Universidade Federal da Paraíba
Centro de Informática
Programa de Pós Graduação em Informática

Redes Residuais: Sobre Quantidade de Camadas e Variância Intraclasse Utilizando Aprendizagem Não Supervisionada

Cecília Flávia da Silva

João Pessoa - PB

Julho de 2020

Cecília Flávia da Silva

Redes Residuais: Sobre Quantidade de Camadas e Variância Intraclasse Utilizando Aprendizagem Não Supervisionada

Dissertação submetida à Coordenação do Curso de Pós-Graduação em Informática da Universidade Federal da Paraíba como parte dos requisitos necessários para obtenção do grau de Mestre em Informática.

Área de Concentração: Sinais, Sistemas Digitais e Gráficos.

Orientadora: Prof. Thaís Gaudencio do Rêgo

João Pessoa - PB

Julho de 2020

Universidade Federal da Paraíba
Centro de Informática
Programa de Pós Graduação em Informática

Cecília Flávia da Silva

Redes Residuais: Sobre Quantidade de Camadas e Variância Intraclasse Utilizando Aprendizagem Não Supervisionada

Trabalho aprovado em: João Pessoa - PB, / /

Cecília Flávia da Silva
Aluno

Thaís Gaudencio do Rêgo
Professora Orientadora

João Pessoa - PB

Julho de 2020

Catalogação na publicação
Seção de Catalogação e Classificação

S586r Silva, Cecília Flávia da.
Redes residuais : sobre quantidade de camadas e variância intraclasse utilizando aprendizagem não supervisionada / Cecília Flávia da Silva. - João Pessoa, 2021.
71 f. : il.

Orientação: Thaís Gaudencio do Rêgo.
Dissertação (Mestrado) - UFPB/CI.

1. Arquitetura de redes. 2. Deep learning. 3. Variância intraclasse. 4. Data augmentation. I. Rêgo, Thaís Gaudencio do. II. Título.

UFPB/BC CDU 004.72

Elaborado por ANNA REGINA DA SILVA RIBEIRO - CRB-15/024

"It is always darkest before the dawn. Shake it out."

(Florence Welch)

Agradecimentos

Cinco anos atrás eu trabalhava com robótica e queria fazer um carro autônomo para o meu trabalho de conclusão de curso. Nesse trabalho meu foco era *hardware*, mas em uma parte grande dele eu tinha que usar visão computacional para identificar placas de trânsito. Porém, há três anos uma professora me mostrou *Deep Learning* para resolver esse problema e, com isso, tive que aprender em um mês, uma técnica que na época não tinha muitas fontes para aprender e não era nada fácil.

Um mês depois eu tinha me graduado em engenharia de computação e me apaixonado por essa área que conhecia tão pouco, mas que me prendeu tanto a atenção. Decidi então mudar toda a minha área de estudos para inteligência artificial e deixar minha pesquisa anterior para aprender puramente *Deep Learning*. Mais do que isso, eu não queria apenas aprender a usar essa técnica, queria aprender o que estava por trás e fazer um mestrado puramente teórico, sem aplicação.

Com essa decisão, procurei a mesma professora que me apresentou a técnica e entramos nesse mestrado com uma pesquisa que era um tiro no escuro, sem muita (ou quase nenhuma) informação sobre na literatura. Foi uma pesquisa arriscada, professores diziam que éramos muito corajosas, quem me conhecia dizia também que eu nunca procuro problemas fáceis, mas essa professora me deu todas as oportunidades do mundo para crescer. Tive a chance de dar aulas, supervisionar vários trabalhos de alunos e participar de bancas de alunos também. Me engajei em alguns projetos, coordeneei outros, e com isso tive a chance de aplicar todos os conhecimentos que aprendi na teoria. Também ganhei um laboratório maravilhoso, em que aprendi como é delicioso trabalhar em equipe, ter um lar no lugar de trabalho, crescer como ser humano e fazer minhas pesquisas sempre com amor. Essa dissertação nasceu não apenas da pesquisa que deu origem a ela, como também da experiência que tive com todo esse conjunto de oportunidades. Agradeço então a essa professora, Thaís Gaudencio, por ter acreditado tanto em mim e me proporcionado tudo isso.

Agradeço também a Andrew Ng, que, em todos os meus estudos teóricos, sempre quando a situação ficava difícil e frustrante, aparecia com sua frase “If you don’t understand don’t worry about it”, e me fazia seguir acreditando que depois de muitos estudos, eu iria conseguir entender tudo.

Além disso, muitas conquistas e dificuldades aparecem em dois anos de pesquisa, mas Diego Alexandre, meu melhor amigo e namorado, esteve sempre presente apoiando com muito orgulho todas as minhas conquistas, bem como ajudando em todos os problemas que apareceram nesse meio tempo. Aprendemos juntos muitos dos experimentos

que existem nessa dissertação, e tive todo o suporte técnico e emocional do mundo para respirar fundo e superar todas as dificuldades. Um agradecimento bem especial para esse Hokage que, quando tudo parecia impossível, sempre me dava muito amor, sorria e me dizia "acredite em mim, que eu acredito em você", e assim nunca desistir.

Também tive muito apoio dos meus amigos em diferentes momentos, não posso citá-los porque não caberia aqui, então agradeço a todos que estiveram comigo de alguma forma nessa jornada. Por fim, mas não menos importante, agradeço a minha família. Minha mãe, meu tio Lúcio e o meu primo Lucas, por todos os ensinamentos de vida, toda a estrutura familiar e princípios passados. Cresci em um ambiente com muito afeto, apesar de todas as condições serem propícias para o oposto, em que pude ser eu mesma, porque, como meu tio gosta sempre de me dizer: "a flor que desabrocha na adversidade é a mais rara e mais bela de todas".

Resumo

Deep Learning é um termo que surgiu na literatura para, originalmente, definir abordagens de Redes Neurais Profundas (i.e. redes que possuem acima de duas camadas ocultas). A partir disso, diferentes arquiteturas de rede foram propostas na literatura e utilizadas como referência para diferentes aplicações, destacando-se redes como VGG-16, GoogLeNet e redes residuais (ResNets). Nesse contexto, redes residuais possuem uma topologia inspirada na VGG-16, utilizando assim pilhas de camadas convolucionais de filtros 3 x 3 e, como contribuição principal, formam blocos residuais por meio de *shortcut connections*, com o objetivo de reduzir problemas de degradação do modelo (i.e. aumento da taxa de erro do modelo conforme aumento de profundidade da rede). A partir disso, foram propostas ResNets de 20 até 1202 camadas, entretanto, a literatura não retrata um padrão de utilização desses modelos de acordo com um determinado contexto. Sendo assim, esse trabalho teve como objetivo a realização de um estudo agnóstico, que analisa o desempenho de cinco redes residuais, assim como o custo computacional, para diferentes bases de dados. Ao todo, 15 bases de dados foram utilizadas e os modelos ResNet 20, 32, 44, 56 e 110 camadas foram treinados e avaliados para cada uma delas, sendo o desempenho dessas redes avaliado por meio da medida *F1*. Posteriormente, realizou-se um teste de significância, utilizando a distribuição t de Student, em que analisou-se a medida *F1* de cada rede, para avaliar se houve melhoria significativa de desempenho com o aumento de profundidade. Entretanto, os experimentos realizados indicaram que, apesar do aumento de custo computacional proporcional a profundidade, não houve melhoria de resultados. Além disso, analisou-se as características de variância intraclasse para cada classe das bases de dados desse estudo, utilizando o algoritmo de aprendizagem não supervisionada *k-means*, sendo os *clusters* formados por este algoritmo avaliados de acordo com o coeficiente de silhueta. Por meio dessa análise, percebeu-se que, dentre as classes com menor medida *F1*, para todas as redes avaliadas, pelo menos uma das classes retornou baixa diversidade, indicando-se assim, como solução, a melhoria de representatividade da base de dados ao invés do aumento de profundidade do modelo. Sendo assim, realizou-se treinamentos com a ResNet-20, tendo como diferencial a adição de diferentes técnicas de *data augmentation*, em que, para todas elas, o desempenho da rede ResNet-20 foi superior ao das demais redes previamente avaliadas. Dessa forma, por meio da análise, percebeu-se que o aumento de profundidade dessas redes resulta em alto custo computacional, sendo este também proporcional a quantidade de amostras na base de dados. Entretanto, este aumento não é proporcional a melhorias de resultados dos modelos de rede avaliados, resultando em medida *F1* sem variações significativas, sendo a utilização de técnicas de *data augmentation* mais eficaz do que o aumento do número de camadas do modelo.

Palavras-chave: *Deep Learning*, *Data Augmentation*, Aprendizagem de Máquina Não Supervisionado, Variância Intraclasse.

Abstract

Deep Learning is a term in the literature to define different approaches of Deep Neural Networks (i.e. networks that have more than two hidden layers). From this, different network architectures have been proposed in the literature. These networks have been used as a reference for different applications, highlighting networks such as VGG-16, GoogLeNet, and residual networks (ResNets). Residual networks have a topology inspired by the VGG-16 since it uses stacks of convolutional layers of 3 x 3 filters. As the main contribution, it forms residual blocks through shortcut connections to reduce problems of model degradation. This problem increases the model error rate as the network depth increases. From this, ResNets of 20 to 1202 layers were proposed. However, the literature does not portray a pattern of use of these models according to a certain context. Therefore, this work aimed to carry out an agnostic study, which analyzes the performance of five residual networks, as well as the computational cost, for different databases. Overall, 15 datasets were used. ResNet models with 20, 32, 44, 56, and 110 layers were trained and evaluated for each dataset. Also, the performance of these networks was evaluated using F1 measure. Afterward, a significance test was performed using t Student distribution. In this test, the F1 measure of each network was analyzed to find out if there was a significant improvement in performance with increasing depth. However, the experiments indicated no improvement in results despite the increase in computational cost. Besides, the intraclass variance was analyzed for each class of the datasets using the k-means unsupervised learning algorithm. Also, the k-means clusters were evaluated according to the silhouette coefficient. Through this analysis, we discovered that at least one of the classes with lower F1 measure scores have low diversity. Thus, this work performed a set of experiments with ResNet-20 and data augmentation techniques to improve the results. The performance of the ResNet-20 network was superior to all previously evaluated networks. Based on this, the increase in depth of these networks results in high computational cost, which is also proportional to the number of samples in the database. However, this increase is not proportional to improvements in the results of the evaluated network models, resulting in an F1 measure without significant variations, with the use of data augmentation techniques being more effective than increasing the number of model layers.

Keywords: Deep learning, Data Augmentation, Unsupervised Learning, Intraclass Variance.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ACC	*Accuracy* (Acurácia)
CNN	*Convolutional Neural Network* (Rede Neural Convolucional)
DL	*Deep Learning*
DNN	*Deep Neural Network* (Rede Neural Profunda)
GPU	*Graphics Processing Unit*
ILSVRC	*ImageNet Large Scale Visual Recognition Competition* (Competição de Reconhecimento Visual em Larga Escala ImageNet)
H	*Height* (Altura)
LSTM	*Long-Short Term Memory*
ML	*Machine Learning* (Aprendizagem de Máquina)
ResNet	*Residual Networks* (Redes Residuais)
RNA	Rede Neural Artificial
SVM	*Suport Vector Machine* (Máquina de Vetor de Suporte)
T.P.	(Tempo de Predição)
T.T.	(Tempo de Treinamento)
W	*Width* (Comprimento)

Sumário

1 Introdução

Deep Learning (DL), especialmente Redes Neurais Convolucionais (do inglês, *Convolutional Neural Networks* - CNNs), é uma abordagem de Aprendizagem de Máquina (do inglês, *Machine Learning* - ML) e, como tal, seu desempenho depende dos dados de entrada que são utilizados (CARVALHO, 2011) (GOODFELLOW; BENGIO; COURVILLE, 2016). Entretanto, essa abordagem diferencia-se das demais por diversos motivos. Entre eles, a representatividade da base de dados (i.e. quantidade e diversidade de amostras) possui relação direta com o seu desempenho. Isto é, quanto maior representatividade, maior desempenho (SUN et al., 2017).

Em algoritmos clássicos de Aprendizagem de Máquina, tais como Redes Neurais Artificiais (RNA) e Máquina de Vetor de Suporte (do inglês, *Suport Vector Machines* - SVM), ainda que exista essa relação, há um limite de performance inferior ao alcançado por uma DL para maioria das aplicações, especialmente no contexto de *big data* (ROSEBROCK, 2017) (NG, 2015). Nesse contexto, Redes Neurais Artificiais Profundas reduzem a diferença de performance existente. Esta abordagem é considerada ponto de partida nas pesquisas de DL na primeira década dos anos 2000 (GOODFELLOW; BENGIO; COURVILLE, 2016) e consiste, principalmente, no aumento da quantidade de neurônios e camadas de uma Rede Neural Artificial.

Contudo, esse aumento resulta, de forma geral, em uma quantidade excessiva de parâmetros do modelo de rede e, consequentemente, custo computacional elevado, uma vez que as conexões dos neurônios são realizadas de forma completamente conectadas e não esparsas (como ocorrem em CNNs por exemplo) (GOODFELLOW; BENGIO; COURVILLE, 2016). Além disso, a profundidade da rede é limitada ao utilizar uma arquitetura profunda devido a determinados fatores como: problemas no algoritmo de retropropagação (*vanish/exploding gradient*) (BENGIO; SIMARD; FRASCONI, 1994) (GLOROT; BENGIO, 2010) e superajuste.

Estes fatores são reduzidos em uma DL, uma vez que essa abordagem proporciona um aprendizado profundo diferenciado com múltiplos níveis de composição, resultando em uma hierarquia de conceitos (GOODFELLOW; BENGIO; COURVILLE, 2016). Dessa forma, quanto maior representatividade e maior complexidade dos dados, maiores níveis de composição podem ser explorados, sendo esta uma das maiores diferenças ao utilizar tal abordagem.

Existem, atualmente, diferentes formas de realizar esse tipo de aprendizagem, destacando-se as CNNs para diferentes contextos e aplicações. Essas redes possuem arquiteturas populares consideradas estado da arte, sendo as Redes Residuais (ResNet) (HE

et al., 2015) um exemplo dessas arquiteturas.

As ResNet trouxeram inovações para Redes Convolucionais que permitiram, sobretudo, o aumento da quantidade de camadas sem aumentar a quantidade de parâmetros totais da rede e sem problemas de degradação (HE; SUN, 2015). Por exemplo, ImageNet (KARPATHY et al., 2014) é uma base que possui 14.197.122 imagens e 1.000 classes, sendo considerada referência para medida de desempenho de CNNs na literatura.

Ao utilizar uma Resnet de 152 camadas, alcança-se um erro de 3,57% nesse banco. Antes da utilização de redes residuais, redes como GoogleNet (SZEGEDY et al., 2014) e VGG-16 (SIMONYAN; ZISSERMAN, 2014) alcançavam, respectivamente, 6,7% e 7,3% de erro. Essas arquiteturas possuem 22 e 19 camadas e estavam entre as que possuíam maior profundidade antes da implementação das redes residuais. Além disso, a performance da ResNet ultrapassa a humana (5,1% de erro) (HE et al., 2015). Entretanto, existem problemas, atualmente, que utilizam DL com uma base de dados que possui menor quantidade de imagens e classes. Consequentemente, tais problemas podem ser solucionados com arquiteturas de redes residuais mais simples.

Ao utilizar redes simplificadas, além de possíveis melhorias de performance, pode-se também reduzir os recursos de *hardware*. DL é uma abordagem que requer, principalmente, a utilização de placas de vídeo (do inglês, *Graphics Processing Unit* - GPU) de alto desempenho. Tal desempenho é necessário devido a dois fatores principais: i) operações matemáticas otimizadas para GPU reduzem consideravelmente o tempo de treinamento das redes e; ii) quanto maior a quantidade de parâmetros da rede, maior a memória gráfica da GPU para armazenar o modelo (NVIDIA, 2020).

Entretanto, placas de vídeo de alto desempenho elevam consideravelmente o financiamento de um projeto com DL. Além disso, dependendo da quantidade de amostras da base de dados, ainda que placas de vídeo sejam utilizadas, um treinamento com rede como a ResNet de 152 camadas demanda tempo de treinamento elevado.

Como cada aplicação de DL possui uma base de dados específica, a literatura não retrata um padrão quanto a arquitetura de rede necessária ou seleção de hiperparâmetros para cada tipo de base. O resultado disso é a realização de três tipos de treinamento principais:

- **Seleção de arquiteturas tendo-se como referência os resultados da ImageNet:** são analisados o desempenho de redes como VGG (SIMONYAN; ZISSERMAN, 2014), GoogleNet (SZEGEDY et al., 2014) e ResNet-152 (HE et al., 2015), que em muitas ocasiões possuem complexidade elevada para a aplicação desejada.
- **Aumento de complexidade do modelo:** são analisadas redes mais complexas gradualmente, até encontrar uma arquitetura coerente com a aplicação desejada.

Esse processo demanda tempo e custo computacional elevados, uma vez que, quanto maior complexidade, maiores são o tempo de treinamento e o consumo de recursos de *hardware*. Dentre as alterações de complexidade do modelo, são frequentes o aumento de camadas, quantidade de filtros e estrutura topológica das mesmas.

- **Aplicação de redes pré-treinadas:** utiliza-se uma rede previamente treinada com bases de dados complexas como ImageNet, alterando-se apenas as últimas camadas da rede, caso necessário, para adequa-la a aplicação (STUDER et al., 2019). Essa abordagem, apesar de funcional, requer um determinado grau de similaridade entre a base original utilizada para treinado da rede, e a base de dados do problema (STUDER et al., 2019). Por exemplo, a ImageNet possui imagens de diferentes tipos de veículos como carro, avião e motos. Sendo assim, uma aplicação para classificar veículos poderia utilizar um modelo pré-treinado com esta base. Entretanto, uma aplicação que almeje a classificação de diferentes fabricantes de carros possivelmente não retornaria bons resultados, uma vez que cada fabricante possui uma característica específica, sendo então recomendável a utilização de uma base de dados com rótulos específicos para essa tarefa.

Assim, visando a redução do espaço amostral de arquiteturas de rede para o segundo cenário, esse trabalho considerou como experimento a utilização de cinco redes residuais, alterando-se apenas a quantidade de camadas dessas redes de forma a verificar qual possui desempenho mais adequado para uma dada base de imagens. Bases de diferentes características foram utilizadas com o objetivo de generalizar o experimento.

Além disso, uma vez que o desempenho de DL é diretamente relacionado com a representatividade da base de dados, é comum que as bases de dados contenham características que afetem os resultados da DL. Dentre esses problemas, tem-se bases de dados com classes muito semelhantes, ocasionando em falsos positivos e negativos. Ademais, uma classe também pode conter baixa diversidade, resultando em erros de generalização do modelo em casos onde a aplicação utiliza cenários diversificados (e.g. ambiente não controlado).

Esses problemas, muitas vezes, não são solucionados alterando-se a arquitetura da rede, sendo necessárias técnicas de pré-processamento como *data augmentation*. Estas técnicas consistem na aplicação de transformações na base de dados, previamente ou durante o treinamento, que simulem as diversidades necessárias para a aplicação, sendo também utilizadas para redução do superajuste do modelo (SHORTEN; KHOSHGOFTAAR, 2019). Nesse contexto, esse estudo também propõe uma medida utilizando aprendizagem não-supervisionada para avaliar a representatividade e identificar possíveis erros das redes causados por características da base de dados.

Dessa forma, utilizando a ResNet e suas derivações, essa dissertação tem como

objetivo analisar o desempenho de cinco redes residuais em diferentes bases de dados de forma a auxiliar na escolha de uma arquitetura de rede residual de acordo com características da base de dados.

1.1 Objetivos Específicos

- Avaliação de custo computacional de redes residuais conforme aumento de profundidade;
- Analisar características de variância intraclasse utilizando aprendizagem não supervisionada. Neste objetivo, busca-se analisar a formação de *clusters* de acordo com a similaridade intraclasse, com o objetivo de avaliar a representatividade da base de dados e identificar como esta representatividade pode impactar a performance das redes residuais avaliadas;
- Comparar o impacto de técnicas de *data augmentation* e aumento da quantidade de camadas do modelo;

1.2 Estrutura da Dissertação

O restante deste trabalho será organizado conforme o modelo descrito a seguir:

- **Capítulo 2:** Fundamentação teórica abordando conceitos importantes de *Deep Learning* e aprendizagem não supervisionada.
- **Capítulo 3:** Trabalhos relacionados retratam pesquisas que tiveram como base a análise comparativa de redes convolucionais e avaliação de representatividade da base de dados.
- **Capítulo 4:** Metodologia utilizada na dissertação representada por quatro etapas principais.
- **Capítulo 5:** Resultados descreve a discussão principal do trabalho segundo a metodologia aplicada na dissertação.
- **Capítulo 6:** Conclusões do trabalho em conjunto com as considerações finais.

2 Fundamentação Teórica

Rede Neural Artificial (RNA) não é um termo recente na literatura ou, até mesmo, na indústria. RNA é um algoritmo de aprendizagem de máquina inspirado no funcionamento biológico do cérebro cujas pesquisas foram iniciadas em meados dos anos 1940 (NORVIG; RUSSELL, 2016).

Posteriormente, com o surgimento de melhorias em RNAs, como o algoritmo de retropopagação (HINTON; RUMELHART; WILLIANS, 1986) (HINTON et al., 2020) e camadas escondidas, essas redes passaram a ser amplamente utilizadas na academia e no mercado, principalmente entre os anos 1990 e 2000 (GOODFELLOW; BENGIO; COURVILLE, 2016).

Entretanto, aplicações que representam o mundo real tendem a possuir uma complexidade elevada, devido a diversidade de situações e cenários. Por exemplo, um reconhecedor facial utilizado em câmeras de segurança deve considerar, além da diversidade de características na face de cada pessoa, os diferentes tipos de iluminação, oclusão, ângulos e posições (SREENU; DURAI, 2019).

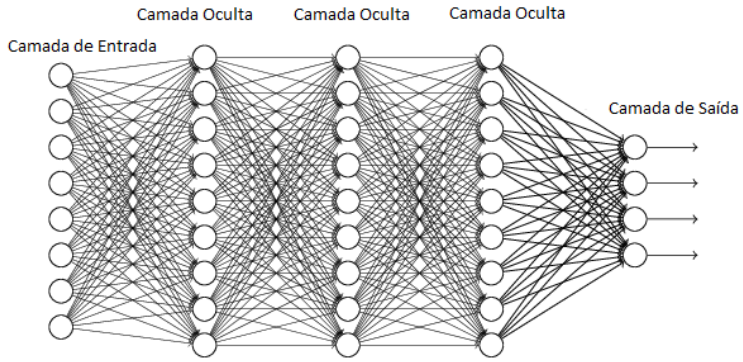
Assim, diversas alterações estruturais em RNAs passaram a ser realizadas de forma que tais redes pudessem ser utilizadas também para esses problemas. Inicialmente, essas alterações eram relacionadas com o aumento de profundidade da rede, o que originou o termo "*Deep Learning*". Com isso, passou-se a utilizar RNAs profundas constituídas por três ou mais camadas ocultas, conforme ilustrado na Figura 1.

Essa estrutura, contudo, resulta em um custo computacional elevado considerando que as conexões entre os neurônios ocorrem de forma que cada neurônio possua n conexões, em que n refere-se a quantidade total de neurônios de uma camada da rede. Sendo assim, quanto maior a profundidade, mais conexões são necessárias, resultando em maior custo computacional e tempo de treinamento. Isso inviabilizou o uso de RNAs profundas apenas aumentando-se a quantidade de camadas.

Por fim, como o aumento de conexões resultou também no aumento da complexidade das operações, a rede passou a ter também problemas de generalização. Assim, além da profundidade, passou-se também a explorar mudanças quanto a estrutura de cada camada da rede. Essas alterações resultaram em abordagens como Redes Neurais Convolucionais (CNNs) (LECUN et al., 1998b) e *Long Short-Term Memory* (LSTM) (HOCHREITER; SCHMIDHUBER, 1997).

Essas e outras abordagens de DL exploram uma arquitetura muitas vezes profunda (i.e. múltiplas camadas), mas que sobretudo possuem diferentes formas de conectividade

Figura 1 – Exemplificação de uma rede neural profunda com três camadas ocultas.

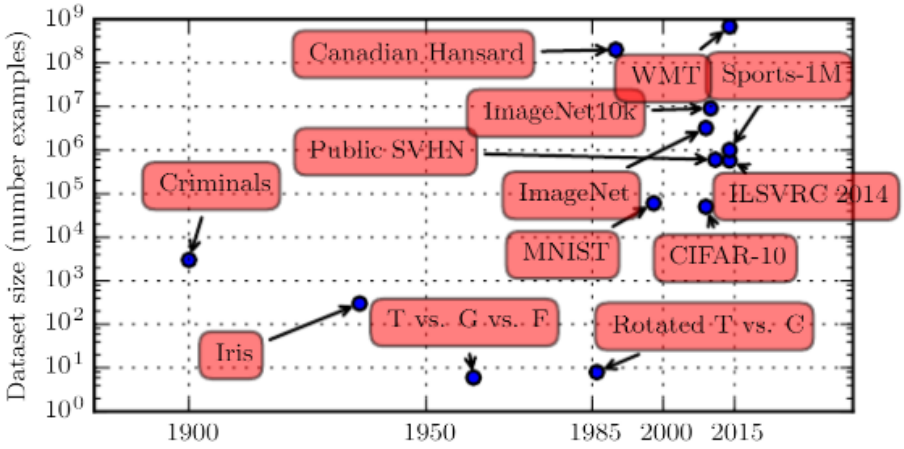


Fonte: (NIELSEN, 2019).

de neurônios, permitindo melhor extração de informação para cada neurônio da rede. Essas formas puderam ser estudadas, principalmente, devido a dois fatores principais: o aumento no tamanho de bases de dados e de recursos computacionais para DL.

Nesse cenário, percebeu-se que com o aumento de acesso a informações e tipos de dados no mundo, mais bases de dados foram disponibilizadas na literatura. O gráfico exposto na Figura 2 ilustra algumas das bases de dados mais populares entre os anos 1900 e 2015, o qual percebe-se um aumento na quantidade de bases a partir de meados dos anos 2000.

Figura 2 – Relação entre o ano de publicação de um banco de imagens e a quantidade de amostras do mesmo.

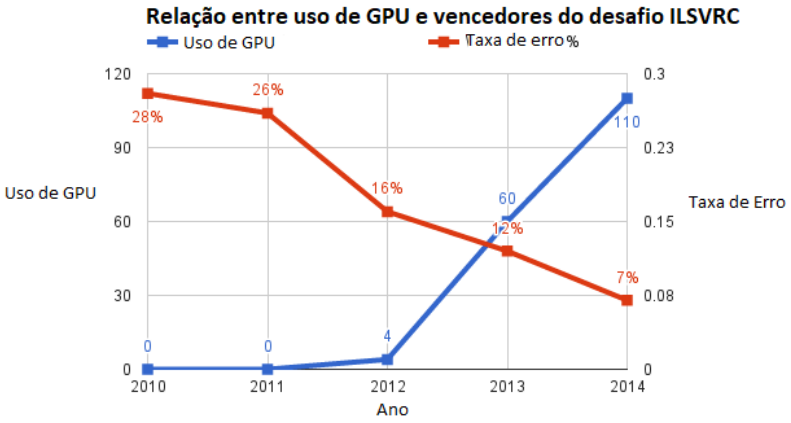


Fonte: (GOODFELLOW; BENGIO; COURVILLE, 2016).

Juntamente com as bases de dados, empresas como a NVIDIA passaram a investir em placas de vídeo voltadas para *Deep Learning*. Essas placas otimizam diversas opera-

ções matemáticas e, reduzem assim, o tempo de treinamento de redes (NVIDIA, 2020). O gráfico ilustrado na Figura 3 exemplifica o aumento na quantidade de pesquisas de DL, conforme aumento de uso de placas gráficas no desafio *ImageNet Large Scale Visual Recognition Competition* (ILSVRC).

Figura 3 – Relação entre o uso de GPU e a taxa de erro das equipes vencedoras do desafio ILSVRC ao longo dos anos 2010 até 2014 em uma escala de 0 a 120.



Fonte: (GOODFELLOW; BENGIO; COURVILLE, 2016).

Tendo como base o exposto, pode-se dizer que, apesar do termo *Deep Learning* ter sido originado pelo aumento de profundidade das RNAs, esse conceito pode ser compreendido, atualmente, como uma forma de aprendizado em níveis hierárquicos de informação (GOODFELLOW; BENGIO; COURVILLE, 2016).

Devido a grande utilização de CNN em diferentes contextos, o trabalho concentrou-se apenas nessa abordagem de DL, sendo CNNs exploradas na Seção 2.1.

2.1 Redes Convolucionais (CNN)

Conforme descrito no início do capítulo, Redes Neurais Artificiais tiveram a estrutura básica das camadas de rede modificadas, principalmente, no que se diz respeito ao número de conexões entre neurônios, de forma a melhorar a performance para problemas mais complexos, dando origem a outros tipos de redes como CNNs.

Tais modificações em uma CNN são marcadas pela presença da operação matemática denominada convolução ilustrada na Equação 2.1. Essa operação comumente é utilizada em áreas como robótica, sinais e processamento digital de imagens (GONZALEZ; WOODS, 2016). No contexto de CNNs, pode-se dizer que x representa os atributos de entrada de uma camada (sejam eles os próprios dados ou resultantes de operações entre camadas anteriores da rede) e W, uma matriz denominada *kernel* ou filtro.

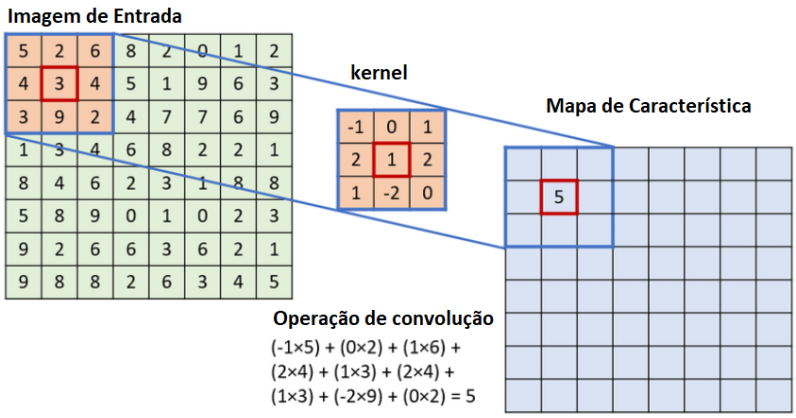
Essa matriz representa os pesos aprendidos durante o treinamento de uma CNN e, usualmente, possui tamanho inferior ao dado de entrada, sendo a mesma transladada (representada por t da equação) até que a operação seja realizada em todos os atributos, como uma janela deslizante.

$$s(t) = (x * w)(t) \tag{2.1}$$

Sendo assim, denomina-se camada convolucional uma camada que realiza tal operação, podendo-se dizer que um filtro representa uma determinada quantidade de pesos da rede, sendo comum em uma camada, a aplicação de diversos filtros. Ao aplicá-los, informações específicas dos dados de entrada são extraídas, sendo o resultado de cada filtro denominado **mapa de características**. Quando uma Rede Neural possui uma ou mais camadas com tais características, pode-se classificar a rede como uma CNN (GOODFELLOW; BENGIO; COURVILLE, 2016).

Um exemplo da operação de convolução em tais mapas de características pode ser visualizado na Figura 4, o qual tem-se uma matriz de entrada de dimensão 8x8 e um *kernel* de dimensão 3x3. Nesse exemplo, a operação de convolução para o valor 3 é aplicada, resultando no valor 5 para o mapa de características resultante.

Figura 4 – Exemplo de operação de convolução em matrizes.



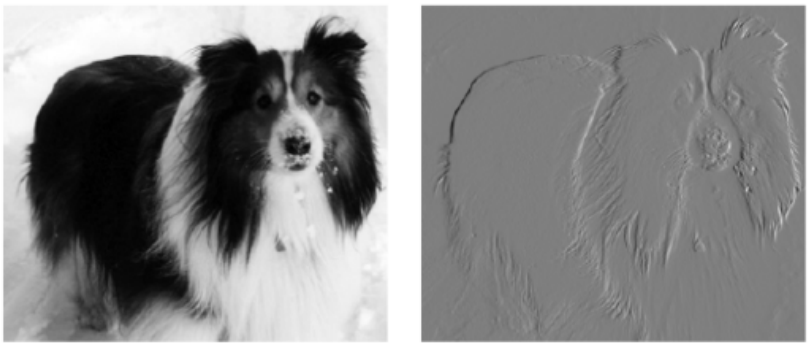
Fonte: (AL, 2011).

Nesse contexto, os mapas são utilizados como dados de entrada para a próxima camada da rede que pode também conter diversos filtros, que serão responsáveis por extrair informações desses mapas, e assim, sucessivamente. Por meio desse processo, cada filtro

pode aprender informações diferentes dos dados e que podem ser refinadas ao utilizar múltiplas camadas convolucionais em sequência, resultando em um aprendizado hierárquico (GOODFELLOW; BENGIO; COURVILLE, 2016).

Por exemplo, a Figura 5 representa a aplicação de um filtro da rede responsável por extrair informações relacionadas a bordas da imagem. Aplicando-se esse filtro na primeira camada da rede, pode-se compreender, inicialmente, as formas básicas de um cachorro.

Figura 5 – Exemplificação da aplicação de um filtro de aguçamento de bordas (direita) em uma imagem original de cachorro (esquerda).



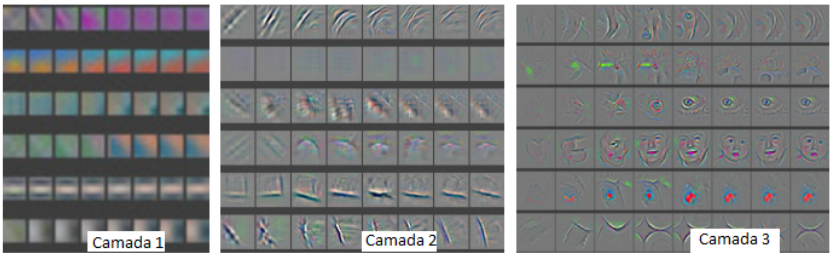
Fonte: (GOODFELLOW; BENGIO; COURVILLE, 2016).

Percebe-se, após a aplicação do filtro, que nenhuma informação sobre as cores do animal foi extraída e as formas principais do pêlo, olhos e focinho foram aguçadas. Da mesma forma, diferentes filtros podem ser aplicados na mesma camada para extrair demais informações básicas da imagem. Posteriormente, a segunda camada aplicará outros filtros que podem destacar tais contornos e que, por sua vez, serão realçados na terceira camada, até que informações simples da imagem possam ser compreendidas pela rede como *feature*, sendo este termo referente a uma combinação dos atributos que resulta em partes da imagem consideradas relevantes para a classificação desejada. Para o dado exemplo, pode-se compreender como *feature* (GONZALEZ; WOODS, 2016), entre outras, o conjunto de atributos que forma os olhos, nariz e boca do cachorro.

A Figura 6 exemplifica a extração de *features* por meio dos mapas de característica das três primeiras camadas de uma CNN de arquitetura similar a AlexNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012) utilizada para classificação das 1.000 classes da base de dados ImageNet (ZEILER; FERGUS, 2014). Percebe-se que os mapas das duas primeiras camadas possuem informações simples, como cor e contornos, enquanto a terceira camada extrai *features* como olhos e rostos.

Nesse contexto, como dito anteriormente, um filtro, usualmente, possui tamanho menor do que os dados de entrada. Para imagem, tamanhos comuns de filtros são 3 x 3, 5 x 5 e 7 x 7. Isso reduz significativamente o custo computacional, uma vez que, para uma

Figura 6 – Visualização dos mapas de característica de uma rede com arquitetura similar a AlexNet utilizando a base de dados ImageNet.



Fonte: (ZEILER; FERGUS, 2014).

imagem 128 x 128, por exemplo, a aplicação de 32 filtros de tamanho 3 x 3 resulta em 128 x 128 x 3 x 3 x 32 = 4.718.592 operações. Em uma RNA tradicional, em uma camada com 32 neurônios seriam necessárias 32 conexões para cada neurônio, resultando em 128 x 128 x 32 x 32 = 16.777.216 operações.

Dessa forma, ao representar os neurônios em filtros e aplicar convolução, tem-se uma redução considerável de custo computacional, sendo esta uma das primeiras motivações para utilização de redes convolucionais na literatura.

Formalmente, essa representação trouxe duas propriedades relevantes: conexões esparsas e compartilhamento de parâmetros. Denomina-se conexão esparsa quando, ao invés de cada camada conter n pesos para n neurônios, tem-se uma quantidade k de pesos, em que k é um número menor que n (GOODFELLOW; BENGIO; COURVILLE, 2016). A Figura 7 ilustra dois tipos de redes em que (a) contém conexões esparsas e (b) possui conexões tradicionais completamente conectadas. Em (a), apenas os pesos s_2, s_3 e s_4 são responsávies por extrair informações da entrada x_3. Por outro lado, para (b) todos os pesos possuem informações dessa mesma entrada, resultando em múltiplas conexões e informações, o que dificulta o aprendizado da rede.

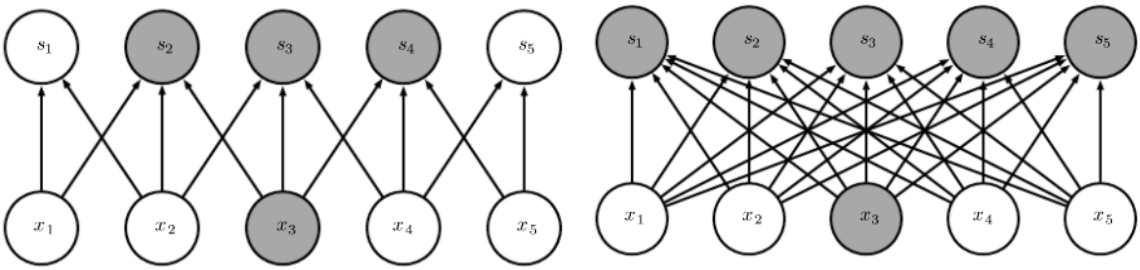
De forma semelhante, pode-se dizer que CNNs possuem a propriedade de compartilhamento de parâmetros porque, o mesmo peso é utilizado repetidamente para diferentes atributos, uma vez que a operação de convolução é realizada como uma janela deslizante ao longo dos dados. Isso reduz, significativamente, a quantidade de parâmetros necessários para o modelo.

Essas duas propriedades são amplamente exploradas na literatura para desenvolvimento de arquiteturas de CNNs mais complexas, que podem ser utilizadas em diferentes aplicações e sistemas.

Atualmente, essas redes são utilizadas não apenas em máquinas de grande desempenho com placa de vídeo dedicada, como também em sistemas de tempo real e máquinas

Figura 7 – Tipos de Conexões Entre Neurônios. Em 24a que apenas os neurônios s_2, s_3 e s_4 aprendem informações do atributo x_3, enquanto em 24b todos os neurônios possuem, cada um, um peso diferente para este atributo.

(a) Conexões Esparsas (b) Conexões Não Esparsas

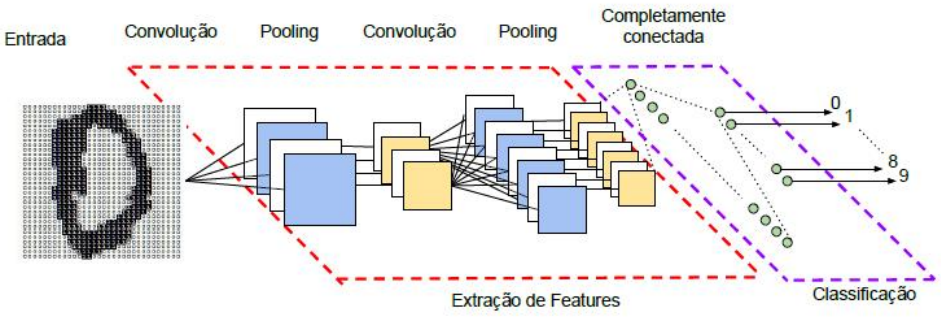


Fonte: (GOODFELLOW; BENGIO; COURVILLE, 2016).

de possuem recursos de *hardware* reduzido, como sistemas embarcados e dispositivos móveis. Dentre as arquiteturas, destacam-se as redes residuais que são descritas na Seção 2.2.

Além dessas arquiteturas, pode-se também implementar uma arquitetura de autoria própria para uma determinada aplicação. A Figura 8 exemplifica uma arquitetura composta de duas camadas convolucionais (conexões esparsas) e uma camada completamente conectada (conexões não esparsas), utilizada para classificação de dígitos. Nessa arquitetura, as camadas convolucionais são utilizadas para extração de *features*, enquanto a camada completamente conectada é utilizada para auxiliar na classificação.

Figura 8 – Exemplo de rede convolucional composta por duas camadas convolucionais e uma camada completamente conectada.



Fonte: (PENHA, 2018).

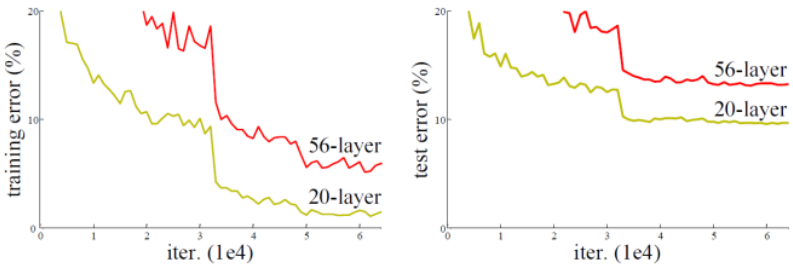
2.2 Redes Residuais

Com o surgimento de CNNs como AlexNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012), VGG-16 (SIMONYAN; ZISSERMAN, 2014) e GoogLeNet (SZEGEDY et

al., 2014), percebeu-se dois fenômenos:

- **Redução de problemas de gradiente:** *vanishing/exploding gradients* são problemas amplamente estudados na literatura desde Redes Neurais tradicionais (BENGIO; SIMARD; FRASCONI, 1994) (GLOROT; BENGIO, 2010) (LECUN et al., 1998a) (SAXE; MCCLELLAND; GANGULI, 2013). Esses problemas resultavam, respectivamente, em um gradiente que tende a zero ou ao infinito e dificultavam a convergência da rede com o acréscimo de camadas. Entretanto, a presença de técnicas de normalização (IOFFE; SZEGEDY, 2015) permitiu um aumento da profundidade da rede reduzindo tais problemas.
- **Aumento de problemas de degradação da acurácia de treinamento:** ao aumentar consideravelmente a quantidade de camadas da rede, a degradação é resultante da saturação da acurácia de treinamento, seguida da redução abrupta da mesma (HE; SUN, 2015) (SRIVASTAVA; GREFF; SCHMIDHUBER, 2015). A Figura 9 exemplifica como ocorre a degradação utilizando arquiteturas ResNet de 20 e 56 camadas sem blocos residuais para a base de dados CIFAR10 (KRIZHEVSK, 2009). Percebe-se que, durante o treinamento, ao aumentar a profundidade de uma rede ResNet sem blocos residuais de 20 para 56 camadas, ocorre o aumento também da porcentagem de erro, simbolizando assim o problema da degradação. Sendo assim, quanto maior a profundidade da rede, maior o erro de treinamento.

Figura 9 – Porcentagem de erro durante o treinamento (esquerda) e teste (direita) para a base de dados CIFAR10.



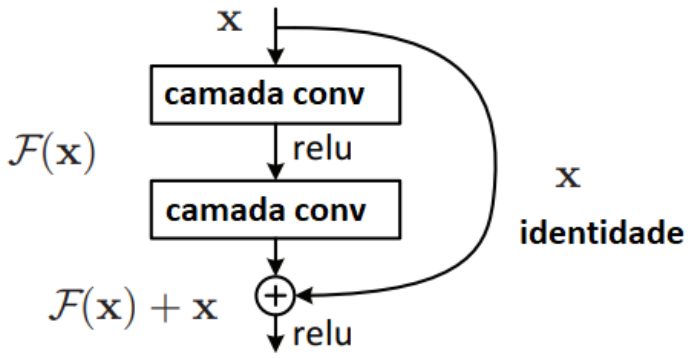
Fonte: (HE et al., 2015).

Levando em consideração esses dois fenômenos, as redes residuais foram desenvolvidas utilizando técnicas de normalização para evitar problemas do gradiente, além da implementação de blocos residuais que permitiam o aumento de profundidade com problemas de degradação reduzidos.

Nesse contexto, blocos residuais são formados por *shortcut connections*. Esse tipo de conexão foi utilizado previamente em diferentes abordagens de redes neurais, como

LSTM (HOCHREITER; SCHMIDHUBER, 1997) e arquiteturas Inception (SZEGEDY et al., 2014). Para ResNets, *shortcut connection* é uma operação não sequencial que executa um mapeamento de identidade (HE et al., 2015), conforme ilustrado na Figura 10, em que o resultado de uma camada **Y** depende não apenas dos mapas de característica da mesma, como também de determinados mapas derivados de camadas **X** específicas, anteriores a **Y**.

Figura 10 – Exemplificação de um bloco residual. Neste bloco, o resultado de uma camada F(x) será a combinação entre os mapas de característica resultantes desta camada e os mapas previamente calculados de uma camada x.



Fonte: (HE et al., 2015).

Essa operação de concatenação é também representada formalmente pela Equação 2.2 em que *F(x)* é o resultado da camada *Y*.

$$F(x) + x \tag{2.2}$$

Ao realizar tal composição, tem-se um mapeamento de identidade, que possui uma simples implementação, de forma que os cálculos de retropropagação ocorram da mesma forma que modelos tradicionais de redes convolucionais, sem aumento de complexidade ou quantidade de parâmetros.

Além de blocos residuais, uma arquitetura ResNet possui pilhas de camadas convolucionais de filtros 3 x 3 e camadas convolucionais de filtro 1 x 1, derivadas, principalmente, das redes VGG e GooleLeNet com blocos Inception.

Nessas redes, verificou-se que o uso de pilhas de camadas convolucionais de filtros 3 x 3, ao invés de uma única camada com filtros 7 x 7, como na AlexNet, reduz consideravelmente a quantidade de parâmetros da rede, alcançando o mesmo campo receptivo de filtros 7 x 7. Por exemplo, assumindo que a entrada e a saída de três camadas convolucionais de filtros 3 x 3 tenham C canais (i.e. profundidade da camada resultante da quantidade de mapas de característica da mesma), seria realizado um total de $3(3^2C^2) = 27C^2$ operações.

Em uma única camada 7 x 7 com a mesma quantidade de canais C, teria-se $7^2C^2 = 49C^2$ operações.

Dessa forma, tem-se menor custo com o mesmo efeito de performance. Além disso, em uma pilha de camadas convolucionais, como cada camada possui uma função de ativação, tem-se diferentes níveis de não linearidade na pilha, o que auxilia na convergência do modelo e aumenta o desempenho do mesmo.

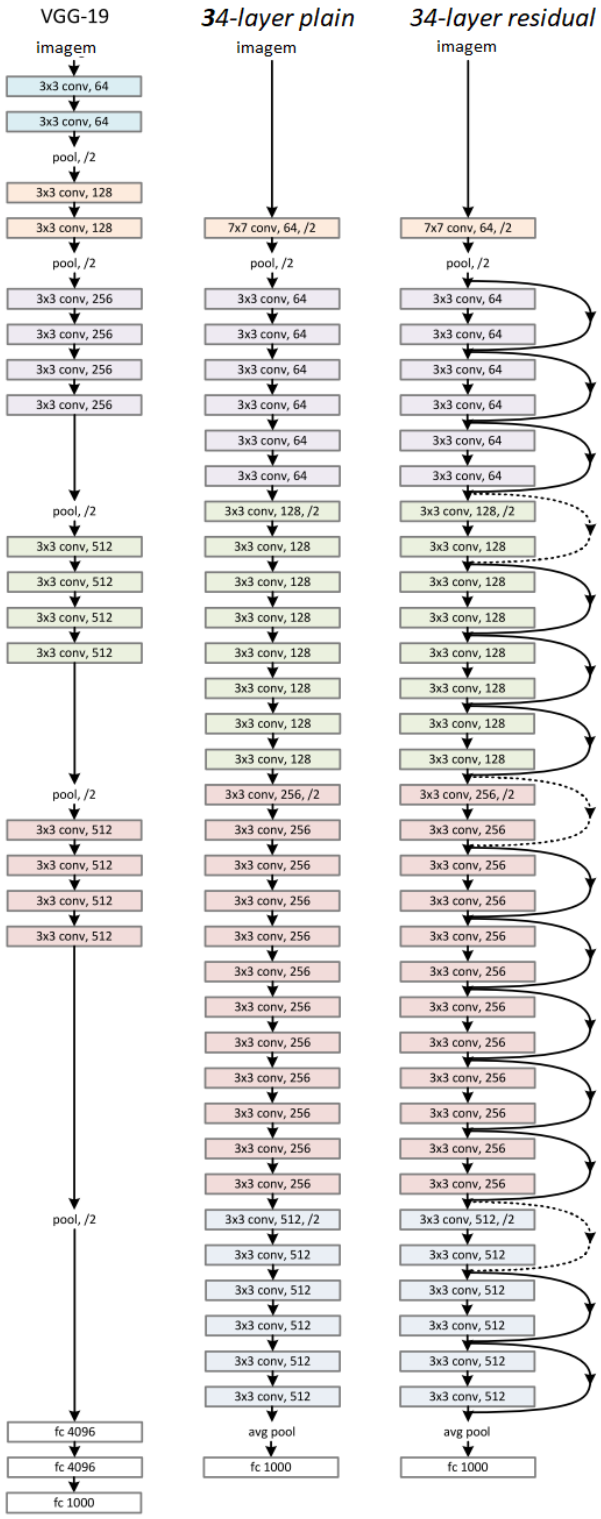
Similarmente, camadas convolucionais com filtro 1 x 1 são adicionadas nessas redes para redução de profundidade de determinadas camadas, previamente a operações de filtro 3 x 3 ou 5 x 5. Ao realizar tal convolução, tem-se as duas primeiras dimensões da entrada (altura e comprimento) inalteradas e apenas a profundidade reduzida. Por este motivo, a operação é denominada também de redução de dimensão ou *cross channel down-sampling*.

Por exemplo, para uma entrada de dimensão 28 x 28 x 32, ao aplicar 32 filtros de dimensão 5 x 5 nessa entrada, seriam aplicadas 28 x 28 x 32 x 5 x 5 x 32 = 20.070.400 operações. Ao invés disso, pode-se aplicar uma camada convolucional de filtros 1 x 1 antes da camada com filtros 5 x 5. Considerando o mesmo exemplo, aplicando-se 8 filtros de dimensão 1 x 1, resulta-se em 28 x 28 x 32 x 1 x 1 x 8 = 200.704 operações e uma entrada de dimensão 28 x 28 x 8, ao invés de 28 x 28 x 32. Posteriormente, aplicando-se 32 filtros de dimensão 5 x 5, tem-se 28 x 28 x 8 x 5 x 5 x 32 = 5.017.600 operações. Sendo assim, ao todo, seriam necessárias 200704 + 5017600 = 5.218.304 operações, o que é significativamente menor do que as 20.070.400 operações originalmente utilizadas com a aplicação de 32 filtros de dimensão 5 x 5.

Assim, associando todos esses conceitos em uma arquitetura ResNet, essa arquitetura é composta por pilhas de camadas convolucionais com filtros de dimensão 3 x 3 e *shortcut connections* entre essas pilhas. Além disso, as camadas convolucionais de uma pilha possuem, obrigatoriamente, a mesma quantidade de filtros. Por fim, a adição da quantidade de filtros entre pilhas é realizada sempre de forma que a próxima pilha contenha o dobro de filtros da pilha anterior.

A Figura 11 exibe a arquitetura de uma VGG-19 e a ResNet 34 camadas. Para maior entendimento, tem-se ilustrada uma ResNet sem blocos residuais (*34-layer plain*) e com os blocos (*34-layer residual)*. Na Figura, *fc* refere-se as camadas completamente conectadas (i.e. neurônios com multiplicação de matriz tradicional sem conexões esparsas), *pool* e *avg pool*, técnicas de redução espacial (do inglês *downsampling*) das dimensões H (altura, do inglês *height*) e W (comprimento, do inglês *width*).

Figura 11 – Arquiteturas VGG-16, ResNet sem blocos residuais (34-layer plain) e ResNet de 34 camadas com blocos residuais.



Fonte: (HE et al., 2015).

Como descrito previamente, a estrutura da ResNet também foi desenvolvida afim

de reduzir problemas de gradiente, sendo assim, utilizou-se normalização do *batch* (IOFFE; SZEGEDY, 2015), após as operações de convolução das camadas e antes da função de ativação.

As imagens de entrada são redimensionadas para dimensões 224 x 224, normalizadas e centralizadas seguindo o mesmo pré-processamento aplicado na rede AlexNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012). Além disso, as mesmas técnicas de *data augmentation* aplicadas para a rede VGG-16 (SIMONYAN; ZISSERMAN, 2014) são utilizadas.

Por fim, os pesos foram inicializados como descrito em (HE et al., 2015), sendo utilizado o otimizador do gradiente descentendente estocástico (do inglês *Stochastic Gradient Descent* - SGD), com taxa de decaimento de 0,0001 e momento de 0,9. A taxa de aprendizagem inicia-se em 0,1 e é recalculada por um fator de 10, seguindo a curva de erro *plateau*. Para realização do treinamento, tornaram-se necessárias mais de 60×10^4 iterações utilizando um *batch* de tamanho igual a 256.

Além da ResNet 34 camadas, redes residuais de diferentes quantidades de camadas foram propostas para classificação utilizando a base de dados ImageNet. A Tabela 1 representa cinco dessas arquiteturas, em que a sequência de pilhas é representada a cada linha, sendo cada pilha uma matriz. Na matriz, cada linha refere-se a uma camada da pilha, enquanto as colunas referem-se, respectivamente, a dimensão dos filtros e a quantidade de filtros aplicados na camada. Além disso, o identificador ($\times N$) indica a quantidade de vezes N que essas pilhas foram repetidas para cada sequência.

18-camadas	34-camadas	50-camadas	101-camadas	152-camadas
$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$

Tabela 1 – Quantidade de pilhas e a quantidade de filtros, assim como a dimensão, de cada camada convolucional dessas pilhas para diferentes arquiteturas ResNets.

Nessas arquiteturas, percebe-se que convoluções 1 x 1 são utilizadas apenas para arquiteturas acima de 50 camadas, uma vez que tais arquiteturas possuem custo computacional alto (Tabela 2). Esse custo é medido em operações de ponto flutuante por segundo

(do inglês *Floating Point Operations Per second* - FLOPs) seguindo a Equação 2.3, em que copf significa contagem de operações de ponto flutuante, e te o tempo de execução.

$$FLOPs = \frac{copf}{te \times 10^6} \tag{2.3}$$

ResNet	Custo (FLOPs)
18-camadas	1,8 $\times 10^9$
34-camadas	3,6 $\times 10^9$
50-camadas	3,8 $\times 10^9$
101-camadas	7,6 $\times 10^9$
152-camadas	11,3 $\times 10^9$

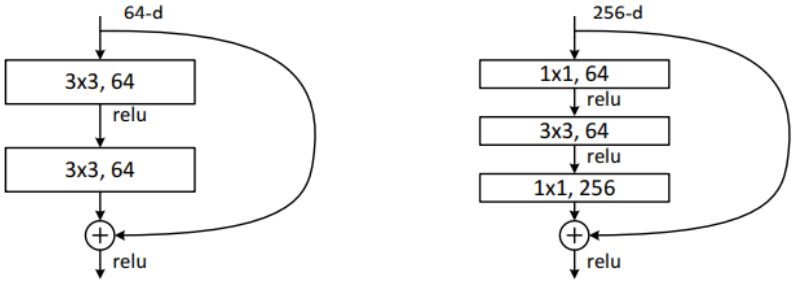
Tabela 2 – Custo computacional em arquiteturas ResNets.

Assim, em uma ResNet, convoluções de filtro 1 x 1 são adicionadas no início de uma pilha convolucional para redução de dimensão, como também no fim da pilha de forma reversa (i.e. para aumento de profundidade), para que a profundidade resultante da camada Y da pilha seja equivalente a profundidade da camada X. Isso tornou-se necessário, uma vez que operações de concatenação requerem que as matrizes tenham mesma profundidade. A Figura 12 representa dois blocos residuais em que, na esquerda, tem-se um bloco simples sem convoluções 1 x 1 e, o oposto, na direita. Além disso, a saída de cada camada convolucional é resultante da função de ativação retificadora ReLU. Essa função torna os valores negativos presentes nos mapas de ativação iguais a zero e sua utilização resulta em melhorias de convergência do modelo (HE et al., 2015).

Nessa representação, percebe-se que no bloco com convoluções 1 x 1, a profundidade dos dados de entrada da pilha era de 256 canais e tais convoluções foram aplicadas, inicialmente, para reduzir a profundidade de 256 para 64 e assim, aplicar uma camada de convolução 5 x 5. Posteriormente, aplicou-se uma nova camada de convolução 1 x 1, de forma que a profundidade dos dados de saída da pilha tenha 256 canais, da mesma forma que os dados de entrada, possibilitando a operação de concatenação.

Essas arquiteturas foram propostas, especificamente, para bases de dados como ImageNet. Entretanto, também foram propostas arquiteturas com quantidade de camadas iguais a 20, 32, 44, 56, 110 e 1202 camadas para bases de dados menores como CIFAR10. Essas arquiteturas seguem a mesma estrutura representada pela Tabela 3, alterando-se apenas a quantidade de filtros dessas camadas. Isto é, ao invés de utilizar 64, 128, 256 e 512 filtros, as pilhas utilizam 16, 32 e 64 filtros.

Figura 12 – Blocos residuais com camadas convolucionais de filtros 1 x 1 (direita) e sem estas camadas (esquerda). Direita. Aplica-se uma camada convolucional de filtros de 1 x 1 no início de uma pilha para reduzir a profundidade de 256 para 64, aplicando-se, posteriormente, outra camada de filtros 1 x 1 para retornar à profundidade de 256 canais e realizar a operação de concatenação. Esquerda. como não há camadas deste tipo, a entrada e a saída da pilha possuem dimensão igual a 64.



(HE et al., 2015)

Todos os experimentos dessas arquiteturas simplificadas foram realizados com *data augmentation* na base CIFAR10. Essa base consiste em 10 classes de diferentes tipos de animais e veículos e os resultados são expostos na Tabela 3, na qual percebe-se a não ocorrência de degradação, ainda que uma rede muito profunda seja utilizada (1202 camadas).

Quantidade de Camadas	Erro (%)	Total de parâmetros (em milhões)
20	8,75	0,27
32	7,51	0,46
44	7,17	0,66
56	6,97	0,85
110	6,43	1,7
1202	7,93	19,4

Tabela 3 – Porcentagem de erro, quantidade de camadas e total de parâmetros de ResNets utilizadas para classificação com a base de dados CIFAR10.

Sendo assim, essa dissertação utilizou estas redes residuais para diferentes bases de dados similares ao CIFAR10, uma vez que não foram utilizadas bases de dados complexas como a ImageNet, tendo como diferencial a não realização de *data augmentation*. Optou-se por não utilizar esta técnica de pré-processamento, inicialmente, para verificar possíveis alterações no desempenho dessas redes, de acordo com a diversidade original das bases de dados. Essa representatividade foi analisada utilizando aprendizagem não supervisionada conforme descrito na Seção 2.3. Entretanto, visando comparar o impacto entre o uso de técnicas de *data augmentation* e o aumento da quantidade de camadas, experimentos posteriores foram realizados neste trabalho, sendo estas técnicas descritas na seção **??**.

2.3 Análise de Agrupamento

Em aprendizagem de máquina, pode-se dividir os dados em grupos, denominados *clusters*, em que cada um diferencia-se, comumente, de acordo com algum(s) atributos dos objetos (CARVALHO, 2011). A semelhança entre objetos para definição de um grupo é calculada pelo algoritmo, de acordo com uma métrica de proximidade e um método de busca que identifique uma estrutura ótima ou subótima na realização do agrupamento.

Essa abordagem não utiliza rótulos associados, sendo o aprendizado denominado não supervisionado. Além disso, a ausência de rótulo torna a análise dos *clusters* subjetiva, dada a diversidade de análises possíveis por um pesquisador ou pelo próprio algoritmo dependendo dos critérios e parâmetros utilizados pelos mesmos (ESTIVILL-CASTRO, 2002).

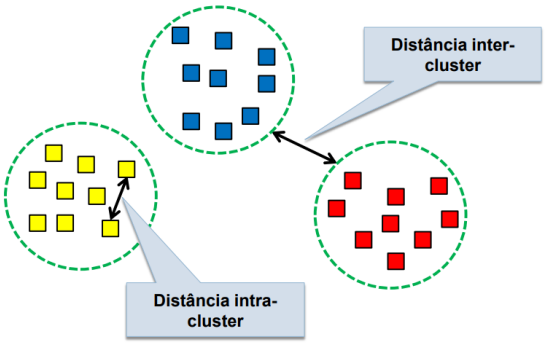
A literatura estabelece diferentes definições formais de *cluster* dependendo do contexto desejado (BARBARA, 2000), tais como *cluster* baseado em densidade, similaridade e em centro. Para esse trabalho, estabelece-se o mesmo como um conjunto de pontos que são similares entre si, enquanto pontos em diferentes *clusters* não são similares.

Essa similaridade pode ser medida utilizando métricas de distância e divide-se em dois tipos principais.

- **Intra-cluster:** a similaridade interna de um determinado grupo, sendo um agrupamento eficaz, idealmente, aquele que possuir alta homogeneidade neste cenário, isto é, dados com características semelhantes.
- **Inter-cluster:** a similaridade entre dois grupos distintos, sendo um agrupamento eficaz, idealmente, aquele que possuir alta heterogeneidade, isto é, com os dados dos outros grupos com características diferentes.

Sendo assim, pode-se dizer que o objetivo de um agrupamento é minimizar a distância intra-cluster e, em contraste, maximizar a distância inter-cluster. A Figura 13 representa a definição desses dois tipos, em que percebe-se que a distância intracluster calcula a distância entre as amostras amarelas, enquanto a distância intercluster avalia a distância entre os grupos de amostras vermelha e azul.

Figura 13 – Distâncias intra (dentro de um mesmo cluster) e inter-cluster (entre clusters distintos) para três grupos.



Fonte: (UBERLANDIA, 2018).

Considerando os diferentes critérios de agrupamento, existe também uma diversidade de algoritmos (JAIN; DUBES, 1988) para realização de agrupamento. Essa dissertação utilizou o *k-means* (Figura 14), algoritmo particional baseado em erro quadrático, devido a utilização do mesmo para agrupamento com base de dados de imagens (DHANACHANDRA; JINACHANU; MANGLEM, 2015) (CHEN; YANG; TIAN, 2015).

Figura 14 – Algoritmo K-means.

Algoritmo *k*-médias

Entrada: Um conjunto de dados $\mathbf{X}_{n \times d}$
Número de *clusters* k
Saída: Uma partição de $\mathbf{X}$ em k *clusters*
1 Escolher aleatoriamente k valores para centroides dos *clusters*
2 **repita**
3 **para cada** *objeto* $\mathbf{x}_i \in \mathbf{X}$ *e cluster* $\mathbf{C}_j, j = 1, \ldots k$ **faça**
4 Calcular a distância entre $\mathbf{x}_i$ e o centroide do *cluster* $\bar{\mathbf{x}}^{(j)}$: $d(\mathbf{x}_i, \bar{\mathbf{x}}^{(j)})$, utilizando uma medida de distância
5 **fim**
6 **para cada** *objeto* $\mathbf{x}_i$ **faça**
7 Associar $\mathbf{x}_i$ ao *cluster* com centroide mais próximo
8 **fim**
9 **para cada** *cluster* $\mathbf{C}_j, j = 1, \ldots k$ **faça**
10 Recalcular o centroide
11 **fim**
12 **até** *não haver mais alteração na associação dos objetos aos clusters*;

Fonte: (CARVALHO, 2011).

Como algoritmo particional (CARVALHO, 2011), o *k-means* utiliza como critério de similaridade o erro quadrático de forma iterativa. Essa iteração inicia-se a partir da escolha aleatória dos k centros de cada *cluster* (centroides), seguido do cálculo da distância

de cada amostra, em relação aos centroides. Uma vez que o objetivo é minimizar a distância intra-cluster, atribui-se cada amostra ao *cluster* de menor distância. Uma vez que todas as amostras foram atribuídas a cada *cluster*, recalcula-se os centroides e repete-se as etapas anteriores, até não haver mais alteração entre os *clusters*.

Essas etapas correspondem ao algoritmo básico do *k-means*, existindo versões modificadas do mesmo para otimizar o agrupamento. Entre elas, pode-se destacar versões otimizadas quanto a escolha dos centroides iniciais, considerando que uma escolha aleatória pode tender a um agrupamento não ótimo.

Ao fim do algoritmo, com os grupos formados, pode-se analisar a qualidade dos mesmos através das distâncias intra e inter-cluster encontrados. Uma das formas populares de realizar tal análise, é a partir do método silhueta (J.ROUSSEEUW, 1987), que calcula o grau de separação entre os *clusters*, de acordo com a Equação 2.4.

Na equação, a^i corresponde a distância média intra-cluster e b^i corresponde a distância média inter-cluster;

$$silhueta = \frac{b^i - a^i}{max(b^i - a^i)} \tag{2.4}$$

Essa equação retorna um coeficiente de similaridade em um intervalo [-1,1] em que:

- um coeficiente próximo de 0 implica dizer que existem amostras que podem corresponder a mais de um *cluster*, podendo haver sobreposição entre os mesmos.
- um coeficiente próximo de 1 indica que o *cluster* é homogêneo e assim, as amostras do mesmo são muito similares entre si e dissimilares com os demais grupos.
- um coeficiente tendendo a -1 indica que o *cluster* contém amostras que não são designadas para o mesmo.

Dado o exposto, em uma aplicação de *Deep Learning*, pode-se realizar agrupamentos e avaliá-los posteriormente, utilizando silhueta, com o objetivo de identificar a representatividade da base de dados, sendo o *k-means* um algoritmo utilizado previamente para auxiliar redes neurais na extração de *features*.

Nesse contexto, em DL, quanto maior a diversidade de amostras em uma classe, mais informações a rede poderá aprender sobre ela. Sendo assim, quanto mais próximo de 1 o coeficiente da silhueta, menor será a representatividade intra-classe.

Essa análise contém custo computacional consideravelmente menor utilizando o *k-means* uma vez que o mesmo possui complexidade O(n), enquanto algoritmos de agrupamento como o Mean Shift (COMANICIU; MEER, 2002) possui O(T$n$log(n)) para dados

de pequenas dimensões e O(Tn^2), para bases de dados de grandes dimensões, o que o torna mais custoso que o *k-means* em ambos os casos.

Além disso, existem abordagens para medir a similaridade entre imagens como a relação sinal-ruído de pico (do inglês, *Peak Signal-to-Noise-Ratio* - PSNR), correlação, o índice de similaridade médio (do inglês *Structural Similarity Image Metric* - SSIM) e variações (WANG et al., 2004b).

Essas abordagens são recomendadas considerando imagens centralizadas, sendo então usualmente utilizadas (WANG et al., 2004a), por exemplo, em análise de qualidade de transmissão de vídeo. Nessas análises, pode-se comparar cada *frame* e, caso dois destes não retornem alta similaridade, a transmissão pode conter queda de *frames* (WANG; SHEIKH; BOVIK, 2003).

Em DL, por outro lado, bases de dados em cenários reais, usualmente, contêm objetos em diferentes posições e rotações, não sendo indicada dessa forma a utilização de tais métodos. Por fim, DL, comumente, utilizam-se técnicas de *data augmentation* para aumento de diversidade de amostras da base de dados, conforme descrito na Seção **??**.

2.4 Data Augmentation

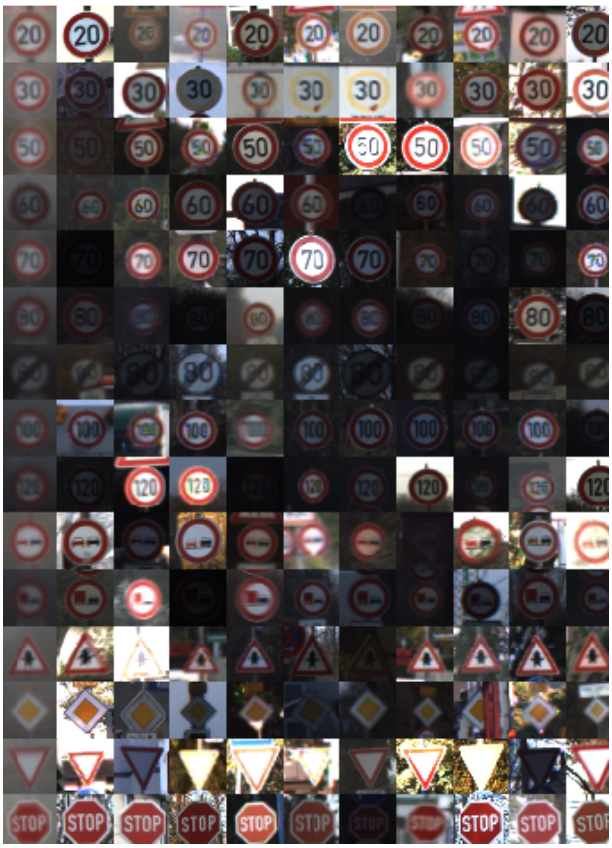
Técnicas de *data augmentation* são utilizadas em aprendizagem de máquina para diferentes utilidades, entre eles, destaca-se o uso para redução de superajuste do modelo e simulação de ambientes naturais (SHORTEN; KHOSHGOFTAAR, 2019). Por exemplo, a Figura 15 ilustra imagens de placas de trânsito. Supondo o desenvolvimento de uma aplicação que vise a identificação de placas de trânsito para carros autônomos, o carro precisa identificar corretamente as placas para tomadas de decisão do sistema em diferentes situações. Isso indica a necessidade de uma base de dados com imagens, por exemplo, em diferentes níveis de exposição, de forma que a identificação possa ser realizada em diferentes horários e condições de iluminação.

Entretanto, comumente existem bases de dados que não possuem tais variações, sendo necessário introduzi-las de forma a simular o ambiente real, o qual a aplicação está inserida. Sendo assim, as técnicas de *data augmentation* aplicam transformações nas amostras da base, como alterações de brilho, contraste e saturação, de forma a criar novas amostras que apresentam as variações necessárias para o problema.

Similarmente, algoritmos de aprendizagem de máquina e DL, dependendo da topologia e hiperparâmetros escolhidos, podem resultar em modelos superajustados (i.e. acurácia de validação menor que a de treinamento). Ao ocorrer superajuste, uma das formas de reduzi-lo é aplicar *data augmentation*, de forma a variar informações presentes nas amostras, sem alterar as características principais delas. Por exemplo, ao rotacionar uma

placa de 80 km/h em 5 graus, apesar de rotacionada, as informações principais, como os números 8 e 0, permanecem presentes na imagem.

Figura 15 – Exemplificação de placas de trânsito em diferentes condições de iluminação.



Fonte: própria.

Dessa forma, pode-se definir *data augmentation* como técnicas que aplicam transformações nas amostras da base de dados, criando-se assim, variações dos dados originais, de forma a diversificar os dados de entrada utilizados em modelos de aprendizagem de máquina .

Dentre essas técnicas, no escopo de imagens, destacam-se as transformações afins (e.g. rotação, translação e escala), como técnicas clássicas, utilizadas amplamente para implementação de redes convolucionais no desafio ImageNet (KARPATHY et al., 2014). Entretanto, pode-se aplicar, também, transformações no espaço de cores (e.g. saturação, inversão de canais, seleção de componentes principais), entre outras como equalização do histograma, filtros de aguçamento, nitidez e alterações de contraste, além de técnicas de remoção de *pixels* como *cutout* e *coarse dropout* (LEE et al., 2014) (BARAN; KUPYN; KRAVCHENKO, 2019).

A escolha das técnicas mais adequadas depende da compreensão e tipos de imagens presentes na base de dados. Por exemplo, transformações de cor não são utilizadas

para bases como MNIST (LECUN et al., 1998b) e Fashion MNIST (XIAO; RASUL; VOLLGRAF, 2017), uma vez que, são bases com imagens binárias e em escala de cinza, respectivamente. Entretanto, independente da técnica utilizada e do objetivo, *data augmentation* aumenta a diversidade de amostras em base de dados. Sendo assim, como um dos objetivos desse trabalho é a realização de análise de diversidade nas bases, em uma primeira etapa serão realizados experimentos sem *data augmentation*. Posteriormente, os resultados serão comparados com experimentos que utilizaram essas técnicas, para analisar o impacto destes no desempenho dos modelos treinados, conforme descrito na metódologia do trabalho (Seção 4).

3 Trabalhos Relacionados

Essa dissertação baseou-se no trabalho de (HE et al., 2015), que deu origem as redes residuais e analizou a utilização delas para classificação e detecção de objetos. Para classificação, o desempenho das cinco redes residuais foi avaliado utilizando a base de dados CIFAR10. No estudo, as arquiteturas de 20, 32, 44, 56, 110 e 1202 (denominadas nessa Seção de I a V, respectivamente) camadas, conforme descritas na Seção 2, foram treinadas com 50.000 imagens desta base de dados, e avaliada com 10.000 imagens. Entretanto, o objetivo desta análise foi apenas identificar problemas de degradação em redes mais profundas utilizando uma base de dados pequena, de forma que não foram realizados testes de custo computacional ou comparações quanto a taxa de erro das arquiteturas.

(KHAN et al., 2018) assemelha-se ao trabalho proposto nesta dissertação quanto a avaliação arquitetural, considerando uma base de imagens para identificação de câncer de mama e outra para detecção de *malware*. No trabalho, as arquiteturas ResNet 18, 50, 101 e 152 são comparadas considerando a acurácia (Acc) de treinamento e teste, além do tempo de predição (T.P.) das mesmas.

Por meio desta análise, percebeu-se que os modelos ResNet 50, 101 e 152 tiveram melhor acurácia para a identificação de câncer de mama, sendo todos iguais a 98%, enquanto a ResNet-158 camadas retornou melhor acurácia, sendo esta igual a 88,36%.

Além da acurácia, o trabalho ilustrou que, por exemplo, o tempo de predição para detecção de *malware* foi de 2.701 segundos para a ResNet 18, enquanto a ResNet 152 obteve um tempo de predição igual a 9.248 segundos. Similarmente, para a identificação de câncer de mama, obteve-se um tempo de execução de 1.131 segundos para a ResNet 18, e 2.131 segundos para a ResNet 152. Esses resultados mostraram o aumento de custo computacional em redes residuais, a medida que modelos mais profundos são utilizados.

Entretanto, não foram realizados testes de custo computacional para o treinamento, apenas para teste (tempo de predição). Além disso, as arquiteturas utilizadas foram as ResNet 18, 50, 101 e 152. Esses modelos foram propostos para bases de dados como a ImageNet, não se adequando a bases mais simples.

Por outro lado, (ELGENDI et al., 2020) compara 16 diferentes modelos de rede considerando a acurácia média por meio de validação cruzada, como também o tempo de treinamento (T.T.). O trabalho realizou a análise apenas com uma única base de imagens contendo exames de raio-X para diagnóstico do vírus COVID-19, sendo avaliadas duas redes residuais (ResNet-18, ResNet-50 e ResNet-101), como também arquiteturas como VGG-19 (SIMONYAN; ZISSERMAN, 2014) e GoogleLeNet (SZEGEDY et al., 2014).

Neste trabalho, avaliou-se a performance apenas em redes pré-treinadas com a base ImageNet, sendo treinada a última camada do modelo, utilizada para classificação. A utilização de modelos pré-treinados tornou-se necessário, principalmente, pela quantidade de amostras do banco de dados, sendo este composto por 85 amostras de pacientes com COVID, e 1,576 amostras de pacientes saudáveis.

Devido a quantidade de amostras, como também o desbalanceamento entre as classes, realizou-se três experimentos para cada modelo de rede, variando-se a proporção de dados de treinamento e validação (50%-50%, 80% treino e 20% validação, 70% treino e 30% validação).

De acordo com estes experimentos, o tempo de treinamento, considerando uma separação de dados de proporção 80% e 20%, foi de 197 segundos para a ResNet-18, 427 segundos para a ResNet-50 e e 732 segundos para a ResNet-110, observando-se assim o aumento de custo computacional entre estas arquiteturas. O mesmo resultado pôde ser observado para as demais separações de dados de treino e validação, podendo-se concluir que o tempo de treinamento, assim como o tempo de predição descrito em (KHAN et al., 2018), aumenta consideravelmente, conforme aumento da quantidade de camadas em redes residuais. Quanto a acurácia, o trabalho definiu que os melhores resultados foram alcançados por meio da ResNet-50, independetemente da proporção de dados utilizados.

Similarmente, os modelos ResNet152 e ResNet50 são explorados juntamente com modelos *Inception* em (KHAN et al., 2019) para avaliar o modelo mais adequado para identificação de imagens não reais para autenticação de sistemas biométricos (e.g. imagens resultantes de captura de tela ou utilização de vídeos gravados anteriormente à autenticação do sistema), sendo este processo conhecido como anti-*spoofing*.

A comparação de resultados neste trabalho foi realizada por meio da métrica acurácia, em que, para cada arquitetura, foram realizados seis experimentos, sendo estes combinações de três fatores principais:

- A utilização de pesos iniciais da base ImageNet, ou inicialização aleatória dos pesos;
- Treinamento de todas as camadas das redes, ou apenas das camadas completamente conectadas, sendo estas as últimas camadas dos modelos; Neste caso, o trabalho visou identificar se, para a identificação de imagens não reais, seria necessário realizar o treinamento completo da rede, ou melhores resultados seriam alcançados por meio de transferência de aprendizado;
- A taxa de aprendizagem, sendo utilizados os valores: 0,001 e 0,00001.

Por meio dessa análise, o trabalho identificou que, dentre as arquiteturas de rede ResNet 152, Inception e ResNet 50, o modelo ResNet 152 obteve maior acurácia para

a base de dados de teste avaliada. Esse trabalho também ilustrou que, a melhor performance foi alcançada utilizando os pesos da ImageNet, treinando apenas as camadas completamente conectadas. Além disso, a taxa de aprendizado igual a 0,001 obteve melhores resultados do que a de 0,00001.

Sendo assim, apesar desses trabalhos compararem o desempenho de diferentes arquiteturas de rede, pode-se dizer que utilizaram uma quantidade reduzida de bases de dados, conforme descrito na Tabela 3. Dessa forma, torna-se inviável generalizar o desempenho de tais arquiteturas utilizando esses estudos. Essa dissertação, por outro lado, propôs um trabalho agnóstico analisando o desempenho dessas arquiteturas. Além disso, não foram realizados testes estatísticos e, exceto o trabalho original da ResNet (HE et al., 2015), nenhum trabalho teve como foco bases de dados simplificadas, utilizando arquiteturas de redes residuais próprias para a ImageNet. Sendo assim, a Tabela 3 compara os trabalhos citados de acordo com a quantidade de bases de dados e de redes residuais, assim como as métricas de avaliação.

Trabalho	**Qtde de bases**	**Redes Residuais**	**Métricas**
(HE et al., 2015)	1	ResNets I-V	Erro (%)
(KHAN et al., 2018)	2	ResNets18,50,101,152	Acc e T.P.
(ELGENDI et al., 2020)	1	ResNet-50, ResNet-152	Acc e T.T.
(KHAN et al., 2019)	1	ResNet50-ResNet-152	Acc
Este trabalho	**15**	**ResNets I-IV**	**F1, T.T.**

Tabela 4 – Relação entre os trabalhos relacionados e o trabalho atual considerando a quantidade de bases de dados estudadas, os modelos de redes residuais e métricas de avaliação.

Além desses trabalhos, essa seção expõe também estudos relacionados a similaridade intra-classe, sendo este conceito utilizado na literatura para propor alterações arquiteturais em redes neurais, além de estratégias para escolha de classes em bases de imagens, como também para *data augmentation*.

Sendo assim, (KAHRAMAN, 2019) analisa a similaridade intra-classe para escolha de 21 classes em uma base de imagens que contém 101 classes, em que cada classe corresponde a um tipo de comida. O objetivo geral deste trabalho foi utilizar arquiteturas de redes neurais convolucionais para prever quantas calorias existem em um prato de comida, de forma a auxiliar a dieta de pessoas com obesidade, uma vez que estas saberiam, em maiores detalhes e rapidamente (e.g. por meio de uma aplicação para dispositivos móveis), a quantidade de calorias ingeridas.

Apesar de existirem aplicações similares no mercado, o trabalho exibe as dificuldades em retornar um cálculo de calorias preciso para o usuário quando as comidas são similares, como dois tipos de carne, ou ainda sopas preparadas de diferentes formas). Sendo

assim, o trabalho analisou a variância intra-classe, calculada por meio da correlação, para 101 diferentes tipos de comidas.

Ao fazer isso, selecionou-se as 21 classes com maior variância intra-classe para treinamento das redes convolucionais. Ao todo, quatro arquiteturas de redes foram avaliadas, sendo elas a ResNet50, GoogleNet (Inception v3), MobileNet and VGG-16. Os modelos foram avaliados de acordo com a medida F1, precisão e *recall*, sendo realizados, para cada rede, três diferentes experimentos, variando-se a proporção de dados de treinamento e teste. Como resultados, a GoogleNet obteve, para medida F1, precisão e *recall*, respectivamente, taxas iguais a 86,60, 88,73 e 87,63, sendo esta a rede com melhores resultados para a base de dados avaliada.

(WEI et al., 2020) propõe uma similaridade intra-classe espectral como estratégia de *data augmentation* para classificação de imagens espectrais. Nestas imagens, usualmente, cada pixel armazena uma curva espectral do objeto, existindo uma diversidade de canais para análise, ao contrário do que ocorre em espaços de cores como RGB, que apenas possui três canais. Sendo assim, imagens espectrais são utilizadas em contextos em que são necessários detalhes mínimos para diferenciar os objetos, tais como aplicações voltadas para identificação e extração de minérios.

Entretanto, tal grau de detalhamento resulta em um processo trabalhoso para criação de rótulos dessas imagens, de forma que as bases de dados, usualmente, possuem poucas amostras. Sendo assim, o trabalho propõs um algoritmo para *data augmentation*, analisando a similaridade intra-classe, com base em métodos de distância, entre um pixel escolhido aleatoriamente e que não possui rótulo, e pixels rotulados, rotulando-o com uma das duas classes com maior coeficiente de similaridade.

Após gerar uma base com maior quantidade de amostras por meio deste método, avaliou-se o algoritmo por meio de três bases de dados, comumente utilizadas para *benchmark* de imagens espectrais: Indian Pines (imagens de 220 canais e resolução 145 $\times$ 145), Pavia University (imagens com 115 canais e resolução 610 $\times$ 340), e a base de dados Salinas (imagens com 224 canais e resolução 512 $\times$ 217).

Após a realização dessa estratégia, uma CNN de três camadas convolucionais e uma camada completamente conectada foi treinada e avaliada, além de modelos SVM e *spatio-spectra Laplacian* SVM. Por fim, os resultados da rede convolucional foram comparados com o estado da arte, os quais possuem menor acurácia.

Por fim, os trabalhos (PILARCZYK; SKARBEK, 2019b) e (PILARCZYK; SKARBEK, 2019a) implementam modificações em redes convolucionais para melhoria de desempenho para problemas com alta variância intra-classe. Nesse contexto, o primeiro utiliza a variância, covariância e média de cada classe para cálculo da função de perdas da rede, e propõe uma camada adicional (denominada Hadamard) para criação de vetores

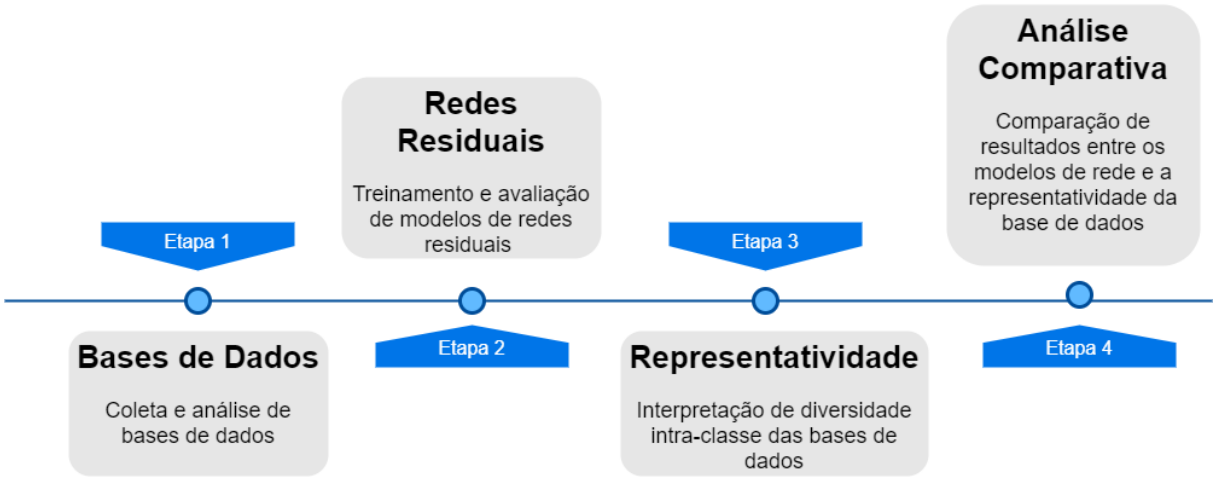
embeddings, cujo objetivo é diferenciar as *features* de maior similaridade e assim reduzir problemas intra-classe. Esse vetor é visualizado no trabalho por meio da Análise de Componentes Principais (PCA do inglês *Principal Component Analysis*), sendo utilizadas as bases de imagens MNIST e Fashion MNIST para avaliação do modelo proposto. O segundo trabalho, por outro lado, propõe uma abordagem denominada *Group Sensitive Triplet Sampling* (GS-TRS), cujo objetivo principal é utilizar a variância intra-classe no cálculo de uma função de perdas de trigêmeos (do inglês *triplet loss)*.

4 Metodologia

Conforme descrito nas Seções 1 e 2, esse trabalho tem como objetivo avaliar a diferença de desempenho entre modelos de redes residuais, assim como investigar possíveis problemas de representatividade, que impactam em tal desempenho.

Sendo assim, para realização desse estudo, a metodologia do trabalho divide-se em quatro etapas principais (Figura 16). Inicialmente, a Seção 4.1 introduz as bases de dados que serão utilizadas e Seção 4.2 ilustra como serão realizados o treinamento e avaliação dos modelos de redes, enquanto as Seções 4.3 e 4.4 focam na representatividade das bases de dados e sua relação com os modelos treinados. Além disso, ainda que técnicas de *data agumentation* não tenham sido utilizadas na Seção 4.2, foram realizados também estudos de caso utilizando tais técnicas, com o objetivo de comparar os resultados, alterando-se a diversidade de amostras da base.

Figura 16 – As quatro etapas principais da metodologia.



Fonte: autoria própria.

4.1 Bases de Dados

Esse trabalho considerou a utilização de bases de dados de domínio público (Tabela 5), diferenciando-se as mesmas, inicialmente, de acordo com a quantidade de amostras por classe, tipos de objetos a serem classificados por essas bases, a utilização de imagens em escala de cinza, binárias e canais RGB, além do balanceamento de classes. O objetivo da utilização desses fatores baseou-se na investigação dos problemas expostos abaixo:

- **Espaço de cores:** a utilização de uma imagem RGB resulta em dimensões H x W x 3, sendo 3 a quantidade de canais da imagem, em contraste a imagens binárias, por exemplo, que possuem dimensões H x W x 1. Sendo assim, para uma imagem de dimensões 32 x 32 x 3, por exemplo, tem-se 3072 atributos inicialmente, enquanto uma imagem binária possui 1024 atributos. De forma semelhante, utilizou-se bases de dados de dimensão 28 x 28 e 128 x 128, para que fosse analisado se o aumento de atributos iniciais poderia resultar na utilização de uma rede residual com maior profundidade;

- **Balanceamento:** em aprendizagem de máquina, bases desbalanceadas podem resultar em erros de generalização. Dessa forma, avaliou-se o impacto dessa característica nos experimentos com diferentes redes e diferentes bases de dados;

- **Quantidade de amostras por classe:** verificou-se possíveis diferenças de resultados entre bases com menor (100 imagens/classe, por exemplo) e maior (1000 imagens/classe, por exemplo) quantidade de amostras por classe;

- **Tipos de objetos:** variou-se os tipos de objetos presentes nas imagens, utilizando bases simples apenas de dígitos como o MNIST (LECUN et al., 1998b), além de bases como CIFAR100 (KRIZHEVSK, 2009) que possuem, por exemplo, classes de diferentes tipos de vegetais, frutas e animais.

Base de Dados	**Tipos de objetos**	**Quantidade de Classes**
Animals10	Animais	10
CIFAR10	Veículos e animais	10
CIFAR100	Veículos, animais, frutas e vegetais, pessoas, árvores e objetos domésticos	100
Fashion MNIST	Roupas	10
GTRSB	Placas de trânsito	42
Man Woman Faces	Rostos de homem e mulher	2
MNIST	Dígitos	10
Mobile Gallery	Carros, memes, montanhas, árvores, imagens do WhatsApp e selfies	5
Natural Images	Veículos, frutas, pessoas e animais	7
EMNIST - Balanced	Dígitos e letras do alfabeto	46
EMNIST - ByClass	Dígitos e letras maiúsculas e minúsculas	61
EMNIST - Digits	Dígitos	10
EMNIST - Letters	Letras do alfabeto	26
EMNIST - MNIST	Dígitos	10
Stanford Dogs	Cachorros	120

Tabela 5 – Descrição das bases de dados e quantidade de classes existentes.

Dentre essas bases, quanto aos canais de cor, Animals10 (ALESSIO, 2019), CIFAR10 e CIFAR100 (KRIZHEVSK, 2009), Mobile Gallery (ANIMESH, 2018), Natural Images (ROY et al., 2018) possuem imagens em RGB, enquanto as demais (LECUN et al., 1998b) (XIAO; RASUL; VOLLGRAF, 2017) (COHEN et al., 2017) e Stanford Dogs (FEI-FEI et al., 2011) estão binarizadas ou em escala de cinza. A Tabela 6 exibe quais tipos de objetos estão contidos em cada base, assim como a quantidade de classes. Percebe-se por meio desta tabela, a existência de bases com apenas 2 classes (Man Woman Faces) e 120 classes (Stanford Dogs), dessa forma, pode-se verificar possíveis impactos no desempenho, de acordo com a quantidade delas.

Além do espaço de cores e quantidade de classes, diferenciam-se as bases quanto ao tamanho (i.e. quantidade de amostras) e a presença ou não de balanceamento. Com isso, pode-se analisar o desempenho das redes com bases muito pequenas, como Man Woman Faces (3338 amostras totais), e também de bases maiores, como EMNIST Digits (250.000 amostras).

Percebe-se, também, a presença de cinco bases de dados de manuscritos EMNIST. Essas bases tiveram como objetivo analisar o desempenho das redes para classificar letras e dígitos individualmente (bases Letters, Digits e MNIST), como também a utilização de bases com os dois tipos de manuscritos em conjunto: Balanced, ByClass, ByMerge). Nesse contexto, investiga-se na base ByClass, possíveis semelhanças considerando letras maiúsculas e minúsculas (ByClass e ByMerge), e a diferença de desempenho utilizando as mesmas classes balanceadas (Balanced e ByMerge).

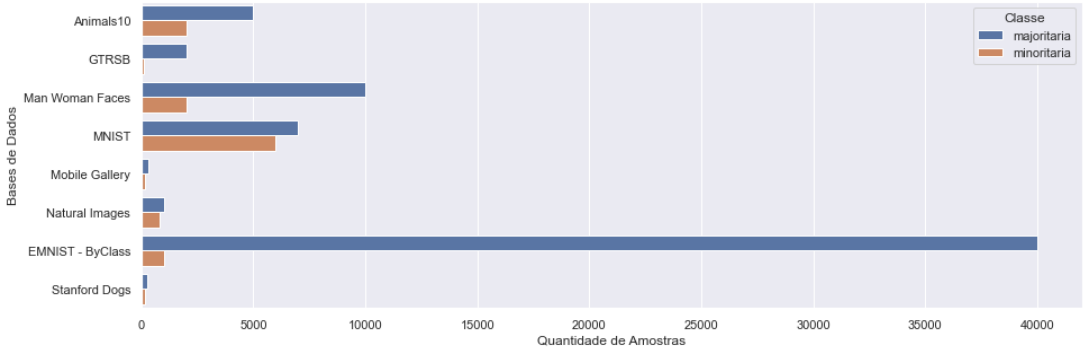
Base de Dados	**Quantidade Totais de Amostras**	**Balanceamento**
Animals10	26178	Desbalanceado
CIFAR10	50000	Balanceado
CIFAR100	50000	Balanceado
Fashion MNIST	60000	Balanceado
GTRSB	39209	Desbalanceado
Man Woman Faces	3338	Desbalanceado
MNIST	60000	Desbalanceado
Mobile Gallery	1266	Desbalanceado
Natural Images	6899	Desbalanceado
EMNIST: Balanced	115000	Balanceado
EMNIST: ByClass	697932	Desbalanceado
EMNIST: Digits	250000	Balanceado
EMNIST: Letters	130000	Balanceado
EMNIST: MNIST	60000	Balanceado
Stanford Dogs	20580	Desbalanceado

Tabela 6 – Quantidade de amostras e status de balanceamento das bases de dados.

Por fim, analisou-se a quantidade de amostras das classes minoritárias e majoritá-

rias para as bases de dados desbalanceadas, objetivando analisar possíveis erros da rede causados por desbalaceamento. A Figura 17 ilustra os gráficos para classes minoritárias e majoritárias de cada base.

Figura 17 – Relação entre Quantidade de Amostras Majoritárias e Minoritárias dos bancos de dados utilizados



Fonte: propria.

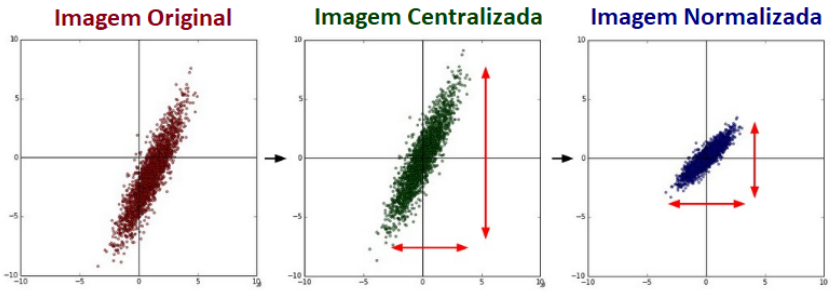
4.2 Redes Residuais

Considerando as arquiteturas de redes residuais descritas na Seção 2.2, uma vez que as bases de dados utilizadas no trabalho são menores do que a base ImageNet (menor dimensão de imagem de entrada, quantidade de imagens na base e total de classes), optou-se pela utilização das arquiteturas simplificadas de ResNet.

Devido a limitação de recursos de *hardware*, a ResNet de 1202 camadas não foi utilizada. Sendo assim, optou-se pelas ResNets de 20, 32, 44, 56 e 110 camadas, em que as mesmas foram treinadas e avaliadas para cada base de dados individualmente. Como ilustrado na Seção 2, esse trabalho não realiza *data augmentation* como treinamento inicial das redes, sendo realizadas apenas duas etapas de pré-processamento ilustradas na Figura 18:

- **Centralização de pixels:** subtraiu-se o valor médio dos dados de entrada para cada atributo dos mesmos. Esse processo centraliza os dados tendo o ponto (0,0) como origem e foi utilizado para redução de problemas do gradiente conforme indicando na metodologia para as arquiteturas originais da ResNet (HE et al., 2015).
- **Normalização:** os dados de entrada foram normalizados em uma escala de intervalo [0,1], para melhorias de convergência do modelo. Esta técnica foi utilizada considerando os valores mínimos e máximos dos dados seguindo a fórmula $\frac{x - x.min()}{x.max() - x.min()}$, sendo utilizado o valor máximo igual a 255, uma vez que o intervalo de pixels no espaço RGB é [0,255].

Figura 18 – Resultado da aplicação da centralização de pixels e normalização em uma imagem



Fonte: (LI, 2020).

Essas etapas foram utilizadas em todas as bases de dados tendo-se, como referência, o pré-processamento padrão realizado na ResNet e as bases foram particionadas em treinamento, validação e teste utilizando a técnica *Holdout*. Adicionalmente, para as bases MNIST e Fashion MNIST, realizou-se uma operação de *padding* para redimensonar os dados de 28 x 28 para 32 x 32. Isso foi necessário devido a incopatilidade das arquiteturas ResNet com imagens de dimensões de entrada inferiores a 32 x 32.

Além disso, os parâmetros descritos na Tabela 7 também foram utilizados considerando os valores padrão de treinamento das ResNets.

Parâmetro	**Valor**
Otimizador	Adam
Taxa de Aprendizagem	0,001
Total de Épocas	200

Tabela 7 – Parâmetros de treinamento

Utilizou-se também funções para monitoramento de treinamento por meio de duas *callbacks*: *EarlyStopping* e redução da taxa de aprendizagem, de acordo com a curva de Plateau. Para *Early Stopping*, uma técnica que monitora e encerra o treinamento após uma determinada quantidade de épocas, cuja quantidade é definida por meio do parâmetro denominado paciência, em que não houve melhorias da acurácia de validação. Utilizou-se uma paciência de 20 épocas monitorando-se a acurácia de validação, enquanto a redução da taxa de aprendizagem utilizou uma paciência de 5.

Por fim, o tamanho do *batch* utilizado para treinamento das arquiteturas variou para cada base e modelo de rede. Essa variação tornou-se necessária devido a requisitos de *hardware*. Uma vez que, quanto maior, valor de *batch*, menor tempo de treinamento da rede

e maior memória gráfica é necessária. Os valores indicados pela Tabela 8 correspondem ao suportado para preenchimento total de memória VRAM da placa gráfica. Entretanto, para verificar possíveis alterações de performance, foram realizados testes utilizando um *batch* fixo de valor igual a 16, optando-se por este valor por ser o menor *batch* exposto na Tabela 8.

O treinamento foi realizado utilizando a API Keras (Versão 2.3.1) e o *framework* Tensorflow (Versão 1.14.0) em uma máquina com processador Intel(R) Core(TM) i5-9400F CPU@2.90GHz, 24GB de memória RAM e uma placa de vídeo RTX 2070 com VRAM de 8GB.

Base de Dados	**ResNet20**	**ResNet32**	**ResNet44**	**ResNet56**	**ResNet110**
Animals10	128	64	64	32	16
CIFAR10	256	256	256	256	256
CIFAR100	256	256	256	256	256
Fashion MNIST	256	256	256	256	256
GTRSB	64	64	64	64	64
Man Woman Faces	128	64	64	32	16
MNIST	256	256	256	256	256
Mobile Gallery	128	64	32	32	16
Natural Images	128	64	64	32	16
EMNIST:Balanced	256	256	256	256	256
EMNIST:ByClass	256	256	256	256	256
EMNIST:ByMerge	256	256	256	256	256
EMNIST:Letters	512	512	512	512	256
EMNIST:MNIST	512	512	512	256	128
Stanford Dogs	128	64	64	32	16

Tabela 8 – Tamanho do *batch* utilizado para cada arquitetura de rede

Após o treinamento, cada modelo foi avaliado considerando como métrica principal a medida *F1* descrita na Equação 4.1. Além disso, obteve-se o tempo de processamento necessário para treinamento de cada rede afim de comparar os recursos computacionais necessários para a mesma.

$$F_1 = 2 \times \frac{precisão \times revocação}{precisão + revocação} \tag{4.1}$$

Posteriormente, realizou-se um teste de significância utilizando como métrica a distribuição t de Student. Esse teste teve como hipótese avaliar se as melhorias de desempenho dos modelos eram significativas e utilizou os parâmetros descritos na Tabela 9.

Parâmetro	**Valor**
alpha	0,05
grau de liberdade	14
t-crítico	2,1315

Tabela 9 – Parâmetros utilizados para realização do teste de significância

O teste foi realizado considerando uma amostra igual a 15, uma vez que 15 bases de dados foram utilizadas. Como a distribuição é válida apenas entre duas sequências, o teste foi realizado de acordo com a abordagem um contra todos, isto é, foram testadas apenas duas sequências por vez, em que cada sequência contém a medida *F1* da amostra para cada modelo de rede. A avaliação foi realizada considerando o p-valor e a probabilidade t de cada teste realizado em que, para que o desempenho de um modelo profundo seja considerado melhor, deve-se ter como premissas que para todos os testes p-valor deverá ser menor que o valor de *alpha*.

Sendo assim, a Figura 19 ilustra o fluxograma principal para treinamento e avaliação dos modelos de rede utilizados em que a Etapa 1 corresponde ao pré-processamento, treinamento do modelo e obtenção da métrica de avalição (medida *F1*), assim como o custo computacional do mesmo. Para cada base de dados, repete-se essa etapa para as cinco arquiteturas de rede avaliadas, sendo assim, como foram utilizadas 15 bases, tem-se um total de $15 \times 5 = 75$ experimentos.

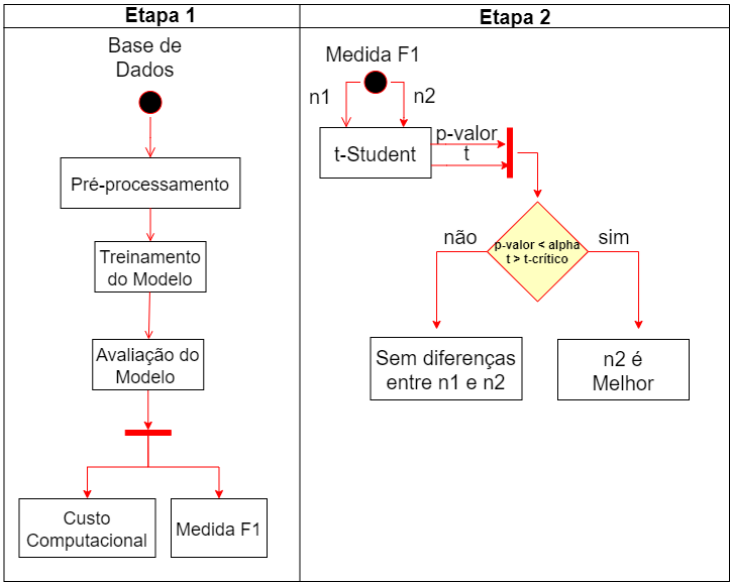
Após a realização de todos os experimentos da primeira fase, a Etapa 2 representa o teste de significância realizado, em que *n1* e *n2* correspondem a todas as medidas *F1* obtidas anteriormente para duas arquiteturas de rede. Além disso, as medidas *F1* de *n1* correspondem a um modelo com menor quantidade de camadas do que as de *n2*. Dessa forma, de acordo com a Etapa 2, será considerado o desempenho de uma arquitetura significativamente melhor apenas caso o p-valor for menor do que o valor de *alpha* e t maior que t-crítico.

Esse processo é repetido até que todas as combinações de modelos de rede sejam realizadas. Sendo assim, para combinações considerando dois modelos de rede, tem-se um total de dez experimentos, uma vez que existem cinco modelos de rede nesse trabalho. Por fim, como 15 bases de dados foram utilizadas, *n1* e *n2* possuem 15 valores de medidas *F1* correspondentes a cada base.

Por exemplo, considerando a base de dígitos MNIST, as redes ResNet 18, 32, 44, 56 e 110 foram treinadas e avaliadas com esta base. Posteriormente, as medidas *F1* das redes de 18 e 32 camadas foram avaliadas de acordo com o teste de significância. O mesmo experimento é realizado para as demais arquiteturas, até que todas as combinações arquiteturais sejam feitas. Se, em nenhuma dessas combinações, p-valor ou t ultrapassarem os limiares estabelecidos, então pode-se dizer que não há comprovação estatística que a

utilização de uma arquitetura de maior profundidade para essa base é relevante.

Figura 19 – Fluxograma correspondente as duas etapas de treinamento e avaliação dos modelos residuais



Fonte: autoria própria.

4.3 Representatividade da Base de Dados

Conforme descrito na Seção 2.3, a representatividade de uma base de dados para DL pode ser interpretada a partir da análise de agrupamento. Para este trabalho, tal representatividade será medida utilizando apenas amostras intraclasse considerando que:

- Uma classe que resulte em melhor desempenho para a rede tenderá a conter uma diversidade de amostras, seja essa diversidade formada por transformações (e.g. rotação, translação) ou tipos de objetos da classe (e.g. diferentes raças de um animal). Neste caso, os *clusters* formados idealmente são heterogêneos com o valor da silhueta próximo ou igual a zero.

- Uma classe que pode estar relacionada com o menor desempenho da rede pode conter amostras muito similares entre si devido a ausência (ou limitação) de diversidade dos objetos da classe. Neste caso, os *clusters* formados, idealmente, tendem a ser homogêneos com valores de silhueta maiores do que zero e tendendo a um.

Sendo assim, para cada classe da base de dados, individualmente, utilizou-se o algoritmo *k-means* para realização do agrupamento. Posteriormente, os grupos foram analisados tendo-se como métrica o valor da silhueta dos mesmos. Com isso, as classes com

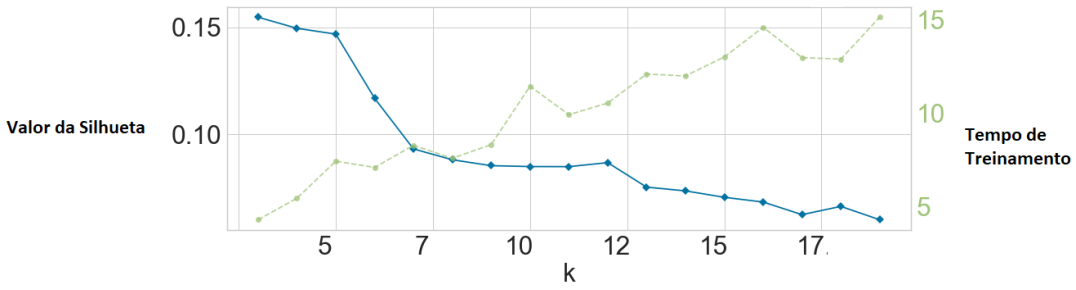
maiores valores da silhueta representam classes que podem conter possíveis problemas de representatividade e, consequentemente, redução de desempenho das redes.

Como no *k-means* a quantidade de k *clusters* é definida previamente, optou-se por um valor de k pequeno e igual a três. Esse valor foi definido considerando-se que, em agrupamento, os valores de k que indicam a formação de grupos com amostras intraclusters similares, são aqueles cujo valor da silhueta estão próximos a zero e, consideravelmente, constantes, formando uma curva de "cotovelo"(do inglês *elbow*) (J.ROUSSEEUW, 1987).

Sendo assim, para avaliação de representatividade, a análise é realizada de forma oposta considerando a formação de *clusters* com amostras dissimilares, optando-se então por selecionar um valor de k previamente à formação do "joelho"para cada classe. Por exemplo, a Figura 20 representa os valores de silhueta para a classe 0 da base de dados Fashion MNIST utilizando k variando de 3 até 20. Percebe-se que o maior valor da silhueta é obtido para k igual a três (0,15) e o "joelho"passa a ser formado a partir de k igual a 6. Além disso, o tempo de treinamento aumenta consideravelmente, conforme aumento de k. Por fim, vale salientar que, para esse trabalho, foram considerados apenas experimentos intraclasse, isto é, o agrupamento foi formado apenas por classe, com o objetivo de compreender a diversidade apenas de cada uma delas.

Dessa forma, a análise poderia ser realizada com k entre 3 e 15 para este banco de dados, utilizando-se o valor três devido ao custo computacional. O mesmo padrão foi observado para todos os bancos utilizados, sendo então este valor adotado como padrão para todos os agrupamentos realizados nesta etapa.

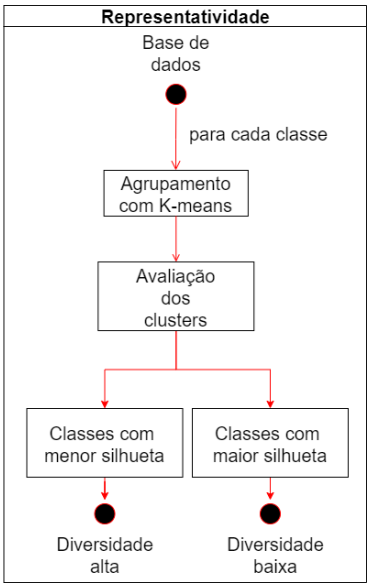
Figura 20 – Valor da silhueta e tempo de treinamento para k clusters variando de 3 até 20 utilizando a base de dados Fashion MNIST



Fonte: autoria própria.

Por fim, a Figura 21 descreve os três passos necessários para esta etapa da metodologia por meio de um fluxograma. Esses passos são repetidos para cada banco de imagens e a performance das classes com maior valor de silhueta serão analisadas na Seção 4.4.

Figura 21 – Fluxograma que ilustra a formação e análise de agrupamento para indicação de representatividade intraclasse



Fonte: autoria própria.

4.4 Análise Comparativa

Após a obtenção das métricas das Seções 4.2 e 4.3, comparou-se os menores valores de medida *F1* e maiores de silhueta, afim de verificar se ambos pertenciam as mesmas classes. Em caso positivo, pode-se ter como possível solução de melhoria de desempenho da rede, a modificação da representatividade da base de dados, ao invés do aumento de profundidade da arquitetura do modelo.

Realizou-se essa avaliação para todas as bases de dados descritas em 4.1 e, como estudo de caso, repetiu-se os treinamentos das bases de dados CIFAR100, Stanford Dogs e Fashion MNIST utilizando técnicas de *data augmentation* para modificar a representatividade deste banco e, verificar assim, possíveis melhorias nas classes indicadas com baixa representatividade.

As técnicas foram aplicadas durante o treinamento do modelo a cada *batch*, utilizando a ferramenta Albumentations (BUSLAEV et al., 2020), sendo avaliados os métodos descritos na Tabela 10. Essas técnicas foram agrupadas em duas categoriais: tradicionais e *heavy*, em que tradicionais correspondem a técnicas de rotação, translação e escala, enquanto o segundo tipo corresponde a abordagens de pré-processamento de imagens adicionais como equalização, alteração de brilho e contraste da imagem *albumentations*.

Nesse contexto, os métodos tradicionais foram aplicados em todos os experimentos e os demais foram escolhidos e aplicados aleatoriamente com uma probabilidade igual a 0,5 por meio da biblioteca *albumentations*.

Como a base de dados CIFAR100 possui imagens no espaço de cores RGB, enquanto as demais possuem imagens em escala de cinza, os métodos relativos a este espaço de cor foram aplicados apenas nesta base.

Esses experimentos foram realizados seguindo as mesmas arquiteturas e parâmetros descritos na Seção 4.2.

Método	**Descrição**	**I**	**II**	**III**
Cutout	Remoção de blocos de pixels da imagem	X	X	X
CoarseDropout	Remoção de blocos de pixels da imagem	X	X	X
RandomBrightness	Aumento ou redução de brilho	X	X	-
RandomSizedCrop	Recorte e escala de uma parte da imagem	X	X	-
RandomContrast	Aumento ou redução de contraste	X	X	X
Equalize	Aplica a equalização do histograma	X	X	-
FancyPCA	Aplica o algoritmo PCA em blocos de pixels	X	-	-
CLAHE	Equalização local do histograma	X	X	X
IAASharpen	Aplicação de filtro de aguçamento	X	X	-
RGBShift	Inversão dos canais de cores RGB	X	-	-

Tabela 10 – Métodos de heavy augmentation utilizados para as bases CIFAR100 (I), StanfordDogs (II) e Fashion MNIST (III).

5 Resultados

Visando analisar o desempenho das redes residuais, assim como possíveis relações entre o mesmo, e a representatividade das bases de dados, esse capítulo divide-se em duas seções principais.

Na Seção 5.1 são ilustrados os resultados, considerando apenas a medida *F1* dos modelos avaliados. Nessa seção são realizados também experimentos relativos ao teste de significância e a análise de custo computacional. Além disso, a Seção 5.2 ilustra a análise de possíveis classes que retornem menor desempenho dos modelos de CNNs, bem como estudos de caso utilizando as bases de dados Fashion MNIST, CIFAR100 e Stanford Dogs.

5.1 Redes Residuais

Seguindo as etapas ilustradas na seção metodológica, as cinco arquiteturas de rede foram treinadas e avaliadas. A base de teste foi obtida utilizando-se 15% dos dados do banco por meio da técnica *Holdout*. A Tabela 11 ilustra a medida *F1* média para cada arquitetura.

Dataset	**ResNet20**	**ResNet32**	**ResNet44**	**ResNet56**	**ResNet110**
Animals10	0,69	**0,70**	0,68	0,68	0,70
CIFAR10	0,79	0,80	**0,82**	0,77	0,81
CIFAR100	0,47	0,48	0,49	0,48	**0,52**
Fashion MNIST	0,90	**0,93**	0,92	0,93	0,92
GTRSB	**1**	0,98	0,99	0,99	0,99
Man Woman Faces	0,89	0,90	0,89	**0,92**	0,90
MNIST	0,9924	0,9908	**0,9942**	0,98	0,9908
Mobile Gallery	0,87	**0,86**	0,77	0,85	0,79
Natural Images	**0,95**	0,92	0,93	0,91	0,93
EMNIST - Balanced	**0,89**	0,89	0,89	0,86	0,86
EMNIST - ByClass	0,87	0,86	0,87	**0,87**	0,87
EMNIST - Digits	**1**	1	1	1	1
EMNIST - Letters	0,94	0,94	0,94	**0,95**	0,95
EMNIST - MNIST	**0,99**	0,99	0,99	0,99	0,99
Stanford Dogs	0,12	0,08	**0,13**	0,11	0,10

Tabela 11 – A medida *F1* média de cada rede residual para as bases de dados utilizadas.

Percebe-se que, de forma geral, para cada base de dados, o desempenho é similar entre as arquiteturas de rede, não havendo um padrão específico de escolha da mesma de acordo com as características da base de dados. Por exemplo, as bases de dados Man

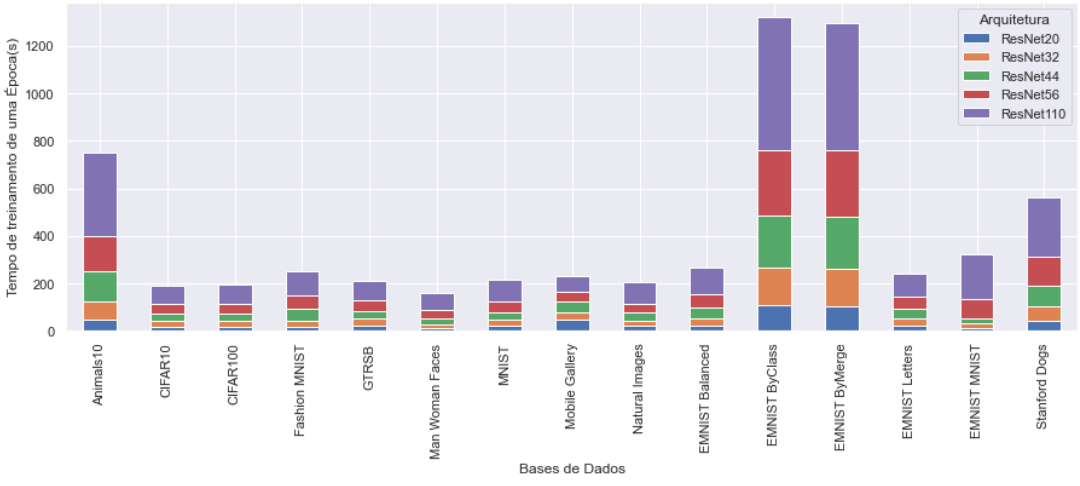
Woman Faces e Mobile Gallery possuem menor quantidade total de amostras (3338 e 1266, respectivamente) do que as demais e o melhor desempenho é alcançado com as redes de 56 e 32 camadas respectivamente. Por outro lado, EMINIST Balanced possui 115000 amostras ao todo e obteve melhor desempenho com a ResNet de apenas 20 camadas.

Similarmente, canais de cor não interferem necessariamente na escolha de uma arquitetura, uma vez que as bases Animals10 e Fashion MNIST alcançam melhores resultados com a ResNet 32 (medidas *F1* de 0,70 e 0,93, respectivamente). Além disso, também não foram encontrados padrões entre bases balanceadas.

As bases CIFAR100 e Stanford Dogs retornaram resultados com medida *F1* abaixo de 0,50 para todas as redes testadas. Essas bases são as que possuem maior quantidade de classes (100 e 120 respectivamente) e pouca quantidade de amostras por classe (500 e 150-250 respectivamente). Sendo assim, a performance pode ser justificada, inicialmente, devido a pequena quantidade de amostras por classe.

Ao realizar os experimentos da Tabela 11, obteve-se também o tempo médio de treinamento por época, ilustrado na Figura 22. Percebe-se que, conforme o aumento de profundidade de rede, maior tempo de treinamento é necessário.

Figura 22 – Gráfico comparativo do tempo de treinamento de uma época para cada ResNet.



Fonte: autoria própria.

Essa diferença é maior para bases de dados EMNIST devido a quantidade de amostras das mesmas, considerando que as bases EMNIST - ByClass e ByMerge possuem 814255 amostras. Nessas bases, enquanto a ResNet56 demanda 276 s/época, a ResNet110 requer 556 s/época, resultando em um aumento de 280 segundos.

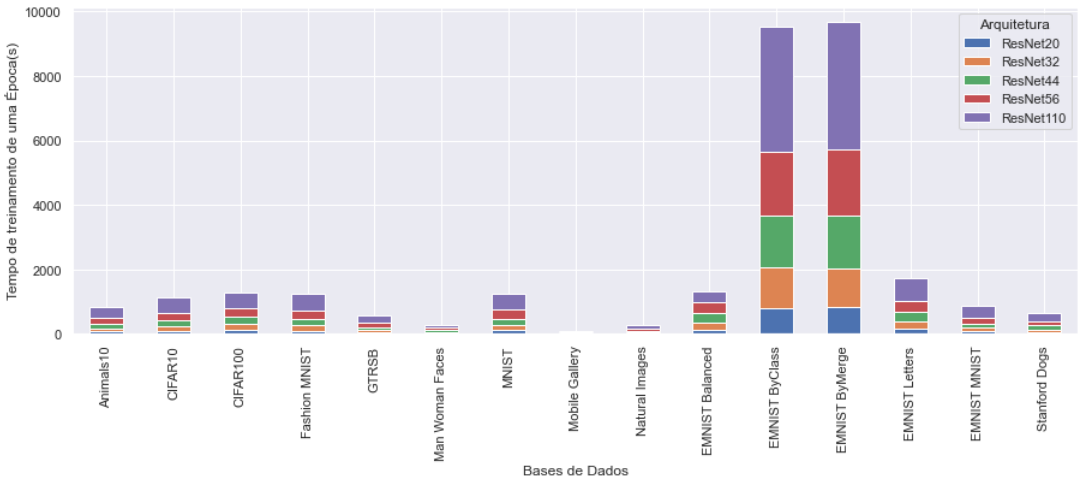
Assim, para a ResNet de maior profundidade, utilizando a *callback Early Stopping*, considerando que o modelo foi treinado com 88 épocas, resultou-se em um tempo de

treinamento total igual a 50688 segundos ou, aproximadamente, 14 horas. Entretanto, ao observar a Tabela 11, o mesmo resultado é alcançado com uma arquitetura de 20 camadas, cujo tempo total de treinamento é de 9504 segundos, ou 2,76 horas para a mesma quantidade de épocas.

O tempo de treinamento ilustrado utilizou a maior quantidade de *batch* variável para cada rede seguindo a Tabela 8, ocupando-se assim um total de 8 GB de memória gráfica. Sendo assim, a mesma análise foi realizada para um *batch* fixo e igual a 16 para todas as redes (Figura 23).

Percebe-se a ocorrência do mesmo comportamento gráfico para todas as bases, entretanto, com maior tempo de treinamento. Sendo assim, para a base EMNIST - ByClass, tem-se 3848 s/época, ao invés de 556 s/época. Esse aumento resulta em 338624 segundos para 88 épocas, ou aproximadamente 4 dias.

Figura 23 – Gráfico comparativo do tempo de treinamento de uma época para cada ResNet para batch fixo e igual a 16.



Fonte: autoria própria.

Essa alteração de *batch* impacta no tempo de treinamento, mas resultados similares de medida *F1* são alcançados. Por exemplo, a Tabela 12 ilustra o desempenho das redes para a base Fashion MNIST variando-se apenas esse parâmetro.

Batch	**ResNet20**	**ResNet32**	**ResNet44**	**ResNet56**	**ResNet110**
256	0,90	0,93	0,92	0,93	0,92
16	0,90	0,93	0,92	0,93	0,92

Tabela 12 – Medida *F1* da base de dados Fashion MNIST variando-se o tamanho do batch.

Sendo assim, percebe-se que o aumento de profundidade é proporcional ao custo computacional, mas não representa necessariamente melhorias de performance.

Por fim, por meio do teste de significância, utilizando a distribuição t de Student, esse estudo analisou as medidas *F1* de cada rede, expostas na Tabela 11, afim de identificar se as melhorias foram significativas.

Conforme metodologia deste trabalho, o teste aceitaria a hipótese de que o aumento de profundidade melhora o desempenho na base de dados, apenas para p-valor menor que 0,05. Entretanto, ao observar os valores da Tabela 13, observa-se que, para todas as arquiteturas testadas, p-valor é superior a 0,9.

p-valor	**Arquiteturas**
0,97	(Resnet20,ResNet32)
0,96	(ResNet20,ResNet44)
0,95	(ResNet20,ResNet56)
0,97	(ResNet20,ResNet110)
0,99	(ResNet32,ResNet44)
0,98	(ResNet32,ResNet56)
0,99	(ResNet32,ResNet110)
0,99	(ResNet44,ResNet56)
0,98	(ResNet44,ResNet110)
0,98	(ResNet56,ResNet110)

Tabela 13 – Teste de significância para medir melhorias de desempenhoria conforme aumento de profundidade do modelo. Realizou-se 10 testes ao todo variando-se as arquiteturas.

Sendo assim, de acordo com o teste de significância, esse estudo estabele que não há comprovações estatísticas de que melhorias de performance são alcançadas apenas aumentando-se a quantidade de camadas do modelo. Além disso, o custo computacional é consideravelmente alto para desempenhos similares, podendo-se então apenas utilizar a ResNet20 de forma geral para a maioria das bases de dados.

5.2 Análise de Representatividade da Base de Dados

Após a realização do agrupamento utilizando algoritmo *k-means*, formou-se três *clusters* para cada classe e a similaridade entre eles foi avaliada utilizando a silhueta como métrica.

De forma geral, percebeu-se que classes com maior representatividade retornavam silhueta em torno de 0 e 0,09, enquanto aquelas com menor representatividade obtiveram o valor de silhueta entre 0,10 e 0,30, conforme indicado no estudo de caso ao fim desta Seção.

Para comparação de desempenho, analisou-se quais as três classes com menor representatividade e quais possuem menor medida *F1*. A Tabela 14 ilustra esses resultados. Percebe-se que, de forma geral, com exceção da base National Images, para todas as bases de dados, no mínimo uma das classes com menor representatividade coincide com aquelas de menor medida *F1*. Nas bases Fashion MNIST e EMNIST - MNIST, em especial, as três classes com menor representatividade também representam aquelas com menor desempenho para as redes.

Base de Dados	Menor Representatividade	Menor Medida F1
Animals10	3, 5, 8	5, 7, 9
CIFAR10	0, 2, 6	2, 3, 5
Fashion MNIST	**2, 4, 6**	**2, 4, 6**
Mobile Gallery	0, 2, 4	0, 3, 4
Natural Images	0, 5, 6	2, 3, 4
EMNIST - Balanced	1, 19, 21,	1, 15, 21
EMNIST - ByClass	1, 18, 46	18, 50, 51
EMNIST - ByMerge	1, 18, 21	21, 40, 44
EMNIST - MNIST	**1, 7, 0**	**1, 7, 0**

Tabela 14 – Classes com menor representatividade medida pelo valor da silhueta e menor medida *F1* das redes avaliadas.

Vale ressaltar que, para as cinco redes testadas, as classes com menor desempenho eram comuns a todas as arquiteturas. Além disso, o estudo dessa seção não contempla as bases MNIST - Digits e GTRSB, uma vez que estas retornaram medida *F1* igual a 1, não existindo assim classes com menor desempenho para essas bases.

Analisou-se também, separadamente, as bases Man Woman Faces, CIFAR100 e Stanford Dogs. Isso tornou-se necessário considerando que a primeira base utiliza apenas duas classes e as outras duas obtiveram desempenho inferior a 50% para todos os modelos testados.

Sendo assim, Man Woman Faces possuiu maior performance com a ResNet56 (medida *F1* igual a 0,92), sendo a classe 1 (mulher) com menor performance (0,86 de medida *F1* contra 0,95 para a classe 0). Ao analisar a representatividade dessas classes, a classe 0 possui silhueta igual a 0,13, enquanto a classe 1 obteve um valor igual a 0,14.

Para a base CIFAR100, todas as classes possuem valor de silhueta entre 0,09 e 0,24, indicando assim a existência de possíveis problemas de representatividade para todas elas. Além disso, essa base foi proposta por (KRIZHEVSK, 2009) com o intuito de conter classes variantes de uma superclasse. Por exemplo, as classes *"orchids"*, *"poppies"*, *"roses"*, *"sunflowers"* e *"tulips"* representam diferentes tipos de flores, sendo todas elas pertencentes a superclasse *"flower"*. Sendo assim, além da possível existência de uma diversidade alta dentro de uma mesma classe, essa base pode conter também, alta diversidade interclasse.

Dessa forma, visando a melhoria de resultados dessa base, realizou-se experimentos com diferentes tipos de *data augmentation* ainda aliados ao uso da rede ResNet-20. Optou-se pela escolha dessa arquitetura considerando o menor custo computacional e pequena variação de performance (medida *F1*) segundo análise de resultados da Seção 5.1. Nesse contexto, a aplicação das técnicas tradicionais rotação, translação, escala e *flip* horizontal retornou uma medida *F1* de 0,58, sendo escolhidas também técnicas adicionais (conhecidas como *heavy augmentation*). A Tabela 15 ilustra os experimentos realizados com tais técnicas, sendo estas escolhidas e aplicadas aleatoriamente, juntamente com as técnicias tradicionais. Percebe-se que, para todos os casos, a medida *F1* foi superior a 0,52, sendo este valor o resultado da ResNet-110 para esta base. Sendo assim, infere-se maior performance através do uso dessas técnicas, ao invés de aumento de profundidade. Na literatura, a performance dessa base possui acurácia a partir de 60% variando-se as técnicas de *augmentation* (BARAN; KUPYN; KRAVCHENKO, 2019), assim como as arquiteturas de rede (HUANG et al., 2016) (HUANG et al., 2019).

Técnicas de augmentation aplicadas	**Medida *F1***
Sem augmentation	0,47
Cutout	0,57
CoarseDropout ou cutout	0,57
CoarseDropout ou cutout; RGBShift	0,55
CoarseDropout ou Cutout; RandomBrightness	0,56
CoarseDropout ou Cutout; RandomSizedCrop	0,52
CoarseDropout ou Cutout; RandomSizedCrop; Equalize; RandomBrithness; RandomContrast	0,54
CoarseDropout ou Cutout; RandomSizedCrop; Equalize; FancyPCA	0,54
CoarseDropout ou Cutout; CLAHE	0,53
CoarseDropout ou Cutout; Equalize; IAASharpen	0,56

Tabela 15 – Resultados de técnicas de data augmentation para a base CIFAR100 utilizando a rede ResNet-20.

Stanford Dogs, por outro lado, possui valores de silhueta entre 0,06 e 0,12, em que 0,06 são classes com melhor desempenho. Por exemplo, as classes 1, 3, 6 e 42 possuem silhueta igual a 0,06 e medida *F1* entre 0,23 e 0,26 para a ResNet-44, enquanto que, para as demais classes, a medida *F1* varia entre 0 e 0,10. Estas quatro classes obtiveram maior medida *F1* entre as 120 classes, juntamente com a classe 87, que foi a única a obter medida *F1* acima de 0,26. Essa classe retornou o valor igual a 0,37 para essa métrica e uma silhueta de 0,07.

Similarmente a base CIFAR100, essa base foi proposta para *fine-grained feature image categorization* (FEI-FEI et al., 2011), isto é, a base contém classes muito similares. Por exemplo, as raças de cachorro Shih-Tzu e Lhasa Apso correspondem a duas classes

para esta base, sendo estas raças muito similares entre si. A Figura 24 ilustra as duas raças descritas, em que percebe-se, dentre outras características semelhantes, altura do cachorro, cor, comprimento e formato dos pêlos.

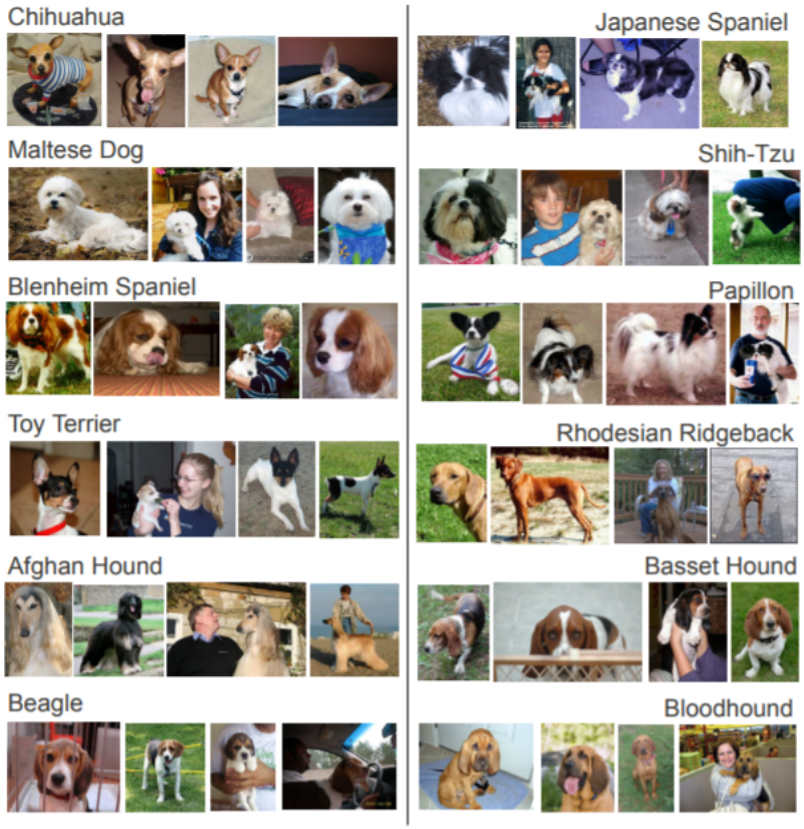
Figura 24 – Exemplificação de variância interclasse a partir das raças de cachorro Shih-Tzu e Lhasa

(a) Raça Shih-Tzu (b) Raça Lhasa

Fonte: própria.

Adicionalmente, a Figura 25 ilustra 12 classes dessa base em que pode-se identificar exemplos da variância intraclasse como a cor e composição dos pêlos, assim como o tamanho do animal e variações do ambiente, como o fundo da imagem e presença de pessoas. Sendo assim, para essa base, conforme descrito em (FEI-FEI et al., 2011), há uma grande variância intra e interclasse que dificulta a generalização do algoritmo, alcançando uma acurácia média de 22% na literatura. Como as classes com menor variação intraclasse obtiveram melhores resultados, estima-se que houve uma variação muito alta para a quantidade de amostras destas classes, uma vez que cada classe possui entre 150 e 250 imagens.

Figura 25 – Exemplos de amostras da base Stanford Dogs.



Fonte: (FEI-FEI et al., 2011).

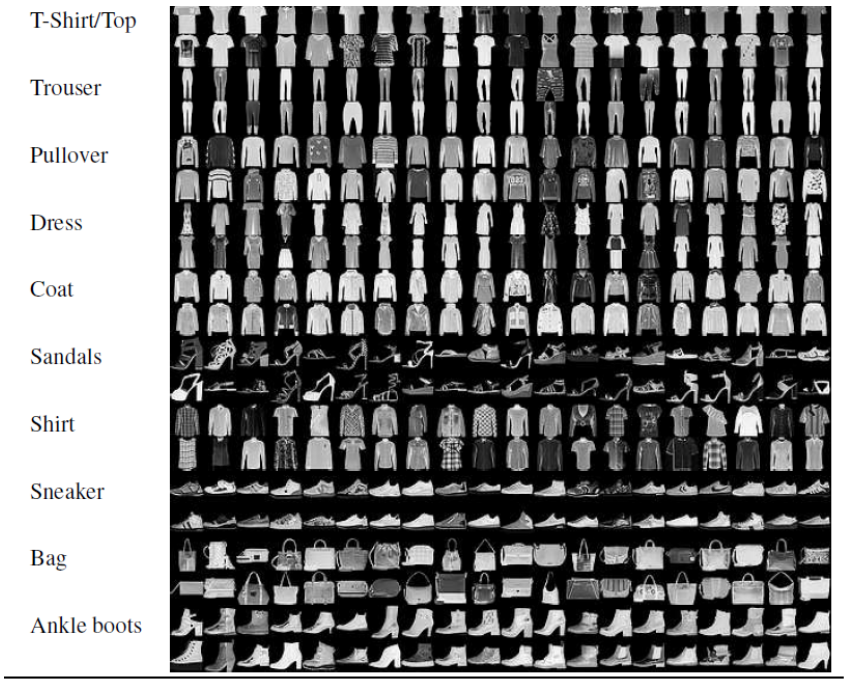
Para alcançar melhores resultados, também foram realizados experimentos utilizando a ResNet-20 e técnicas de *augmentation*, ilustrados na Tabela 16. Percebe-se que, da mesma forma que a base CIFAR100, os resultados com *augmentation* obtiveram maior medida *F1* do que aqueles com aumento de profundidade da rede.

Técnicas de augmentation aplicadas	**Medida *F1***
Sem Augmentation	0,12
Cutout	0,17
CoarseDropout ou cutout	0,21
CoarseDropout ou Cutout; RandomBrightness	0,17
CoarseDropout ou Cutout; RandomSizedCrop	0,25
CoarseDropout ou Cutout; RandomSizedCrop; Equalize; RandomBrithness; RandomContrast	0,29
CoarseDropout ou Cutout; CLAHE	0,23
CoarseDropout ou Cutout; Equalize; IAASharpen	0,26

Tabela 16 – Resultados de técnicas de data augmentation para a base CIFAR100 utilizando a rede ResNet-20.

Por fim, considerando que, para a base Fashion MNIST, as classes com maior silhueta são aquelas com menores resultados de medida *F1*, analisou-se os resultados de cada classe individualmente, sendo também aplicadas técnicas de *augmentation*. A Figura 26 ilustra as classes dessa base, assim como exemplos de amostras das mesmas.

Figura 26 – Exemplos de amostras da base Fashion MNIST.



Fonte: (XIAO; RASUL; VOLLGRAF, 2017).

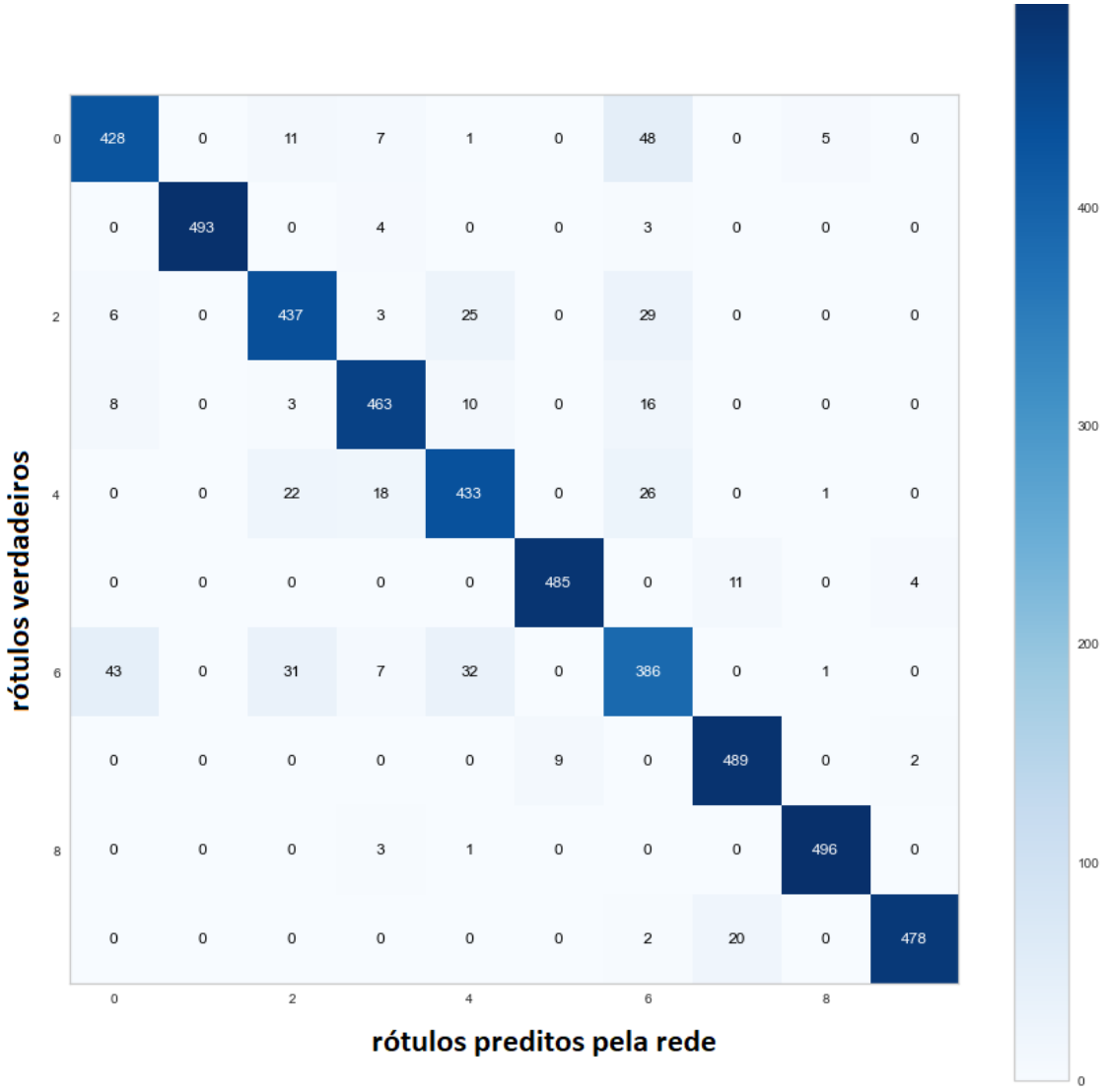
Percebe-se visualmente que as classes "*T-shirt/Top*", "*Pullouver*", "*Coat*"e "*Shirt*"são semelhantes, principalmete por possuírem variações como estampa e tamanho da manga, sendo todas correspondentes a diferentes tipos de camisas. Ao analisar a silhueta de cada classe (Tabela 17), percebe-se que essas quatro classes possuem os maiores valores desta métrica.

Índice	**Classes**	**Silhueta**
0	**T-Shirt**	**0,15**
1	Trouser	0,13
2	**Pullover**	**0,19**
3	Dress	0,14
4	**Coat**	**0,18**
5	Sandals	0,09
6	**Shirt**	**0,23**
7	Sneaker	0,12
8	Bag	0,18
9	Ankle boots	0,14

Tabela 17 – Valores de silhueta para cada classe da base de dados Fashion MNIST.

Sendo assim, dado os valores da silhueta, essas classes retornam os maiores erros para os modelos de rede. Esse fato pode ser confirmado por meio da Figura 27 que ilustra a matriz de confusão da ResNet-20 para esta base, sendo a maior quantidade de erros na mesma proveniente destas classes. Além disso, na matriz, as classes estão representadas por seus respectivos índices.

Figura 27 – Matriz de Confusão da rede ResNet-20 para base Fashion MNIST.



Fonte: própria.

Visando melhorar o desempenho dessas classes, comparou-se a medida *F1* de cada classe da ResNet-20 com o treinamento original (I) e quatro novos treinamentos com *data augmentation* expostos na Tabela 18. Para esses experimentos, aplicou-se as técnicas tradicionais (II) realizadas para as bases CIFAR100 e Stanford Dogs, além dos métodos RandomContrast (III), CLAHE (IV), e o método RandomContrast, juntamente com CoarseDropout ou Cutout (V), sendo a escolha dessas técnicas aleatória.

Classes	**Original**	**Tradicional**	**RandomContrast**	**Clahe**	**RandomContrast; CoarseDropout ou Cutout**
T-Shirt	0,82	0,88	0,88	0,88	0,89
Trouser	0,98	0,99	0,99	0,99	1
Pullover	0,84	0,91	0,90	0,90	0,91
Dress	0,9	0,94	0,94	0,93	0,95
Coat	0,84	0,90	0,91	0,89	0,90
Sandals	0,95	0,98	0,99	0,99	0,99
Shirt	0,74	0,81	0,81	0,80	0,82
Sneaker	0,94	0,97	0,97	0,97	0,97
Bag	0,97	0,99	0,99	0,98	0,98
Ankle boots	0,97	0,98	0,98	0,98	0,98
F1 médio	0,90	0,93	0,94	0,94	0,94

Tabela 18 – Medida *F1* das classes da base Fashion MNIST para treinamento original e com quatro tipos de augmentation utilizando a ResNet-20.

Da mesma forma que para as bases anteriormente analizadas, independemente da técnica utilizada, o *F1* médio é igual ou superior do que aqueles ilustrados na Seção 5.1, com arquiteturas de redes residuais de maior profundidade. Além disso, as classes com maior silhueta da base Fashion MNIST estão entre aquelas com maiores melhorias de desempenho ao utilizar *augmentation*.

Analisando-se também a realização de *augmentation* apenas nas quatro classes com menor medida *F1*, percebe-se também melhoria nos resultados por meio da Tabela 19. Para esses experimentos, foram utilizadas apenas as duas técnicas com melhores resultados da Tabela 18. Baseado nisso, a combinação destes e outros métodos em classes específicas podem indicar também possíveis melhorias.

Classe	**Original**	**RandomContrast; CoarseDropout ou Cutout**	**RandomContrast**
T-Shirt	**0,82**	**0,84**	**0,87**
Trouser	0,98	0,98	0,98
Pullover	**0,84**	**0,89**	**0,87**
Dress	0,90	0,90	0,89
Coat	**0,84**	**0,85**	**0,87**
Sandals	0,95	0,99	0,98
Shirt	**0,74**	**0,75**	**0,78**
Sneaker	0,94	0,96	0,97
Bag	0,97	0,98	0,98
Ankle boots	0,97	0,98	0,98
F1 médio	0,90	0,91	0,92

Tabela 19 – Medida *F1* das classes da base Fashion MNIST para treinamento original e aplicando *augmentation* apenas nas classes *T-Shirt*, *Pullouver*, *Coat* e *Shirt*.

Sendo assim, por meio dos resultados analisados neste capítulo, de forma geral, o aumento de profundidade em redes residuais não possui, estatisticamente, impacto considerável na performance do modelo apesar do alto custo computacional proporcional a este aumento. Além disso, os experimentos realizados indicam que as classes com maiores valores de silhueta intraclasse estão entre aquelas com maiores erros para essas redes. Ademais, o uso de técnicas de *augmentation* retornam melhores resultados do que o aumento de profundidade.

6 Considerações Finais

Esse trabalho teve como objetivo analizar o desempenho de redes residuais de diferentes profundidades, sendo realizados experimentos com as redes ResNet de 20, 32, 44, 56 e 110 camadas. Essas redes foram treinadas e testadas com 15 bases de dados de diferentes características, como canais de cores, quantidade de amostras e balanceamento.

Por meio da análise, percebeu-se que o aumento de profundidade resulta em alto custo computacional, sendo este também proporcional a quantidade de amostras na base de dados. Entretanto, este aumento não é proporcional a melhorias de resultados dos modelos de rede avaliados, resultando em medida *F1* sem variações significativas, sendo essa hipótese concluída a partir do teste de significância realizado com a distribuição t de Student.

Adicionalmente, foram realizados testes com diferentes técnicas de *data augmentation* para três dessas bases (Fashion MNIST, CIFAR100 e Stanford Dogs) utilizando a rede ResNet-20 como estudo de caso. Nesses experimentos, notou-se que, independetemente da técnica utilizada, todos os resultados foram superiores do que aqueles com redes residuais com maior profundidade, podendo assim concluir que a aplicação de tais técnicas retornam melhores resultados nessas redes do que o aumento de camadas.

Por fim, analisou-se a variância intraclasse das 15 bases de dados desse estudo, através da aplicação do algoritmo de agrupamento *k-means* nas classes de cada um dos conjuntos, sendo os grupos formados por este algoritmo avaliados de acordo com a métrica silhueta. Nesse estudo, percebeu-se que há uma possível relação entre classes com menor variedade intraclasse e as classes com menor medida *F1* das cinco redes avaliadas, indicando a existência de, no mínimo, uma classe com problemas de representatividade intraclasse.

Como trabalhos futuros, o menor desempenho de uma rede pode ser causado por problemas tanto intra quanto interclasse, sendo assim, podem ser realizados estudos para elaboração de métricas interclasse que, em conjunto com o método intraclasse realizado nesta dissertação, retorne maior parte das classes que representam erros para as redes. A partir dessa combinação, pode-se focar em melhorias na base de dados, como *data augmentation*, ao invés de modelos de redes convolucionais de maior profundidade, uma vez que estas melhorias são computacionalmente menos custosas do que treinamentos de redes profundas.

Adicionalmente, os experimentos expostos podem ser replicados em diferentes arquiteturas de redes convolucionais afim de identificar a existência de possíveis padrões com os resultados obtidos com as redes residuais.

Referências

AL, D. P. et. *Best Practice Guide - Deep Learning*. [S.l.]: Partnership for Advanced Computing in Europe, DOI:10.13140/RG.2.2.31564.05769, 2011. Citado na página 21.

ALESSIO, C. *Animals10*. 2019. Kaggle. Disponível em: <https://www.kaggle-.com/alessiocorrado99/animals10>. Acesso em: 29.06.2020. Citado na página 45.

ANIMESH, A. *Mobile Gallery*. 2018. Kaggle. Disponível em: <https://www.kaggle.com-/n0obcoder/mobile-gallery-image-classification-data>. Acesso em: 29.06.2020. Citado na página 45.

BARAN, I.; KUPYN, O.; KRAVCHENKO, A. Safe augmentation: Learning task-specific transformations from data. *arXiv preprint arXiv:1907.12896*, 2019. Citado 2 vezes nas páginas 36 e 59.

BARBARA, D. *An introduction to cluster analysis for data mining*. 2000. NVIDIA Developer. Disponível em: <https://www-users.cs.umn.edu/~hanxx023/dmclass-/cluster_survey_10_02_00.pdf>. Acesso em: 29.06.2020. Citado na página 32.

BENGIO, Y.; SIMARD, P.; FRASCONI, P. Learning long-term dependencies with gradient descent is difficult. In: *IEEE Transactions on Neural Networks. Volume 5, Issue: 2, pp. 157 - 166*. [S.l.: s.n.], 1994. Citado 2 vezes nas páginas 14 e 25.

BUSLAEV, A. et al. Albumentations: Fast and flexible image augmentations. *Information*, v. 11, n. 2, 2020. ISSN 2078-2489. Disponível em: <https://www.mdpi-.com/2078-2489/11/2/125>. Citado na página 52.

CARVALHO, A. *Inteligência Artificial - Uma Abordagem de Aprendizado de Máquina*. [S.l.]: LTC, 2011. Citado 3 vezes nas páginas 14, 32 e 33.

CHEN, S.; YANG, X.; TIAN, Y. Discriminative hierarchical k-means tree for large-scale image classification. *IEEE Transactions on Neural Networks and Learning Systems. Volume: 26. Issue: 9*, 2015. Citado na página 33.

COHEN, G. et al. Emnist: an extension of mnist to handwritten letters. *arXiv preprint arXiv:1702.05373*, 2017. Citado na página 45.

COMANICIU, D.; MEER, P. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2002. Citado na página 34.

DHANACHANDRA, N.; JINACHANU, Y.; MANGLEM, K. Image segmentation using k -means clustering algorithm and subtractive clustering algorithm. *Procedia Computer Science*, 2015. Citado na página 33.

ELGENDI, M. et al. The evaluation of deep neural networks and x-ray as a practical alternative for diagnosis and management of covid-19. *Preprint from medRxiv DOI: 10.1101/2020.05.12.20099481*, 2020. Citado 2 vezes nas páginas 38 e 40.

ESTIVILL-CASTRO, V. Why so many clustering algorithms: a position paper. *ACM SIGKDD Explorations Newsletter. Volume 4, Issue 1*, 2002. Citado na página 32.

FEI-FEI, L. et al. Novel dataset for fine-grained image categorization. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011. Citado 4 vezes nas páginas 45, 59, 60 e 61.

GLOROT, X.; BENGIO, Y. Understanding the difficulty of training deep feedforward neural networks. *In AISTATS*, 2010. Citado 2 vezes nas páginas 14 e 25.

GONZALEZ, R. C.; WOODS, R. E. *Processamento Digital De Imagens*. [S.l.]: MIT Press, 2016. `http://www.deeplearningbook.org`. Citado 2 vezes nas páginas 20 e 22.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. `http://www.deeplearningbook.org`. Citado 8 vezes nas páginas 14, 18, 19, 20, 21, 22, 23 e 24.

HE, K. et al. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *IEEE International Conference on Computer Vision (ICCV)*, 2015. Citado 3 vezes nas páginas 15, 29 e 30.

HE, K.; SUN, J. Convolutional neural networks at constrained time cost. *CVPR*, 2015. Citado 2 vezes nas páginas 15 e 25.

HE, K. et al. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015. Citado 8 vezes nas páginas 15, 25, 26, 28, 31, 38, 40 e 46.

HINTON, G. E. et al. Backpropagation and the brain. *Nature, 335–346*, 2020. Citado na página 18.

HINTON, G. E.; RUMELHART, D. E.; WILLIANS, R. J. Learning representations by back-propagating errors. *Nature, 323, 533–536*, 1986. Citado na página 18.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. *Neural computation. Edition 9 (8) pp. 1735–1780*, 1997. Citado 2 vezes nas páginas 18 e 26.

HUANG, G. et al. Deep networks with stochastic depth. *arXiv preprint arXiv:1603.09382*, 2016. Citado na página 59.

HUANG, G. et al. Densely connected convolutional networks. *arXiv preprint arXiv:1608.06993*, 2019. Citado na página 59.

IOFFE, S.; SZEGEDY, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *In ICML*, 2015. Citado 2 vezes nas páginas 25 e 29.

JAIN, A. K.; DUBES, R. C. Algorithms for clustering data. *Prentice Hall*, 1988. Citado na página 33.

J.ROUSSEEUW, P. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 1987. Citado 2 vezes nas páginas 34 e 51.

KAHRAMAN, M. T. N. Comparison of cnn tolerances to intra class variety in food recognition. *IEEE International Symposium on INnovations in Intelligent SysTems and Applications (INISTA)*, 2019. Citado na página 40.

KARPATHY, A. et al. Imagenet large scale visual recognition challenge. *arXiv preprint arXiv:1409.0575*, 2014. Citado 2 vezes nas páginas 15 e 36.

KHAN, R. U. et al. Evaluating the performance of resnet model based on image recognition. *Proceedings of the 2018 International Conference on Computing and Artificial Intelligence - ICCAI*, 2018. Citado 3 vezes nas páginas 38, 39 e 40.

KHAN, R. U. et al. A performance evaluation of convolutional neural networks for face anti spoofing. *9 International Joint Conference on Neural Networks*, 2019. Citado 2 vezes nas páginas 39 e 40.

KRIZHEVSK, A. Learning multiple layers of features from tiny images. *Tech Report*, 2009. Citado 4 vezes nas páginas 25, 44, 45 e 58.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. In: *Proceedings of the Neural Information Processing Systems Conference*. [S.l.: s.n.], 2012. Citado 3 vezes nas páginas 22, 24 e 29.

LECUN, Y. et al. Efficient backprop. in neural networks: Tricks of the trade. *pp. 9–50. Springer,*, 1998. Citado na página 25.

LECUN, Y. et al. Gradient-based learning applied to document recognition. *IEEE Procedings*, 1998. Citado 4 vezes nas páginas 18, 37, 44 e 45.

LEE, C.-Y. et al. Deeplysupervised nets. *arXiv preprint arXiv:1409.5185*, 2014. Citado na página 36.

LI, F.-F. *Setting up the data and the model*. 2020. Notas de aula. Stanford University CS231n: Convolutional Neural Networks for Visual Recognition. Disponível em: <https://cs231n.github.io/neural-networks-2/>. Acesso em: 25.06.2020. Citado na página 47.

NG, A. *What Data Scientists Should Know About Deep Learning*. 2015. Extract Data Conference. Disponível em: <https://www.slideshare.net/ExtractConf>. Acesso em: 27.06.2020. Citado na página 14.

NIELSEN, M. *Neural Networks and Deep Learning*. 2019. Kaggle. Disponível em: <http://neuralnetworksanddeeplearning.com>. Acesso em: 29.06.2020. Citado na página 19.

NORVIG, P.; RUSSELL, S. *Inteligência artificial*. [S.l.]: MIT Press, 2016. `http://www.deeplearningbook.org`. Citado na página 18.

NVIDIA, C. *NVIDIA AI Platform for Developers*. 2020. NVIDIA Developer. Disponível em: <https://developer.nvidia.com/deep-learning>. Acesso em: 27.06.2020. Citado 2 vezes nas páginas 15 e 20.

PENHA, D. D. P. *REDE NEURAL CONVOLUCIONAL APLICADA À IDENTIFICAÇÃO DE EQUIPAMENTOS RESIDENCIAIS PARA SISTEMAS DE MONITORAMENTO NÃO-INTRUSIVO DE CARGA*. [S.l.]: UNIVERSIDADE FEDERAL DO PARÁ, Programa de pós graduação em engenharia elétrica, 2018. Citado na página 24.

PILARCZYK, R.; SKARBEK, W. Incorporating intra-class variance to fine-grained visual recognition. *Foundations of Computing and Decision Sciences. Volume 44. Issue 3*, 2019. Citado na página 41.

PILARCZYK, R.; SKARBEK, W. On intra-class variance for deep learning of classifiers. *Foundations of Computing and Decision Sciences. Volume 44. Issue 3*, 2019. Citado na página 41.

ROSEBROCK, A. Deep learning for computer vision with python. 2017. Citado na página 14.

ROY, P. et al. Effects of degradations on deep neural network architectures. *arXiv preprint arXiv:1807.10108v4*, 2018. Citado na página 45.

SAXE, A. M.; MCCLELLAND, J. L.; GANGULI, S. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013. Citado na página 25.

SHORTEN, C.; KHOSHGOFTAAR, T. M. A survey on image data augmentation for deep learning. *Journal of Big Data volume 6, Article number 60*, 2019. Citado 2 vezes nas páginas 16 e 35.

SIMONYAN, K.; ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556v6*, 2014. Citado 4 vezes nas páginas 15, 24, 29 e 38.

SREENU, G.; DURAI, M. A. S. Intelligent video surveillance: a review through deep learning techniques for crowd analysis. *Journal of Big Data, article number: 48*, 2019. Citado na página 18.

SRIVASTAVA, R. K.; GREFF, K.; SCHMIDHUBER, J. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015. Citado na página 25.

STUDER, L. et al. A comprehensive study of imagenet pre-training for historical document image analysis. *arXiv preprint arXiv:1905.09113*, 2019. Citado na página 16.

SUN, C. et al. Revisiting unreasonable effectiveness of data in deep learning era. *ICCV*, 2017. Citado na página 14.

SZEGEDY, C. et al. Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*, 2014. Citado 4 vezes nas páginas 15, 25, 26 e 38.

UBERLANDIA, U. F. de. *Notas de aula sobre Análise de Agrupamentos*. 2018. Disponível em: <https://www.studocu.com/pt-br/document/universidade-federal-de-uberlandia-/reconhecimento-de-padroes/resumos/aula09-agrupamentos/8308492/view>. Acesso em: 20.06.2020. Citado na página 33.

WANG, H.; SHEIKH, H. R.; BOVIK, A. C. Objective video quality assessment. *The Handbook of Edeo Databases: Design and Applicarions*, 2003. Citado na página 35.

WANG, Z. et al. Image quality assessment: From error measurement to structural similarity. *IEEE Trans. IiaagePiocessing*, 2004. Citado na página 35.

WANG, Z. et al. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 2004. Citado na página 35.

WEI, W. et al. Intra-class similarity structure representation based hyperspectral imagery classification with few samples. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing PP(99)*, 2020. Citado na página 41.

XIAO, H.; RASUL, K.; VOLLGRAF, R. *Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms*. 2017. Citado 3 vezes nas páginas 37, 45 e 62.

ZEILER, M. D.; FERGUS, R. Visualizing and understanding convolutional neural networks. *In ECCV*, 2014. Citado 2 vezes nas páginas 22 e 23.