técnicos até considerações éticas. As pesquisas exploram diversas abordagens, incluindo a aplicação de modelos de linguagem em larga escala, a utilização de *datasets* sintéticos e a implementação de técnicas de interpretabilidade. Essas evoluções têm contribuído significativamente para a obtenção de uma compreensão mais profunda das informações visuais e linguísticas, resultando em sistemas mais capazes e socialmente responsáveis.

Entretanto, desafios substanciais ainda permanecem. Exemplos desses incluem a limitação de *datasets* que muitas vezes são insuficientes para treinar e avaliar modelos complexos, a restrição das análises a cenários exclusivamente internos em imagens, o que restringe a gama de perguntas passíveis de exploração, bem como as dificuldades impostas por condições extremas de iluminação, que dificultam até mesmo a avaliação humana das perguntas. Além disso, vieses têm sido identificados tanto nas imagens quanto nas perguntas formuladas.

Ainda que diversas abordagens interpretáveis tenham sido propostas, permanece incerto quais fatores realmente impulsionam melhorias no desempenho. Além disso, a distinção das contribuições individuais de dados textuais e de imagem é uma tarefa complexa, dificultando a avaliação da validade dos algoritmos sob as razões corretas. É possível que esses algoritmos demonstrem bom desempenho em *benchmarks* sem que resolvam

efetivamente o problema (KAFLE; SHRESTHA; KANAN, 2019).

Adicionalmente, a determinação da melhor metodologia para avaliar sistemas de VQA ainda permanece uma questão em aberto. Cada estratégia de avaliação possui vantagens e desvantagens intrínsecas, cuja escolha depende da forma como o *dataset* foi construído, do grau de viés presente e dos recursos disponíveis. Essa diversidade de abordagens traz consigo desafios na comparação de algoritmos, dada a variação nos métodos de avaliação empregados (KAFLE; KANAN, 2017). O Quadro 4 apresenta uma compilação dos trabalhos revisados, destacando os *datasets*, modelos e métricas utilizados nas abordagens analisadas.

Referência	Datasets	Modelos	Métricas
Agrawal et al. (2017)	VQA 1.0	BoW, LSTM, VGG-Net	Acurácia
Chen et al. (2015)	Toronto COCO-QA, DAQUAR, VQA 1.0	LSTM, VGG-19	Acurácia, Wu-Palmer similarity measure Set (WUPS)
Goyal et al. (2017)	VQA 2.0	LSTM, CNN	Recall@5, Acurácia
Li e Jia (2016)	COCO-QA, VQA 1.0	GRU, VGG	Acurácia, WUPS
Agrawal et al. (2018)	VQA-CP v1, VQA-CP v2, VQA 1.0, VQA 2.0	LSTM, Glove, VGG-Net	Acurácia
Li et al. (2019)	VQA 2.0, VQA-CP v2, Visual Genome	Glove, GRU, GAT, Faster R-CNN, ResNet-101	Acurácia
Vedantam et al. (2019)	CLEVR, SHAPES	LSTM, ResNet-101, CNN	Acurácia
Wu e Mooney (2019)	VQA-X, VQA 2.0	GRU, LSTM, GloVe, GradCAM, LIME	BLEU-4, METEOR, ROUGE-L, CIDEr, SPICE, EMD, Faithful score
Singh et al. (2019)	VQA 2.0, VizWiz, TextVQA	Glove, LSTM, FastText, ResNet-152, Faster-RCNN, Rosetta OCR	Acurácia
Riquelme et al. (2020)	VQA 2.0, VQA-X, VQA-HAT	Glove, LSTM, ResNet-152	Acurácia, Rank-Correlation, BLEU-4, METEOR, ROUGE-L, CIDEr, SPICE, EMD
Dancette et al. (2021)	VQA 2.0, VQA-CP v2	Faster R-CNN	Acurácia
Sharma e Jalal (2021b)	VQA 1.0, VQA 2.0	LSTM, GRU, GloVe, ResNet101, GNN	Acurácia
Rahman et al. (2021)	VQA 2.0	Glove, LSTM, Faster R-CNN, ResNet-101	Acurácia
Nuthalapati et al. (2021)	GQA	GAT, GRU	Acurácia, Consistency, Validity, Plausibility
Liang, Jiang e Liu (2021)	GQA	GCN, GINE, GAT	Acurácia, Consistency, Validity, Plausibility, Distribution
Changpinyo et al. (2022)	VQA 2.0, GQA, OK-VQA	T5-XXL, ResNet-152, Faster R-CNN	Acurácia
Cascante-Bonilla et al. (2022)	ThreeDWorld-VQA, Real-VQA, Hypersim-VQA	Glove, LSTM, Faster-RCNN, CLIP, ResNet-50, ResNet-101	Acurácia
Miao et al. (2022)	GQA, VQA 2.0	Glove, GRU, GAT, Faster R-CNN	Acurácia, Consistency, Validity, Plausibility, Distribution, Ground truth
Tiong et al. (2022)	VQA 2.0, GQA, OK-VQA	BLIP, GradCAM	Acurácia, Answer hit rate (AHR)
Deuser et al. (2022)	VizWiz	CLIP	Acurácia
Koshti, Gupta e Kalla (2022)	VQA 2.0	BERT, Faster R-CNN, ResNet-101	Acurácia
Liu et al. (2022)	VQA-RAD, SLAKE	LSTM, GloVe, GRU, BAN, ResNet-50, Grad-CAM	Acurácia
Cao e Jiang (2023)	GQA e VQA 2.0	OWL, MDETR, CLIP	Acurácia
Guo et al. (2023)	VQA 2.0, OK-VQA, A-OKVQA	T5, OPT, BLIP, GradCAM	Acurácia, AHR, Answer noise rate (ANR)
Hu et al. (2024)	RAD-VQA, Mimic-VQA	Glove, GRU, Faster R-CNN, GAT	AUC, Acurácia
Ravi et al. (2023)	OK-VQA, A-OKVQA	SBERT, Fast RCNN, YOLOv5, VL-BERT	Acurácia
Koshti et al. (2024)	EDUVQA	ResNet50, BERT	Acurácia
Mañas, Krojer e Agrawal (2024)	VQA 2.0, VG-QA, OK-VQA	Flan-T5-XXL, Vicuna-v1.3-13B, GPT-3.5-Turbo	LAVE
Qian et al. (2024)	NuScenes-QA	GloVe, biLSTM, ResNet	Top-1 accuracy

Quadro 4 – *Datasets*, modelos e métricas usados nos trabalhos da literatura VQA avaliada.

Diante do cenário exposto, este trabalho tem como objetivo melhorar a precisão dos resultados obtidos para o *dataset* VizWiz. Para isso, serão propostos dois métodos e quatorze abordagens fundamentadas nesses métodos. O primeiro método consiste em representar os dados de entrada (imagem e pergunta) de forma conjunta utilizando grafos e aplicar GAT para alcançar a resposta desejada. O segundo método baseia-se na abordagem apresentada por Deuser et al. (2022), na qual os atributos representativos dos dados de entrada são concatenadas e, em seguida, o problema é resolvido por meio de uma MLP.

As abordagens propostas envolvem a utilização de diferentes elementos, como os modelos para representar os atributos das imagens e perguntas, incluindo BERT, CLIP, BLIP2, entre outros. Além disso, aspectos como a quantidade de classes utilizadas, a forma como essas classes são definidas, as funções de perda empregadas, dentre outros

representam os diferenciais entre as abordagens. Um detalhamento mais aprofundado sobre os métodos e abordagens será apresentado no Capítulo 4.

4 Abordagem Proposta

Neste capítulo, são apresentados os dois métodos desenvolvidos neste trabalho, bem como as abordagens implementadas que os utilizam. As Seções 4.1 e 4.2 descrevem detalhadamente os métodos propostos, estruturados com base nas principais estratégias discutidas na literatura e descritas na Seção 2.2. A Seção 4.3 resume as principais diferenças entre os métodos propostos. Finalmente, a Seção 4.4 apresenta os elementos que compõem as 14 abordagens desenvolvidas, ressaltando suas características específicas e os diferenciais de cada uma.

4.1 Método 1

O Método 1 é composto por quatro etapas principais, detalhadas nas próximas seções: a Seção 4.1.1 aborda o pré-processamento dos dados da imagem e da pergunta; a Seção 4.1.2 descreve o processo de extração dos atributos; a Seção 4.1.3 discute a fusão dos atributos; e, por fim, a Seção 4.1.4 explica a etapa de obtenção das respostas. A Figura 31 ilustra de forma geral essas etapas.

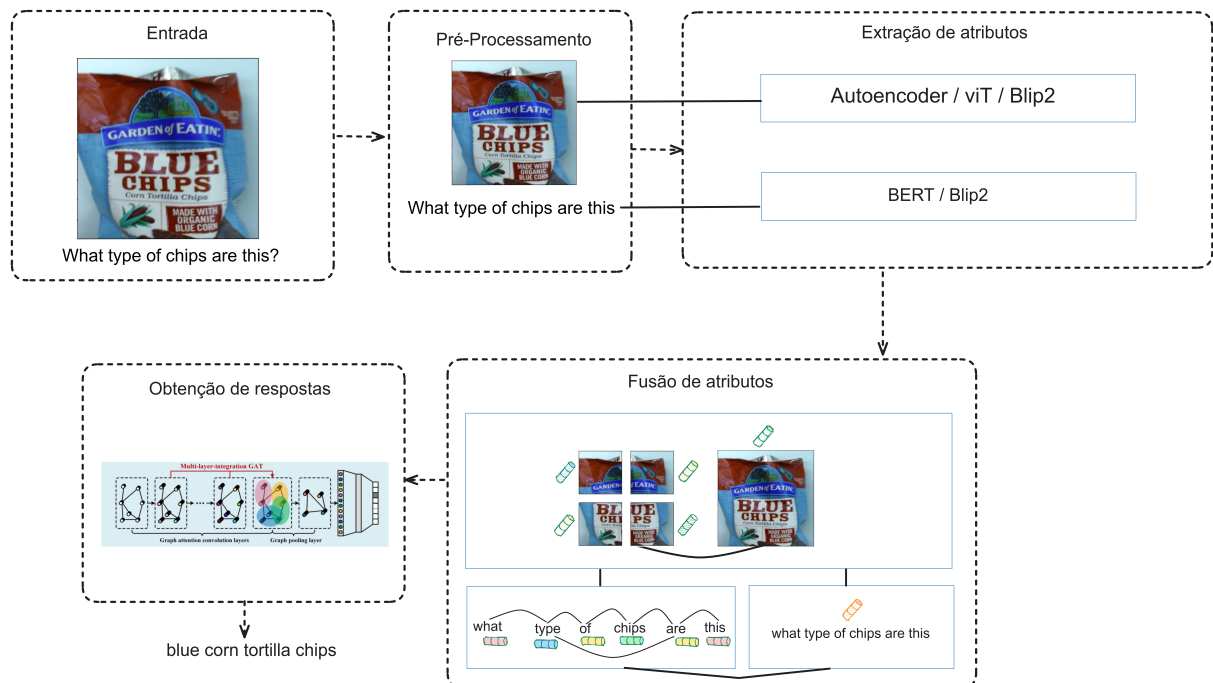


Figura 31 – Pipeline do Método 1 exemplificado com 4 patches.

4.1.1 Pré-processamento

O pré-processamento está dividido em dois segmentos: um voltado para a pergunta e outro para a imagem. Em PLN, essa etapa é essencial para melhorar a qualidade e a eficiência dos modelos, transformando textos brutos em dados mais estruturados e padronizados. Para realizar o pré-processamento textual, foram utilizadas duas bibliotecas principais: *contractions* e *regex (re)*.

O pré-processamento das perguntas implementado neste trabalho é simplificado e ocorre em quatro etapas principais: padronização para letras minúsculas, expansão de abreviações, remoção de pontuações e remoção de espaços extras. A seguir, é apresentado um exemplo do resultado desse processo.

- Frase original:
 - *What is this? Yeah, I can't it though. It vibrated. Let's see. It*
- Etapa 1: Padronização para letras minúsculas
 - *what is this? yeah, i can't it though. it vibrated. let's see. it*
- Etapa 2: Expansão de abreviações
 - *what is this? yeah, i cannot it though. it vibrated. let us see. it*
- Etapa 3: Remoção de pontuações
 - *what is this yeah i cannot it though it vibrated let us see it*
- Etapa 4: Remoção de espaços extras
 - *what is this yeah i cannot it though it vibrated let us see it*

No exemplo apresentado, a frase resultante, *what is this yeah i cannot it though it vibrated let us see it*, torna-se uma das entradas para a próxima etapa do método.

O pré-processamento da imagem envolve dois passos principais: redimensionamento e normalização. Modelos de *machine learning* geralmente requerem entradas de tamanho fixo. O redimensionamento das imagens para uma dimensão específica garante uniformidade no formato, o que facilita o processamento e previne erros. Além disso, essa etapa pode reduzir a quantidade de dados a serem processados, diminuindo o tempo de treinamento sem, frequentemente, comprometer a qualidade dos resultados, dependendo da tarefa e da resolução ideal.

A normalização das imagens, geralmente realizada ao converter os valores dos pixels para um intervalo entre 0 e 1 ou -1 e 1, assegura que todas as entradas estejam em uma escala semelhante. Isso minimiza o impacto de valores extremos, melhorando a estabilidade do treinamento e, potencialmente, permitindo uma convergência mais rápida (CORLEY et al., 2024). Neste método, optou-se por normalizar os valores dos pixels para o intervalo entre -1 e 1. Para isso, utilizou-se a função *Normalize* da biblioteca *torchvision*, conforme descrito na Equação 4.

$$output[channel] = \frac{input[channel] - mean[channel]}{std[channel]} \quad (4)$$

4.1.2 Extração de atributos

A extração de atributos da pergunta será realizada utilizando os modelos pré-treinados BERT ou BLIP2. A predição com um modelo pré-treinado aproveita o aprendizado adquirido a partir de grandes volumes de dados textuais, permitindo que o modelo desenvolva uma compreensão profunda das relações complexas entre palavras e contextos.

No caso do BERT, o texto é processado em *tokens* (subpalavras ou palavras), incluindo *tokens* especiais como [CLS] e [SEP], que estruturam a entrada do modelo. Cada *token* é convertido em um vetor de *embeddings*, refinado pelas camadas de *encoders* do BERT. Essas camadas aprimoram o entendimento contextual ao integrar informações de cada palavra ao contexto global da frase, gerando uma representação vetorial contextualizada para cada *token*.

Já no caso do BLIP2, o texto é fornecido diretamente ao modelo, sem a necessidade de etapas adicionais de preparação. Ambos os modelos, BERT e BLIP2, produzem *embeddings* com dimensão 768. Assim, serão obtidos *embeddings* para representar tanto a pergunta pós pré-processamento quanto cada palavra que a compõe, conforme ilustrado na Figura 32.

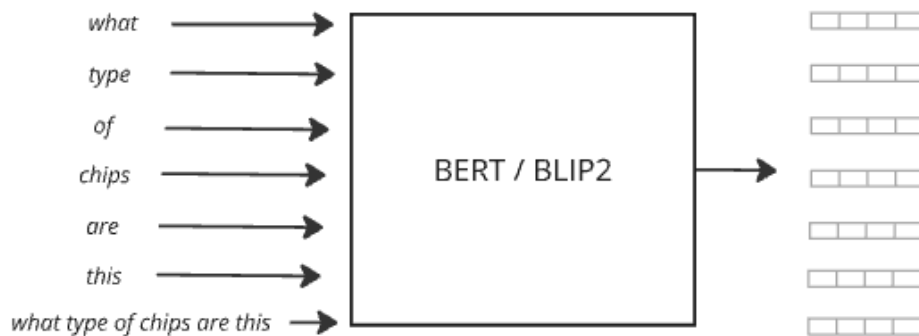


Figura 32 – Exemplo de obtenção de *embeddings* para a pergunta e as palavras que a compõem.

A extração de atributos das imagens será realizada por meio de modelos pré-treinados e modelos treinados de forma *end-to-end*. Os modelos pré-treinados utilizados serão o ViT e o BLIP2, enquanto o modelo treinado *end-to-end* selecionado é um *Auto-encoder*. Maiores detalhes sobre o treinamento deste modelo podem ser encontrados no Apêndice A.

O funcionamento do ViT é semelhante ao do BERT, pois requer a adição de um *token* especial no processo de obtenção dos *embeddings*. Nos demais modelos, não são necessárias etapas adicionais; basta fornecer a imagem já pré-processada. Todos os

modelos retornam um *embedding* de tamanho 768, assim como o BERT e o BLIP2. Os *embeddings* serão obtidos tanto para a imagem inteira quanto para seus *patches*.

Um *patch* de imagem é uma pequena porção ou segmento de uma imagem maior, geralmente em formato retangular ou quadrado. Essa técnica é útil em redes neurais que processam imagens, pois reduz a complexidade computacional e permite que o modelo capture detalhes específicos de cada parte da imagem. A Figura 33 ilustra o processo de obtenção dos *embeddings* para a imagem completa e para seus *patches*.

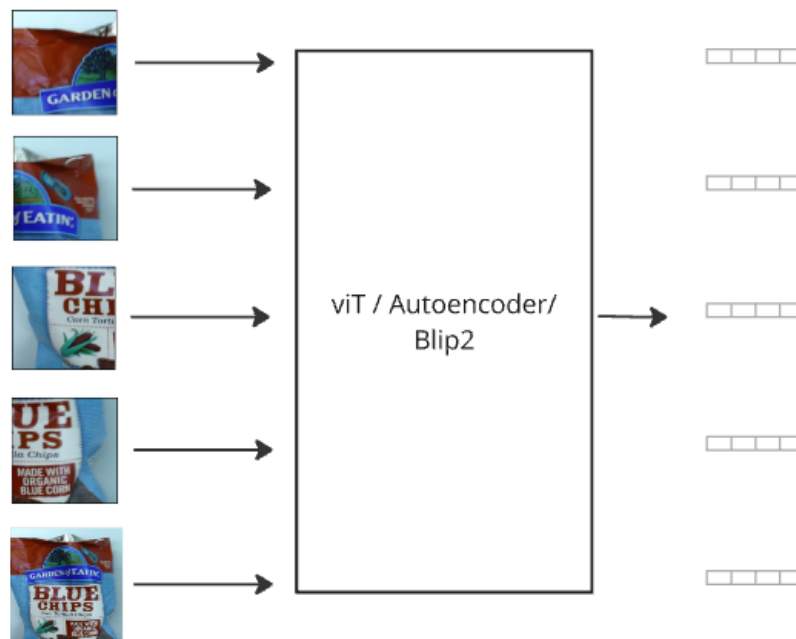


Figura 33 – Exemplo de obtenção dos *embeddings* para a imagem e seus *patches*, sendo considerados quatro neste caso.

4.1.3 Fusão dos atributos

Após a extração dos atributos, estas são organizadas em uma estrutura unificada modelada como um grafo não direcionado. Esse grafo é formado por um conjunto de vértices interligados por arestas bidirecionais, o que significa que a conexão entre dois vértices permite a navegação em ambos os sentidos. A estrutura do grafo é composta por diferentes tipos de vértices:

- Um vértice que representa a imagem como um todo;
- p vértices, cada um correspondente a um dos *patches* resultantes da divisão da imagem em p partes iguais;
- Um vértice que representa a pergunta completa;
- n vértices, cada um associado a uma palavra individual da pergunta.

Cada vértice possui um *embedding* associado, previamente calculado na etapa anterior. As conexões entre esses vértices são estabelecidas pelas arestas do grafo, seguindo

as seguintes regras:

- Cada um dos vértices estará conectado a si mesmo (*self-loop*);
- Os vértices associados aos *patches* da imagem serão conectados aos demais *patches* de acordo com sua disposição na imagem. Assim, cada vértice dos *patches* poderá estabelecer conexões apenas com os vértices dos *patches* localizados imediatamente acima, abaixo, à direita e à esquerda, conforme ilustrado no exemplo da Figura 34. Além disso, esses vértices estarão interligados aos vértices da imagem completa, à pergunta e às palavras que a compõem, conforme demonstrado na Figura 35;

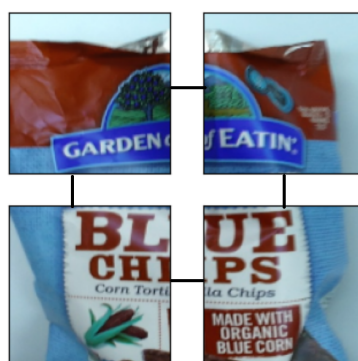


Figura 34 – As conexões entre os *patches* são demonstradas no exemplo, que utiliza apenas 4 *patches*.

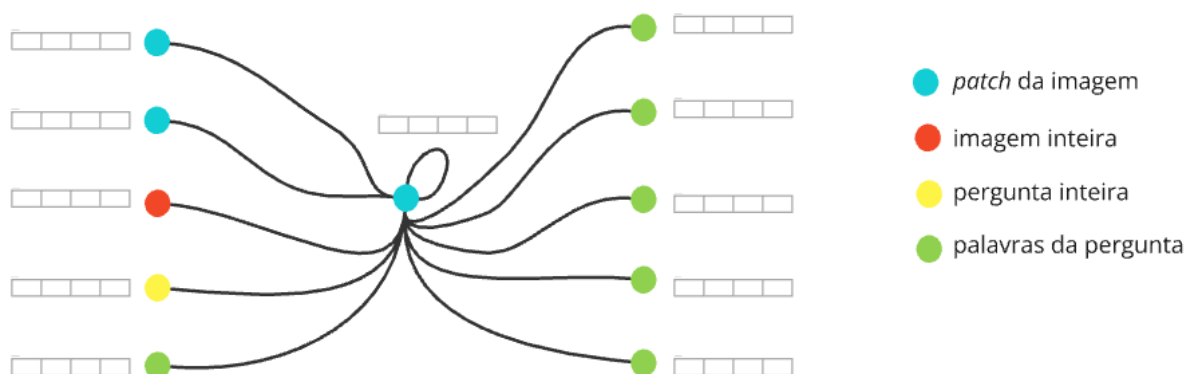


Figura 35 – Exemplo das conexões de um *patch* da imagem.

- O vértice que representa a imagem completa estará interligado a todos os outros vértices, assim como o vértice correspondente à pergunta. As Figuras 36 e 37 ilustram essas conexões;
- As conexões entre os vértices que representam as palavras da pergunta serão estabelecidas de duas maneiras: bidirecionalmente e por meio de *dependency parsing*, utilizando o modelo de melhor acurácia, *en_core_web_trf*, da biblioteca *spaCy*. *Dependency parsing* refere-se à análise das relações gramaticais entre as palavras de uma frase, determinando quais palavras são dependentes de outras, como sujeito-verbo e

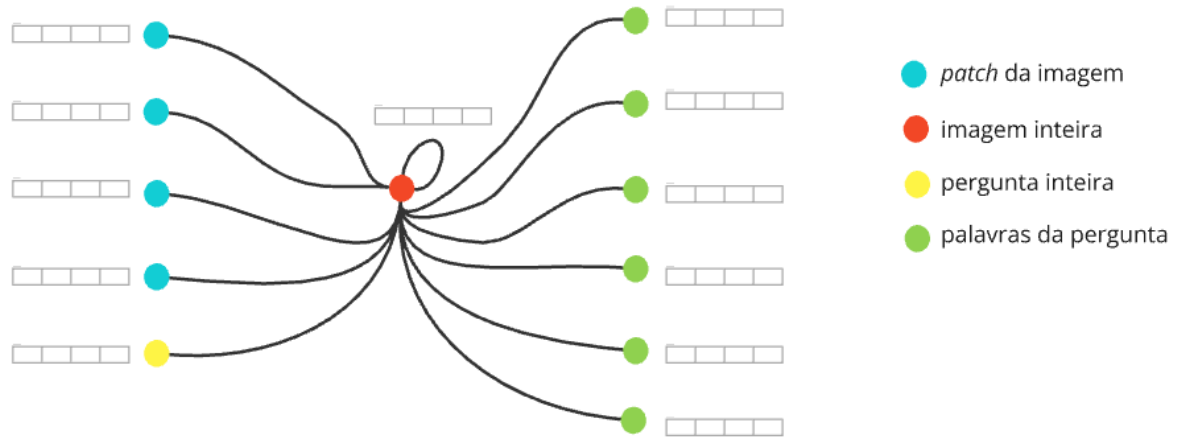


Figura 36 – Exemplo das conexões da imagem completa.

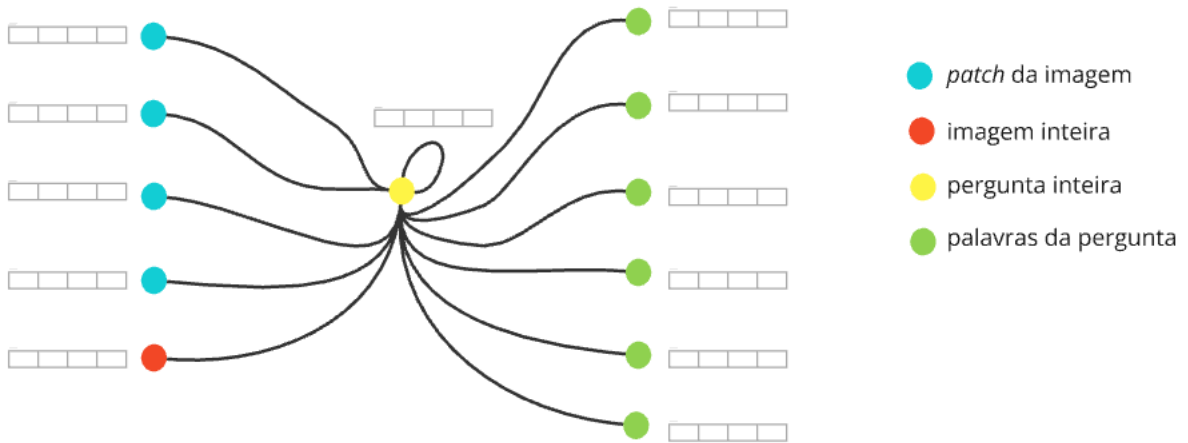
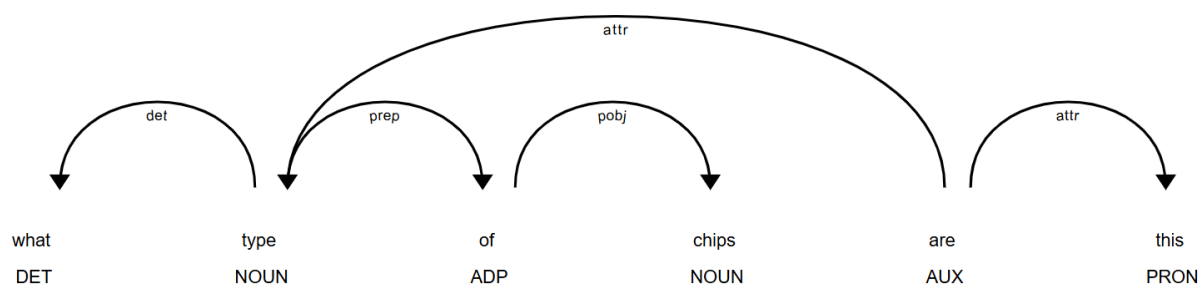
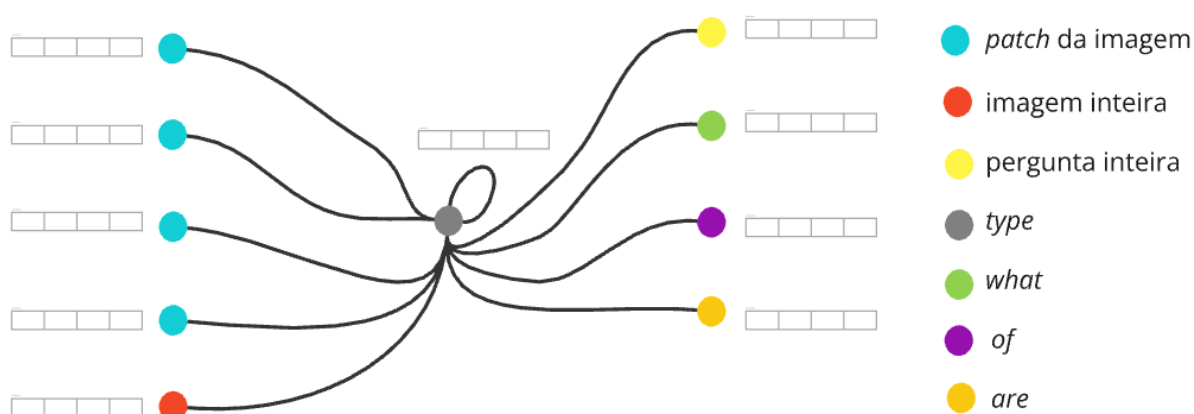


Figura 37 – Exemplo das conexões da pergunta inteira.

objeto-verbo. Para o exemplo apresentado na Figura 31, a relação obtida por meio de *dependency parsing* pode ser visualizada na Figura 38. Cada vértice correspondente a uma palavra estará conectado tanto aos vértices das palavras anteriores e posteriores quanto às suas dependências sintáticas, além de estar interligado a todos os outros vértices. O exemplo da Figura 39 ilustra as conexões para o vértice que representa a palavra *type*. Este vértice está conectado às palavras anterior e posterior, *what* e *of*, respectivamente, e também às suas dependências sintáticas, que incluem os vértices *of* e *are*, bem como a si mesmo e aos demais vértices que representam a imagem, a pergunta e os *patches*.

4.1.4 Obtenção das respostas

Após obter a representação do par imagem/pergunta como um grafo, a geração de respostas é tratada como uma tarefa de classificação, uma abordagem amplamente adotada na literatura (FARAZI; KHAN; BARNES, 2021). Para isso, é necessário definir as

Figura 38 – Exemplo de análise de *dependency parser*.Figura 39 – Exemplo das conexões para o vértice representando a palavra *type*.

classes utilizadas no treinamento do modelo, o que pode ser feito de diferentes maneiras, conforme será discutido na Seção 4.4.

Com as classes definidas, inicia-se o treinamento do modelo, utilizando conjuntos de treinamento e validação. O conjunto de treinamento permite que o modelo aprenda padrões ajustando seus parâmetros para minimizar a função de perda, enquanto o conjunto de validação é usado para ajustar hiperparâmetros e evitar *overfitting*, servindo também como *checkpoint* para avaliar continuamente o desempenho do modelo.

A arquitetura escolhida, GAT, atribui pesos distintos às conexões entre os vértices, permitindo que a rede aprenda a relevância de cada vizinho na agregação de atributos de cada vértice. Essa escolha se deve à capacidade do modelo de capturar relações complexas e contextuais no grafo, ajustando dinamicamente a importância de cada aresta durante o treinamento. Esse ajuste pode contribuir para a interpretabilidade dos resultados, uma vez que possibilita a análise das arestas que tiveram maior peso em uma determinada resposta.

Após a agregação, cuja profundidade é determinada pelo número de camadas da rede, aplica-se a operação de *global_mean_pool* para consolidar as representações dos vértices em um único vetor. Essa operação calcula a média dos *embeddings* dos vértices, resultando em uma representação global do grafo. Por fim, essa representação é processada por uma camada final, que pode ou não ser totalmente conectada, gerando

assim a resposta predita.

4.2 Método 2

O Método 2, fundamentado no trabalho de [Deuser et al. \(2022\)](#), segue a estrutura descrita na Seção 2.2, com a visão geral do processo ilustrada na Figura 40. A Seção 4.2.1 detalha o procedimento para a representação dos dados de entrada, enquanto a Seção 4.2.2 discute a fusão dos atributos extraídas na etapa anterior. Por fim, a Seção 4.2.3 apresenta a implementação do processo de obtenção de respostas.

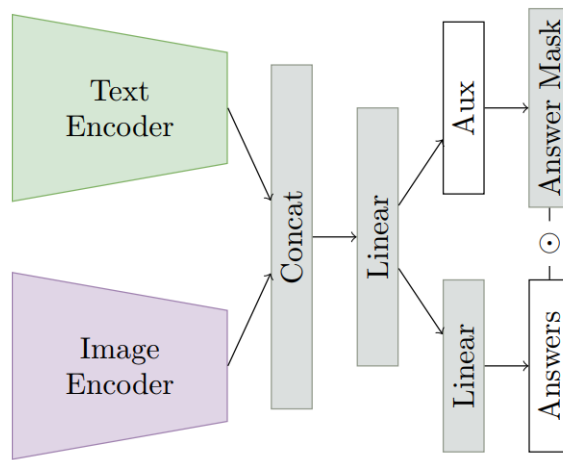


Figura 40 – Exemplo de *pipeline* do Método 2 ([DEUSER et al., 2022](#)).

4.2.1 Extração de atributos

Uma das principais distinções entre o trabalho de [Deuser et al. \(2022\)](#) e o método apresentado neste estudo está nesta etapa. Enquanto [Deuser et al. \(2022\)](#) emprega a técnica de *data augmentation* antes de obter a representação numérica das imagens, neste trabalho optamos por não utilizá-la, visando tornar o modelo computacionalmente mais eficiente. A *data augmentation* é amplamente aplicada em aprendizado de máquina, especialmente na área de visão computacional, com o objetivo de expandir o tamanho e a diversidade de um *dataset* sem a necessidade de coletar novos exemplos. Essa técnica gera variações nos dados existentes por meio de transformações que simulam condições reais, como rotações, ampliações, cortes (*crops*), translações e inversões (*flip*), entre outras.

Neste método, os atributos representativos das imagens são extraídas utilizando modelos pré-treinados, como CLIP, BLIP2 ou ResNet-152. No caso dos modelos CLIP e BLIP2, basta fornecer as imagens, e as representações são obtidas diretamente. Por outro lado, para o modelo ResNet-152, as imagens precisam ser redimensionadas para 224×224 *pixels*, que é a dimensão para a qual a rede foi originalmente treinada. Além disso, é necessário normalizar as imagens para garantir que os valores de entrada estejam

no mesmo intervalo utilizado durante o treinamento, o que contribui para a estabilidade da inferência. A normalização típica aplicada a imagens na ResNet-152 pré-treinada no *ImageNet* considera a média $[0.485, 0.456, 0.406]$ e o desvio padrão $[0.229, 0.224, 0.225]$ do conjunto de treinamento do *ImageNet*.

A representação das perguntas também é realizada por meio de atributos extraídas com modelos pré-treinados, como CLIP, BLIP2 e BERT. Apenas o modelo BERT exige ajustes na entrada, conforme detalhado na Seção 4.1.2, enquanto os demais modelos requerem apenas a pergunta para gerar as representações. Outra diferença significativa entre o método proposto e o trabalho de Deuser et al. (2022) está na escolha dos modelos utilizados. Enquanto Deuser et al. (2022) emprega exclusivamente o modelo CLIP para representar tanto as imagens quanto as perguntas, o método atual adota uma variedade mais ampla de modelos, ampliando a capacidade de capturar representações diversificadas.

4.2.2 Fusão dos atributos

Após a obtenção das representações numéricas para cada par imagem/pergunta, os atributos são combinados por meio de concatenação, conforme ilustrado na Figura 20a. Os modelos CLIP, BLIP2 e BERT geram *embeddings* com dimensão 768, enquanto o ResNet-152 produz representações com dimensão 1000. Contudo, como a fusão é realizada por concatenação, a diferença nas dimensões não representa um problema.

4.2.3 Obtenção das respostas

Após a junção dos atributos na etapa anterior, procede-se à obtenção das respostas por meio de uma classificação. O treinamento será realizado utilizando uma rede MLP, empregando não apenas os conjuntos de treinamento e validação, mas também reservando 5% do conjunto de treinamento para formar um conjunto de teste. Esse conjunto adicional tem como objetivo fornecer uma avaliação imparcial da capacidade de generalização do modelo em relação a novos dados durante o treinamento.

O modelo pode ser configurado para treinar até três tarefas simultaneamente, dependendo da abordagem adotada, conforme será detalhado na Seção 4.4. Nesse cenário, a função de perda a ser minimizada durante o treinamento considerará as tarefas de forma integrada. As tarefas são: (i) obtenção da resposta para a pergunta fornecida (objetivo principal), (ii) identificação do tipo de resposta e (iii) verificação da adequação da imagem para que a pergunta possa ser respondida.

Para a tarefa (i), as classes de resposta do modelo são definidas conforme será especificado na Seção 4.4. Na tarefa (ii), as classes possíveis incluem: *yes/no*, *number*, *other* e *unanswerable*. As previsões geradas para os tipos de resposta são projetadas linearmente em um vetor cuja dimensão corresponde ao número de classes possíveis da

tarefa (i). Após a passagem por uma camada sigmóide, esse vetor é multiplicado pelos *logits* do vocabulário de respostas, conforme ilustrado na Figura 41. Esse processo permite mascarar respostas inadequadas ao tipo de resposta identificado durante a inferência.

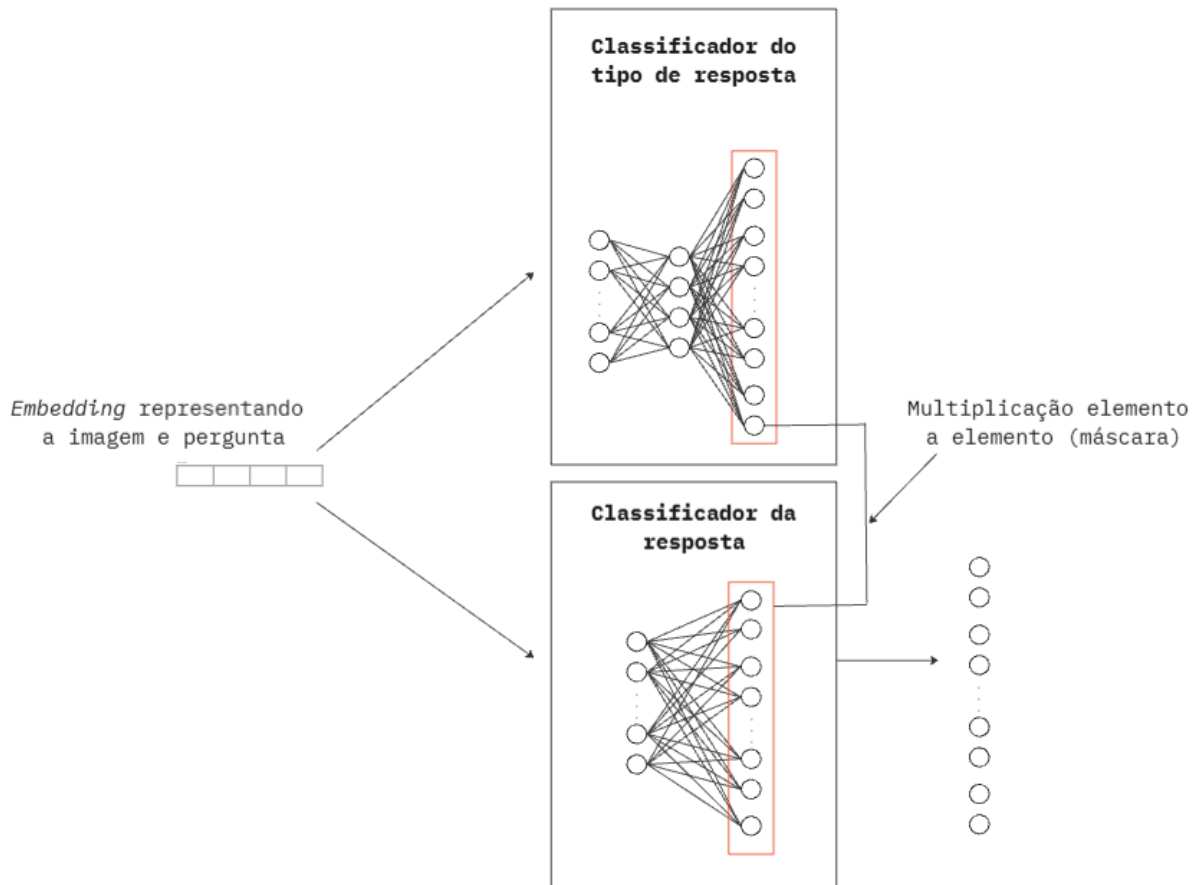


Figura 41 – Exemplo da ação conjunta das tarefas (i) e (ii).

Na tarefa (iii), as classes de treinamento são 0 (*unanswerable*) e 1 (*answerable*). Comparando com a abordagem de Deuser et al. (2022), a tarefa (ii) considerou oito tipos de resposta: *other*, *numbers*, *yes*, *no*, *color*, *unsuitable* e *unanswerable*. Esses tipos foram identificados por meio de expressões regulares, baseando-se na correspondência com a melhor resposta selecionada para cada par imagem/pergunta, enquanto as classes para o Método 2 já estavam definidas no *dataset* utilizado.

4.3 Comparação dos métodos

A principal diferença entre os métodos está nas etapas de extração e fusão de atributos. No Método 1, a fusão dos atributos é realizada por meio de grafos, utilizando uma rede GAT para a obtenção das respostas. Por outro lado, o Método 2 adota a concatenação dos atributos e emprega uma MLP como mecanismo para gerar as respostas.

Além disso, o Método 1 incorpora partes específicas da imagem e da pergunta, com o objetivo de aprimorar a representação de ambos. Em contraste, o Método 2 se destaca ao integrar informações adicionais disponibilizadas no *dataset* de estudo durante a etapa de obtenção das respostas. O Quadro 5 ilustra as principais diferenças entre os métodos propostos.

	Fusão dos atributos	Obtenção da resposta	Utiliza partes da imagem e da pergunta	Utiliza informações sobre tipo de resposta e validação da pergunta
Método 1	Grafos	GAT	x	-
Método 2	Concatenação	MLP	-	x

Quadro 5 – Diferenças entre os métodos 1 e 2.

4.4 Abordagens

Esta seção apresenta as 14 abordagens implementadas com base nos Métodos 1 e 2 descritos anteriormente, incorporando também a técnica de *ensemble learning*. Vale destacar que a construção dessas abordagens envolveu escolhas específicas relacionadas às 13 características descritas a seguir. Essas escolhas foram determinadas com base em testes preliminares, cujo objetivo foi identificar a melhor configuração.

1. **Técnica:** A técnica abordada refere-se à metodologia aplicada para resolver o problema de VQA. Neste contexto, foram utilizados o Método 1, Método 2 e o *ensemble learning*. O *ensemble learning* é uma técnica que combina as previsões de múltiplos modelos com o objetivo de melhorar a precisão e a robustez dos resultados, reduzindo os erros individuais e aumentando a capacidade de generalização do sistema (BROWN, 2017). Duas variantes de *ensemble learning* foram exploradas neste trabalho. A primeira variante (*Ensemble 1*) realiza uma combinação entre duas abordagens, selecionando, para cada par imagem/pergunta do conjunto de teste, a resposta com maior valor de probabilidade entre as abordagens. Na segunda variante (*Ensemble 2*), que combina três abordagens, caso as respostas das três abordagens sejam diferentes, seleciona-se a resposta com a maior probabilidade; caso duas ou mais abordagens apresentem a mesma resposta, essa será a resposta final.
 - **Método 1:** Abordagens 1 a 5;
 - **Método 2:** Abordagens 6 a 10;
 - **Ensemble 1:** Abordagem 11 - combinação das abordagens 6 e 10; Abordagem 12 - combinação das abordagens 6 e 7; Abordagem 13 - combinação das abordagens 7 e 10;
 - **Ensemble 2:** Abordagem 14 - combinação das abordagens 6, 7 e 10.
2. **Extração de atributos da imagem:** Os modelos que fornecem a representação numérica das imagens ou de seus *patches*, conforme o método adotado, incluem viT, *AutoEncoder*, BLIP2, CLIP e Resnet-152.

- **viT**: Abordagens 1 e 2;
 - **AutoEncoder**: Abordagens 3 e 4;
 - **BLIP2**: Abordagens 5 e 7;
 - **CLIP**: Abordagens 6, 8 e 10;
 - **Resnet-152**: Abordagem 9.
3. **Extração de atributos da pergunta**: A extração dos atributos para representar a pergunta e suas palavras é realizada por meio dos modelos BERT, BLIP2 e CLIP.
- **BERT**: Abordagens 1 a 4, 9 e 10;
 - **BLIP2**: Abordagens 5 e 7;
 - **CLIP**: Abordagens 6 e 8.
4. **Normalização dos atributos**: As abordagens 1 e 2 aplicaram a normalização L2 após a extração dos atributos. A normalização L2, também chamada de normalização Euclidiana, ajusta os vetores para que a soma dos quadrados de seus valores seja igual a 1, o que é útil em aprendizado de máquina, pois torna as características dos dados mais comparáveis, reduz a variância entre os atributos e acelera o treinamento dos modelos ([HAAS](#); [YOLLAND](#); [RABUS](#), 2022).
- **Normalização L2**: Abordagens 1 e 2;
 - **Sem normalização**: Abordagens 3 a 10.
5. **Definição das classes**: Em modelos de aprendizado de máquina, uma classe é uma categoria ou rótulo que o modelo tenta prever ou classificar. Ela representa as diferentes saídas ou possíveis categorias do problema em análise, especialmente em tarefas de classificação. Para isso, as classes foram definidas com base em cinco estratégias distintas: (1) 3.000 classes, (2) 3.000 classes com filtro na resposta, (3) 1.000 classes, (4) 5.239 classes e (5) 3.000 classes com filtro na confiança.
- Na estratégia 1, foram selecionadas as 3.000 respostas mais frequentes do conjunto de treinamento. Para a estratégia 2, foi realizada uma filtragem que remove pares de imagem/pergunta cuja resposta mais frequente fosse “*unanswerable*”; em seguida, foram escolhidas as 3.000 respostas mais frequentes. Na estratégia 3, foram selecionadas as 1.000 respostas mais frequentes do conjunto de treinamento.
- Para a estratégia 4, foi aplicada a metodologia de [Deuser et al. \(2022\)](#), em que a resposta mais comum com maior pontuação para cada par imagem/pergunta é escolhida. Se essa seleção resultar em múltiplas respostas, a resposta com maior frequência em todo o conjunto de treinamento é usada. Em caso de empate, utiliza-se a distância de *Levenshtein* entre pares para identificar a resposta mais representativa entre as demais. A distância de *Levenshtein*, ou distância de edição, mede a diferença entre duas sequências de caracteres, como palavras ou frases, quantificando o número mínimo de operações necessárias para transformar uma sequência em outra. Esse processo resultou na definição de 5.239 classes.
- Por fim, na estratégia 5, foram selecionadas as 3.000 respostas mais frequentes do

conjunto de treinamento, considerando apenas aquelas com confiança indicada como *yes* e *maybe*.

- **Estratégia 1:** Abordagens 1 e 3;
- **Estratégia 2:** Abordagem 2;
- **Estratégia 3:** Abordagens 4 e 5;
- **Estratégia 4:** Abordagens 6, 7, 9 e 10;
- **Estratégia 5:** Abordagem 8;

6. **Arquitetura da rede:** A arquitetura de uma rede neural define a estrutura de camadas e conexões entre neurônios, determinando como a informação é processada ao longo da rede. A escolha dessa arquitetura impacta diretamente o desempenho da rede em diferentes tipos de tarefas. No caso do Método 1, as camadas principais utilizadas foram: *GATv2Conv*, *dropout*, *global_mean_pool* e *Linear*. A camada *GATv2Conv*, representando a GAT, possui dois parâmetros relevantes: *out_channels* e *heads*. O parâmetro *out_channels* especifica as dimensões dos atributos de saída, variando entre 200 e 2000 neurônios nessa implementação, enquanto o parâmetro *heads* determina o número de cabeças de atenção, possibilitando *multi-head attention* com valores entre 1 e 8.

A camada *dropout* é uma técnica de regularização aplicada para reduzir o *overfitting* em redes neurais. Durante o treinamento, ela desativa aleatoriamente uma proporção de neurônios em cada camada (geralmente entre 20% e 50%), o que previne a dependência excessiva em neurônios específicos. A camada *global_mean_pool* realiza uma operação de *pooling* usada em GNNs, calculando a média dos vetores de características de todos os nós de um grafo e gerando uma única representação para o grafo inteiro. Já a camada *Linear* realiza uma transformação linear nos dados de entrada, multiplicando-os por uma matriz de pesos e somando um vetor de bias. Formalmente, essa operação é expressa por $y = xW + b$, onde x é o vetor de entrada, W a matriz de pesos, b o vetor de bias, e y a saída. Essa camada permite que a rede aprenda relações complexas entre entradas e saídas ao preparar os dados para as camadas subsequentes.

Para o Método 2, as camadas utilizadas foram *LayerNorm*, *dropout* e *Linear*. A camada *LayerNorm* (*Layer Normalization*) normaliza as ativações de cada camada em relação à média e ao desvio padrão de cada exemplo individual, o que melhora a estabilidade e o desempenho durante o treinamento. Neste caso, o *dropout* foi configurado para 0.5. Os trechos de código que definem as arquiteturas usadas estão descritos no Apêndice B.

7. **Otimizador:** O otimizador é um algoritmo responsável por ajustar os pesos dos neurônios com o objetivo de minimizar o erro entre as previsões da rede e os valores reais. Entre os otimizadores mais utilizados, destacam-se o *Adam* e o *SGD*, que também são empregados neste trabalho.

- **Adam:** Abordagens 1, 2 e 6 a 10;
 - **SGD:** Abordagens 3 a 5.
8. **Taxa de aprendizagem:** A taxa de aprendizagem (*learning rate*, lr) é um hiperparâmetro fundamental em redes neurais, responsável por controlar a amplitude dos ajustes nos pesos do modelo durante o processo de treinamento. No presente estudo, as taxas de aprendizagem adotadas variam de 1^{-5} a 0.005.
- 0.005 : Abordagens 1 e 2;
 - 1^{-5} : Abordagens 3 a 5;
 - 5^{-4} : Abordagens 6 a 10.
9. **Função de perda:** A função de perda, ou *loss function*, é uma métrica fundamental para avaliar o desempenho de um modelo, calculando a diferença entre as previsões do modelo e os valores reais do *dataset*. Esse valor representa uma “penalidade” que orienta o modelo a ajustar seus parâmetros durante o treinamento, buscando minimizar essa diferença e, assim, aprimorar sua precisão. Um valor mais baixo da função de perda indica que o modelo está se ajustando melhor aos dados. Neste trabalho, foram utilizadas a *Cross-Entropy Loss* e a *Soft Cross-Entropy Loss*, sendo esta última uma variação da *Cross-Entropy Loss* que, ao invés de utilizar rótulos binários fixos (como 0 e 1), permite o uso de rótulos em formato de distribuições de probabilidade. Dessa forma, a *Soft Cross-Entropy Loss* possibilita uma “suavização” dos rótulos, permitindo que o modelo produza respostas menos rígidas.
- **Soft Cross-Entropy Loss:** Abordagens 1 e 2;
 - **Cross-Entropy Loss:** Abordagens 3 a 10.
10. **Pesos na função de perda:** A utilização de pesos na função de perda possibilita atribuir maior relevância a determinados exemplos, contribuindo para que o modelo alcance um aprendizado mais equilibrado. Essa abordagem é particularmente vantajosa em *datasets* desbalanceados, nos quais algumas classes possuem maior representatividade. Dessa forma, assegura-se que o modelo dedique atenção suficiente às classes menos frequentes. Os pesos adotados nas abordagens foram determinados com base na proporcionalidade de cada classe dentro do *dataset*.
- **Função de perda com pesos:** Abordagens 1, 2 e 3;
 - **Função de perda sem pesos:** Abordagens 4 a 10.
11. **Tarefas resolvidas:** Essas tarefas referem-se àquelas descritas na Seção 4.2.3, associadas ao Método 2: (i) obtenção da resposta para a pergunta fornecida (objetivo principal), (ii) identificação do tipo de resposta e (iii) verificação da adequação da imagem para que a pergunta possa ser respondida. A função de perda incorpora essas tarefas de maneira integrada, somando as contribuições de cada uma. Para fins de simplificação, a tarefa (i) será rotulada como VQA na Tabela 3, a tarefa (ii) como *answer_type* e a tarefa (iii) como *answerability*.
- **vqa, answer_type:** Abordagem 8;

- **vqa, answer_type, answerability:** Abordagens 6, 7, 9 e 10.
12. **Tamanho do *batch*:** O *batch* refere-se a um conjunto de exemplos de dados processados simultaneamente em uma única passada pela rede neural. Neste estudo, o tamanho do *batch* utilizado nos conjuntos de treinamento e validação varia entre 16 e 100.
- ***batch* de treinamento**
 - **16:** Abordagem 2;
 - **32:** Abordagens 1 e 6 a 10;
 - **64:** Abordagem 3;
 - **100:** Abordagens 4 e 5.
 - ***batch* de validação**
 - **16:** Abordagem 2;
 - **32:** Abordagens 1 e 3 a 10.
13. **Épocas de treinamento:** A quantidade de épocas corresponde a uma iteração completa sobre todo o conjunto de treinamento. Em cada época, o modelo processa todos os exemplos de entrada e ajusta seus parâmetros (pesos) com base nos erros calculados, com o objetivo de minimizar a função de perda. Neste trabalho, o número de épocas variou entre 50 e 200.
- **50:** Abordagens 6 a 10;
 - **200:** Abordagens 1 a 5.

A Tabela 3 sintetiza os elementos empregados em cada abordagem.

Abord.	Técnica	Rep. da pergunta	Rep. da imagem	Norm. dos atributos	Classes	Otim.	lr	Função de perda	Pesos na função de perda	Tarefas resolvidas	batch treino	batch valida.	Épocas
1	Método 1	BERT	viT	Sim	3.000	Adam	0.005	SoftCrossEntropyLoss	Sim	NA	32	32	200
2	Método 1	BERT	viT	Sim	3.000	Adam	0.005	SoftCrossEntropyLoss	Sim	NA	16	16	200
3	Método 1	BERT	AutoEncoder	Não	3.000	SGD	1e-5	CrossEntropyLoss	Sim	NA	64	32	200
4	Método 1	BERT	AutoEncoder	Não	1.000	SGD	1e-5	CrossEntropyLoss	Não	NA	100	32	200
5	Método 1	BLIP2	BLIP2	Não	1.000	SGD	1e-5	CrossEntropyLoss	Não	NA	100	32	200
6	Método 2	CLIP	CLIP	Não	5.239	Adam	5e-4	CrossEntropyLoss	Não	VQA, answer_type, answerability	32	32	50
7	Método 2	BLIP2	BLIP2	Não	5.239	Adam	5e-4	CrossEntropyLoss	Não	VQA, answer_type, answerability	32	32	50
8	Método 2	CLIP	CLIP	Não	3.000	Adam	5e-4	CrossEntropyLoss	Não	VQA, answer_type	32	32	50
9	Método 2	BERT	Resnet-152	Não	5.239	Adam	5e-4	CrossEntropyLoss	Não	VQA, answer_type, answerability	32	32	50
10	Método 2	BERT	CLIP	Não	5.239	Adam	5e-4	CrossEntropyLoss	Não	VQA, answer_type, answerability	32	32	50
11	Ensemble learning (Abordagens 6 e 10)	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
12	Ensemble learning (Abordagens 6 e 7)	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
13	Ensemble learning (Abordagens 7 e 10)	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
14	Ensemble learning (Abordagens 6, 7 e 10)	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA

Tabela 3 – Composição dos elementos utilizados nas abordagens.

5 Resultados

Este capítulo apresenta os resultados dos experimentos computacionais realizados para avaliar o desempenho das abordagens propostas. A Seção 5.1 descreve a metodologia utilizada nos experimentos, detalhando o *dataset* empregado, a métrica utilizada para medir o desempenho das abordagens e o local de submissão dos resultados. Por fim, a Seção 5.2 apresenta os resultados obtidos, acompanhados de uma discussão que compara esses resultados com os encontrados na literatura.

5.1 Metodologia dos Experimentos

Para a avaliação das abordagens propostas na Seção 4.4, foram realizados experimentos utilizando o *dataset* VizWiz¹. O VizWiz foi escolhido por seu papel social em auxiliar no desenvolvimento de tecnologias assistivas voltadas para pessoas com deficiência visual.

Os experimentos foram conduzidos com a partição *test-standard* (test-std), pois ela está disponível para suporte à avaliação de algoritmos durante todo o ano. As anotações do VizWiz-std estão organizadas em três arquivos JSON: *train.json*, *val.json* e *test.json*, cujo formato é ilustrado na Figura 42. Além disso, a imagem correspondente ao exemplo descrito na Figura 42 é apresentada na Figura 43.

```
{'image': 'VizWiz_train_00000000.jpg',
 'question': "What's the name of this product?",
 'answers': [{'answer_confidence': 'yes', 'answer': 'basil leaves'},
 {'answer_confidence': 'yes', 'answer': 'basil leaves'},
 {'answer_confidence': 'yes', 'answer': 'basil'},
 {'answer_confidence': 'yes', 'answer': 'basil'},
 {'answer_confidence': 'yes', 'answer': 'basil leaves'},
 {'answer_confidence': 'yes', 'answer': 'basil leaves'},
 {'answer_confidence': 'yes', 'answer': 'basil leaves'},
 {'answer_confidence': 'yes', 'answer': 'basil leaves'},
 {'answer_confidence': 'yes', 'answer': 'basil leaves'},
 {'answer_confidence': 'yes', 'answer': 'basil'}],
 'answer_type': 'other',
 'answerable': 1}
```

Figura 42 – Exemplo de anotação para um par imagem/pergunta do VizWiz-std.

As anotações incluem cinco campos, conforme ilustrado na Figura 42, sendo eles:

- *image*: Armazena o nome da imagem;
- *question*: Contém a pergunta correspondente à imagem;
- *answers*: Armazena as informações obtidas na curadoria realizada, fornecendo 10 respostas e o grau de confiança para cada uma, detalhadas a seguir:
 - *answer*: Resposta atribuída por um curador ao par imagem/pergunta fornecido;

¹ O *dataset* está disponível no link: <https://vizwiz.org/tasks-and-datasets/vqa/>



Figura 43 – Imagem *VizWiz_train_00000000.jpg* da base *VizWiz-std*.

- *answer_confidence*: Grau de confiança que o curador teve ao atribuir à resposta, podendo ser *yes*, *no* ou *maybe*;
- *answer_type*: Indica a categoria da resposta, podendo ser:
 - *number*: Para respostas numéricas;
 - *yes/no*: Para respostas binárias;
 - *unanswerable*: Quando a resposta predominante é *unanswerable*, o que ocorre quando não é possível responder à pergunta devido à baixa qualidade da imagem ou outros fatores;
 - *other*: Para outros tipos de resposta, que não cabem em uma categoria específica.
- *answerable*: Indica se a pergunta é respondível ou não.

A avaliação dos métodos propostos que utilizam o VizWiz é comumente realizada utilizando o conjunto de teste disponibilizado. Entretanto, esse conjunto não inclui as respostas fornecidas pelos curadores. Portanto, para avaliar a performance, é necessário inscrever-se no *VizWiz VQA Challenge*², lançado anualmente.

Uma limitação relacionada a essa avaliação é que cada equipe pode submeter no máximo cinco arquivos de resultados, com o limite de um resultado por dia. A métrica fornecida no desafio é a acurácia geral (*overall accuracy*), que corresponde à média da acurácia por imagem, conforme apresentado na Equação 2. Além disso, o desafio também disponibiliza as acurácias segmentadas por tipo de resposta: *number*, *yes/no*, *unanswerable* e *other*, permitindo uma avaliação mais detalhada das capacidades dos métodos desenvolvidos.

Os experimentos foram realizados em um computador equipado com o sistema ope-

² Link do desafio de 2024: <https://eval.ai/web/challenges/challenge-page/2185/leaderboard/5394>

racional Ubuntu 20.04.6 LTS, processador Intel® Xeon(R) E-2174G CPU @ 3.80GHz × 8, 125,6 GB de memória RAM e GPU NVIDIA Corporation GP106GL [Quadro P2000] / Quadro P2000/PCIe/SSE2. As implementações foram desenvolvidas na linguagem *Python*, escolhida por sua ampla gama de bibliotecas que facilitam a construção de modelos de *machine learning*, como o *framework PyTorch*, empregado neste trabalho, além de outras bibliotecas detalhadas no Apêndice C. Para garantir a reprodutibilidade dos experimentos, os códigos utilizados estão disponíveis no seguinte repositório: <https://github.com/araceli252/VQA>.

5.2 Resultados Experimentais e Comparações

Os resultados obtidos pelas abordagens propostas foram comparados com os trabalhos do estado da arte, destacando-se aqueles apresentados por Gurari et al. (2018), Kim et al. (2018), Singh et al. (2019), Huang et al. (2021) e Deuser et al. (2022). Alguns desses estudos, como os de Gurari et al. (2018) e Deuser et al. (2022), reportaram resultados para diversos métodos. Para a análise comparativa realizada neste trabalho, foram selecionados os métodos que apresentaram os melhores desempenhos em cada estudo.

Os resultados das abordagens propostas, submetidas ao *VizWiz VQA Challenge*, estão apresentados na Tabela 4, que também inclui os resultados reportados na literatura avaliada. Ao analisar os resultados, observa-se que as abordagens de 1 a 5 apresentaram desempenhos idênticos. Todas essas abordagens empregaram o método 1 em comum e utilizaram uma quantidade de classes menor em comparação às abordagens que adotaram o método 2, o que pode ter contribuído para seus desempenhos inferiores. Além disso, diferenças na configuração de outros elementos não tiveram impacto significativo nos resultados. Apesar disso, essas abordagens se destacaram na categoria *unanswerable*, acertando todas as respostas desta categoria. Também, na categoria *number*, apresentaram resultados superiores aos reportados por Kim et al. (2018) e Gurari et al. (2018). Entre as abordagens de 6 a 10, que utilizam o método 2, a abordagem 6 obteve o melhor desempenho na maioria das categorias. Por outro lado, as abordagens de 11 a 14, que aplicam *ensemble learning*, mostraram resultados bem próximos, com cada uma delas se destacando em uma categoria específica.

Considerando os resultados por tipo de resposta, as abordagens de 6 a 14 superaram o trabalho de Huang et al. (2021), que apresentava o melhor desempenho na literatura para a categoria *other*. Na categoria *unanswerable*, todas as abordagens propostas superaram o resultado de Kim et al. (2018), anteriormente o melhor da literatura. Para a categoria *number*, as abordagens 6 e de 8 a 14 também superaram os resultados de Huang et al. (2021), e todas as abordagens apresentaram desempenho superior aos trabalhos de Kim et al. (2018) e Gurari et al. (2018).

Na categoria *yes/no*, nenhuma das abordagens propostas conseguiu superar os

Metodologia	overall	other	unanswerable	yes/no	number
Abordagem 1	46,19	25,95	100,00	35,16	23,58
Abordagem 2	46,19	25,95	100,00	35,16	23,58
Abordagem 3	46,19	25,95	100,00	35,16	23,58
Abordagem 4	46,19	25,95	100,00	35,16	23,58
Abordagem 5	46,19	25,95	100,00	35,16	23,58
Abordagem 6	60,70	47,71	94,68	61,98	32,11
Abordagem 7	57,24	43,86	92,17	59,86	25,28
Abordagem 8	56,11	44,97	86,30	51,46	29,67
Abordagem 9	54,33	38,19	94,18	66,79	29,11
Abordagem 10	60,49	47,50	93,27	70,29	29,19
Abordagem 11	61,43	48,56	94,71	65,67	31,79
Abordagem 12	61,50	48,45	95,53	63,04	34,47
Abordagem 13	61,64	48,77	94,68	68,14	29,67
Abordagem 14	62,07	49,13	95,43	67,02	31,63
Deuser et al. (2022)	60,15	-	-	-	-
Singh et al. (2019)	54,72	-	-	-	-
Huang et al. (2021)	53,40	35,40	82,60	73,40	26,70
Kim et al. (2018)	52,00	31,60	86,20	69,10	19,10
Gurari et al. (2018)	47,50	29,40	77,60	66,90	22,00

Tabela 4 – Comparação dos resultados das abordagens com a literatura.

resultados da literatura. No entanto, a abordagem 10 apresentou desempenho próximo ao de Huang et al. (2021), cujo uso do detector de objetos *Faster R-CNN* parece ter sido decisivo para o desempenho superior, considerando que respostas *yes/no* frequentemente se referem à presença ou ausência de objetos na cena. É relevante destacar que a abordagem 10 superou os resultados de Kim et al. (2018) e Gurari et al. (2018), enquanto as abordagens 13 e 14 superaram os resultados de Gurari et al. (2018).

Na categoria *overall*, as abordagens 6 e de 10 a 14 superaram os resultados da literatura, representada por Deuser et al. (2022). Um ponto interessante é que a abordagem 6 se diferencia do método de Deuser et al. (2022) apenas pelo uso de *data augmentation*, evidenciando que a não utilização dessa técnica foi benéfica para os resultados.

De maneira geral, a abordagem 14 foi a que apresentou o melhor desempenho. Essa abordagem utilizou um *ensemble* formado pelas abordagens 6, 7 e 10, que foram treinadas com uma quantidade maior de classes. As diferenças entre as abordagens consistem apenas na representação da imagem e da pergunta, os demais elementos foram os mesmos utilizados no trabalho de Deuser et al. (2022)³. Conforme destacado por

³ A implementação está disponível em <https://www.kaggle.com/code/yousefkotp/visual-question-answering/notebook>.

Deuser et al. (2022) e Anderson et al. (2018), o uso de técnicas de *ensemble learning* pode melhorar a acurácia, principalmente devido à combinação das capacidades dos modelos de representação e à maior diversidade de classes aprendidas. Essa estratégia permitiu que a abordagem 14 superasse inclusive o melhor resultado encontrado na literatura, também referente a uma abordagem de Deuser et al. (2022) baseada em *ensemble learning*.

6 Considerações Finais

Visual Question Answering (VQA) é um problema multimodal que combina duas áreas fundamentais da inteligência artificial: visão computacional e processamento de linguagem natural. Seu objetivo é gerar respostas para perguntas formuladas sobre imagens. Neste trabalho, foi realizada uma ampla revisão da literatura de VQA, abrangendo artigos publicados desde 2014, quando surgiram os primeiros conjuntos de dados dedicados a esse problema, até os dias atuais. Essa revisão discute tanto os principais desafios da área quanto os *datasets* e métricas mais utilizados. Observou-se que abordagens clássicas baseadas em redes neurais, como CNN e LSTM, são frequentemente empregadas na solução do problema. No entanto, métodos mais recentes exploram modelos avançados, como BLIP, BAN e GPT, além de arquiteturas que utilizam redes neurais para grafos.

Além da revisão da literatura, este trabalho propõe dois métodos para a solução do problema, seguindo as principais etapas descritas na literatura. O primeiro método realiza a fusão dos atributos da imagem, da pergunta e de suas partes por meio de grafos, utilizando a rede neural para grafos GAT para a obtenção das respostas. O segundo método, por sua vez, emprega a fusão dos atributos via concatenação, com as respostas sendo geradas por uma rede neural clássica, MLP.

Com base nesses métodos, foram desenvolvidas quatorze abordagens distintas, diferenciadas por aspectos de implementação, como a quantidade de classes utilizadas e a forma de representação dos atributos de imagem e pergunta. No entanto, devido às limitações de memória da máquina utilizada, não foi possível explorar modelos mais robustos, como PaLI e BEiT-3, que possuem altos requisitos computacionais. Os experimentos foram conduzidos com o *dataset* VizWiz, e os resultados foram comparados com cinco trabalhos da literatura, utilizando a métrica de acurácia para avaliação.

A maioria das abordagens propostas superou os resultados da literatura analisada em quase todas as categorias avaliadas (*overall*, *number*, *yes/no*, *other* e *unanswerable*). A abordagem de maior destaque foi a 14, que utilizou *ensemble learning* para combinar os resultados de outras abordagens, atingindo um *overall* de 62,07, enquanto o maior valor reportado na literatura para essa categoria era 60,15. Como observado no trabalho de [Deuser et al. \(2022\)](#) e em outros estudos, a utilização da técnica de *ensemble learning* se mostrou vantajosa, pois permite combinar as capacidades de diferentes modelos, resultando em um desempenho superior. Destaca-se, ainda, que mesmo as abordagens mais simples baseadas no método 1 apresentaram resultados competitivos, especialmente nas categorias *number* e *unanswerable*.

Com base nos estudos realizados e nos resultados obtidos, propõem-se as seguintes

direções para trabalhos futuros:

- Expandir os experimentos das abordagens propostas, avaliando-as em outros *data-sets*;
- Modificar as abordagens de 1 a 5, explorando ajustes na construção dos grafos, na definição das classes, entre outros aspectos;
- Investigar e implementar técnicas que aprimorem a interpretabilidade dos resultados obtidos;
- Construir um *dataset* VQA em português.

Referências

ABACHA, A. B. et al. Vqa-med: Overview of the medical visual question answering task at imageclef 2019. In: 9-12 SEPTEMBER 2019. **Proceedings of CLEF (Conference and Labs of the Evaluation Forum) 2019 Working Notes**. [S.l.], 2019. Citado 3 vezes nas páginas 4, 42 e 44.

AGRAWAL, A. et al. Don't just assume; look and answer: Overcoming priors for visual question answering. In: IEEE. **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.], 2018. p. 4971–4980. Citado 3 vezes nas páginas 10, 41 e 50.

AGRAWAL, A. et al. Vqa: Visual question answering. **International Journal of Computer Vision**, Springer, v. 123, n. 1, p. 4–32, 2017. Citado 9 vezes nas páginas v, 4, 7, 8, 9, 22, 40, 41 e 50.

ANDERSON, P. et al. Bottom-up and top-down attention for image captioning and visual question answering. In: INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS (IEEE). **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.], 2018. p. 6077–6086. Citado na página 72.

AUER, S. et al. Dbpedia: A nucleus for a web of open data. **Lecture Notes in Computer Science**, Springer, v. 4825, p. 722–735, 2007. Citado na página 18.

AYYUBI, H. A. et al. Generating rationales in visual question answering. **arXiv e-prints**, p. arXiv–2004, 2020. Citado na página 24.

BANK, D.; KOENIGSTEIN, N.; GIRYES, R. Autoencoders. **Machine Learning for Data Science Handbook: Data Mining and Knowledge Discovery Handbook**, Springer, p. 353–374, 2023. Citado 3 vezes nas páginas vi, 33 e 34.

BAO, H. et al. Vlmo: Unified vision-language pre-training with mixture-of-modality-experts. **Advances in Neural Information Processing Systems**, v. 35, p. 32897–32912, 2022. Citado na página 21.

BAZI, Y. et al. Vision–language model for visual question answering in medical imagery. **Bioengineering (Basel, Switzerland)**, v. 10, n. 3, 2023. Citado na página 26.

BEN-YOUNES, H. et al. Mutan: Multimodal tucker fusion for visual question answering. In: IEEE. **2017 IEEE International Conference on Computer Vision (ICCV)**. [S.l.], 2017. p. 2631–2639. Citado na página 21.

BROWN, G. Ensemble learning. In: SAMMUT, CLAUDE AND WEBB, GEOFFREY I. **Encyclopedia of Machine Learning and Data Mining**. Boston, MA: Springer US, 2017. p. 393–402. ISBN 978-1-4899-7687-1. Disponível em: <https://doi.org/10.1007/978-1-4899-7687-1_252>. Citado na página 62.

CAO, R.; JIANG, J. Modularized zero-shot vqa with pre-trained models. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. **Findings of the Association for Computational Linguistics: ACL 2023**. [S.l.], 2023. p. 58–76. Citado 3 vezes nas páginas 46, 47 e 50.

CAO, Y. T. et al. What's different between visual question answering for machine "understanding" versus for accessibility? In: **Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing**. [S.l.: s.n.], 2022. p. 1025–1034. Citado na página 45.

CASCANTE-BONILLA, P. et al. Simvqa: Exploring simulated environments for visual question answering. In: IEEE. **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.], 2022. p. 5056–5066. Citado 2 vezes nas páginas 44 e 50.

CHANGPINYO, S. et al. All you may need for vqa are image captions. In: **Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies**. [S.l.: s.n.], 2022. p. 1947–1963. Citado 2 vezes nas páginas 44 e 50.

CHEN, K. et al. Abc-cnn: An attention based convolutional neural network for visual question answering. **arXiv e-prints**, p. arXiv–1511, 2015. Citado 2 vezes nas páginas 40 e 50.

CHEN, L. et al. Counterfactual samples synthesizing for robust visual question answering. In: IEEE. **2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.], 2020. p. 10797–10806. Citado na página 21.

CHEN, X. et al. Pali-x: On scaling up a multilingual vision and language model. **arXiv e-prints**, p. arXiv–2305, 2023. Citado na página 21.

CHEN, X. et al. Pali: A jointly-scaled multilingual language-image model. In: **The Eleventh International Conference on Learning Representations**. [S.l.: s.n.], 2022. Citado na página 21.

CHEN, X. et al. Chatffa: interactive visual question answering on fundus fluorescein angiography image using chatgpt. **Available at SSRN 4578568**, 2023. Citado na página 24.

CLARK, C.; YATSKAR, M.; ZETTLEMOYER, L. Don't take the easy way out: Ensemble based methods for avoiding known dataset biases. In: **Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)**. [S.l.: s.n.], 2019. p. 4069–4082. Citado na página 21.

CORLEY, I. et al. Revisiting pre-trained remote sensing model benchmarks: resizing and normalization matters. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2024. p. 3162–3172. Citado na página 53.

COUPRIE, C. et al. Indoor semantic segmentation using depth information. In: **First International Conference on Learning Representations (ICLR 2013)**. [S.l.: s.n.], 2013. p. 1–8. Citado na página 15.

DANCETTE, C. et al. Beyond question-based biases: Assessing multimodal shortcut learning in visual question answering. In: IEEE. **Proceedings of the IEEE/CVF International Conference on Computer Vision**. [S.l.], 2021. p. 1574–1583. Citado 2 vezes nas páginas 44 e 50.

- DEUSER, F. et al. Less is more: Linear layers on clip features as powerful vizwiz model. **arXiv e-prints**, p. arXiv–2206, 2022. Citado 12 vezes nas páginas [vi](#), [21](#), [46](#), [50](#), [59](#), [60](#), [61](#), [63](#), [70](#), [71](#), [72](#) e [73](#).
- DO, T. et al. Compact trilinear interaction for visual question answering. In: IEEE COMPUTER SOCIETY. **2019 IEEE/CVF International Conference on Computer Vision (ICCV)**. [S.l.], 2019. p. 392–401. Citado na página [21](#).
- DOSOVITSKIY, A. An image is worth 16x16 words: Transformers for image recognition at scale. **arXiv preprint arXiv:2010.11929**, 2020. Citado 3 vezes nas páginas [37](#), [38](#) e [39](#).
- DRIESS, D. et al. Palm-e: An embodied multimodal language model. In: PMLR. **International Conference on Machine Learning**. [S.l.], 2023. p. 8469–8488. Citado na página [21](#).
- FARAZI, M.; KHAN, S.; BARNES, N. Accuracy vs. complexity: a trade-off in visual question answering models. **Pattern Recognition**, Elsevier, v. 120, p. 108106, 2021. Citado na página [57](#).
- FUKUI, A. et al. Multimodal compact bilinear pooling for visual question answering and visual grounding. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. **Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing**. [S.l.], 2016. p. 457–468. Citado na página [21](#).
- GANJU, S.; RUSSAKOVSKY, O.; GUPTA, A. What’s in a question: Using visual questions as a form of supervision. In: IEEE COMPUTER SOCIETY. **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.], 2017. p. 241–250. Citado na página [24](#).
- GAO, P. et al. Question-guided hybrid convolution for visual question answering. **Computer Vision–ECCV 2018**, Springer International Publishing, p. 485–501, 2018. Citado na página [21](#).
- GAT, I. et al. Removing bias in multi-modal classifiers: Regularization by maximizing functional entropies. **Advances in Neural Information Processing Systems**, v. 33, p. 3197–3208, 2020. Citado na página [21](#).
- GILMER, J. et al. Neural message passing for quantum chemistry. In: PMLR. **International conference on machine learning**. [S.l.], 2017. p. 1263–1272. Citado na página [35](#).
- GOYAL, Y. et al. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In: IEEE COMPUTER SOCIETY. **2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.], 2017. p. 6325–6334. Citado 3 vezes nas páginas [10](#), [41](#) e [50](#).
- GUO, J. et al. From images to textual prompts: Zero-shot visual question answering with frozen large language models. In: IEEE COMPUTER SOCIETY. **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.], 2023. p. 10867–10877. Citado 2 vezes nas páginas [47](#) e [50](#).
- GURARI, D. et al. Vizwiz grand challenge: Answering visual questions from blind people. In: IEEE. **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.], 2018. p. 3608–3617. Citado 6 vezes nas páginas [4](#), [11](#), [21](#), [41](#), [70](#) e [71](#).

- HAAS, J.; YOLLAND, W.; RABUS, B. Linking neural collapse and l2 normalization with improved out-of-distribution detection in deep neural networks. **arXiv e-prints**, p. arXiv–2209, 2022. Citado na página 63.
- HAMILTON, W.; YING, Z.; LESKOVEC, J. Inductive representation learning on large graphs. **Advances in neural information processing systems**, v. 30, 2017. Citado na página 36.
- HAMILTON, W. L. **Graph representation learning**. [S.l.]: Morgan & Claypool Publishers, 2020. Citado na página 35.
- HAN, X. et al. Greedy gradient ensemble for robust visual question answering. In: IEEE. **Proceedings of the IEEE/CVF international conference on computer vision**. [S.l.], 2021. p. 1584–1593. Citado na página 21.
- HAYKIN, S. **Redes neurais: princípios e prática**. [S.l.]: Bookman Editora, 2001. Citado 2 vezes nas páginas 28 e 29.
- HE, J. et al. Pefomed: Parameter efficient fine-tuning on multimodal large language models for medical visual question answering. **arXiv e-prints**, p. arXiv–2401, 2024. Citado na página 21.
- HE, K. et al. Deep residual learning for image recognition. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2016. p. 770–778. Citado 2 vezes nas páginas 36 e 37.
- HIROTA, Y.; NAKASHIMA, Y.; GARCIA, N. Gender and racial bias in visual question answering datasets. **Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency**, p. 1280–1292, 2022. Citado na página 4.
- HU, R. et al. Language-conditioned graph networks for relational reasoning. In: IEEE COMPUTER SOCIETY. **2019 IEEE/CVF International Conference on Computer Vision (ICCV)**. [S.l.], 2019. p. 10293–10302. Citado na página 21.
- HU, R. et al. Modeling relationships in referential expressions with compositional modular networks. In: IEEE. **2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.], 2017. p. 4418–4427. Citado na página 21.
- HU, X. et al. Expert knowledge-aware image difference graph representation learning for difference-aware medical visual question answering. In: **Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining**. [S.l.: s.n.], 2023. p. 4156–4165. Citado na página 24.
- HU, X. et al. Interpretable medical image visual question answering via multi-modal relationship graph learning. **Medical Image Analysis**, Elsevier, v. 97, p. 103279, 2024. Citado 3 vezes nas páginas 3, 47 e 50.
- HU, Y. et al. Promptcap: Prompt-guided image captioning for vqa with gpt-3. In: IEEE COMPUTER SOCIETY. **Proceedings of the IEEE/CVF International Conference on Computer Vision**. [S.l.], 2023. p. 2963–2975. Citado na página 24.
- HU, Y. et al. Visual program distillation: Distilling tools and programmatic reasoning into vision-language models. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2024. p. 9590–9601. Citado na página 21.

- HUANG, H. et al. Answer-checking in context: A multi-modal fully attention network for visual question answering. In: IEEE. **2020 25th International Conference on Pattern Recognition (ICPR)**. [S.l.], 2021. p. 1173–1180. Citado 3 vezes nas páginas 21, 70 e 71.
- HUANG, P. et al. Multi-grained attention with object-level grounding for visual question answering. In: **Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics**. [S.l.: s.n.], 2019. p. 3595–3600. Citado na página 26.
- HUDSON, D.; MANNING, C. D. Learning by abstraction: The neural state machine. **Advances in neural information processing systems**, v. 32, 2019. Citado na página 21.
- HUDSON, D. A.; MANNING, C. D. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In: IEEE. **Proceedings of the IEEE/CVF conference on computer vision and pattern recognition**. [S.l.], 2019. p. 6700–6709. Citado 2 vezes nas páginas 12 e 24.
- ISHMAM, M. F.; SHOYON, M. S. H.; DEY, N. From image to language: A critical analysis of visual question answering (vqa) approaches, challenges, and opportunities. **Information Fusion**, v. 106, p. 102–270, 2024. ISSN 1566-2535. Citado na página 9.
- JHA, S. et al. A novel approach on visual question answering by parameter prediction using faster region based convolutional neural network. **International Journal of Interactive Multimedia and Artificial Intelligence**, v. 5, 2019. ISSN 19891660. Citado na página 26.
- JIANG, H. et al. In defense of grid features for visual question answering. In: IEEE. **2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.], 2020. p. 10264–10273. Citado na página 24.
- JIN, W. et al. A good prompt is worth millions of parameters: Low-resource prompt-based learning for vision-language models. In: **Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**. [S.l.: s.n.], 2022. p. 2763–2775. Citado na página 24.
- JOHNSON, J. et al. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In: IEEE. **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.], 2017. p. 2901–2910. Citado na página 14.
- KAFLE, K.; KANAN, C. Answer-type prediction for visual question answering. In: IEEE. **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.], 2016. p. 4976–4984. Citado 2 vezes nas páginas 21 e 23.
- KAFLE, K.; KANAN, C. Visual question answering: Datasets, algorithms, and future challenges. **Computer Vision and Image Understanding**, Elsevier, v. 163, p. 3–20, 2017. Citado 8 vezes nas páginas 1, 3, 4, 5, 9, 22, 40 e 50.
- KAFLE, K.; SHRESTHA, R.; KANAN, C. Challenges and prospects in vision and language research. **Frontiers in Artificial Intelligence**, v. 2, p. 28–28, 2019. Citado 3 vezes nas páginas 4, 5 e 50.
- KAMATH, A. et al. Mdetr-modulated detection for end-to-end multi-modal understanding. In: IEEE. **Proceedings of the IEEE/CVF International Conference on Computer Vision**. [S.l.], 2021. p. 1780–1790. Citado na página 21.

- KENTON, J. D. M.-W. C.; TOUTANOVA, L. K. Bert: Pre-training of deep bidirectional transformers for language understanding. In: MINNEAPOLIS, MINNESOTA. **Proceedings of naacL-HLT**. [S.l.], 2019. v. 1, p. 2. Citado 2 vezes nas páginas 30 e 31.
- KIM, J.-H. et al. Bilinear attention networks for vizwiz challenge. In: **Proceedings of the ECCV Workshop on VizWiz Grand challenge**. [S.l.: s.n.], 2018. Citado 3 vezes nas páginas 21, 70 e 71.
- KIM, J.-H. et al. Hadamard product for low-rank bilinear pooling. In: INTERNATIONAL CONFERENCE ON LEARNING REPRESENTATIONS, ICLR. **5th International Conference on Learning Representations**. [S.l.], 2016. Citado na página 21.
- KIPF, T. N.; WELLING, M. Semi-supervised classification with graph convolutional networks. **arXiv e-prints**, p. arXiv-1609, 2016. Citado na página 36.
- KOSHTI, D.; GUPTA, A.; KALLA, M. Bert based hierarchical alternating co-attention visual question answering using bottom-up features. **International Journal of Intelligent Systems and Applications in Engineering**, v. 10, n. 3s, p. 158–168, 2022. Citado 2 vezes nas páginas 45 e 50.
- KOSHTI, D. et al. Eduvqa—visual question answering: An educational perspective. **Journal of Advanced Research in Applied Sciences and Engineering Technology**, v. 42, n. 1, p. 144–157, 2024. Citado 3 vezes nas páginas 3, 49 e 50.
- KUMARI, V. et al. Visual question answering analysis: Datasets, methods, and image featurization techniques. In: . [S.l.: s.n.], 2023. v. 1. ISSN 21844313. Citado na página 10.
- KUNLAMAI, T. et al. Improving visual question answering for bridge inspection by pre-training with external data of image–text pairs. **Computer-Aided Civil and Infrastructure Engineering**, Wiley Online Library, v. 39, n. 3, p. 345–361, 2024. Citado na página 48.
- KUZNETSOVA, A. et al. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. **International journal of computer vision**, Springer, v. 128, n. 7, p. 1956–1981, 2020. Citado na página 14.
- LAU, J. J. et al. A dataset of clinically generated visual questions and answers about radiology images. **Scientific data**, Nature Publishing Group, v. 5, n. 1, p. 1–10, 2018. Citado na página 13.
- LI, C. et al. mplug: Effective and efficient vision-language learning by cross-modal skip-connections. In: **Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing**. [S.l.: s.n.], 2022. p. 7241–7259. Citado na página 21.
- LI, H. et al. Visual question answering as reading comprehension. In: IEEE COMPUTER SOCIETY. **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.], 2019. p. 6319–6328. Citado na página 21.
- LI, J. et al. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In: PMLR. **International conference on machine learning**. [S.l.], 2023. p. 19730–19742. Citado 3 vezes nas páginas v, 32 e 33.
- LI, L. et al. Relation-aware graph attention network for visual question answering. In: IEEE. **Proceedings of the IEEE/CVF international conference on computer vision**. [S.l.], 2019. p. 10313–10322. Citado 2 vezes nas páginas 42 e 50.

- LI, P. et al. Masked vision and language pre-training with unimodal and multimodal contrastive losses for medical visual question answering. In: SPRINGER. **International Conference on Medical Image Computing and Computer-Assisted Intervention**. [S.l.], 2023. p. 374–383. Citado na página 21.
- LI, P. et al. Self-supervised vision-language pretraining for medial visual question answering. In: IEEE. **2023 IEEE 20th International Symposium on Biomedical Imaging (ISBI)**. [S.l.], 2023. p. 1–5. Citado na página 21.
- LI, R.; JIA, J. Visual question answering with question representation update (qru). **Advances in Neural Information Processing Systems**, v. 29, 2016. Citado 2 vezes nas páginas 40 e 50.
- LI, Y. et al. Visual question generation as dual task of visual question answering. In: IEEE. **2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.], 2018. p. 6116–6124. Citado na página 24.
- LIANG, W.; JIANG, Y.; LIU, Z. Graghvqa: Language-guided graph neural networks for graph-based visual question answering. In: **Proceedings of the Third Workshop on Multimodal Artificial Intelligence**. [S.l.: s.n.], 2021. p. 79–86. Citado 2 vezes nas páginas 43 e 50.
- LIN, W. et al. Pmc-clip: Contrastive language-image pre-training using biomedical documents. In: SPRINGER. **International Conference on Medical Image Computing and Computer-Assisted Intervention**. [S.l.], 2023. p. 525–536. Citado na página 21.
- LIU, B. et al. Medical visual question answering via conditional reasoning and contrastive learning. **IEEE Transactions on Medical Imaging**, IEEE, 2022. Citado 2 vezes nas páginas 44 e 50.
- LU, J. et al. Hierarchical question-image co-attention for visual question answering. **Advances in neural information processing systems**, v. 29, 2016. Citado na página 21.
- LU, P. et al. Co-attending free-form regions and detections with multi-modal multiplicative feature embedding for visual question answering. In: **Proceedings of the AAAI conference on artificial intelligence**. [S.l.: s.n.], 2018. v. 32, n. 1. Citado na página 21.
- MA, Z. et al. Joint embedding vqa model based on dynamic word vector. **PeerJ Computer Science**, v. 7, 2021. ISSN 23765992. Citado na página 26.
- MALINOWSKI, M.; FRITZ, M. A multi-world approach to question answering about real-world scenes based on uncertain input. **Advances in neural information processing systems**, v. 27, 2014. Citado 2 vezes nas páginas 4 e 15.
- MAÑAS, O.; KROJER, B.; AGRAWAL, A. Improving automatic vqa evaluation using large language models. In: **Proceedings of the AAAI Conference on Artificial Intelligence**. [S.l.: s.n.], 2024. v. 38, n. 5, p. 4171–4179. Citado 2 vezes nas páginas 48 e 50.
- MANMADHAN, S.; KOVOOR, B. C. Visual question answering: a state-of-the-art review. **Artificial Intelligence Review**, Springer Science+Business Media B.V., v. 53, p. 5705–5745, 12 2020. ISSN 15737462. Citado 4 vezes nas páginas viii, 2, 3 e 4.
- MANMADHAN, S.; KOVOOR, B. C. Visual question answering: a state-of-the-art review. **Artificial Intelligence Review**, Springer Netherlands Dordrecht, v. 53, n. 8, p. 5705–5745, 2020. Citado na página 9.

- MAO, J. et al. The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. In: ICLR. **International Conference on Learning Representations**. [S.l.], 2018. Citado na página 21.
- MARINO, K. et al. Ok-vqa: A visual question answering benchmark requiring external knowledge. In: IEEE. **Proceedings of the IEEE/cvf conference on computer vision and pattern recognition**. [S.l.], 2019. p. 3195–3204. Citado na página 17.
- MASCHARKA, D. et al. Transparency by design: Closing the gap between performance and interpretability in visual reasoning. In: IEEE. **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.], 2018. p. 4942–4950. Citado na página 21.
- MIAO, Y. et al. Research on visual question answering based on gat relational reasoning. **Neural Processing Letters**, Springer, v. 54, p. 1435–1448, 4 2022. ISSN 1573773X. Citado 3 vezes nas páginas 3, 45 e 50.
- MOHOLKAR, E. a. K. P. Visual question answering using convolutional neural networks. **Turkish Journal of Computer and Mathematics Education (TURCOMAT)**, v. 12, 2021. Citado na página 26.
- NARASIMHAN, M.; LAZEBNIK, S.; SCHWING, A. Out of the box: Reasoning with graph convolution nets for factual visual question answering. **Advances in neural information processing systems**, v. 31, 2018. Citado na página 21.
- NARASIMHAN, M.; SCHWING, A. G. Straight to the facts: Learning knowledge base retrieval for factual visual question answering. In: SPRINGER. **European Conference on Computer Vision**. [S.l.], 2018. p. 460–477. Citado na página 21.
- NGUYEN, B. X. et al. Coarse-to-fine reasoning for visual question answering. In: IEEE COMPUTER SOCIETY. **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.], 2022. p. 4558–4566. Citado na página 21.
- NUTHALAPATI, S. V. et al. Lightweight visual question answering using scene graphs. In: . [S.l.]: Association for Computing Machinery, 2021. p. 3353–3357. ISBN 9781450384469. Citado 3 vezes nas páginas 43, 44 e 50.
- PANDHRE, S.; SODHANI, S. Survey of recent advances in visual question answering. **arXiv e-prints**, p. arXiv–1709, 2017. Citado na página 2.
- PATRO, B.; PATEL, S.; NAMBOODIRI, V. Robust explanations for visual question answering. In: IEEE. **Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision**. [S.l.], 2020. p. 1577–1586. Citado na página 24.
- PUSTOKHIN, D. A. et al. An effective deep residual network based class attention layer with bidirectional lstm for diagnosis and classification of covid-19. **Journal of Applied Statistics**, Taylor & Francis, v. 50, n. 3, p. 477–494, 2023. Citado 2 vezes nas páginas vi e 37.
- QIAN, T. et al. Nuscenes-qa: A multi-modal visual question answering benchmark for autonomous driving scenario. In: **Proceedings of the AAAI Conference on Artificial Intelligence**. [S.l.: s.n.], 2024. v. 38, n. 5, p. 4542–4550. Citado 3 vezes nas páginas 48, 49 e 50.

RADFORD, A. et al. Learning transferable visual models from natural language supervision. In: PMLR. **International conference on machine learning**. [S.l.], 2021. p. 8748–8763. Citado 3 vezes nas páginas [v](#), [31](#) e [32](#).

RAHMAN, M. K. Training sensitivity in graph isomorphism network. In: **Proceedings of the 29th ACM International Conference on Information & Knowledge Management**. [S.l.: s.n.], 2020. p. 2181–2184. Citado na página [35](#).

RAHMAN, T. et al. An improved attention for visual question answering. In: IEEE. **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.], 2021. p. 1653–1662. Citado 2 vezes nas páginas [43](#) e [50](#).

RAVI, S. et al. Vlc-bert: Visual question answering with contextualized commonsense knowledge. In: IEEE COMPUTER SOCIETY. **Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision**. [S.l.], 2023. p. 1155–1165. Citado 2 vezes nas páginas [47](#) e [50](#).

REN, M.; KIROU, R.; ZEMEL, R. Exploring models and data for image question answering. **Advances in neural information processing systems**, v. 28, 2015. Citado 2 vezes nas páginas [4](#) e [15](#).

RIQUELME, F. et al. Explaining vqa predictions using visual grounding and a knowledge base. **Image and Vision Computing**, Elsevier, v. 101, p. 103968, 2020. Citado 2 vezes nas páginas [43](#) e [50](#).

SCARSELLI, F. et al. The graph neural network model. **IEEE transactions on neural networks**, IEEE, v. 20, n. 1, p. 61–80, 2008. Citado na página [35](#).

SELVARAJU, R. R. et al. Grad-cam: Visual explanations from deep networks via gradient-based localization. In: IEEE. **2017 IEEE International Conference on Computer Vision (ICCV)**. [S.l.], 2017. p. 618–626. Citado na página [5](#).

SHAH, M. et al. Cycle-consistency for robust visual question answering. In: IEEE. **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.], 2019. p. 6649–6658. Citado na página [24](#).

SHAO, Z. et al. Prompting large language models with answer heuristics for knowledge-based visual question answering. In: IEEE. **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.], 2023. p. 14974–14983. Citado na página [21](#).

SHARMA, H.; JALAL, A. S. Image captioning improved visual question answering. **Multimedia tools and applications**, Springer, p. 1–22, 2021. Citado na página [24](#).

SHARMA, H.; JALAL, A. S. Visual question answering model based on graph neural network and contextual attention. **Image and Vision Computing**, Elsevier, v. 110, p. 104165, 2021. Citado 2 vezes nas páginas [43](#) e [50](#).

SHI, Y. et al. Question type guided attention in visual question answering. In: SPRINGER. **Computer Vision—ECCV 2018: 15th European Conference, Munich, Germany, September 8–14, 2018, Proceedings, Part IV 15**. [S.l.], 2018. p. 158–175. Citado na página [26](#).

SINGH, A. et al. Towards vqa models that can read. In: IEEE. **Proceedings of the IEEE/CVF conference on computer vision and pattern recognition**. [S.l.], 2019. p. 8317–8326. Citado 6 vezes nas páginas 14, 21, 41, 50, 70 e 71.

SONG, H. et al. Clip models are few-shot learners: Empirical studies on vqa and visual entailment. In: **Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**. [S.l.: s.n.], 2022. p. 6088–6100. Citado na página 26.

SPEER, R.; CHIN, J.; HAVASI, C. Conceptnet 5.5: An open multilingual graph of general knowledge. In: ASSOCIATION FOR THE ADVANCEMENT OF ARTIFICIAL INTELLIGENCE (AAAI). **Proceedings of the AAAI conference on artificial intelligence**. [S.l.], 2017. v. 31, n. 1. Citado na página 18.

SUO, W. et al. S3c: Semi-supervised vqa natural language explanation via self-critical learning. In: IEEE COMPUTER SOCIETY. **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.], 2023. p. 2646–2656. Citado na página 24.

TALAFHA, B.; AL-AYYOUB, M. Just at vqa-med: A vgg-seq2seq model. In: . [S.l.: s.n.], 2018. v. 2125. ISSN 16130073. Citado na página 26.

TAN, H.; BANSAL, M. Lxmert: Learning cross-modality encoder representations from transformers. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. **Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)**. [S.l.], 2019. p. 5100–5111. Citado na página 21.

TANDON, N.; MELO, G. D.; WEIKUM, G. Webchild 2.0: Fine-grained commonsense knowledge distillation. In: ACL. **Proceedings of ACL 2017, System Demonstrations**. [S.l.], 2017. p. 115–120. Citado na página 18.

TASCON-MORALES, S.; MÁRQUEZ-NEILA, P.; SZNITMAN, R. Logical implications for visual question answering consistency. In: IEEE. **2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.], 2023. p. 6725–6735. Citado na página 24.

TENEY, D.; WU, Q.; HENGEL, A. van den. Visual question answering: A tutorial. **IEEE Signal Processing Magazine**, IEEE, v. 34, n. 6, p. 63–75, 2017. Citado na página 8.

TIONG, A. M. H. et al. Plug-and-play vqa: Zero-shot vqa by conjoining large pretrained models with zero training. In: **Findings of the Association for Computational Linguistics: EMNLP 2022**. [S.l.: s.n.], 2022. p. 951–967. Citado 2 vezes nas páginas 46 e 50.

VAIDEESWARAN, R. et al. Towards reasoning-aware explainable vqa. In: **Workshop on Trustworthy and Socially Responsible Machine Learning, NeurIPS 2022**. [S.l.: s.n.], 2022. Citado na página 24.

VASWANI, A. et al. Attention is all you need. **Advances in neural information processing systems**, v. 30, 2017. Citado 3 vezes nas páginas 29, 31 e 40.

VEDANTAM, R. et al. Probabilistic neural symbolic models for interpretable visual question answering. In: PMLR. **International Conference on Machine Learning**. [S.l.], 2019. p. 6428–6437. Citado 3 vezes nas páginas 42, 43 e 50.

- VEDD, N. et al. Guiding visual question generation. In: **Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies**. [S.l.: s.n.], 2022. p. 1640–1654. Citado na página [24](#).
- VELICKOVIC, P. et al. Graph attention networks. **stat**, v. 1050, p. 4, 2018. Citado na página [36](#).
- VENTURA, V. d. A. **Representação de Imagens Através de Grafos Utilizando o Algoritmo Split And Merge Combinado Com Descritores de Cor e Textura**. Dissertação (Mestrado) — Universidade Federal do Espírito Santo, 2009. Citado 2 vezes nas páginas [34](#) e [35](#).
- WANG, J. et al. Tag: Boosting text-vqa via text-aware visual question-answer generation. **arXiv e-prints**, p. arXiv–2208, 2022. Citado na página [21](#).
- WANG, P. et al. One-peace: Exploring one general representation model toward unlimited modalities. **arXiv e-prints**, p. arXiv–2305, 2023. Citado na página [21](#).
- WANG, P. et al. Fvqa: Fact-based visual question answering. **IEEE transactions on pattern analysis and machine intelligence**, IEEE, v. 40, n. 10, p. 2413–2427, 2017. Citado 2 vezes nas páginas [18](#) e [21](#).
- WANG, W. et al. Image as a foreign language: Beit pretraining for vision and vision-language tasks. In: IEEE. **2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.], 2023. p. 19175–19186. Citado na página [21](#).
- WANG, Z.; JI, S. Learning convolutional text representations for visual question answering. In: SIAM. **Proceedings of the 2018 SIAM international conference on data mining**. [S.l.], 2018. p. 594–602. Citado na página [26](#).
- WANG, Z. et al. Interpretable visual reasoning via induced symbolic space. In: INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS INC. **18th IEEE/CVF International Conference on Computer Vision, ICCV 2021**. [S.l.], 2021. p. 1858–1867. Citado na página [21](#).
- WEST, D. B. et al. **Introduction to graph theory**. [S.l.]: Prentice hall Upper Saddle River, 2001. v. 2. Citado 2 vezes nas páginas [vi](#) e [34](#).
- WU, J.; CHEN, L.; MOONEY, R. J. Improving vqa and its explanations by comparing competing explanations. **arXiv e-prints**, p. arXiv–2006, 2020. Citado na página [24](#).
- WU, J.; MOONEY, R. Faithful multimodal explanation for visual question answering. In: **Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP**. [S.l.: s.n.], 2019. p. 103–112. Citado 3 vezes nas páginas [42](#), [43](#) e [50](#).
- WU, Q. et al. Image captioning and visual question answering based on attributes and their related external knowledge. **arXiv preprint arXiv:1603.02814**, v. 2, 2016. Citado na página [21](#).
- WU, Q. et al. Visual question answering: A survey of methods and datasets. **Computer Vision and Image Understanding**, Elsevier, v. 163, p. 21–40, 2017. Citado 2 vezes nas páginas [8](#) e [9](#).

- WU, Q. et al. Ask me anything: Free-form visual question answering based on knowledge from external sources. In: IEEE. **2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.], 2016. p. 4622–4630. Citado na página [21](#).
- WU, Z.; PALMER, M. Verb semantics and lexical selection. In: NEW MEXICO STATE UNIVERSITY. **32nd Annual Meeting of the Association for Computational Linguistics**. [S.l.], 1994. p. 133–138. Citado na página [23](#).
- XI, Y. et al. Visual question answering model based on visual relationship detection. **Signal Processing: Image Communication**, Elsevier, v. 80, p. 115648, 2020. Citado na página [21](#).
- YANG, Z. et al. Stacked attention networks for image question answering. In: IEEE. **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.], 2016. p. 21–29. Citado na página [21](#).
- YANG, Z. et al. Tap: Text-aware pre-training for text-vqa and text-caption. In: IEEE COMPUTER SOCIETY. **Proceedings of the IEEE/CVF conference on computer vision and pattern recognition**. [S.l.], 2021. p. 8751–8761. Citado na página [21](#).
- YI, K. et al. Neural-symbolic vqa: Disentangling reasoning from vision and language understanding. **Advances in neural information processing systems**, v. 31, 2018. Citado na página [21](#).
- YU, J. et al. Cross-modal knowledge reasoning for knowledge-based visual question answering. **Pattern Recognition**, Elsevier, v. 108, p. 107563, 2020. Citado na página [21](#).
- ZENG, Y. et al. X 2-vlm: All-in-one pre-trained model for vision-language tasks. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, IEEE, 2023. Citado na página [21](#).
- ZHANG, P. et al. Yin and yang: Balancing and answering binary visual questions. In: IEEE. **2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.], 2016. Citado 2 vezes nas páginas [1](#) e [4](#).
- ZHANG, X. et al. Pmc-vqa: Visual instruction tuning for medical visual question answering. **arXiv e-prints**, p. arXiv–2305, 2023. Citado na página [21](#).
- ZHAO, Z. et al. Proto: Program-guided transformer for program-guided tasks. **Advances in neural information processing systems**, v. 34, p. 17021–17036, 2021. Citado na página [21](#).
- ZHU, Y. et al. Visual7w: Grounded question answering in images. In: IEEE. **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.], 2016. p. 4995–5004. Citado na página [16](#).
- ZOU, Y.; XIE, Q. A survey on vqa: Datasets and approaches. In: IEEE. **2020 2nd International Conference on Information Technology and Computer Application (ITCA)**. [S.l.], 2020. p. 289–297. Citado 3 vezes nas páginas [9](#), [10](#) e [11](#).

Apêndices

APÊNDICE A – *Autoencoder*

O treinamento da rede *Autoencoder* foi realizado de forma *end-to-end*, ou seja, iniciado do zero e ajustado especificamente para a base de dados utilizada. Para esse processo, foram utilizadas as divisões de treinamento e validação disponibilizadas no conjunto VizWiz. As imagens foram redimensionadas para dimensões de 112x112x3. O *Encoder* é constituído por cinco camadas de convolução 2D, cada uma com um *kernel* de tamanho 3, intercaladas com a função de ativação Gelu. Após essas camadas convolucionais, a rede inclui uma camada de *flatten*, seguida por uma camada totalmente conectada com tamanho de saída de 768, que corresponde ao tamanho do *embedding* gerado pelo BERT. Essa saída representa a codificação latente da imagem.

O *Decoder* foi projetado como uma inversão do *Encoder*, conforme ilustrado na Figura 44, que detalha a arquitetura implementada. Os parâmetros empregados no treinamento do *Autoencoder* estão descritos na Tabela 5.

```
Autoencoder(
  (encoder): Encoder(
    (net): Sequential(
      (0): Conv2d(3, 112, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
      (1): GELU(approximate='none')
      (2): Conv2d(112, 112, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (3): GELU(approximate='none')
      (4): Conv2d(112, 224, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
      (5): GELU(approximate='none')
      (6): Conv2d(224, 224, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (7): GELU(approximate='none')
      (8): Conv2d(224, 224, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
      (9): GELU(approximate='none')
      (10): Flatten(start_dim=1, end_dim=-1)
      (11): Linear(in_features=43904, out_features=768, bias=True)
    )
  )
  (decoder): Decoder(
    (linear): Sequential(
      (0): Linear(in_features=768, out_features=43904, bias=True)
      (1): GELU(approximate='none')
    )
    (net): Sequential(
      (0): ConvTranspose2d(224, 224, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), output_padding=(1, 1))
      (1): GELU(approximate='none')
      (2): Conv2d(224, 224, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (3): GELU(approximate='none')
      (4): ConvTranspose2d(224, 112, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), output_padding=(1, 1))
      (5): GELU(approximate='none')
      (6): Conv2d(112, 112, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (7): GELU(approximate='none')
      (8): ConvTranspose2d(112, 3, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), output_padding=(1, 1))
      (9): Tanh()
    )
  )
)
```

Figura 44 – Rede do *Autoencoder*.

Na etapa de inferência do *Autoencoder* para a obtenção dos atributos que representam a imagem completa, a imagem deve ser redimensionada para 112x112x3. Por outro lado, para extrair os atributos correspondentes aos *patches* da imagem, a imagem original é

Parâmetro	Valor
<i>batch</i> de treino	32
<i>batch</i> de validação	32
Otimizador	Adam
Função de Perda	MSE
Taxa de aprendizado	1e-5
Épocas	200

Tabela 5 – Parâmetros utilizados no treinamento da rede *Autoencoder*.

redimensionada para 224x224x3 e, em seguida, dividida em quatro partes, cada uma com dimensões de 112x112x3, a partir das quais são gerados os respectivos *embeddings*.

APÊNDICE B – Arquitetura das Abordagens

Os trechos de código relacionados à definição da arquitetura das abordagens, incluindo as camadas utilizadas e a forma como estão conectadas, serão descritos a seguir.

- Abordagem 1:

```
self.gat1 = GATv2Conv(dim_in, 200, heads=8)
self.gat2 = GATv2Conv(200*8, 300, heads=4)
self.layer_3 = torch.nn.Linear(300*4, dim_out)
```

```
def forward(self, x, edge_index, batch):
```

```
    x = self.gat1(x, edge_index)
    x = self.gat2(x, edge_index)
    x = global_mean_pool(x, batch)
    return self.layer_3(x)
```

- Abordagem 2:

```
self.gat1 = GATv2Conv(dim_in, 300, heads=8)
self.dropout3 = nn.Dropout(0.2)
self.layer_3 = torch.nn.Linear(300*8, dim_out)
```

```
def forward(self, x, edge_index, batch):
```

```
    x = self.gat1(x, edge_index)
    x = global_mean_pool(x, batch)
    x = self.dropout3(x)
    return self.layer_3(x)
```

- Abordagem 3:

```
self.gat1 = GATv2Conv(dim_in, 300, heads=8)
self.dropout3 = nn.Dropout(0.2)
self.gat2 = GATv2Conv(300*8, 300, heads=4)
self.dropout4 = nn.Dropout(0.2)
self.layer_3 = torch.nn.Linear(300*4, dim_out)
```

```
def forward(self, x, edge_index, batch):
```

```
    x = self.gat1(x, edge_index)
```

```

x = self.dropout3(x)
x = self.gat2(x, edge_index)
x = global_mean_pool(x, batch)
x = self.dropout4(x)
return self.layer_3(x)

```

- Abordagem 4:

```

self.gat1 = GATv2Conv(dim_in, 2000)
self.gat2 = GATv2Conv(2000, 1000)
self.lin = Linear(1000, dim_out)

```

```

def forward(self, x, edge_index, batch):
    x = F.elu(self.gat1(x, edge_index))
    x = F.elu(self.gat2(x, edge_index))
    x = global_mean_pool(x, batch)
    x = F.dropout(x, p=0.5)
    return self.lin(x)

```

- Abordagem 5:

```

self.gat1 = GATv2Conv(dim_in, 2000)
self.gat2 = GATv2Conv(2000, 1000)
self.lin = Linear(1000, dim_out)

```

```

def forward(self, x, edge_index, batch):
    x = F.elu(self.gat1(x, edge_index))
    x = F.elu(self.gat2(x, edge_index))
    x = global_mean_pool(x, batch)
    x = F.dropout(x, p=0.5)
    return self.lin(x)

```

- Abordagem 6:

```

self.linear_layer1 = nn.Sequential(
    nn.LayerNorm(self.clip_model.visual.output_dim + self.clip_model.text_projection.shape[1]),
    nn.Dropout(p=0.5),
    nn.Linear(self.clip_model.visual.output_dim + self.clip_model.text_projection.shape[1],
              hidden_size))
self.linear_layer2 = nn.Sequential(
    nn.LayerNorm(hidden_size),
    nn.Dropout(p=0.5),

```

```

nn.Linear(hidden_size, num_classes))
self.answer_type_layer = nn.Linear(hidden_size, 4)
self.answer_mask_layer = nn.Linear(4, num_classes)
self.sigmoid = nn.Sigmoid()
self.answerability_linear_layer = nn.Sequential(
nn.LayerNorm(self.clip_model.visual.output_dim + self.clip_model.text_projection.shape[1]),
nn.Linear(self.clip_model.visual.output_dim + self.clip_model.text_projection.shape[1],
hidden_size))
self.answerability_final_layer = nn.Linear(hidden_size, 1)
self.answerability_sigmoid = nn.Sigmoid()

def forward(self, image, question):
    features = torch.cat((torch.flatten(image, start_dim=1), torch.flatten(question, start_dim=1)),
dim=1)
    # Answerability score
    answerability_score = self.answerability_linear_layer(features)
    answerability_score = self.answerability_final_layer(answerability_score)
    answerability_score = self.answerability_sigmoid(answerability_score).squeeze()
    # First linear layer
    features = self.linear_layer1(features)
    # Answer type and mask
    answer_type = self.answer_type_layer(features)
    answer_mask = self.sigmoid(self.answer_mask_layer(answer_type))
    # Second linear layer and apply mask
    output = self.linear_layer2(features)
    output = output * answer_mask
    return output, answer_type, answerability_score

```

- Abordagem 7:

```

self.linear_layer1 = nn.Sequential(
nn.LayerNorm(1536),
nn.Dropout(p=0.5),
nn.Linear(1536, hidden_size))
self.linear_layer2 = nn.Sequential(
nn.LayerNorm(hidden_size),
nn.Dropout(p=0.5),
nn.Linear(hidden_size, num_classes))
self.answer_type_layer = nn.Linear(hidden_size, 4)
self.answer_mask_layer = nn.Linear(4, num_classes)

```

```

self.sigmoid = nn.Sigmoid()
self.answerability_linear_layer = nn.Sequential(
    nn.LayerNorm(1536),
    nn.Linear(1536, hidden_size))
self.answerability_final_layer = nn.Linear(hidden_size, 1)
self.answerability_sigmoid = nn.Sigmoid()

def forward(self, image, question):
    features = torch.cat((torch.flatten(image, start_dim=1), torch.flatten(question, start_dim=1)),
dim=1)
    # Answerability score
    answerability_score = self.answerability_linear_layer(features)
    answerability_score = self.answerability_final_layer(answerability_score)
    answerability_score = self.answerability_sigmoid(answerability_score).squeeze()
    # First linear layer
    features = self.linear_layer1(features)
    # Answer type and mask
    answer_type = self.answer_type_layer(features)
    answer_mask = self.sigmoid(self.answer_mask_layer(answer_type))
    # Second linear layer and apply mask
    output = self.linear_layer2(features)
    output = output * answer_mask
    return output, answer_type, answerability_score

```

- Abordagem 8:

```

self.linear_layer1 = nn.Sequential(
    nn.LayerNorm(self.clip_model.visual.output_dim + self.clip_model.text_projection.shape[1]),
    nn.Dropout(p=0.5),
    nn.Linear(self.clip_model.visual.output_dim + self.clip_model.text_projection.shape[1],
hidden_size))
self.linear_layer2 = nn.Sequential(
    nn.LayerNorm(hidden_size),
    nn.Dropout(p=0.5),
    nn.Linear(hidden_size, num_classes))
self.answer_type_layer = nn.Linear(hidden_size, 4)
self.answer_mask_layer = nn.Linear(4, num_classes)
self.sigmoid = nn.Sigmoid()

```

```

def forward(self, image, question):

```

```

        features = torch.cat((torch.flatten(image, start_dim=1), torch.flatten(question, start_dim=1)),
dim=1)
        # Passing the features through the first linear layer
        features = self.linear_layer1(features)
        # Answer type and mask
        answer_type = self.answer_type_layer(features)
        answer_mask = self.sigmoid(self.answer_mask_layer(answer_type))
        # Second linear layer and apply mask
        output = self.linear_layer2(features)
        output = output * answer_mask
        return output, answer_type

```

- Abordagem 9:

```

self.linear_layer1 = nn.Sequential(
    nn.LayerNorm(1768),
    nn.Dropout(p=0.5),
    nn.Linear(1768, hidden_size))
self.linear_layer2 = nn.Sequential(
    nn.LayerNorm(hidden_size),
    nn.Dropout(p=0.5),
    nn.Linear(hidden_size, num_classes))
self.answer_type_layer = nn.Linear(hidden_size, 4)
self.answer_mask_layer = nn.Linear(4, num_classes)
self.sigmoid = nn.Sigmoid()
self.answerability_linear_layer = nn.Sequential(
    nn.LayerNorm(1768),
    nn.Linear(1768, hidden_size))
self.answerability_final_layer = nn.Linear(hidden_size, 1)
self.answerability_sigmoid = nn.Sigmoid()

```

```

def forward(self, image, question):

```

```

        features = torch.cat((torch.flatten(image, start_dim=1), torch.flatten(question, start_dim=1)),
dim=1)
        # Calculating the answerability score
        answerability_score = self.answerability_linear_layer(features)
        answerability_score = self.answerability_final_layer(answerability_score)
        answerability_score = self.answerability_sigmoid(answerability_score).squeeze()
        # Passing the features through the first linear layer
        features = self.linear_layer1(features)

```

```

# Passing the features to get 4 answer types
answer_type = self.answer_type_layer(features)
# Expanding answer mask to the same size as the number of classes (vocab size)
answer_mask = self.sigmoid(self.answer_mask_layer(answer_type))
# Passing the features through the second linear layer
output = self.linear_layer2(features)
# Applying the answer mask to the output
output = output * answer_mask
return output, answer_type, answerability_score

```

- Abordagem 10:

```

# First linear layer for feature combination (image + question)
self.linear_layer1 = nn.Sequential(
    nn.LayerNorm(1768),
    nn.Dropout(p=0.5),
    nn.Linear(1768, hidden_size))
# Second linear layer for final classification output
self.linear_layer2 = nn.Sequential(
    nn.LayerNorm(hidden_size),
    nn.Dropout(p=0.5),
    nn.Linear(hidden_size, num_classes))
# Layer for determining answer type (4 types)
self.answer_type_layer = nn.Linear(hidden_size, 4)
# Masking layer for answer vocabulary size
self.answer_mask_layer = nn.Linear(4, num_classes)
# Sigmoid activation
self.sigmoid = nn.Sigmoid()
# Answerability prediction layer
self.answerability_linear_layer = nn.Sequential(
    nn.LayerNorm(1768),
    nn.Linear(1768, hidden_size))
# Final layer for answerability score
self.answerability_final_layer = nn.Linear(hidden_size, 1)
# Sigmoid for final answerability score
self.answerability_sigmoid = nn.Sigmoid()

```

```

def forward(self, image, question):

```

```

    # Flattening and concatenating the image and question features
    features = torch.cat((torch.flatten(image, start_dim=1), torch.flatten(question, start_dim=1)),

```

```
dim=1)
    # Calculating the answerability score
    answerability_score = self.answerability_linear_layer(features)
    answerability_score = self.answerability_final_layer(answerability_score)
    answerability_score = self.answerability_sigmoid(answerability_score).squeeze()
    # Passing the features through the first linear layer
    features = self.linear_layer1(features)
    # Passing the features to get 4 answer types
    answer_type = self.answer_type_layer(features)
    # Expanding answer mask to the same size as the number of classes (vocab size)
    answer_mask = self.sigmoid(self.answer_mask_layer(answer_type))
    # Passing the features through the second linear layer for final classification
    output = self.linear_layer2(features)
    # Applying the answer mask to the output
    output = output * answer_mask
    return output, answer_type, answerability_score
```

APÊNDICE C – Bibliotecas

As bibliotecas utilizadas e suas versões estão listadas a seguir:

CoCa-pytorch==0.1.0

clip==1.0

contractions==0.1.73

gensim==4.3.2

Levenshtein==0.23.0

json==2.0.9

numpy==1.22.2

pandas==1.4.1

pillow==10.2.0

regex==2023.12.25

requests==2.31.0

salesforce-lavis==1.0.2

scikit-learn==1.0.2

sentencepiece==0.1.99

spacy==3.7.2

timm==0.4.12

tokenizers==0.13.3

torch==2.1.2

torch-geometric==2.4.0

torchvision==0.16.2

transformers==4.26.1

tqdm==4.42.1

wget==3.2