

UM ESTUDO DO GARGALO DE INFORMAÇÃO EM UMA REDE NEURAL ARTIFICIAL

GABRIEL LIMA SERTÃO

TRABALHO DE CONCLUSÃO DE CURSO
EM ENGENHARIA DE REDES DE COMUNICAÇÃO

DEPARTAMENTO DE ENGENHARIA ELÉTRICA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASÍLIA

Universidade de Brasília
Faculdade de Tecnologia
Departamento de Engenharia Elétrica

Um estudo do Gargalo de informação em uma rede neural artificial

Gabriel Lima Sertão

TRABALHO DE CONCLUSÃO DE CURSO SUBMETIDA AO DEPARTAMENTO DE ENGENHARIA ELÉTRICA DA UNIVERSIDADE DE BRASÍLIA COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE ENGENHEIRO(A) DE REDES DE COMUNICAÇÃO.

APROVADA POR:

Daniel Guerreiro e Silva, Professor Adjunto (Universidade de Brasília) (Orientador)

Alexandre Ricando Soares Romariz, Professor Associado (Universidade de Brasília) (Examinador Interno)

Eduardo Peixoto Fernandes da Silva, Professor Adjunto (Universidade de Brasília) (Examinador Interno)

Brasília/DF, fevereiro de 2025.

FICHA CATALOGRÁFICA

LIMA SERTÃO, GABRIEL

Um estudo do Gargalo de informação em uma rede neural artificial. [Brasília/DF] 2025.

xxx, nnnp., 210 x 297 mm (ENE/FT/UnB, Engenheiro(a) de Redes de Comunicação, Trabalho de Conclusão de Curso, 2025).

Universidade de Brasília, Faculdade de Tecnologia, Departamento de Engenharia Elétrica.

Departamento de Engenharia Elétrica

1. Aprendizado profundo
2. Aprendizado de máquina
3. Gargalo da informação
4. Teoria da informação
5. Auto codificadores
6. Divergência KL
7. Inteligência artificial
8. *Variational Information Bottleneck*

I. ENE/FT/UnB

II. Uso da Teoria do Gargalo de informação

REFERÊNCIA BIBLIOGRÁFICA

LIMA SERTÃO, GABRIEL (2025). Um estudo do Gargalo de informação em uma rede neural artificial. Trabalho de Conclusão de Curso, Publicação ENE.XXXXX/2025, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, xxxxp.

CESSÃO DE DIREITOS

AUTOR: Gabriel Lima Sertão

TÍTULO: Um estudo do Gargalo de informação em uma rede neural artificial.

GRAU: Engenheiro(a) de Redes de Comunicação ANO: 2025

É concedida à Universidade de Brasília permissão para reproduzir cópias deste Trabalho de Conclusão de Curso e para emprestar ou vender tais cópias somente para propósitos acadêmicos e científicos. O autor reserva outros direitos de publicação e nenhuma parte deste trabalho de conclusão de curso pode ser reproduzido sem autorização por escrito do autor.

Gabriel Lima Sertão

Universidade de Brasília (UnB)

Campus Darcy Ribeiro

Faculdade de Tecnologia - FT

Departamento de Engenharia Elétrica (ENE)

Brasília - DF CEP 70919-970

Sic Parvis Magna

AGRADECIMENTOS

Agradecimentos aos pais, irmã, madrinha e familiares que deram apoio durante este percurso universitário, desde o início até o momento da entrega deste documento concretizador.

Aos amigos e colegas que ajudaram de uma forma ou outra, não apenas no âmbito acadêmico, em especial agradecimentos a: Alexander A. Alemi, Beatriz Hanae, Israel Borges, todos os amigos do aerotrem.

Aos professores da UnB por compartilharem os seus conhecimentos, ajudando a formar uma base sólida nas mais diversas áreas do curso, mesmo aqueles conhecimentos que precisaram de mais empenho e repetições, algumas muitas repetições, são apreciados. Um agradecimento em especial ao professor Daniel Guerreiro e Silva, por ter aceitado ser o professor orientador deste projeto final e ter guiado bem a realização do mesmo, tanto em conteúdos quanto em correções.

RESUMO

Este relatório de projeto final de graduação do curso de Engenharia de Redes de Comunicação tem como objetivo o estudo do princípio do gargalo da informação, aplicado ao aprendizado profundo. Esse princípio faz parte da teoria da informação e tem como uma de suas aplicações a redução da dimensionalidade. Por meio de uma revisão teórica e de testes experimentais, este projeto visa ilustrar a validade da aplicação do princípio na área de aprendizado profundo, tendo em vista que esta é uma das áreas que apresentou grande crescimento nos últimos anos e que pode ser beneficiada pelas aplicações derivadas do uso deste fundamental teórico.

O princípio do gargalo da informação sugere que, para a compreensão e processamento de dados de forma mais eficiente, deve-se reduzir a quantidade de dados irrelevantes e focar apenas nos dados essenciais para a realização da tarefa desejada. No contexto estudado, o aprendizado profundo, isso pode melhorar a eficiência dos modelos e sistemas, diminuindo a complexidade computacional e aumentando a capacidade de generalização da rede neural. Este estudo explora uma abordagem em que a teoria do gargalo pode ser utilizada para otimizar arquiteturas de redes neurais profundas, avaliando o impacto no desempenho e na precisão do modelo.

Além disso, será feita uma breve abordagem teórica do conceito e uso de autocodificadores variacionais, considerando que o modelo aplicado aqui, *Variational Information Bottleneck*, pode ser encarado como uma ramificação do auto codificador variacional. Essa abordagem se faz necessária para um melhor entendimento do modelo estudado e testado. Por fim, são apresentados alguns resultados práticos e métricas coletadas que servem como ferramentas para fazer uma análise do uso deste ferramental teórico

Palavras-chave: Aprendizado profundo, Aprendizado de máquina, Gargalo da informação, Teoria da informação, autocodificadores.

ABSTRACT

This final project report for the Communication Networks Engineering course aims to study the principle of the information bottleneck, applied to deep learning. This principle is part of information theory and has among its applications the reduction of dimensionality. Through a theoretical review and experimental tests, this project seeks to illustrate the validity of applying the principle in the field of deep learning, considering that this is one of the areas that has experienced significant growth in recent years and can benefit from applications derived from this theoretical framework.

The information bottleneck principle suggests that for the efficient understanding and processing of data, irrelevant information should be reduced, focusing only on the essential data required for performing the desired task. In the studied context—deep learning—this principle can enhance the efficiency of models and systems by reducing computational complexity and increasing the generalization capacity of neural networks. This study will explore an approach where the bottleneck theory can be applied to optimize deep neural network architectures, evaluating its impact on model performance and accuracy.

Furthermore, a brief theoretical discussion of the concept and use of variational autoencoders will be provided, considering that the model applied here, the Variational Information Bottleneck, can be seen as a branch of the variational autoencoder. This discussion is necessary for a better understanding of the studied and tested model. Finally, practical results and collected metrics will be presented, serving as tools for analyzing the use of this theoretical framework.

Keywords: Deep learning, Machine learning, Information Bottleneck, Information Theory, Autoencoder.

SUMÁRIO

Sumário i

Lista de figuras iii

Lista de tabelas iv

Glossário v

Capítulo 1 – Introdução 1

1.1 Objetivo 2
1.2 Organização do trabalho 2

Capítulo 2 – Fundamentação Teórica 4

2.1 Teoria da informação e o Gargalo da Informação 4
2.2 Divergência de Kullback-Leibler 5
2.3 Aprendizado profundo 6
2.4 Cadeias de Markov 8
2.5 Aplicação do gargalo da informação em aprendizado profundo. 8
2.6 Auto Codificadores 12
2.7 Autocodificadores Variacionais 14
2.8 Deep Variational Information Bottleneck 16

Capítulo 3 – Metodologia 18

3.1 Preparação de dados 18
3.2 Ferramentas e ambiente de desenvolvimento 22
3.2.1 Arquitetura dos modelos 22
3.2.1.1 VIB 22
3.2.1.2 Autocodificador Modificado 24
3.2.2 Implementação 26

Capítulo 4 – Resultados 27

4.1 Resultados do modelo AEM 27
4.1.1 Primeiro conjunto 27
4.1.2 Segundo conjunto 29
4.1.3 Terceiro conjunto 31
4.2 Resultados do modelo VIB 32
4.2.1 Primeiro conjunto 32
4.2.2 Segundo conjunto 34
4.2.3 Terceiro conjunto 35
4.3 Análise comparativa 36
4.4 Interpretação dos resultados 37
4.5 Considerações 39

Capítulo 5 – Conclusões 40

5.1 Limitações 40
5.2 Conclusões 41

Referências 42

LISTA DE FIGURAS

2.1 Estrutura de uma DNN (ALEMI *et al.*, 2017). 10

2.2 Arquitetura de um auto codificador. 13

2.3 Comparativo AE e VAE. 15

2.4 Modelo VAE padrão (KINGMA; WELLING, 2013b). 16

3.1 Exemplos imagens MNIST. 20

3.2 Exemplos imagens Fashion MNIST. 21

3.3 Exemplos imagens CIFAR-10. 21

3.4 Arquitetura VIB. 23

4.1 Gráficos de métricas para o conjunto MNIST. 28

4.2 Gráficos de métricas para o conjunto Fashion MNIST. 30

4.3 Exemplos de classificações corretas para Fashion MNIST. 30

4.4 Gráficos de métricas para o conjunto CIFAR-10. 31

4.5 Exemplos de classificações incorretas para CIFAR-10. 32

4.6 Gráficos de métricas VIB para o conjunto MNIST. 33

4.7 Exemplos de classificações corretas do modelo VIB para MNIST. 33

4.8 Gráficos de métricas VIB para o conjunto Fashion MNIST. 34

4.9 Gráficos de métricas VIB para o conjunto CIFAR-10. 35

4.10 Exemplos de classificações incorretas do modelo VIB para CIFAR-10. 36

4.11 Matriz de confusão do modelo VIB para o CIFAR-10. 38

LISTA DE TABELAS

3.1 Especificações VIB 24

3.2 Especificações auto codificador Modificado 25

4.1 Métricas finais modelo AEM MNIST no conjunto de validação 27

4.2 Comparativo métricas modelo MNIST. 28

4.3 Métricas finais modelo AEM Fashion MNIST no conjunto de validação 29

4.4 Métricas finais modelo AEM CIFAR-10 no conjunto de validação 31

4.5 Métricas finais modelo VIB MNIST no conjunto de validação 32

4.6 Métricas finais modelo VIB Fashion MNIST no conjunto de validação 34

4.7 Métricas finais modelo VIB CIFAR-10 no conjunto de validação 35

4.8 Comparação dos modelos AEM e VIB em diferentes datasets 37

GLOSSARY

AP	Aprendizado Profundo
AM	Aprendizado de Máquina
IA	Inteligência Artificial
IB	*Information Bottleneck*
AE	*Autoencoder*
AEM	Autoencoder modificado
VAE	*Variational Autoencoder*
VIB	*Variational Information Bottleneck*
DVIB	*Deep Variational Information Bottleneck*

CAPÍTULO 1

INTRODUÇÃO

Aprendizado profundo (*Deep Learning*) é um ramo especializado de aprendizado de máquina (*Machine Learning*) que tem como uma de suas características principais o potencial de retirar características de alto nível de dados de entrada brutos. As aplicações de Aprendizado Profundo (AP) vêm rapidamente substituindo vários sistemas convencionais presentes no cotidiano, como reconhecimento de voz e imagens, procuras na web, filtragem de e-mails, etc. As técnicas de AP também mostram uma capacidade de auxiliar na performance científica das mais variadas áreas possíveis. O aumento da velocidade de processamento e acurácia proporcionado pelos modelos de AP podem beneficiar praticamente todas as aplicações nos quais eles sejam usados, desde que sejam treinados de maneira adequada e com um banco de dados robusto e suficiente para a tarefa no qual será aplicado (CHOUDHARY *et al.*, 2022).

Um conceito importante na área de AP é o da "maldição da dimensionalidade". Esse conceito refere-se aos vários fenômenos que aparecem a medida que aumentamos a dimensionalidade dos dados que estamos analisando ou organizando. Na área de aprendizado de máquina, este conceito se faz muito presente, uma vez que grandes quantidades de dados são necessários para a realização do treinamento. Uma boa prática disseminada na área é a de que geralmente são necessários 5 exemplos de treinamento para cada dimensão na representação (WIKIPEDIA, 2024b). É importante ter conceitos como esse em mente na hora da modelagem e estudo de redes neurais, uma vez que modelos de AP, com suas múltiplas camadas, acabam sendo muito influenciados pela complexidade dos dados (JIM; MARK, 2024).

O foco deste projeto é em avaliar a aplicação da técnica introduzida por Tishby *et al.* (1999), conhecida como *Information Bottleneck Method*, ou Método do gargalo da informação, em tradução livre. Esse método, que por vezes também é chamado de princípio do gargalo da informação (*Information bottleneck principle*), originado da teoria da informação, tem como uma de suas aplicações em aprendizado de máquina a redução de dimensionalidade ao extrair

as informações mais relevantes para dada tarefa, que por consequência aumentaria a eficiência de processamento e a performance do modelo. Essas características são muitíssimo desejadas e bem vistas, já que o treinamento de modelos robustos requer uma grande quantidade de dados de entrada, o que faz com que, à medida que se aumenta a complexidade dos dados usados, mais computacionalmente custoso fica para processá-los. Então, ter uma possibilidade de redução de dimensionalidade dos dados, focando no que é necessário para a atividade proposta e ignorando o que não é usado, ajuda no processo de construção do modelo de aprendizado de máquina (AM) (GRAPHITE-NOTE, 2024).

1.1 OBJETIVO

Os principais objetivos deste trabalho são estudar, investigar e avaliar a aplicação do Princípio do Gargalo da Informação em modelos de aprendizado profundo, observando sua capacidade de melhoria da eficiência e generalização de modelos neurais. Para alcançar esses objetivos, foi feita uma pesquisa sobre métodos existentes que se baseiem nesse princípio, bem como a experimentação prática de modelos de redes neurais. Para ter uma base comparativa mais robusta, foi realizada uma análise comparativa entre um modelo baseado no princípio do gargalo da informação, o *Deep Variational Information Bottleneck*, e um modelo de autocodificador modificado, ambos usados para classificação de imagens em conjuntos de dados. Durante a comparação foram avaliadas métricas de desempenho, como acurácia, perda e complexidade computacional.

O projeto visou, além da comparação prática, realizar também uma abordagem de forma lacônica sobre os embasamentos teóricos necessários que se iniciam na teoria da informação de Shannon, até o ponto final que seria o modelo de rede neural aqui aplicado, conceitos que são desenvolvidos no capítulo de fundamentação teórica.

1.2 ORGANIZAÇÃO DO TRABALHO

Este trabalho é desenvolvido em 6 capítulos. O capítulo 1 serve como uma breve introdução ao tema e objetivos do projeto. O capítulo 2 é constituído pela revisão bibliográfica, onde são apresentados os principais trabalhos e estudos na área, discussões das contribuições e limitações

dos estudos existentes, identificação de desafios existente e mostrar como este trabalho se encaixa na temática estudada. No capítulo 3 é feita a fundamentação teórica, onde são dados os principais conceitos para a utilização da teoria do gargalo da informação na área de DL, pilares os quais também são usados para realizar a parte prática deste projeto. O capítulo 4 mostra a metodologia utilizada durante este projeto final, com a descrição e justificativa dos métodos e procedimentos adotados, e são dados detalhes dos experimentos realizados. O capítulo 5 é reservado para a apresentação e análise dos resultados obtidos, com uma comparação perante a literatura presente até o momento e uma discussão das implicações destes resultados. Por fim, no capítulo 6, é feita uma síntese dos principais temas e resultados do estudo, são apresentadas as limitações encontradas, sugestões para pesquisas futuras e considerações finais.

CAPÍTULO 2

FUNDAMENTAÇÃO TEÓRICA

Para um bom estudo da teoria do gargalo da informação, faz-se necessária uma boa base teórica em alguns conceitos da matemática e computação. Este capítulo de fundamentação teórica visa expor os conceitos e correntes de pesquisa existentes na literatura nesta área, bem como uma breve revisão bibliográfica, fornecendo uma base para a realização da pesquisa deste projeto final.

2.1 TEORIA DA INFORMAÇÃO E O GARGALO DA INFORMAÇÃO

O nascimento da teoria da informação pode ser marcado pelas proposições de Claude Shannon referentes às leis fundamentais de compressão e transmissão de dados (SHANNON, 1948). Um conceito importante em teoria da informação é o de entropia, que pode ser definida para uma variável aleatória como o nível médio de informação, "surpresa"ou incerteza para os possíveis resultados da variável, definida pela seguinte equação

$$H(x) = -\sum p(x) \log p(x). \tag{2.1}$$

O somatório é dado pela soma das probabilidades de todos os valores possíveis da variável e a base logarítmica pode variar dependendo da finalidade, a base 2, por exemplo, representa bits (CHEN, 2016).

Ao formalizarmos ideais intuitivas sobre informação, surge um problema fundamental que é o de dar uma noção quantitativa sobre o que são informações "relevantes"ou "significativas". Problemas que foram deixados de fora da teoria da informação de forma intencional por Shannon, que focou principalmente no problema da transmissão de informação ao invés de seu valor. Com o intuito de focar justamente na parte da importância da informação para uma certa tarefa, Tishby *et al.* (1999) apresenta um método para encontrar representações do sinal, no caso, que capturem apenas a estrutura relevante.

Em seu artigo, Tishby *et al.* (1999) faz algumas definições matemáticas para o que ele denomina de como quantização relevante e depois usa essa medida de relevância através da distorção, chamada de Rate Distortion Theory (Teoria da Taxa de Distorção), termo apresentado por Shannon (1948), para depois apresentar o conceito de relevância através de outra variável, que ele define como o Information Bottleneck (IB), gargalo da informação.

A formulação matemática feita por Tishby resulta na seguinte equação,

$$\mathcal{L}[p(\hat{x}|x)] = I(X;\hat{X}) - \beta I(\hat{X};Y). \tag{2.2}$$

Onde um dado conjunto de dados é dado pela distribuição conjunta $p(\hat{x}|x)$, e o objetivo é encontrar uma variável intermediária $\hat{X}$ (também chamada de representação latente) que seja uma compressão da variável de entrada X, mas que retenha as informações mais relevantes sobre Y. A distribuição $p(\hat{x}|x)$ define como $\hat{X}$ é gerado a partir de X, enquanto $I(X;\hat{X})$ e $I(\hat{X};Y)$ são definidas como as informações mútuas de X e $\hat{X}$, e $\hat{X}$ e Y, respectivamente. Informação mútua sendo uma medida de quantidade de informação que um parâmetro tem sobre o outro. Por fim, β é um parâmetro de regularização que controla o equilíbrio entre a compressão de X e a preservação de informações úteis para Y (TISHBY *et al.*, 1999).

Tishby *et al.* (1999) finaliza seu artigo dizendo sobre o aspecto definido como "fascinante"do princípio do gargalo da informação. O fascínio viria do *framework* unificado para diferentes problemas de processamento, incluindo predição, filtragem e aprendizado. Citando ao fim algumas aplicações do método em problemas mais cotidianos, entre eles, codificação neural e clusterização de palavras da língua inglesa.

2.2 DIVERGÊNCIA DE KULLBACK-LEIBLER

Uma divergência representa o grau de separação entre dois pontos P e Q. A divergência de Kullback–Leibler (muitas vezes também referenciada como entropia relativa), é um tipo de distância estatística, uma medida do quão diferente um modelo de distribuição de probabilidade P é diferente de sua real distribuição de probabilidade Q. Matematicamente, é definida na equação 2.3 (KULLBACK; LEIBLER, 1951).

A divergência é um valor não negativo e real, com valor nulo se e e somente se as duas distribuições em questão são idênticas. Tendo várias aplicações teóricas e práticas, em campos

como o da teoria da informação.

A divergência Kullback-Leibler de P para Q de uma variável aleatória discreta X é definida pela equação,

$$D_{KL}(P|Q) = \sum_{x \in X} p(x) \log(\frac{p(x)}{q(x)}). \tag{2.3}$$

Onde $p(x)$ é a distribuição de probabilidade verdadeira e $q(x)$ é a distribuição de probabilidade aproximada, o log da razão entre a probabilidade verdadeira e a aproximada pode ser visto como a medida de discrepância ponto a ponto entre P e Q (MACKAY, 2003).

2.3 APRENDIZADO PROFUNDO

Um dos grandes desafios da inteligência artificial é solucionar tarefas que os seres humanos executam com facilidade, mas que são difíceis de descrever em termos de regras explícitas, pois ocorrem de maneira intuitiva e automática. Exemplos clássicos incluem o reconhecimento de palavras faladas ou a identificação de rostos em imagens, habilidades naturais para humanos, mas complexas para serem reproduzidas computacionalmente. O aprendizado profundo surge como uma alternativa de solução para esses problemas, ao permitir que os computadores aprendam com a experiência e compreendam o mundo em função de conceitos hierárquicos. Essa abordagem evita que interventores humanos precisem especificar todo o conhecimento que o computador necessita (GOODFELLOW *et al.*, 2016).

O aprendizado profundo permite que modelos computacionais compostos por múltiplas camadas de processamento possam aprender representações de dados com múltiplos níveis de abstração. Esses métodos vêm melhorando ao longo dos anos o estado da arte nas áreas de reconhecimento de fala, reconhecimento de objetos, detecção de objetos e muitos outras áreas de aplicação, como descobrimento de novas drogas. O aprendizado profundo consegue descobrir estruturas complexas em grandes conjuntos de dados ao fazer uso do algoritmo de retro-propagação para indicar como a máquina deve mudar seus parâmetros internos que são usados para computar a representação em cada camada da representação na camada anterior (LECUN *et al.*, 2015).

Aprendizado representacional é um conjunto de métodos que permite que a máquina seja alimentada com dados brutos e descubra automaticamente as representações necessárias para

detecção ou classificação. Métodos de aprendizagem profunda são métodos de aprendizado representacional com múltiplas camadas de representação. A área de aprendizado profundo está fazendo grandes avanços resolvendo problemas que resistiam às melhores tentativas da comunidade de inteligência artificial por muitos anos (LECUN *et al.*, 2015).

Os modelos de aprendizado profundo se fazem presentes no meio acadêmico há décadas. Os anos recentes mostraram um crescimento no desenvolvimento e estudo nesta área, podendo ter o crescimento atribuído a dois fatores principalmente. Primeiramente, o aumento dos conjuntos de dados reduziu o problema de generalização para as redes neurais. O segundo fator pode ser atribuído ao crescimento do poder computacional e da melhor infraestrutura de software (GOODFELLOW *et al.*, 2016).

Uma arquitetura de aprendizado profundo é uma pilha multicamadas de módulos simples, todos (ou quase) são sujeitos a treinamento, e muitos dos quais computam mapeamentos de entrada-saída não lineares. Cada módulo na pilha transforma sua entrada para aumentar tanto a seletividade como a invariância da representação. Com múltiplas camadas não lineares, entre 5 até 20, um sistema pode implementar funções extremamente complexas que são simultaneamente sensíveis a detalhes minuciosos, como diferenciar um samoieda de um lobo branco, e insensíveis a grandes variações como o fundo, pose, iluminação e objetos no ambiente (LECUN *et al.*, 2015).

O uso de aprendizado profundo já começa a ser utilizado nas mais diversas áreas. Um área notável de interesse é o de sistemas de comunicação, mais especificamente na camada física, como mostra o artigo de O'Shea & Hoydis (2017), no qual a camada física é considerada como um autocodificador, usando com a finalidade de otimizar o sistema em que é aplicado. Outra área que vem utilizando *deep learning* em suas aplicações é a de ciência de materiais, como fica exemplificado no artigo de Choudhary *et al.* (2022), onde mostram que o aprendizado profundo é utilizado para fazer a análise de dados não estruturados e também é feita a identificação automática de características dos materiais.

As aplicações de aprendizado profundo são muitas, seus resultados e sucesso são inquestionáveis. Mas existem alguns problemas na área que merecem ser citados. Um dos principais problemas em aprendizado profundo é que os modelos precisam de muitos dados para serem bem treinados em geral. Grandes conjuntos de dados são necessários para se ter uma garantia

que as máquinas entreguem resultados desejados e acurados. Quanto mais poderoso for o sistema que se quer produzir, mais parâmetros precisam ser considerados e consequentemente se precisa de mais dados (KUMAR; REKHA., 2020).

Para resolverem problemas do mundo real usando aprendizado profundo, a máquina precisa estar equipada com poder computacional adequado. Para garantir eficiência e consumir menos tempo, pesquisadores precisam de GPUs de múltiplos núcleos de alta performance, bem como unidades de processamento. Essas unidades são caras e consomem muita energia (KUMAR; REKHA., 2020). Isso também está causando o aumento de preço das placas de vídeo e valorizando empresas do ramo, como a NVIDIA (DIGITAL-TRENDS, 2023).

2.4 CADEIAS DE MARKOV

Uma cadeia de Markov é um modelo matemático que descreve um processo estocástico em que o estado futuro do sistema depende unicamente do estado atual, sendo independente dos estados anteriores. Em outras palavras, trata-se de um processo com a estrutura markoviana, ou seja, um processo sem memória. Uma cadeia de Markov pode ser definida como uma sequência de variáveis aleatórias $X_1, X_2, X_3, ..., X_n$, onde cada X_n assume valores em um conjunto finito e enumerável de estados S (MEYN; TWEEDIE, 2009).

A estrutura markoviana pode ser definida pela equação,

$$Pr(X_{n+1} = x_{n+1}|X_n = x_n, X_{n-1} = x_{n-1}, ..., N_1 = n_1) = Pr(X_{n+1}) = x_{n+1}|X_n = x_n) \quad (2.4)$$

para todo x_1,$x_2, ..., x_n, x_{n+1} \in X$ (COVER; THOMAS, 2012).

2.5 APLICAÇÃO DO GARGALO DA INFORMAÇÃO EM APRENDIZADO PROFUNDO.

Com os conceitos fundamentais da Teoria da Informação, do Princípio do Gargalo da Informação e do aprendizado profundo já estabelecidos nas seções anteriores, esta seção se concentra na aplicação prática do gargalo da informação em modelos de aprendizado profundo. O Princípio do Gargalo da Informação, introduzido por Tishby *et al.* (1999), propõe que a eficiência no

processamento de informações pode ser alcançada ao focar apenas nos dados mais relevantes e reduzir os irrelevantes. Essa abordagem é particularmente útil no contexto de aprendizado profundo, onde a complexidade dos modelos e a quantidade de dados podem ser significativas.

Nesta seção, serão discutidos estudos recentes que aplicaram o Princípio do Gargalo da Informação em diferentes contextos de aprendizado profundo. Serão analisadas as metodologias utilizadas, os resultados obtidos e as implicações dessas aplicações.

O objetivo de qualquer aprendizado supervisionado é o de capturar e eficientemente representar a informação relevante na variável de entrada sobre a variável de saída. Isto pode ser entendido à luz da obtenção da estatística suficiente mínima (ALEMI *et al.*, 2017). A interpretação da teoria da informação sobre este conceito sugere uma maneira para se fazer isso: encontre um mapeamento comprimido ao máximo da variável de entrada que preserve o máximo de informação possível sobre a variável de saída. E isto é justamente o objetivo do método do gargalo da informação (TISHBY; ZASLAVASKY, 2015).

Alguns problemas interessantes surgem quando se aplica este princípio em DNNs, que pode ter sua arquitetura observada na Figura 2.1. Primeiramente, a estrutura em camadas da rede gera uma cadeia de Markov sucessiva de representações intermediárias, que juntas formam, aproximadamente, estatísticas suficientes. Cada camada da rede pode agora ser quantizada pela quantidade de informação que ela retêm sobre a variável de entrada, sobre a (desejada) variável de saída, bem como sobre a saída prevista da rede. A estrutura Markoviana e desigualdades de processamento de dados permitem que possa ser analisada a eficiência das representações internas das camadas escondidas da rede, o que não é possível com outras medidas de distorção/erro (TISHBY; ZASLAVASKY, 2015).

As redes neurais profundas (DNNs) são compostas por múltiplas camadas de neurônios artificiais, ou unidades, projetadas para aprender representações hierárquicas dos dados, tornando-as altamente eficazes em diversas aplicações de aprendizado de máquina. Tipicamente, a entrada X é uma variável de alta dimensionalidade que representa os dados em um formato de baixo nível, como os pixels de uma imagem, que são passados pelas camadas da rede neural até chegar na camada final $\hat{Y}$, camada que representa as categorias ou valores desejados. O sucesso das DNNs em identificar essas características está relacionado ao processamento sequencial dos dados, onde cada camada oculta transforma a saída da camada anterior, permitindo a

construção de representações distribuídas e abstratas em níveis progressivamente mais elevados (TISHBY; ZASLAVASKY, 2015).

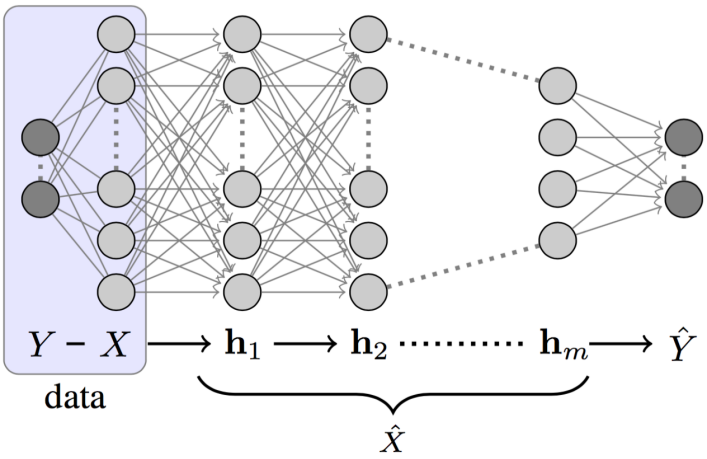


Figura 2.1. Estrutura de uma DNN (ALEMI *et al.*, 2017).

Na Figura 2.1, os círculos representam os neurônios da rede neural, que são as unidades responsáveis pelo processamento das informações. Cada neurônio recebe dados da camada anterior, e transmite o resultado para os neurônios da próxima camada. As setas, por sua vez, simbolizam as conexões entre os neurônios, representando os pesos sinápticos que determinam a influência de um neurônio sobre outro. Esses pesos são ajustados durante o treinamento para otimizar o desempenho da rede. A informação flui na direção das setas, começando na camada de entrada, passando por camadas ocultas, onde ocorre a extração progressiva de características relevantes, e terminando na camada de saída, que gera a predição final do modelo. Esse processo permite que a rede aprenda representações cada vez mais abstratas e eficazes dos dados de entrada, tornando-a capaz de resolver tarefas complexas, como classificação de imagens e reconhecimento de padrões.

A habilidade computacional de uma unidade simples na rede é limitada, e é muitas vezes modelada com um neurônio sigmoide. Neurônios sigmoides são um tipo de unidade funcional em redes neurais artificial que fazem uso da função sigmoide como função de ativação. É amplamente utilizado por conta de sua capacidade de mapear entradas em valores contínuos no intervalo entre 0 e 1, o que é útil quando se está modelando probabilidades ou decisões binárias (IBM, 2024).

Dada uma arquitetura particular, treinar uma rede é reduzido a aprender os pesos entre cada camada. Isso geralmente é feito por métodos estocásticos de descida por gradiente, que visam a minimizar alguns erros de predição, ou distorção, entre as saídas desejadas e previstas, Y e $\hat{Y}$, dada a entrada X (TISHBY; ZASLAVASKY, 2015).

O método do gargalo da informação foi introduzido como um princípio da teoria da informação para extrair informações relevantes que uma variável aleatória de entrada $X \in \mathbf{X}$ contenha sobre uma variável aleatória de saída $Y \in \mathbf{Y}$. Dada sua distribuição de união $p(X,Y)$, a informação relevante é definida como a informação mutua I (X;Y), onde assumimos uma dependência estatística entre X e Y. Neste caso, Y implicitamente determina as características relevantes e irrelevantes em X. Uma representação ótima de X capturaria as características relevantes e comprimiria X ao descartar as partes irrelevantes que não contribuem para a predição de Y (TISHBY; ZASLAVASKY, 2015).

Matematicamente, a parte relevante de X que diz respeito a Y, denotada por $\hat{X}$, é a estatística suficiente mínima de X a respeito de Y. Isto é, é o mapeamento mais simples de X que captura a informação mútua I(X;Y). Podemos então assumir a cadeia de Markov $Y \rightarrow X \rightarrow \hat{X}$ e minimizar a informação mutua $I(X,\hat{X})$ para obter as estatísticas mais simples, sob uma limitação em $I(\hat{X}, Y)$. Encontrar uma representação ótima de $\hat{X} \in \hat{\mathbf{X}}$ é formulado como a minimização do lagrangiano da equação 2.2, sujeitado à limitação da cadeia de Markov. O multiplicador positivo lagrangiano β opera como um parâmetro de troca entre a complexidade (taxa) da representação, $R = I(X; \hat{X})$, e a quantidade de informação relevante preservada $Iy = I(\hat{X}; Y)$. Otimamente, as DNNs devem aprender a extrair as características mais informacionalmente eficientes, ou as estatísticas suficientes mínimas, com a arquitetura mais compacta (menor número de camadas, com o menor número de unidades entre cada camada) (TISHBY; ZASLAVASKY, 2015).

Como mostrado na Figura 2.1, cada camada da DNN processa entradas apenas de sua camada anterior, o que significa que as camadas da rede formam uma cadeia de Markov. Uma consequência do que é conhecido como *data processing inequality*(DPI) é que a informação sobre Y que for perdida em uma camada não pode ser recuperada pelas camadas superiores. Isto é, para qualquer $i \geq j$ temos que (com $\mathbf{h}$ sendo a saída de cada camada)

$$I(Y;X) \geq I(Y;\mathbf{h}_j) \geq I(Y;\mathbf{h}_i) \geq I(Y;\hat{Y}). \tag{2.5}$$

Alcançar igualdade na Eq. 2.5 é possível se e somente se cada camada é uma estatística suficiente de sua entrada. Para se requisitar não apenas a informação mais relevante em cada camada, mas também a representação mais concisa da entrada, cada camada deve tentar maximizar $I(Y;\mathbf{h}_i)$ enquanto minimiza $I(\mathbf{h}_{i-1};\mathbf{h}_i)$ (TISHBY; ZASLAVASKY, 2015).

Pode-se considerar $I(Y;\hat{Y})$ como o quantificador natural da qualidade do DNN, já que mede precisamente quanto das características preditivas em X para Y é capturado pelo modelo. Reduzir $I(\mathbf{h}_{i-1};\mathbf{h}_i)$ também tem um interpretação clara de aprendizado como sendo a da descrição mínima do comprimento da camada (TISHBY; ZASLAVASKY, 2015).

2.6 AUTO CODIFICADORES

Auto codificadores são um tipo de arquitetura de rede neural, na qual usamos a rede para a tarefa de aprendizado de representação, onde os valores de saída são iguais aos de entrada, ou seja, $y(i) = x(i)$. Especificamente, projetaremos uma arquitetura de rede neural de modo a impor um gargalo na rede que força uma representação de conhecimento compactada da entrada original. Caso os recursos de entrada sejam independentes um do outro, essa representação seria muito difícil ou impossível de ser encontrada, pois não haveria uma correlação entre os dados. No entanto, se houver algum tipo de estrutura e correlação nos dados, essa estrutura poderá ser aprendida e aproveitada ao forçar a entrada de dados no gargalo da rede (ACADEMY, 2024).

Esse tipo de rede é composta por duas partes:

- Codificador: É a parte da rede que compacta a entrada em uma representação de espaço latente (codificando a entrada). Pode ser representado por uma função de codificação $h = f(x)$.
- Decodificador: Esta parte tem como objetivo reconstruir a entrada da representação do espaço latente. Pode ser representado por uma função de decodificação $r = g(h)$.

Uma representação da arquitetura de um auto codificador pode ser observada na Figura 2.2

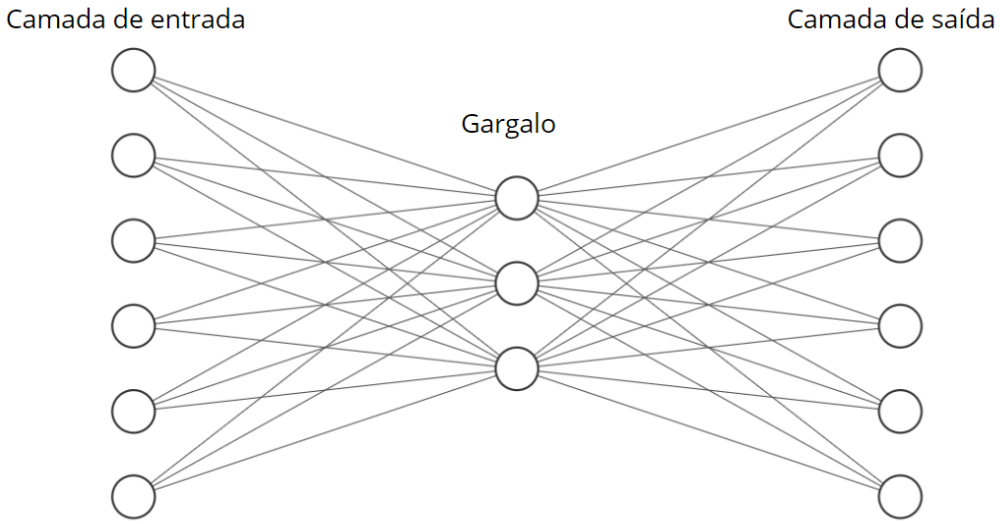


Figura 2.2. Arquitetura de um auto codificador.

Partindo do pressuposto de que os auto codificadores copiem a entrada para a saída, eles não seriam muito úteis. Para encontrar propósito para o treinamento deles, a camada latente deve ter propriedades úteis.

Essa utilidade pode ser alcançada ao se criar certar restrições na tarefa de cópia. Por exemplo, restringindo a camada latente h a ter dimensões menores que X, neste caso, o auto codificador é chamado de incompleto. Quando se treina uma representação incompleta, força-se o auto codificador a aprender os recursos mais importantes dos dados de treinamento. Caso haja muita capacidade no auto codificador, ou uma camada latente com a mesma dimensionalidade da camada de entrada, o auto codificador poderá aprender a realizar a tarefa de cópia sem aprender nenhuma informação útil sobre a distribuição dos dados (ACADEMY, 2024).

Os autocodificadores apresentam muitas aplicações, entre elas estão *denoising* (remoção de ruídos) e a redução de dimensionaldiade para visualização de dados (BERMANN; STRYKER, 2023). Com a realização dos treinamentos de forma correta, os auto codificadores podem aprender projeções da dados mais interessantes que técnicas mais básicas, como o PCA (Principal Component Analysis).

Os auto codificadores aprendem automaticamente a partir de exemplos de dados, tornando fácil o treinamento de instâncias especializadas do algoritmo que terão um bom desempenho em um tipo específico de entrada e que não requererão nenhuma nova técnica de recursos,

apenas dados de treinamento apropriados. Auto codificadores são treinados para preservar o máximo de informações possíveis quando uma nova entrada é alimentada ao codificador e depois repassada ao decodificador, mas também é almejado um treinamento que faça com que essas representações tenham propriedades úteis (ACADEMY, 2024).

Existem diferentes tipos de auto codificadores que servem diferentes tipos de finalidades, a seguir serão apresentados os tipos que são englobados pelo propósito deste projeto.

2.7 AUTOCODIFICADORES VARIACIONAIS

Os auto codificadores variacionais (VAEs) são uma classe específica de autocodificadores, introduzidos por Kingma e Welling em 2013 (KINGMA; WELLING, 2013b), os VAEs combinam princípios de autocodificadores tradicionais com técnicas de inferência variacional, proporcionando um método poderoso para a modelagem de distribuições complexas e a geração de dados sintéticos. No contexto do *Deep Learning*, os VAEs são amplamente utilizados para tarefas de redução de dimensionalidade, geração de imagens e processamento de sinais (BERMANN; STRYKER, 2023).

Os VAEs não apenas capturam as características mais relevantes dos dados de entrada, mas também permitem a manipulação desses dados no espaço latente, o que pode ser extremamente útil para diversas aplicações, como a geração de novas amostras semelhantes às do conjunto de treinamento. Nesta seção, será explorado o funcionamento dos autocodificadores variacionais, destacando suas principais características e diferenças em relação aos autocodificadores tradicionais.

Um autocodificador variacional combina conceitos de aprendizado profundo e estatística bayesiana para aprender representações latentes contínuas de dados e gerar novas amostras semelhantes aos dados originais (ACADEMY, 2024). Na figura 2.3 pode ser observado um comparativo simplificado dos processos de um auto codificador clássico e um variacional.

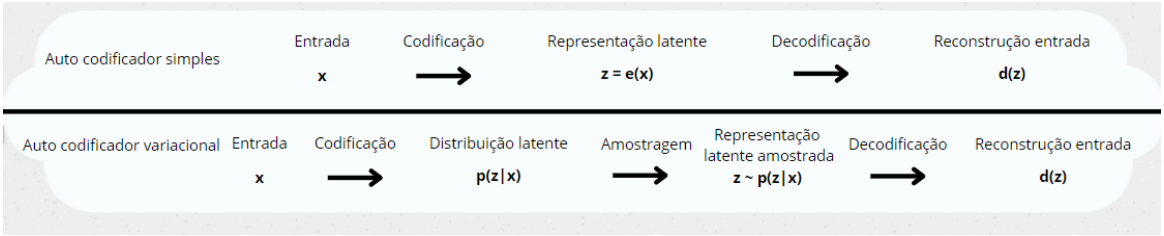


Figura 2.3. Comparativo AE e VAE.

Explicando a Figura 2.3 ambos os modelos recebem o parâmetro x como dado de entrada. As diferenças a parecem a partir da codificação, onde o autocodificador simples transforma a entrada x em uma representação determinística z, enquanto o autoencoder variacional aprende uma distribuição probabilística. Existe um processo exclusivo ao modelo variacional que é o de amostragem, esse processo assegura que z seja uma amostra da distribuição $q(z|x)$, mas ainda seja diferenciável em relação aos parâmetros do codificador. Por fim, no processo de decodificação, o autocodificador simples mapeia z de volta para o espaço de entrada x, tentando a recontrução de x a partir de z. O decodificador do autocodificador variacional também mapeia z para o espaço de entrada, mas neste caso, a reconstrução é modelada como uma distribuição.

A base matemática dos VAEs tem pouca relação com a dos autocodificadores clássicos. Eles são chamados de "autocodificadores"apenas porque o objetivo final de treinamento que deriva desta configuração tem um codificador e um decodificador, e remete a um autocodificador tradicional (DOERSCH, 2016).4 Primeiramente, como se escolhem as variáveis latentes z de forma que se capture informação latente? Ele precisa não apenas o dígito, mas o ângulo no qual o dígito é desenhado, a largura do traço, e também propriedades estilísticas abstratas. Idealmente, queremos evitar decidir manualmente quais informações cada dimensão z codifica. Também quer-se evitar explicitamente descrever as dependências, estrutura latente, entre as dimensões de z. Os VAEs tomam uma abordagem não usual para lidar com este problema: eles assumem que não existe interpretação simples das dimensões de z, então afirmam que amostras de z podem ser desenhadas a partir de uma simples distribuição normal multivariada, $\mathcal{N}(0,I)$, onde 0 é o vetor de médias, indicando que a média de cada dimensão da distribuição é zero e I é a matriz indentidade, que funciona como a matriz de covariância, usada para regularização e normalização (DOERSCH, 2016). O desenvolvimento matemático pode ser visualizado em (DOERSCH, 2016; KINGMA; WELLING, 2013a) e uma representação gráfica do modelo de

um auto codificador variacional pode ser visualizado na Figura 2.4.

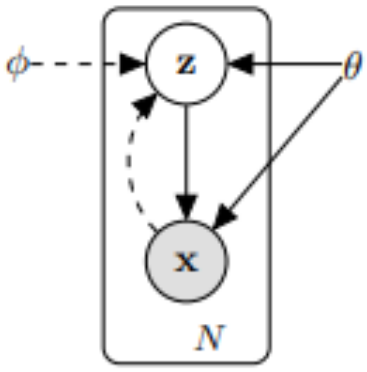


Figura 2.4. Modelo VAE padrão (KINGMA; WELLING, 2013b).

.

Na Figura 2.4, as linhas sólidas denotam o modelo generativo e as linhas tracejadas denotam a aproximação variacional.

O símbolo ϕ representa os parâmetros do modelo codificador, responsáveis por modelar a distribuição aproximada. Recebe um dado de entrada x e estima uma distribuição probabilística sobre a variável latente z, geralmente modelada como uma normal multivariada.

Já o simbolo θ, representa os parâmetros do decodificador. Toma uma amostra da variável latente z e produz uma reconstrução ou geração dos dados x. Os parâmetros θ são ajustados para maximizar a probabilidade dos dados reconstruídos ou gerados, tentando garantir com que o modelo capture da melhor forma a estrutura dos dados originais.

Os parâmetros variacionais ϕ são aprendidos conjuntamente com os parâmetros θ do modelo generativo. Essas duas variáveis trabalham em conjunto para realizar o aprendizado no VAE, buscando o equilíbrio entra a reconstrução dos dados e a regularização do espaço latente (KINGMA; WELLING, 2013a).

2.8 DEEP VARIATIONAL INFORMATION BOTTLENECK

A parte prática usa como base o modelo proposto no artigo "*Deep Variational Information Bottleneck*"(ALEMI *et al.*, 2017). Nele, pesquisadores do Google propõem um método chamado *Deep Variational Information Bottleneck* (VIB), ou Deep VIB, que realiza uma aproximação

variacional para o gargalo da informação de Tishby *et al.* (1999).

Dado um conjunto de dados de entradas X e saídas Y, define-se uma representação intermediária Z. Para alcançar o objetivo de predizer Y a partir de Z, utiliza-se a técnica de descida por gradiente. É desejável maximizar a informação mútua $I(Z,Y)$. Esse objetivo tem sentido lógico, já que não quer se perder nenhuma informação importante (ALEMI *et al.*, 2017).

Outro objetivo a ser alcançado é o de minimizar a informação passando por Z. Para atingir tal objetivo, a informação mútua $I(Z,X)$ deve ser minimizada, fazendo presente os conceitos teóricos do gargalo da informação. Basicamente, Z deve manter a informação relevante sobre Y, mas descartar o resto de X (ALEMI *et al.*, 2017).

A informação mútua é difícil de se medir, uma vez que requer o conhecimento das distribuições de probabilidade ótimas. Neste caso, usa-se uma aproximação delas por meio de inferência variacional, o qual é o método proposto por Alemi *et al.* (2017).

O modelo assume uma cadeia de Markov $Y \rightarrow X \rightarrow Z$. Como dito anteriormente, o objetivo do modelo VIB é maximizar $I(Z,Y)$ enquanto minimiza $I(Z,X)$. Para tornar este problema tratável, os pesquisadores fizeram uso de inferência variacional para aproximar $I(Z,Y)$ e $I(Z,X)$. Isso é feito através de distribuições aproximadas e reparametrizações, que facilitam a otimização através do gradiente estocástico. O desenvolvimento matemático das funções, principalmente da função perda, pode ser visto com detalhes no artigo de Alemi *et al.* (2017).

A aplicação do modelo VIB pode ser também visualizada no artigo de Dai *et al.* (2018), que utiliza desta técnica para realizar a compressão de redes neurais, e também no artigo de Srivastava *et al.* (2020), que utiliza a técnica para a compressão de redes sequenciais utilizadas para reconhecimento de ações humanas.

CAPÍTULO 3

METODOLOGIA

Este capítulo é dedicado à parte de metodologia, onde são detalhados os procedimentos e técnicas adotadas para a realização deste trabalho, cujo objetivo é avaliar a aplicação do Princípio do Gargalo da Informação em um modelo de aprendizado profundo. Essa pesquisa foi realizada seguindo procedimentos experimentais fundamentados, com o intuito de proporcionar resultados replicáveis. Os conjuntos de dados usados, que podem ser visualizados nas Figuras 3.1 à 3.3, para o treinamento dos modelos são consolidados no meio de pesquisa de aprendizado de máquina. A escolha de um conjunto já existente se dá tendo em vista que a geração de um novo conjunto de dados próprios levaria um tempo demasiado longo para ser feita, não condizente com o tempo usualmente definido para um projeto de graduação, também foram coletadas métricas de acurácia e perda para a comparação entre os modelos.

A perda e a acurácia são métricas fundamentais na avaliação do desempenho de modelos de aprendizado de máquina. A perda (loss), ou custo, quantifica o erro do modelo ao comparar suas previsões com os valores reais, sendo calculada por uma função matemática específica, como a Cross-Entropy Loss para classificação ou Mean Squared Error (MSE) para regressão. Quanto menor a perda, melhor o modelo está ajustado aos dados. Já a acurácia mede a proporção de previsões corretas feitas pelo modelo em relação ao total de amostras avaliadas, expressa em percentual. Enquanto a perda fornece uma medida contínua do erro, permitindo ajustes mais refinados durante o treinamento, a acurácia dá uma visão mais direta da eficácia do modelo, especialmente em tarefas de classificação.

3.1 PREPARAÇÃO DE DADOS

Os dados foram coletados utilizando os conjuntos de dados MNIST, Fashion MNIST e CIFAR-10, amplamente utilizados em pesquisas de Deep Learning para tarefas de classificação

de imagens.

MNIST é um grande banco de dados de pequenas imagens de 28x28 pixels de dígitos escritos à mão, dígitos que variam entre 0 e 9. Consiste de um total de 70.000 imagens, com o conjunto de treinamento tendo 60.000 e o conjunto de testes tendo 10.000. Todas as imagens são rotuladas com os respectivos dígitos que elas representam (WIKIPEDIA, 2024d).

Fashion MNIST foi criado como um substituto mais desafiador para o clássico conjunto de dados MNIST. Ele consiste em 70.000 imagens em tons de cinza, distribuídas em 10 categorias diferentes de artigos de moda, como camisetas, calçados, bolsas, entre outros. Cada imagem tem uma resolução de 28x28 pixels (WIKIPEDIA, 2024c).

O CIFAR-10 (Canadian Institute for Advanced Research) é um dos datasets mais populares na área de aprendizado de máquina e visão computacional. Ele consiste em 60.000 imagens coloridas de 32x32 pixels, distribuídas em 10 classes diferentes, com 6.000 imagens por classe. As classes são balanceadas e incluem objetos comuns como aviões, automóveis, pássaros, gatos, cervos, cães, sapos, cavalos, navios e caminhões. O CIFAR-10 é relativamente mais desafiador que os conjuntos anteriores porque as imagens são pequenas e contém variações significativas em termos de ângulo, iluminação e plano de fundo, o que requer modelos mais robustos para alcançar alto desempenho (WIKIPEDIA, 2024a).

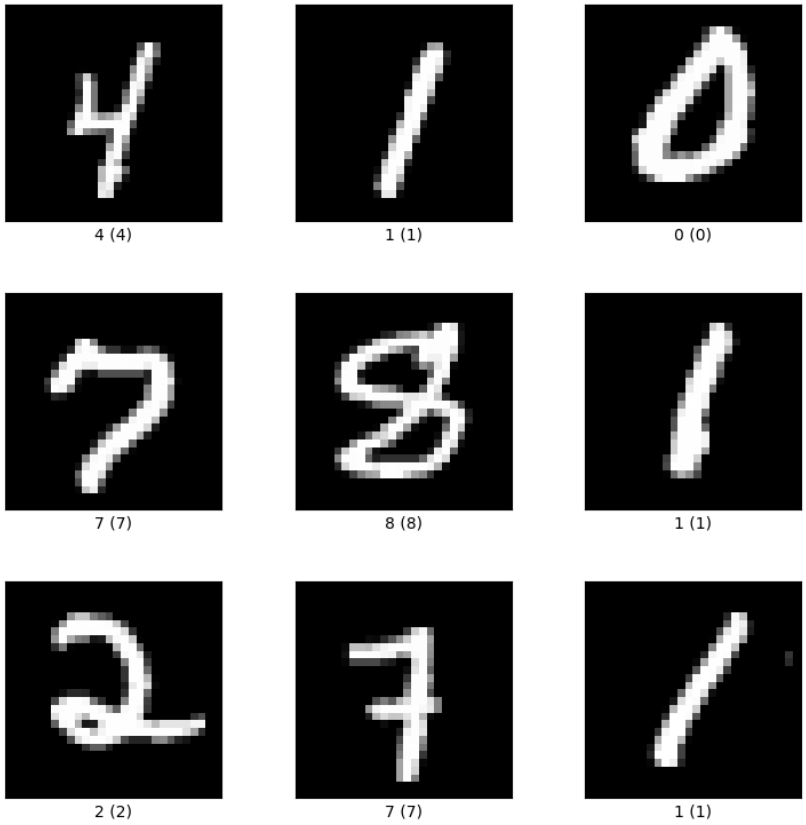


Figura 3.1. Exemplos imagens MNIST.

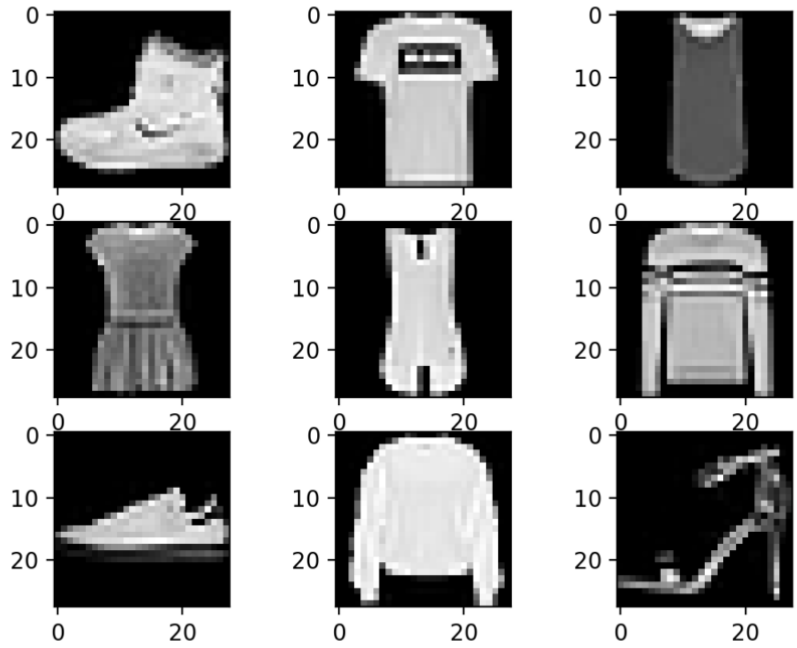


Figura 3.2. Exemplos imagens Fashion MNIST.

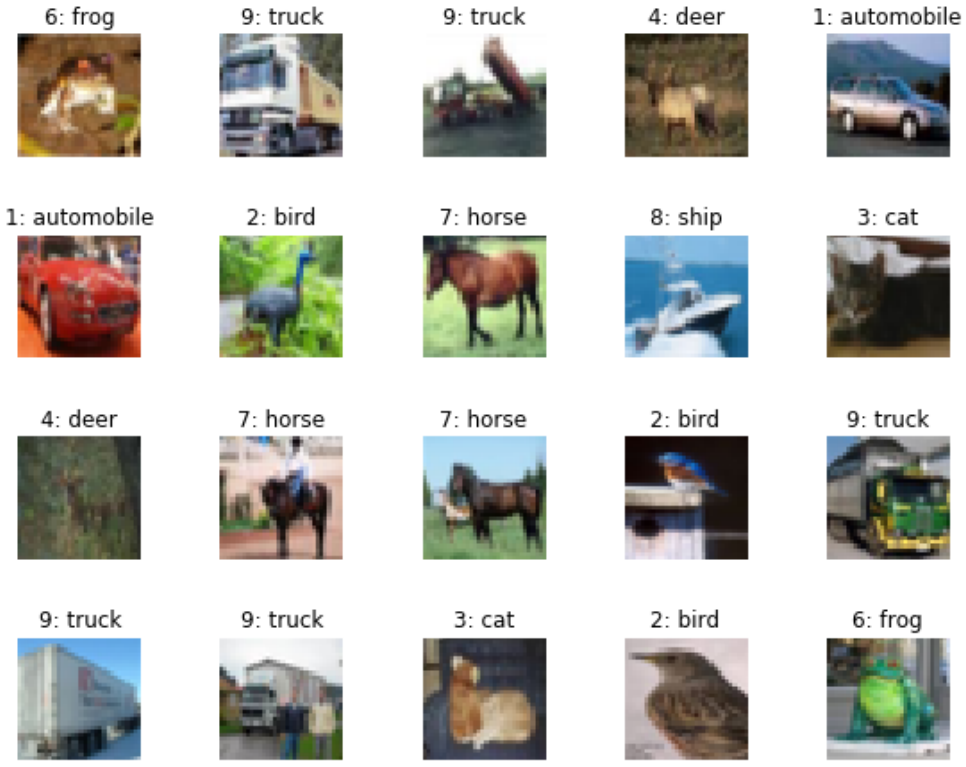


Figura 3.3. Exemplos imagens CIFAR-10.

3.2 FERRAMENTAS E AMBIENTE DE DESENVOLVIMENTO

Para a realização dos experimentos computacionais, foi utilizado o ambiente Google Colaboratory (Colab). Colab é um serviço hospedado de Jupyter Notebook. O ambiente não precisa de configurações prévias e provê acesso gratuito a recursos computacionais, incluindo GPUs e TPUs. É uma ferramenta muito útil para aprendizado de máquina, ciência de dados e educação em geral. O código usado para a implementação do modelo foi escrito em python, usando a biblioteca pytorch, muito utilizada para aprendizado de máquina, e outras bibliotecas usuais para realização de operações matemáticas e visualização de dados.

Como base do código, foi utilizado o código cedido pelo pesquisador Udeepa Meepegama em seu GitHub (MEEPEGAMA, 2020). O código faz uma implementação basilar da teoria do VIB proposta por Alemi *et al.* (2017). No código, tem-se a criação do modelo utilizando o método VIB para o banco de dados MNIST. A partir deste código, fez-se a implementação de algumas métricas úteis para uma melhor observação da aplicação do VIB, além um cenário de implementação próprio.

Nesse cenário próprio, faz-se a comparação do modelo treinado usando VIB com um autocodificador "clássico"com a camada Z de dimensão equivalente a do VIB. Porém, ao invés de usar um decodificador na saída desta camada latente, como é o usual em autocodificadores, conectamos à uma camada SoftMax para realizar a classificação dos dados. Os resultados de ambas as arquiteturas serão comparados no capítulo de resultados.

Com a realização deste experimento tem-se como objetivo estudar a aplicabilidade do modelo VIB. Buscando mostrar, por meio dos dados coletados durante os cenários experimentais, se foi possível ou não chegar à alguma conclusão que mostre que os benefícios propostos pela teoria foram alcançados e justificados.

3.2.1 Arquitetura dos modelos

3.2.1.1 VIB

O modelo proposto por Alemi *et al.* (2017), e aqui implementado, segue a arquitetura que pode ser visualizada na Figura 3.1. A parte do modelo correspondente ao codificador estocástico

é implementado como um MLP (Multilayer Perceptron) com as dimensões de suas camadas seguindo a ordem $784 - 1024 - 1024 - 2K$, onde K é o tamanho do gargalo.

Diferente dos autocodificadores tradicionais, este modelo não possui um decodificador convencional que reconstrói os dados de entrada. Em vez disso, a saída da camada latente é diretamente conectada a uma camada softmax, cuja função é classificar os dados em diferentes categorias. O motivo dessa escolha arquitetural é que, no *Variational Information Bottleneck* (VIB), o foco não está na reconstrução dos dados originais, mas sim na extração e compressão das informações mais relevantes para a tarefa de classificação. Dessa forma, o modelo aprende uma representação latente que preserva apenas os aspectos essenciais da entrada, descartando informações redundantes ou irrelevantes

Esse modelo foi aplicado com $K = 256$ nos seguintes conjuntos de dados: MNIST, Fashion MNIST e CIFAR-10

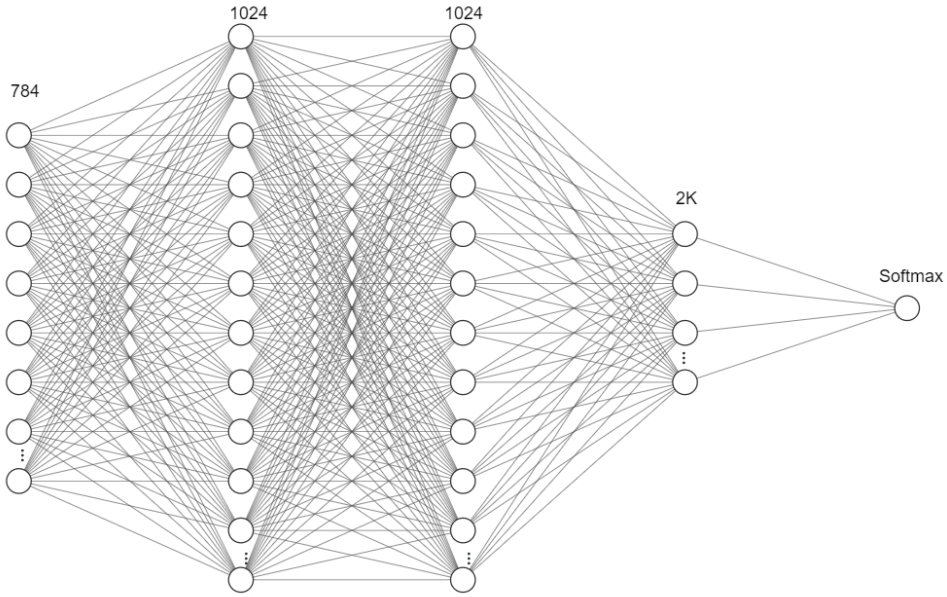


Figura 3.4. Arquitetura VIB.

Na Tabela 3.1 podem ser visualizadas as especificações do modelo.

Encoder	
Camada linear	$784 \rightarrow 1024$
Função de ativação	ReLU
Camada linear	$1024 \rightarrow 1024$
Função de ativação	ReLU
Camada Latente	
fc mu	Camada Linear $1024 \rightarrow 256$
fc std	Camada Linear $1024 \rightarrow 256$ + Função Softplus
Decoder	
Camada linear	$256 \rightarrow$ N classes
Função de ativação	Softmax
Função perda	
Perda combinada	Cross Entropy Loss (CE) + KL Divergence (β * KL)

Tabela 3.1. Especificações VIB

3.2.1.2 Autocodificador Modificado

Para o segundo modelo de experimentação, foi utilizado uma arquitetura nomeada "Auto codificador modificado"(AEM). A arquitetura do modelo foi adaptada para fazer classificação usando elementos de um autocodificador, mas com uma mudança importante: ao contrário de decodificar a saída da camada latente de volta para a entrada original (como em um auto codificador tradicional), a saída da camada latente é conectada a uma camada softmax para classificação. Esse mudança arquitetural fez-se com o intuito de aproximar a arquitetura de um autocodificador clássico à do modelo VIB, para uma melhor análise comparativa.

As dimensões das camadas do autocodificador modificado são as mesmas do VIB, para manter uma mesma base de comparação entre os dois modelos.

Na Tabela 3.2 podem ser visualizadas as especificações do modelo.

Encoder	
Camada linear	784 (achatada) → 1024
Função de ativação	ReLU
Camada linear	1024 → 1024
Função de ativação	ReLU
Camada latente	1024 → z
Camada de classificação	
Camada linear	z → N classes
Função de ativação	Softmax
Função perda	
Perda	Cross Entropy Loss (CE)

Tabela 3.2. Especificações auto codificador Modificado

No terceiro conjunto de dados (CIFAR-10) foram necessárias algumas alterações no modelo em relação aos outros cenários. As mudanças na arquitetura do auto codificador modificado ao adaptá-lo do MNIST e FMNIST para o CIFAR-10 foram necessárias devido à maior complexidade e à natureza das imagens no CIFAR-10, o CIFAR-10 possui imagens coloridas de 32x32 pixels, com três canais de cor (RGB).

Dado que as imagens do CIFAR-10 são maiores, coloridas e mais complexas em comparação com o MNIST, o codificador precisa ser mais robusto e capaz de capturar características mais complexas. Porém, a camada latente ainda mantêm a mesma dimensão dos conjuntos de dados anteriores, para manter a consistência no gargalo. As mudanças feitas envolveram alterações para trabalhar com três canais de cores, uma vez que as imagens não são mais em escala de cinza como as anteriores, além do uso de uma camada diferente da que estava sendo utilizada. As camadas lineares foram substituídas por camadas conv2D, pois geralmente esse tipo de camada trabalha melhor com dados de imagem, nos cenários anteriores não se fizeram necessárias pois os dados mostraram-se simples

Para acomodar essa diferença, as dimensões das camadas do codificador foram ampliadas, aumentando a capacidade do modelo para capturar as características mais complexas presentes nas imagens coloridas e de maior resolução do CIFAR-10. Houve um aumento na dimensão da camada de entrada, pois os dados agora têm 3072 de tamanho, dados pela multiplicação $32x32x3$, os dois primeiros termos referem-se à escala das novas imagens e o terceiro termo

refere-se aos 3 canais de cores (RGB), pois as imagens agora são coloridas. A camada latente foi mantida com o tamanho de 256.

As mudanças referentes ao dataset CIFAR-10 também foram necessárias de serem feitas no modelo VIB, alterando seus parâmetros de entrada para atenderem às mudanças do novo conjunto de dados.

3.2.2 Implementação

Ambos os modelos implementados foram treinados nos três conjuntos de dados citados anteriormente. Para os conjuntos de dados MNIST e Fashion MNIST não foi necessária nenhuma alteração além dos dados usados nos modelos. No conjunto CIFAR-10, porém, foi necessário realizar algumas alterações em ambos os modelos, pois o uso da arquitetura original, testada nos dois primeiros cenários anteriores, indicou para este conjunto um desempenho aquém do razoável.

Para uma melhor avaliação dos experimentos, foram calculadas as seguintes métricas durante a execução dos mesmos: acurácia e perda, tanto de treinamento quanto de validação, tempo de simulação. Os modelos foram treinados em cada cenário, usando 30 épocas com os conjuntos MNIST e Fashion MNIST, e para o conjunto CIFAR-10 foram treinados em 50 épocas.

CAPÍTULO 4

RESULTADOS

Neste capítulo são apresentados os resultados obtidos a partir dos experimentos realizados durante a aplicação dos modelos, com foco principalmente no modelo VIB.A análise dos resultados permite verificar a eficácia do modelo, bem como a validade de sua aplicação, comparando o desempenho dos modelos utilizados sob diferentes métricas, como acurácia, perda e uso de recursos computacionais. Inicialmente, serão expostos os resultados do modelos treinado sem a aplicação do Princípio do Gargalo da Informação, servindo como modelo base de comparação com o que aplica o princípio. Em seguida, são discutidos os resultados do modelo que incorpora essa abordagem, destacando as melhorias ou possíveis limitações observadas. Neste capítulo também será feita uma interpretação dos resultados, tentando traçar uma relação com o cumprimento, ou não, dos objetivos propostos, bem como com a fundamentação teórica previamente apresentada.

4.1 RESULTADOS DO MODELO AEM

4.1.1 Primeiro conjunto

Os primeiros resultados coletados para o modelo de autocodificador modificado referem-se ao conjunto de dados MNIST. Ao final de 30 épocas de treinamento as métricas coletadas mostram que o modelo obteve êxito em sua função de classificação, as quais podem ser observadas na Tabela 4.1. O experimento foi realizado em 10 execuções independentes e a média calculada.

Métrica	**Valor**
Acurácia	98.30%
Tempo de treinamento	610 s

Tabela 4.1. Métricas finais modelo AEM MNIST no conjunto de validação

Os resultados para este conjunto de dados mostraram-se bastantes satisfatórios. Deve-se levar em consideração que é um conjunto é simples, mas que serve como uma forma inicial de validação deste modelo proposto para comparar com o modelo VIB. Na Figura 4.1 podem ser observados os gráficos para perda e acurácia por época, de forma a se analisar a evolução das métricas durante o treinamento. As curvas dos gráficos da figura 4.1 exibem resultados positivos em suas curvas,

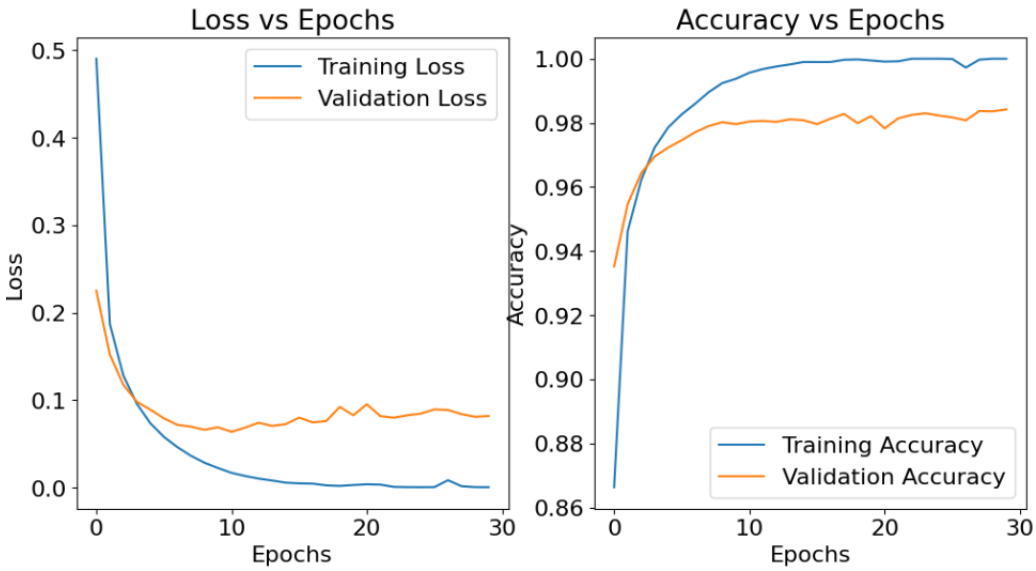


Figura 4.1. Gráficos de métricas para o conjunto MNIST.

O tempo de treinamento faz referência apenas ao treinamento do modelo em si, o experimento como um todo dura alguns minutos a mais, pois é necessário fazer o download do conjuntos de dados toda vez que a sessão no ambiente colab é encerrada. De qualquer forma, pode-se entender como um tempo de treinamento aceitável, aproximadamente 10 minutos, para um treinamento simples.

A Tabela 4.2 exibe as métricas para a primeira e última épocas.

Métrica	Valor	Época
Perda da validação	0.2251	Inicial
Acurácia da validação	93.52%	Inicial
Perda da validação	0.0816	Final
Acurácia da validação	98.30%	Final

Tabela 4.2. Comparativo métricas modelo MNIST.

Ao fim do treinamento houve, para as métricas de validação, uma redução de 63,75% e aumento de 5,24% para perda e acurácia, respectivamente. Relembrando que para a métrica de perda, quando mais próximo de 0 o valor, melhor. Então, houve uma melhora efetiva em todos as métricas neste primeiro modelo.

4.1.2 Segundo conjunto

O segundo cenário faz referência ao treinamento com o conjunto de dados Fashion MNIST, o qual foi feito também em 30 épocas. Ao final do treinamento, os resultados obtidos também foram satisfatórios, possivelmente devido a este conjunto de dados ser ligeiramente mais complexo que o MNIST clássico, foi obtida uma acurácia ligeiramente menor. O experimento foi realizado em 10 execuções independentes e a média calculada.

Ao final do treinamento foram coletadas as mesmas métricas do primeiro conjunto, que podem ser observadas na Tabela 4.3.

Métrica	**Valor**
Acurácia	89,63%
Tempo de treinamento	586 s

Tabela 4.3. Métricas finais modelo AEM Fashion MNIST no conjunto de validação

Na Figura 4.2 podem ser observados os gráficos para perda e acurácia por época. Os gráficos aqui também apresentam resultados positivos em suas curvas, mostrando que não foi atingido um ponto de *overfitting* ou *underfitting* durante a realização do experimento.

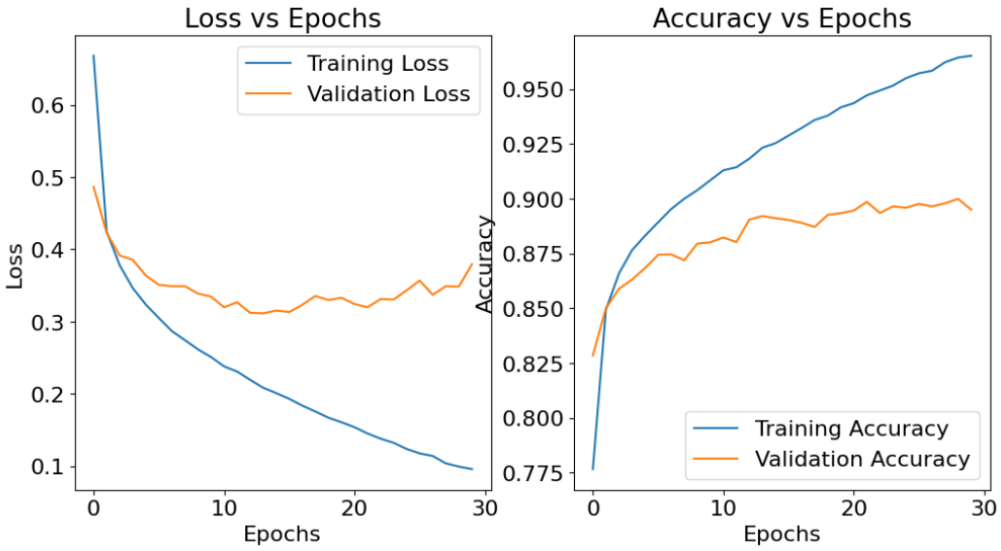


Figura 4.2. Gráficos de métricas para o conjunto Fashion MNIST.

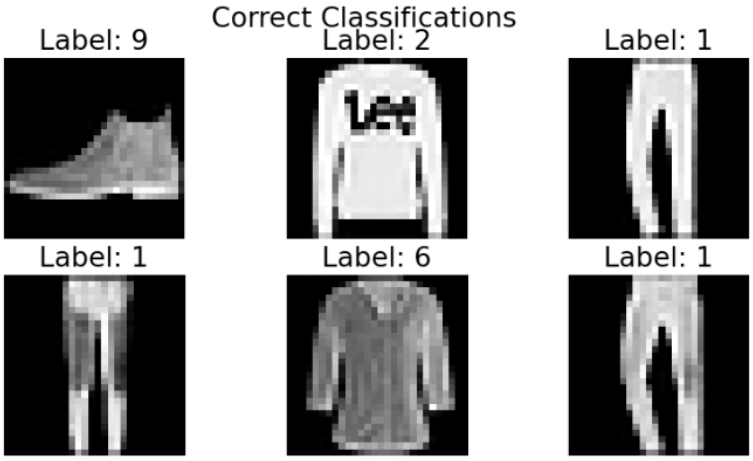


Figura 4.3. Exemplos de classificações corretas para Fashion MNIST.

Na Figura 4.3 podem ser observados algumas imagens classificadas corretamente pelo modelo AEM, para se ter uma confirmação visual dos resultados obtidos. Apesar do segundo cenário apresentar dados ligeiramente mais detalhados, o tempo de treinamento se manteve semelhante ao do primeiro cenário, sendo até ligeiramente menor.

4.1.3 Terceiro conjunto

Para o terceiro conjunto avaliado, CIFAR-10, treinou-se o modelo com 60 épocas. O experimento foi realizado em 10 execuções independentes e a média calculada. As métricas ao fim do experimento podem ser observadas na tabela 4.4.

Métrica	**Valor**
Acurácia	71,33%
Tempo de treinamento	8400 s

Tabela 4.4. Métricas finais modelo AEM CIFAR-10 no conjunto de validação

Nas figuras 4.4 e 4.5 podem ser observados os gráficos de perda e acurácia por época e alguns resultados incorretos das classificações do modelo, respectivamente.

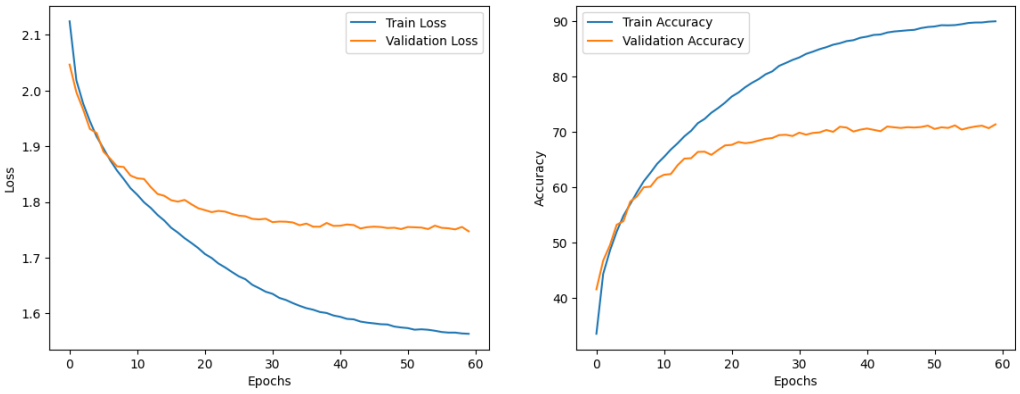


Figura 4.4. Gráficos de métricas para o conjunto CIFAR-10.



Figura 4.5. Exemplos de classificações incorretas para CIFAR-10.

Aqui se pode observar uma grande diferença já em um conjunto de dados um pouco mais complexo. Pelo conjunto CIFAR-10 quantidade de cores e imagens mais robustas, a acurácia do modelo reduziu bastante comparando com os conjuntos anteriores, o valor da perda também cresce bastante quando comparado, chegando em valores unitários, além de o tempo de execução médio ter aumentado consideravelmente.

4.2 RESULTADOS DO MODELO VIB

4.2.1 Primeiro conjunto

A ordem de testes para o modelo VIB foi a mesma do modelo AEM. Começando pelo conjunto de dados MNIST treinado em 30 épocas, também foram realizadas 10 execuções para o cálculo da média, as métricas finais do modelo VIB podem ser visualizadas na Tabela 4.5.

Métrica	**Valor**
Acurácia	98.27%
Tempo de treinamento	736 s

Tabela 4.5. Métricas finais modelo VIB MNIST no conjunto de validação

Aqui novamente os resultados foram positivos. O modelo conseguiu treinar bem com o conjunto de dados, chegando a uma boa acurácia ao fim do treinamento. As Figuras 4.6 e 4.7 exibem os gráficos de perda e acurácia por época e algumas classificações corretas do modelo, respectivamente.

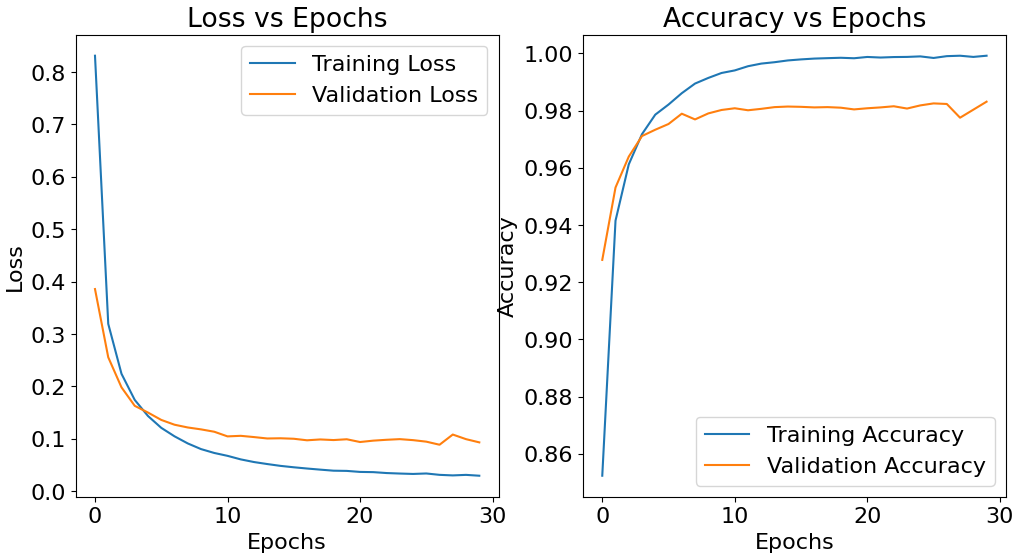


Figura 4.6. Gráficos de métricas VIB para o conjunto MNIST.

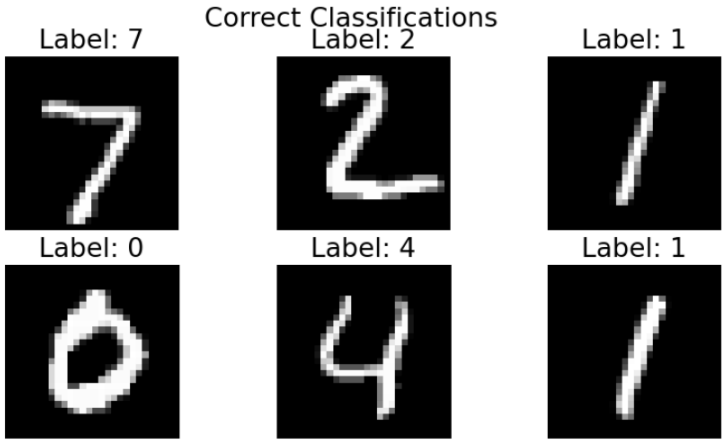


Figura 4.7. Exemplos de classificações corretas do modelo VIB para MNIST.

Os gráficos da Figura 4.6 mostram que o modelo apresentou um comportamento positivo durante seu treinamento, evidenciado pela forma das curvas, não apresentando problemas de performance que evidenciassem problemas na generalização do modelo. Já na Figura 4.7, podem ser observadas uma comprovação visual das classificações corretas do modelo.

4.2.2 Segundo conjunto

A seguir são apresentados gráficos, imagens e comparações feitas sobre o modelo VIB utilizando o conjunto de dados Fashion MNIST. Na Tabela 4.6 podem ser visualizadas as métricas finais do modelo após 30 épocas de treinamento, com 10 execuções independentes para o cálculo da média.

Métrica	Valor
Acurácia	89.20%
Tempo de treinamento	695 s

Tabela 4.6. Métricas finais modelo VIB Fashion MNIST no conjunto de validação

Os resultados finais mostraram-se aceitáveis, havendo pequenos decrementos negativos nas métricas, mas lembrando que a complexidade do problema aumentou. Pode-se observar na Figura 4.8 os gráficos para perda e acurácia por época.

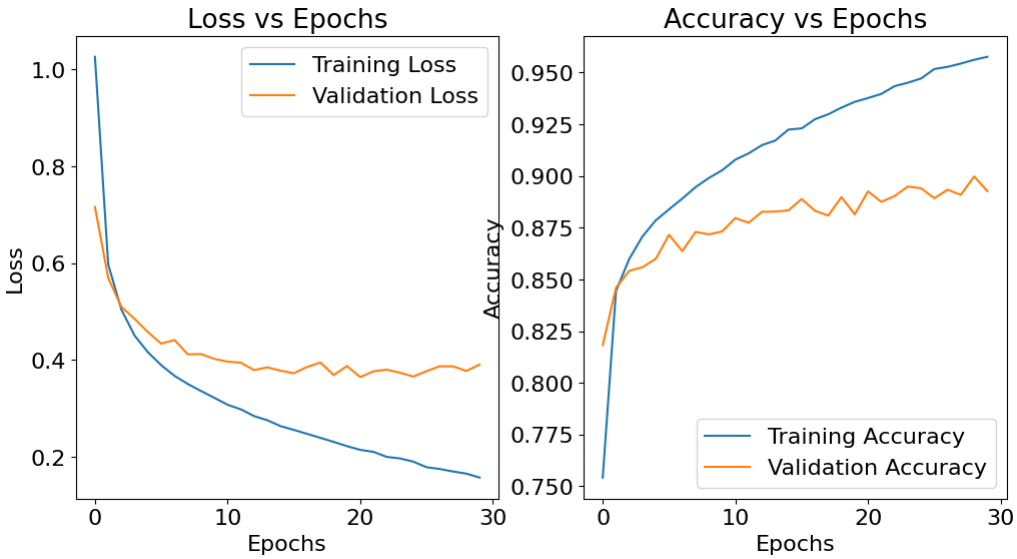


Figura 4.8. Gráficos de métricas VIB para o conjunto Fashion MNIST.

As curvas dos gráficos também mostram que não houve nenhum problema relacionado à performance do modelo durante o treinamento, mostrando até agora resultados positivos para o modelo VIB, mantendo-se próximos ao do modelo AEM.

4.2.3 Terceiro conjunto

Para a última parte expositiva de resultados, temos as métricas coletadas para o modelo VIB com o conjunto de dados CIFAR-10. Ele foi treinado em 60 épocas, e foi calculada a média de 10 execuções independentes do treino e seus resultados podem ser observados na Tabela 4.7.

Métrica	**Valor**
Acurácia	69,29%
Tempo de treinamento	6300 s

Tabela 4.7. Métricas finais modelo VIB CIFAR-10 no conjunto de validação

Repetindo o que se observou no modelo AEM, para o terceiro conjunto de dados o modelo teve uma queda significativa nas métricas. Novamente, os dados do CIFAR-10 são significativamente mais complexos que os dos modelos MNIST, apesar de serem simples. Na Figura 4.9 pode-se observar que apesar da redução da acurácia e do aumento da perda, o modelo ainda manteve curvas que indicam que não houve problemas de perfomance no treinamento, como *underfitting* ou *overfitting*. Já na Figura 4.10 podem ser vistos alguns exemplos incorretos de classificações feitas pelo modelo. O modelo errou a classificação do última figura, que era um cachorro e foi classificado como um veado. O significado de cada label pode ser conferido em Krizhevsky *et al.* (2009)

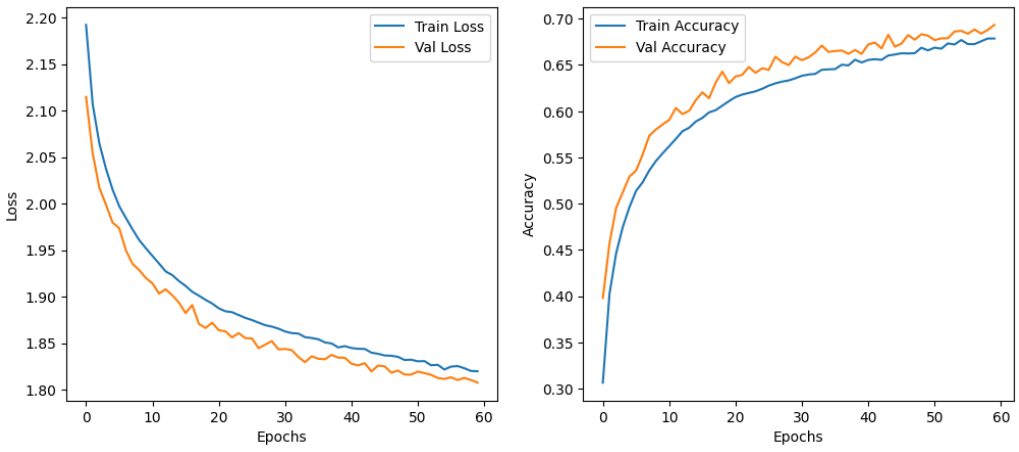


Figura 4.9. Gráficos de métricas VIB para o conjunto CIFAR-10.



Figura 4.10. Exemplos de classificações incorretas do modelo VIB para CIFAR-10.

4.3 ANÁLISE COMPARATIVA

Após terem sido expostos os resultados individuais dos modelos durante seus treinamentos com os três conjuntos de imagens, será feita nesta seção uma comparação entre os dois. As comparações têm como objetivo identificar as diferenças de desempenho, eficiência e consumo de recursos de ambos os modelos, utilizando as métricas coletadas. A análise é necessária para validar e fornecer questões sobre a aplicabilidade da técnica aqui estudada. A Tabela 4.8 exibe as métricas finais para cada modelo ao fim de seu treinamento.

Métrica	MNIST	Fashion MNIST	CIFAR-10
AEM			
Perda da validação	0,0816	0,3795	1,9246
Acurácia média da validação	98,30%	89,63%	73,33%
Tempo de execução	610 s	586 s	8400 s
VIB			
Perda da validação	0,0933	0,3903	1,6118
Acurácia média da validação	98,27%	89,20%	69,29%
Tempo de execução	736 s	695 s	6300 s

Tabela 4.8. Comparação dos modelos AEM e VIB em diferentes datasets

Como pode ser observado, os modelos são praticamente equivalentes com os respectivos conjuntos. Com o modelo AEM entregando resultados ligeiramente melhores que os do VIB, mas nada que possa ser um diferencial. Uma coisa interessante a se notar, porém, é o tempo de execução dos modelos. O modelo VIB apresenta um tempo de execução maior que o do AEM nos dois primeiros cenários observados. Porém, no terceiro cenário, o VIB apresentou um tempo de execução consideravelmente menor.

4.4 INTERPRETAÇÃO DOS RESULTADOS

Os resultados obtidos serviram como uma forma de exemplificar o funcionamento do VIB, que mostrou resultados satisfatórios em todos os cenários apresentados, tendo sua menor acurácia no conjunto de dados CIFAR-10, onde atingiu uma média de 69,29%.

É interessante observar as classificações erradas para termos uma ideia de se os erros do classificador se deram com classes parecidas, mais passíveis de erros, ou classes com diferenças

mais gritantes. Segue na Figura 4.11 a matriz de confusão do VIB.

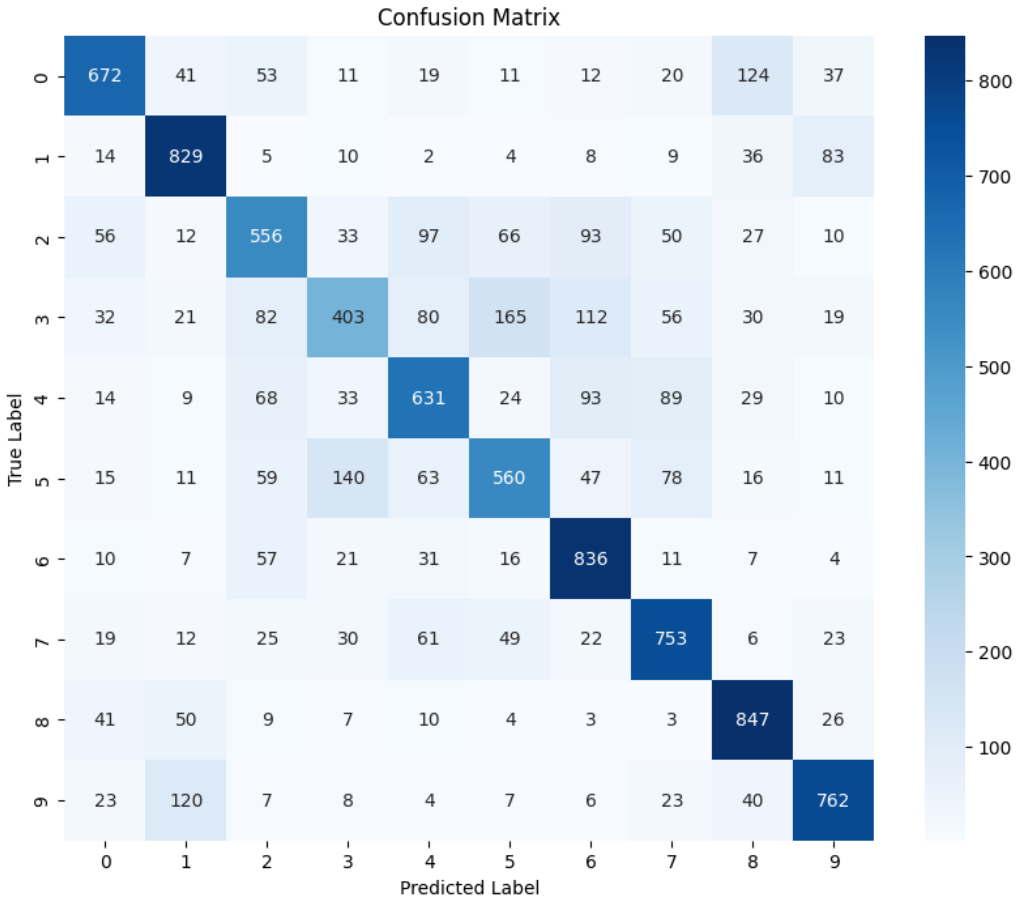


Figura 4.11. Matriz de confusão do modelo VIB para o CIFAR-10.

Pode-se observar que um dos principais erros de classificação do modelo foram com os rótulos 3 e 5, que referem-se às imagens de gatos e cachorros, respectivamente. Por serem classes mais parecidas, dadas as características dos animais, um maior número de confusão entre esses rótulos se faz compreensível. Outra maior ocorrência de erros é entre o rótulo 0 e o rótulo 8, que são para aeronaves e navios, respectivamente. Além da similaridade entre os rótulos, outro fato de erro pode ser dado pela forma das imagens, que não apresentam uma resolução elevada. Os rótulos do CIFAR-10 seguem a seguinte forma: avião, carro, pássaro, gato, veado, cachorro, sapo, cavalo, navio, caminhão. Com os rótulos variando de 0 a 9.

4.5 CONSIDERAÇÕES

Realizados todos os testes experimentais, o modelo alvo de estudo deste projeto mostrou-se satisfatório. Nos dois primeiros cenários, com MNIST e Fashion MNIST, o modelo VIB conseguiu alcançar resultados praticamente iguais aos do AEM, não havendo uma evidente motivação para a escolha de um em detrimento do outro. Porém, o teceiro cenário, com o conjunto CIFAR-10, trouxe um fator interessante de ser analisado, que foi o tempo de execução do VIB ser um quarto menor do que o do AEM.

O menor tempo de execução, apresentado no terceiro cenário, não pode ter seu melhor resultado atribuído ao princípio do gargalo da informação apenas com os resultados aqui apresentados, sendo necessários estudos mais profundos em experimentos mais estruturados e com outros tipos de modelos para comparação.

Como já exposto na fundamentação teórica, o VIB busca transmitir apenas a informação relevante, "comprimindo"a representação dos dados na variável latente z. Essa compactação pode reduzir a quantidade de dados necessários durante o treinamento, descartando as partes consideradas irrelevantes. Isso pode levar a um treinamento mais rápido e, consequentemente, à uma maior eficiência computacional. Como o AEM não possui um mecanismo tão explícito para limitar a quantidade de informação transmitida, suas representações latentes podem ser mais ricas em detalhes, mas que, novamente, esses resultados não podem ser afirmados apenas com os dados coletados durante a realização deste único projeto.

Os conjuntos aqui trabalhados são mais simples, mas serviram seu propósito de mostrar o funcionamento do modelo VIB e sua comparação com um modelo de arquitetura semelhante. Porém, para um conjunto de dados mais robusto, o trabalho de Alemi *et al.* (2017) utiliza os princípios do VIB no conjunto de imagens ImageNet, um dos mais completos e robustos disponíveis, e os resultados de acurácia chegaram à 80%, apresentando resultados ligeiramente superiores aos outros modelos usados de comparação no artigo. Resultado que mostra que os princípios aqui estudados podem almejar aplicações mais complexas.

CAPÍTULO 5

CONCLUSÕES

5.1 LIMITAÇÕES

Serão comentadas brevemente algumas limitações e dificuldades aplicadas a esse projeto, podendo ser levadas em conta em projetos futuros que usem este trabalho como referencial teórico.

A principal limitação encontrada foi a de execução dos códigos referentes aos experimentos. Quando se trabalha com treinamento de redes neurais, a robustez dos equipamentos precisa aumentar conforme a complexidade dos modelos também aumenta, bem como as tarefas que estão sendo pesquisadas ou desenvolvidas. Para a realização deste projeto, foi utilizada a plataforma Google Colab, que permitiu que todos os objetivos fossem alcançados. Porém, continua sendo uma plataforma online e eventuais problemas de conexão podem ocasionar atrasos no desenvolvimento. O Colab também precisa sempre ser executado novamente quando a sessão do navegador é encerrada, fazendo que muitas tarefas se tornem repetitivas.

Um outro limitante importante de se ressaltar é o do conjunto de dados, neste projeto foram utilizados três conjuntos com uma complexidade pouco maior entre um e outro. Serviram seu propósito de estudo, porém, para um treinamento mais robusto e que vise maiores aplicações na realidade, conjuntos mais complexos devem ser utilizados. Existem muitos conjuntos de dados distribuídos entre comunidades pela internet, se fazem muito úteis e economizam o trabalho de se criar um banco de dados próprio, que pode ser muitas vezes mais trabalhoso que a realização do modelo. Os conjuntos de dados devem ser escolhidos de forma a melhor atender os objetivos do projeto.

5.2 CONCLUSÕES

Este trabalho visou investigar a aplicação do Principio do Gargalo da Informação em um modelo de aprendizado de máquina, visando observar a sua capacidade em melhorar a eficiência computacional e sua capacidade de generalização para tarefas de classificação. A partir de uma revisão teórica, que visou abordar os fundamentos da aplicação final, e de experimentos realizados com conjuntos de dados amplamente utilizados, MNIST, Fashion MNIST e CIFAR-10, foi possível analisar os efeitos dessa técnica em comparação com um modelo de classificação mais tradicional.

Os resultados demonstraram que o uso do modelo *Deep Variational Information Bottleneck* proporcionou classificações com resultados satisfatórios em todos os cenários, mostrando que sua aplicação para as tarefas de classificação pode ser justificada. O modelo apresentou resultados semelhantes ao AEM nos dois primeiros cenários, MNIST e Fashion MNIST, e um tempo de treinamento consideravelmente mais baixo no cenário com o CIFAR-10, apesar de ser um resultado positivo, não se pode dizer que foi exclusivamente devido aos princípios do gargalo da informação usados na construção do modelo.

Os resultados obtidos neste trabalho não serviram como uma forma de validação final e total para o método VIB, devido ao escopo e propósito geral do trabalho. O propósito geral era o de estudo de teoria do gargalo da informação e o de comparar um modelo que fizesse a aplicação desses conceitos na prática com um modelo que seguisse um modelo de arquitetura mais comum, o que foi feito por meio do modelo AEM.

Conclui-se que o Princípio do Gargalo da Informação apresenta-se como uma técnica promissora para otimizar redes neurais de aprendizado profundo, sendo um campo que oferece ainda inúmeras possibilidades de exploração e pesquisas. Estudos como o de Alemi *et al.* (2017) mostram que bons resultados podem ser obtidos quando se tem a teoria do gargalo da informação como cerne da pesquisa e da modelagem, a partir de estudos como este, espera-se que novas pesquisas possam expandir a aplicação deste princípio em outras áreas, além do aprendizado de máquina, visando à construção de modelos mais eficientes, econômicos e melhores.

REFERÊNCIAS BIBLIOGRÁFICAS

ACADEMY, D. S. *Deep Learning Book*. 2024. Acesso em: 8 ago. 2024. Disponível em: <https://www.deeplearningbook.com.br/introducao-aos-autoencoders/#:~:text=Os%20Autoencoders%20s%C3%A3o%20uma%20t%C3%A9cnica,conhecimento%20compactada%20da%20entrada%20original.> Citado 3 vezes nas páginas 12, 13, and 14.

ALEMI, A. A.; FISCHER, I.; DILLON, J. V.; MURPHY, K. Deep variational information bottleneck. *International Conference on Learning Representations 2017*, ICLR, p. 1–19, 2017. Citado 8 vezes nas páginas iii, 9, 10, 16, 17, 22, 39, and 41.

BERMANN, D.; STRYKER, C. *What is an Autoencoder?* 2023. Acesso em: 25 dez. 2024. Disponível em: <https://www.ibm.com/think/topics/autoencoder>. Citado 2 vezes nas páginas 13 and 14.

CHEN, R. X. F. A brief introduction to shannon's information theory. Hefei University of Technology, 2016. Citado na página 4.

CHOUDHARY, K.; DECOST, B.; CHEN, C.; AL. et. Recent advances and applications of deep learning methods in materials science. *npj Comput Mater*, Nature, v. 8, p. 97–111, 2022. Citado 2 vezes nas páginas 1 and 7.

COVER, T.; THOMAS, J. *Elements of Information Theory*. Wiley, 2012. ISBN 9781118585771. Disponível em: <https://books.google.com.br/books?id=VWq5GG6ycxMC>. Citado na página 8.

DAI, B.; ZHU, C.; WIPF, D. Compressing neural networks using the variational information bottleneck. 2018. Citado na página 17.

DIGITAL-TRENDS. *Nvidia GPUs see massive price hike and huge demand from AI*. 2023. Acesso em: 14 jul. 2024. Disponível em: <https://www.digitaltrends.com/computing/nvidia-ai-gpus-shortage-price-hikes/>. Citado na página 8.

DOERSCH, C. Tutorial on variational autoencoders. 2016. Citado na página 15.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>. Citado 2 vezes nas páginas 6 and 7.

GRAPHITE-NOTE. *How much data is needed for machine learning*. 2024. Acesso em: 11 jul. 2024. Disponível em: <https://graphite-note.com/how-much-data-is-needed-for-machine-learning/>. Citado na página 2.

IBM. *What is a Neural Network?* 2024. Acesso em: 23 dez. 2024. Disponível em: <https://www.ibm.com/think/topics/neural-networks>. Citado na página 10.

JIM, H.; MARK, S. *What is Deep Learning*. 2024. Acesso em: 12 dez. 2024. Disponível em: <https://www.ibm.com/topics/deep-learning>. Citado na página 1.

KINGMA, D.; WELLING, M. Auto-encoding variational bayes. p. 1–14, 2013. Citado 2 vezes nas páginas 15 and 16.

KINGMA, D. P.; WELLING, M. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2013. Disponível em: <https://api.semanticscholar.org/CorpusID:216078090>. Citado 3 vezes nas páginas iii, 14, and 16.

KRIZHEVSKY, A.; NAIR, V.; HINTON, G. Cifar-10 (canadian institute for advanced research). 2009. Disponível em: <http://www.cs.toronto.edu/~kriz/cifar.html>. Citado na página 35.

KULLBACK, S.; LEIBLER, R. A. On Information and Sufficiency. *The Annals of Mathematical Statistics*, Institute of Mathematical Statistics, v. 22, n. 1, p. 79 – 86, 1951. Disponível em: <https://doi.org/10.1214/aoms/1177729694>. Citado na página 5.

KUMAR, A.; REKHA., G. Challenges of applying deep learning in real world applications. 2020. Citado na página 8.

LECUN, Y.; BENGIO, Y.; HINTON., G. Deep learning. *Nature*, Nature, p. 436–444, 2015. Citado 2 vezes nas páginas 6 and 7.

MACKAY, D. J. *Information Theory, Inference, and Learning Algorithms*. 2003. Citado na página 6.

MEEPEGAMA, U. *VIB*. 2020. Acesso em: 18 jul. 2024. Disponível em: <https://github.com/udeepam/vib/blob/master/vib.ipynb>. Citado na página 22.

MEYN, S.; TWEEDIE, R. *Markov Chains and Stochastic Stability*. Cambridge University Press, 2009. (Cambridge Mathematical Library). ISBN 9780521731829. Disponível em: <https://books.google.com.br/books?id=Md7RnYEPkJwC>. Citado na página 8.

O'SHEA, T.; HOYDIS, J. An introduction to deep learning for the physical layer. *IEEE TRANSACTIONS ON COGNITIVE COMMUNICATIONS AND NETWORKING*, 2017. Citado na página 7.

SHANNON, C. A mathematical theory of communication. *Bell Syst. Tech. J*, Bell, p. 379–423, 623–656, 1948. Citado 2 vezes nas páginas 4 and 5.

SRIVASTAVA, A.; DUTTA, O.; AP, P.; AGARWAL, S.; GUPTA, J. A variational information bottleneck based method to compress sequential networks for human action recognition. *IEEE WACV 2021*, IEEE, 2020. Citado na página 17.

TISHBY, N.; PEREIRA, F. C.; BIALEK, W. The information bottleneck method. *The 37th annual Allerton Conference on Communication, Control, and Computing*, Allerton Conference, p. 368–377, 1999. Citado 5 vezes nas páginas 1, 4, 5, 8, and 17.

TISHBY, N.; ZASLAVASKY, N. Deep learning and the information bottleneck principle. *IEEE ITW 2015*, IEEE, p. 1–5, 2015. Citado 4 vezes nas páginas 9, 10, 11, and 12.

WIKIPEDIA. *CIFAR-10*. 2024. Acesso em: 12 ago. 2024. Disponível em: <https://en.wikipedia.org/wiki/CIFAR-10>. Citado na página 19.

WIKIPEDIA. *Curse of dimensionality*. 2024. Acesso em: 14 dez. 2024. Disponível em: <https://en.wikipedia.org/wiki/Curse_of_dimensionality>. Citado na página 1.

WIKIPEDIA. *Fashion MNIST*. 2024. Acesso em: 18 jul. 2024. Disponível em: <https://en.wikipedia.org/wiki/Fashion_MNIST>. Citado na página 19.

WIKIPEDIA. *MNIST database*. 2024. Acesso em: 18 jul. 2024. Disponível em: <https://en.wikipedia.org/wiki/MNIST_database>. Citado na página 19.