UNIVERSIDADE DE SÃO PAULO
ESCOLA DE ENGENHARIA DE SÃO CARLOS

Kaio Augusto de Oliveira

Avaliação de desempenho de redes neurais convolucionais para classificação de imagens de texturas rotacionadas.

São Carlos
2018

KAIO AUGUSTO DE OLIVEIRA

Avaliação de desempenho de redes neurais convolucionais para classificação de imagens de texturas rotacionadas.

Monografia apresentada ao Curso de Engenharia Elétrica, da Escola de Engenharia de São Carlos da Universidade de São Paulo, como parte dos requisitos para obtenção do título de Engenheiro Eletricista/Eletrônico.

Orientador: Prof. Dr. Adilson Gonzaga

São Carlos

2018

Ficha catalográfica elaborada pela Biblioteca Prof. Dr. Sérgio Rodrigues Fontes da EESC/USP com os dados inseridos pelo(a) autor(a).

Oliveira, Kaio Augusto de

O48a

Avaliação de desempenho de redes neurais convolucionais para classificação imagens de texturas rotacionadas. / Kaio Augusto de Oliveira; orientador Adilson Gonzaga. São Carlos, 2018.

Monografia (Graduação em Engenharia Elétrica com ênfase em Eletrônica) -- Escola de Engenharia de São Carlos da Universidade de São Paulo, 2018.

1. Redes Neurais Convolucionais. 2. Classificação de textura. 3. Invariância à rotação. 4. Aprendizado profundo. I. Título.

Eduardo Graziosi Silva - CRB - 8/8907

FOLHA DE APROVAÇÃO

Nome: Kaio Augusto de Oliveira

Título: "Avaliação de desempenho de redes neurais convolucionais para classificação de imagens de texturas rotacionadas por diferentes métodos"

Trabalho de Conclusão de Curso defendido e aprovado em 26/ 11 /2018,

com NOTA 10,0 (DEZ, ZERO), pela Comissão Julgadora:

Prof. Associado Adilson Gonzaga - Orientador - SEL/EESC/USP

Dra. Carolina Toledo Ferraz - Pós-Doutorado/Faculdade Campo Limpo Paulista

Mestre Adriane Cavichiolli - Centro Estadual de Educação Tecnológica Paula Souza/Faculdade de Tecnologia de Presidente Prudente

Coordenador da CoC-Engenharia Elétrica - EESC/USP:
Prof. Associado Rogério Andrade Flauzino

À minha família por todo cuidado, carinho, incentivo e apoio constantes.

AGRADECIMENTOS

Agradeço, especialmente, minha família por todo o carinho, amor, força e incentivo. Aos meus pais, Jose Paulo e Enelita Damiana, que lutaram muito para que eu pudesse concluir o curso de graduação e que, com esforço, sempre me mantiveram no caminho da educação. Às minhas irmãs que sempre estiveram dispostas a colaborar com a minha educação e por me oferecem suporte nos momentos difíceis.

Agradeço aos meus amigos, colegas e a todos que, de alguma forma, participaram diretamente e indiretamente da minha formação.

A todos os professores que compartilharam seus conhecimentos ao longo da minha vida acadêmica. E um agradecimento especial ao meu Orientador, Professor Doutor Adilson Gonzaga, pelo suporte e orientações oferecidos durante a elaboração, execução e escrita deste trabalho.

“Que todos os nossos esforços estejam sempre focados no desafio à impossibilidade. Todas as grandes conquistas humanas vieram daquilo que parecia impossível” (Charles Chaplin).

RESUMO

De Oliveira, K. A. **Avaliação de desempenho de redes neurais convolucionais para classificação de imagens de texturas rotacionadas.**, 2018. Monografia (Trabalho de Conclusão de Curso de Engenharia Elétrica) – Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2018.

Este trabalho propõe a avaliação de desempenho de redes neurais convolucionais para classificação de imagens de texturas rotacionadas por diferentes métodos. Foram utilizadas duas redes profundas: a AlexNet e a VGG-16, para a classificação de imagens de texturas rotacionadas do Álbum de Brodatz rotacionado. Essa base de dados contém imagens de textura naturais que foram rotacionadas fisicamente no momento da captura e rotacionadas computacionalmente. Cinco métodos de interpolação são utilizados: Lanczos, B-spline, Cúbica, Linear e *Nearest Neighbor.* Foram criados conjuntos de treino e teste para cada método de rotação a partir do recorte sem sobreposição das imagens contidas na base de dados. Foram realizados experimentos para avaliar o desempenho das redes quando são treinadas e testadas com imagens rotacionadas através do mesmo método, nos quais a rotação via *hardware* obteve 100% de acurácia para ambas as redes, resultado não igualado por nenhuma interpolação, demonstrando que a rotação computacional acarreta em perdas para a classificação de texturas rotacionadas utilizando CNNs. Em seguida foi realizado o teste com a utilização de amostras rotacionadas por outros métodos, para este caso as imagens rotacionadas fisicamente obtiveram os melhores resultados de acurácia para todas as redes treinadas com amostras rotacionadas computacionalmente. Já para as redes treinadas com amostras rotacionadas fisicamente, o teste com imagens rotacionadas computacionalmente acarreta em grande queda de acurácia causada por *overfitting.* A seguir, as redes foram treinadas com imagens rotacionadas por todos os métodos sem alterar o tamanho do conjunto de treinamento, os testes mostraram que não existe variação significativa na acurácia entre quando a rede é treinada e testada para cada método de rotação individualmente, ou quando a rede é treinada com todos os métodos e testada para cada método individualmente. Por fim as redes foram treinadas com um conjunto geral, formado por todos os conjuntos de treinamento de todos os métodos de rotação. As acurácias obtidas por estas redes aumentaram para todos os métodos de interpolação utilizados nos testes, demonstrando que *data augmentation* no conjunto

de treinamento através da adição de amostras rotacionadas por diversos métodos aumenta o desempenho das CNNs na tarefa de classificação de imagens de texturas rotacionadas.

Palavras-chave: Redes neurais convolucionais. Classificação de textura. Invariância à rotação. Aprendizado profundo.

ABSTRACT

De Oliveira, K. A. **Performance evaluation of neural networks in classifying rotated texture images.**, 2018. Monography (term paper for the course electrical engineering) – Engineering school of São Carlos, University of São Paulo, São Carlos, 2018.

This work come up with the performance evaluation of convolutional neural networks to classify texture images rotated by different methods. The two deep networks that were used to classify texture images rotated by different methods were Alexnet and VGG-16. This database contains natural texture images that were physically rotated at the time of capture and rotated computationally. Five interpolation methods are used: Lanczos, B-spline, Cubic, Linear and Nearest Neighbor. Trainig and testing sets were criated for each rotation method from the non-overlapping images clipping contained on the database. Experiments were made to evaluate the networks performance when trained and tested with rotated images using the same method, as a results the hardware got 100% of accuracy using both networks, result which was not achieved by any interpolation method, showing that computational rotation results in losses to the classification of rotationed textures using CNN. Then, a test using rotated samples by other methods was performed, in which the best accuracy results were achieved from physically rotationed images by all the networks trained computationally. However, the test with computationally rotated images leads to great loss of accuracy because of overfitting when the networks are trained with physically rotated images. After that, the networks were trained with rotated images by all the methods without changing the size of the training sets, the tests showed that there is no substantial variation of accuracy if the network is trained and tested individually for each method of rotation, or if the network is trained by all the methods and, then, tested individually for each method. In the end, the networks were trained with a general set compound by all the other training sets from all the rotation methods, the accuracy achieved by the networks used increases for all interpolation methods used during the tests. This, demonstrate that data augmentation in the training set through addition of rotated samples by diverse methods increases the performance of CNNs in classifying rotated texture images.

Keywords: Convolutional neural networks. Texture classification. Invariance to rotation. Deep learning

LISTA DE ILUSTRAÇÕES

Figura 1: Processo de geração dos mapas de características.....32
Figura 2: processo da aplicação da função MAX pooling.....34
Figura 3: 32 imagens escolhidas do Álbum de Brodatz 48
Figura 4 :Exemplo de digitalização de textura em 21 ângulos de rotação. 49
Figura 5: Estrutura da rede Alexnet.....50
Figura 6: Estrutura da rede VGG-16 52
Figura 7: Comparação entre acurácias obtidas pela AlexNet e VGG-16, para as redes treinadas e testadas com imagens rotacionadas através do mesmo método .62
Figura 8: Comparação entre os resultados obtidos pelas CNNs AlexNet e VGG-16 treinadas por cada método de rotação e testadas com o conjunto contendo todos os seis métodos. 68
Figura 9: Comparação entre acurácias obtidas pelas duas redes treinadas com imagens rotacionadas através dos 6 métodos (16,66%). E teste realizado com imagens rotacionadas por cada método individualmente, e com um conjunto geral composto por amostras rotacionadas pelos 6 métodos. 70

LISTA DE TABELAS

Tabela 1: Arquitetura da rede AlexNet 50
Tabela 2: Arquitetura da rede VGG-16.....52
Tabela 3: Acurácia obtida pela rede AlexNet treinada e testada com imagens rotacionadas através do mesmo método 61
Tabela 4: Acurácia obtida pela rede VGG-16 treinada e testada com imagens rotacionadas através do mesmo método 62
Tabela 5: Acurácia média obtida pelo descritor CLMP_S/M(24,3) (VIEIRA,2017) testado com 50 amostras de tamanho 130 x 130 pixels da base de dados Brodatz Texture Rotation Database.....63
Tabela 6: Acurácia obtida pela rede AlexNet treinada com imagens rotacionadas via hardware e testada com imagens rotacionadas através de cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos.....64
Tabela 7: Acurácia obtida pela rede AlexNet treinada com imagens rotacionadas por interpolação B-spline e testada com imagens rotacionadas através de cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos. 64
Tabela 8: Acurácia obtida pela rede AlexNet treinada com imagens rotacionadas por interpolação Cúbica e testada com imagens rotacionadas através de cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos. 64
Tabela 9: Acurácia obtida pela rede AlexNet treinada com imagens rotacionadas por interpolação Lanczos e testada com imagens rotacionadas através de cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos. 64
Tabela 10: Acurácia obtida pela rede AlexNet treinada com imagens rotacionadas por interpolação Linear e testada com imagens rotacionadas através de cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos. 65
Tabela 11: Acurácia obtida pela rede AlexNet treinada com imagens rotacionadas por interpolação Nearest Neighbor e testada com imagens rotacionadas através de

cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos 65
Tabela 12: Acurácia obtida pela rede VGG-16 treinada com imagens rotacionadas via hardware e testada com imagens rotacionadas através de cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos. 66
Tabela 13:Acurácia obtida pela rede VGG-16 treinada com imagens rotacionadas por interpolação B-spline e testada com imagens rotacionadas através de cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos..... 66
Tabela 14: Acurácia obtida pela rede VGG-16 treinada com imagens rotacionadas por interpolação Cúbica e testada com imagens rotacionadas através de cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos..... 66
Tabela 15: Acurácia obtida pela rede VGG-16 treinada com imagens rotacionadas por interpolação Lanczos e testada com imagens rotacionadas através de cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos..... 66
Tabela 16: Acurácia obtida pela rede VGG-16 treinada com imagens rotacionadas por interpolação Linear e testada com imagens rotacionadas através de cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos..... 67
Tabela 17: Acurácia obtida pela rede VGG-16 treinada com imagens rotacionadas por interpolação Nearest Neighbor e testada com imagens rotacionadas através de cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos 67
Tabela 18: Acurácia obtida pela rede AlexNet treinada com imagens rotacionadas através dos 6 métodos (16,67%). E teste realizado com imagens rotacionadas por cada método individualmente, e com um conjunto geral composto por amostras rotacionadas pelos 6 métodos. 69
Tabela 19: Acurácia obtida pela rede VGG-16 treinada com imagens rotacionadas através dos 6 métodos (16,67%). E teste realizado com imagens rotacionadas por cada método individualmente, e com um conjunto geral composto por amostras rotacionadas pelos 6 métodos. 69

Tabela 20: Acurácia obtida pela rede AlexNet treinada com imagens rotacionadas através dos 6 métodos (100%). E teste realizado com imagens rotacionadas por cada método individualmente, e com um conjunto geral composto por amostras rotacionadas pelos 6 métodos.....70
Tabela 21: Acurácia obtida pela rede VGG-16 treinada com imagens rotacionadas através dos 6 métodos (100%). E teste realizado com imagens rotacionadas por cada método individualmente, e com um conjunto geral composto por amostras rotacionadas pelos 6 métodos.....71

LISTA DE ABREVIATURAS E SIGLAS

BS	*B-spline*
CB	Cúbica de Terceira ordem
CLBP	*Completed Local Binary Pattern*
CLMP	*Completed Local Mapped Pattern*
CNN	*Convolutional Neural Network*
HW	*Hardware*
ILTPDFT	*Improved Local Ternary Pattern Discrete Fourier Transform*
LBP	*Local Binary Pattern*
LN	Linear
LTPDFT	*Local Ternary Pattern Discrete Fourier Transform*
LZ	Lanczos
NN	*Nearest Neighbor*
ReLU	*Rectified Linear Units*
SLMP_M	*Sampled Local Mapped Pattern Magnitude*

SUMÁRIO

1 INTRODUÇÃO 27

1.1 REDES NEURAIS CONVOLUCIONAIS 27

1.2 MOTIVAÇÃO 28

1.3 OBJETIVOS 29

1.4 ORGANIZAÇÃO DO TRABALHO 30

2 REVISÃO BIBLIOGRÁFICA 31

2.1 ARQUITETURA DAS REDES NEURAIS CONVOLUCIONAIS 31

2.1.1 Camada de entrada 31

2.1.2 Camada de Convolução 31

2.1.3 Camada de ativação 33

2.1.4 Camada de *pooling* 34

2.1.5 Camada totalmente conectada 35

2.1.6 Camada de classificação 36

2.2 TREINAMENTO 37

2.3 MÉTODOS DE ROTAÇÃO DE IMAGENS 38

2.3.1 Rotação por *software* 38

2.3.1.1 *Nearest Neighbor* 38

2.3.1.2 Linear 38

2.3.1.3 Cúbica de terceira ordem 39

2.3.1.4 Cúbica *B-spline* 39

2.3.1.5 Lanczos 39

2.3.2 Rotação por *hardware* 39

2.4 TRABALHOS RELACIONADOS 40

2.4.1 Melhorando a classificação de textura baseada em CNN por balanceamento de Cores 40

2.4.2 Reconhecimento de Espécies Florestais usando Redes Neurais Convolucionais Profundas 41

2.4.3 Utilização de bancos de filtros em redes neurais convolucionais para classificação de texturas 43

2.4.4 Avaliação de Redes Neurais Convolucionais para Reconhecimento de Textura Rotacionada 45

3 MATERIAIS E MÉTODOS 47

3.1 FERRAMENTAS UTILIZADAS 47

3.2 SELEÇÃO DA BASE DE DADOS 47

3.2.1 Álbum de Brodatz Rotacionado 47

3.3 ARQUITETURA DAS CNNs UTILIZADAS 50

3.3.1 Alexnet 50

3.3.2 VGG-16 52

3.3.3 Alterações nas arquiteturas 54

3.4 CONJUNTOS GERADOS PARA TREINO E TESTE 55

3.5 PROCESSO DE TREINAMENTO 56

3.6 EXPERIMENTOS 56

3.6.1 Considerações iniciais 56

3.6.2 Experimento 1 57

3.6.3 Experimento 2 57

3.6.4 Experimento 3 58

3.6.5 Experimento 4 59

4 RESULTADOS 61

4.1 RESULTADO DO EXPERIMENTO 1 61

4.2 RESULTADO DO EXPERIMENTO 2 64

4.3 RESULTADO DO EXPERIMENTO 3 68

4.4 RESULTADO DO EXPERIMENTO 4 70

5 CONCLUSÃO 73

REFERÊNCIAS BIBLIOGRÁFICAS 75

APÊNDICE A - ROTINA BASE PARA TREINAR E TESTAR CNN 82

1 INTRODUÇÃO

1.1 REDES NEURAIS CONVOLUCIONAIS

A visão computacional é um tema amplamente discutido, estudado e difundido atualmente, além de ser utilizada em diversas áreas de aplicação de inteligência artificial, com resultados importantes na identificação online de imagens, segurança de aparelhos eletrônicos, reconhecimento facial, diagnósticos de doenças, contagens em laboratórios, reconhecimento de objetos, pedestres, entre outros. Neste sentido, as Redes Neurais Convolucionais, do inglês *Convolutional Neural Network* (CNN) entraram em cena acelerando o processo de reconhecimento de padrões nas imagens.

As CNNs foram primeiramente propostas por Lecun et al. (1998), quando os autores desenvolveram a LeNet, para o reconhecimento de dígitos manuscritos. Esta arquitetura estabeleceu um novo estado da arte ao atingir 99,20% de acurácia na base de dados MNIST (LECUN; CORTES; BURGES, 1998). Porém apenas no ano de 2012 as CNNs ganharam perspectiva na área de visão computacional, quando Krizhevsky, Sutskever e Hilton (2012) estabeleceram um marco na área ao propor a AlexNet, arquitetura vencedora do ILSVRC (*ImageNet Large Scale Visual Recognition Challenge*), competição que avalia performance de algoritmos para reconhecimento de imagens.

Nos anos seguintes as CNNs obtiveram resultados expressivos no ILSVRC, em 2014 o primeiro e segundo lugar foram ocupados por GoogLeNet (SZEGEDY et al., 2015) e VGG Net (SIMONYAN; ZISSERMAN, 2014) respectivamente, e em 2015 a vencedora foi a ResNet (HE et al., 2016). Desde então, a aplicação mais popular relacionada a CNNs é reconhecer padrões em imagens.

O funcionamento das redes neurais convolucionais foi baseado nas redes de Perceptons de múltiplas camadas. Por terem sido inspiradas nas redes neurais tradicionais, as CNNs, acabam por ser, estruturalmente, semelhantes, em alguns aspectos, as redes tradicionais: ambas possuem neurônios que contêm pesos e *bias* e devem ser treinadas para funcionar do modo esperado.

Sua arquitetura foi inspirada em processos biológicos, mais especificamente, na organização do córtex visual dos animais. Neurônios corticais individuais respondem a estímulos apenas em regiões restritas do campo de visão, conhecidas

como campos receptivos. Os campos receptivos de diferentes neurônios se sobrepõem parcialmente de forma a cobrir todo o campo de visão (HUBEL; WIESEL, 1968). Estas redes possuem diversas camadas de filtros que atuam como os campos receptivos dos neurônios, dividindo imagens em mapas de características, estes mapas são, então, utilizados para reconhecer e classificar outras imagens.

1.2 MOTIVAÇÃO

Classificação de textura consiste em uma tarefa de visão computacional e processamento de imagens, que pode ser aplicada em diversas áreas como: diagnóstico médico (DAS et al., 2015), classificação de objetos (FERRAZ; GONZAGA, 2016), reconhecimento de face (LATIF et al.,2016), classificação de espécies florestais (FILHO et al.,2012).

Uma imagem de textura pode ser definida como uma distribuição de intensidade de cor não uniforme e não espacialmente variável (HADID et al., 2011). Portanto, a tarefa de classificação de textura apresenta algumas semelhanças com a classificação de objeto, como a forte correlação de intensidades de pixel no espaço 2D, sendo possível que algumas técnicas de reconhecimento de imagens sejam aplicáveis em ambos. Porém, existem características únicas das texturas, como realizar a classificação utilizando apenas um pequeno fragmento da imagem (CAVALIN; OLIVEIRA, 2017).

Um problema recorrente na classificação de texturas se deve ao fato de que texturas podem ser encontradas em resoluções espaciais e rotações arbitrarias, além de diversas condições de iluminação (VIEIRA, 2017).

Entre os primeiros estudos realizados para a classificação de textura invariante à rotação está o método proposto por Kashyap e Khotanzad (1986) que utiliza um modelo auto regressivo circular.

Com o passar dos anos foram estudados modelos com formulações mais simples, principalmente descritores baseados na codificação de micropadrões presentes nas texturas. O descritor *Local Binary Pattern* (LBP), proposto por Ojala, Pietikäinen e Mäenpää (2002), foi amplamente estudado, gerando diversas variações de seu algoritmo original. O LBP trabalha com a relação de cada pixel e seus vizinhos por um padrão binário, e para a classificação utiliza o histograma dos padrões de textura. Uma variação do LBP que ganhou destaque devido ao seu desempenho e

simplicidade de implementação foi o Co*mpleted Local Binary Pattern* (CLBP) (GUO, ZHANG; ZHANG, 2010). Este descritor trabalha com a utilização de vizinhanças circulares.

Um desafio na tarefa de classificação de texturas rotacionadas é o método utilizado para a rotação. No mundo digital de hoje, grande parte das imagens a serem classificadas têm suas propriedades como: resolução espacial, rotação e iluminação, alteradas computacionalmente.

Vieira e Gonzaga (2016) realizaram um estudo do efeito dos métodos de interpolação sobre o reconhecimento de texturas rotacionadas por software. Para isto, foram testadas duas variações do LBP: os descritores LTP^{DFT} e $ILTP^{DFT}$ propostos por Kylberg e Sintorn (2016), além do descritor CLBP. Os autores mostraram que o método de interpolação influencia na capacidade de reconhecimento desses descritores.

São poucos os estudos realizados para classificação robusta aos métodos de interpolação de texturas rotacionadas. Vieira (2017) obteve ótimos resultados nessa tarefa ao propor a utilização dos descritores *Completed Local Mapped Pattern* (CLMP) e *Sampled Local Mapped Pattern* Magnitude (SLMP_M), cujas metodologias utilizam a informação de magnitude da diferença entre os níveis de cinza dos pixels da vizinhança e o pixel central do micropadrão analisado.

Devido aos poucos estudos realizados para a classificação robusta aos métodos de interpolação de texturas rotacionadas, é motivação deste trabalho estudar a aplicação, nesta tarefa, de CNNs, que são apontadas como estado da arte em diversas aplicações.

1.3 OBJETIVOS

O objetivo deste trabalho é avaliar o desempenho de CNNs na tarefa de classificação robusta aos métodos de rotação de texturas rotacionas. Os métodos avaliados são rotação via *hardware* e rotação via *software* pelos métodos de interpolação: Cúbica de terceira ordem, Cúbica *B-spline*, Lanczos, Linear e *Nearest Neighbor*. *Estudar* o comportamento de CNNs quando são treinadas e testadas com imagens de textura rotacionadas através de apenas um método. Avaliar a performance da rede quando são utilizadas amostras rotacionadas por outros

métodos no teste. E também, avaliar o desempenho das CNNs quando são utilizadas imagens rotacionadas através de diversos métodos no treinamento.

1.4 ORGANIZAÇÃO DO TRABALHO

Este trabalho está dividido em 5 capítulos:

O primeiro trata da introdução ao assunto abordado neste trabalho, neste capítulo são apresentados: a história das CNNs, a problematização e histórico sobre a classificação de texturas rotacionadas, além do objetivo e organização do trabalho.

O segundo apresenta a revisão bibliográfica necessária para o entendimento deste trabalho.

O terceiro contém o método proposto para a avaliação do desempenho de CNNs na tarefa de classificação robusta aos métodos de rotação de texturas rotacionadas.

No quarto são apresentados e discutidos os resultados obtidos neste trabalho.

E por fim, o quinto capítulo apresenta a conclusão do trabalho.

2 REVISÃO BIBLIOGRÁFICA

2.1 ARQUITETURA DAS REDES NEURAIS CONVOLUCIONAIS

A arquitetura de redes neurais convolucionais é formada por camadas em sequência, cada uma com uma função distinta. Sua arquitetura básica é composta por: camada de convolução, camada de *pooling* e camada totalmente conectada. Apenas as camadas de convolução e camadas totalmente conectadas possuem pesos.

É comum uma CNN apresentar após as camadas de convolução, uma camada de ativação (LI; JOHNSON; YEUNG, 2017). Ainda é possível a adição de outros tipos de camadas para melhorar o desempenho da rede. Ao empilhar essas diversas camadas, é formada uma arquitetura completa de rede neural convolucional. Abaixo serão explicadas diversas camadas utilizadas para a formação de uma arquitetura completa de uma CNN.

2.1.1 Camada de entrada

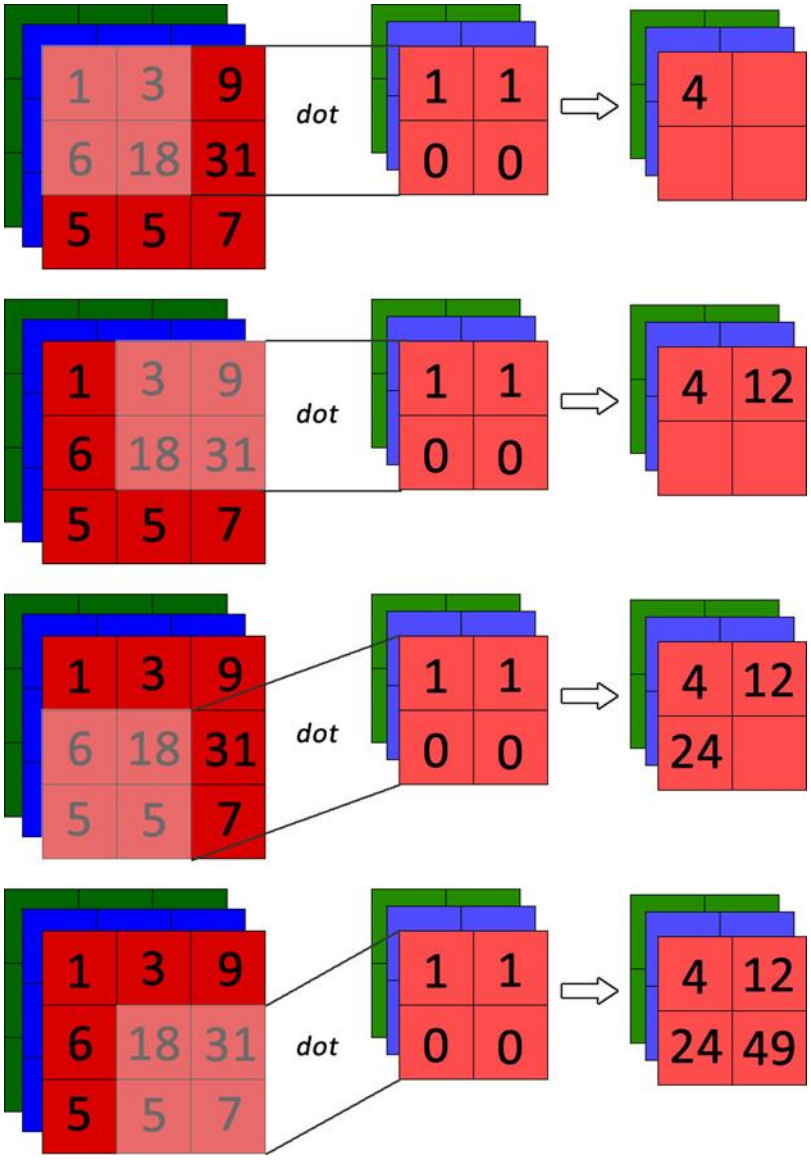
A primeira camada de uma CNN é a entrada, esta camada recebe uma imagem que é interpretada como uma matriz de intensidades com três dimensões: altura, largura e profundidade. O valor que a profundidade assume corresponde ao número de canais da imagem de entrada, ou seja, para imagens de tons de cinza esse valor é igual a 1, já para imagens coloridas a profundidade vale 3, número que representa os canais R, G e B (ZHENG et al., 2016).

2.1.2 Camada de Convolução

Nessa camada são aplicados diferentes filtros em cada porção da camada anterior, estabelecendo pesos para cada uma dessas conexões. Cada filtro percorre várias regiões da imagem até que seja completamente visitada, formando assim um mapa de características. Nesse processo há um compartilhamento de pesos durante o treinamento, o que resulta na redução do custo computacional, já que restringe o número de parâmetros que devem ser aprendidos.

Cada filtro é responsável pela formação de um mapa de características, a junção dos mapas gerados pelos diferentes filtros será a saída da camada de convolução. Os mapas de características gerados para camadas de convolução superiores contém informações de padrões mais simples, como bordas horizontais e verticais, enquanto as camadas mais profundas contêm informações de padrões mais complexos. Dessa forma essa camada é a principal responsável por tornar as CNNs invariantes no espaço. A Figura 1 exemplifica o processo da geração dos mapas de características.

Figura 1: Processo de geração dos mapas de características



Fonte: (PACHECO, 2017)

Variáveis importantes em relação aos filtros são: o tamanho e o passo (*stride*) no qual o filtro se movimenta pela imagem. O tamanho do filtro define a região que cada neurônio processará, e o passo define quantos pixels são transpostos. Nas camadas de convolução das CNN são hiperparâmetros (parâmetros definidos pelo programador), apenas os parâmetros da arquitetura do filtro, ou seja, o tamanho do filtro, o passo, a quantidade de filtros a serem aplicadas na camada e o *padding* (um preenchimento das bordas da imagem a qual será aplicada os filtros, geralmente feito com zeros, para que se possa percorrer mais da imagem, muitas vezes com a intenção de que a saída gerada após a convolução tenha o mesmo tamanho da entrada).

Para uma imagem de entrada com dimensões definidas como W_1 x H_1 x D_1, o mapa gerado, no final da camada de convolução, forma uma matriz, cujas dimensões podem ser representadas por: W_2 x H_2 x D_2. Ao considerar o tamanho do filtro como “F”, passo “S”, quantidade de filtros “K” e *padding* “P”, pode-se utilizar as seguintes equações 1, 2 e 3 (LI; JOHNSON; YEUNG, 2017) para alcançar as dimensões da matriz de saída gerada por essa camada:

$$W_2 = \frac{W_1 - F + 2P}{S} + 1 \qquad (1)$$

$$H_2 = \frac{H_1 - F + 2P}{S} + 1 \qquad (2)$$

$$D_2 = K \qquad (3)$$

2.1.3 Camada de ativação

Uma função de ativação é, frequentemente, aplicada logo após a camada de convolução, essa função é responsável por decidir se o “neurônio” foi acionado ou não.

O único hiperparâmetro nessa camada é a função a ser utilizada. Existem diversas funções que podem ser empregadas para a ativação, normalmente, funções de ativação não-lineares como: Sigmoid e Tanh. Porém a mais comum em redes neurais convolucionais é a função *Rectified Linear Units* (ReLU).

A função ReLU é definida pela equação 4, na qual os valores positivos assumem uma função linear e os valores negativos assumem o valor 0:

$$f(x) = max(0, x) \qquad (4)$$

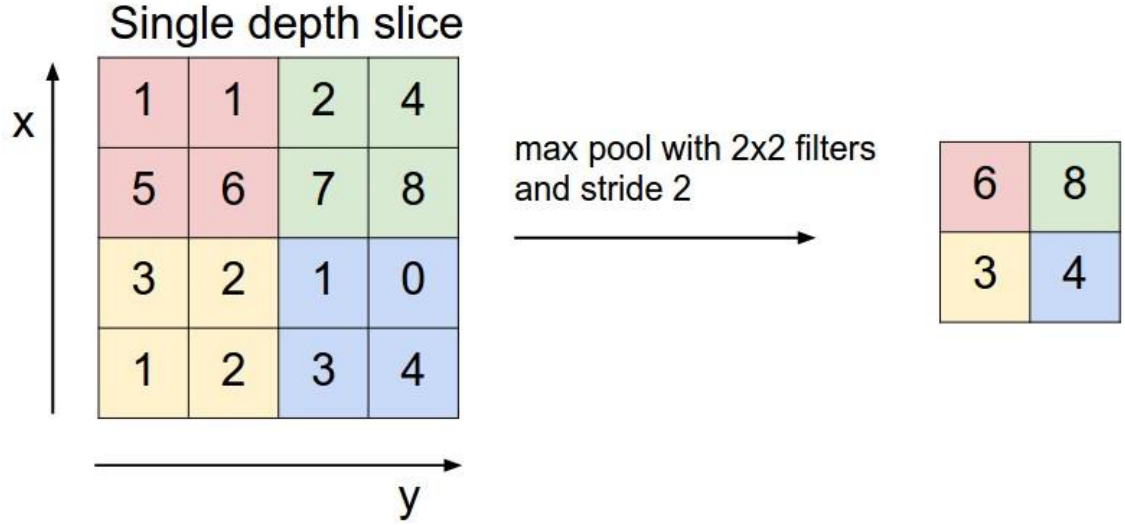
2.1.4 Camada de *pooling*

Também conhecida como camada de agrupamento, as camadas *pooling* são utilizadas para reduzir o tamanho das matrizes (operação conhecida como *downsampling*), reduzindo as medidas de largura e altura, porém, a profundidade não é alterada, diminuindo a quantidade de dados na rede, dessa forma reduzindo o custo computacional da CNN.

Esta camada ainda funciona como uma espécie de regularização, ou seja, combate o *overfitting* (quando a rede tem ótimos resultados para o conjunto de treinamento, mas não obtém grande desempenho quando testada com novas amostras).

Apesar de existirem muitas funções para serem utilizadas nesta camada, a que vem apresentando melhores resultados é a *MAX pooling* (SCHERER; MÜLLER; BEHNKE, 2010*)*. A função *MAX pooling* é aplicada utilizando-se filtros que captam apenas o valor máximo de cada vizinhança, preservando dessa forma os maiores valores obtidos após as camadas de convolução e ativação, que representam as principais características extraídas. A Figura 2 exemplifica o processo da aplicação da função *MAX pooling*.

Figura 2: processo da aplicação da função *MAX pooling*.



Fonte: (LI; JOHNSON; YEUNG, 2017)

Nas camadas de *pooling* das CNNs são hiperparâmetros, apenas os parâmetros da arquitetura do filtro, ou seja, o tamanho do filtro, o passo e o *padding* a ser utilizado.

Com uma matriz de entrada com dimensões definidas como W_1 x H_1 x D_1, a saída do filtro forma uma matriz cujas dimensões podem ser representadas por: W_2 x H_2 x D_2. Ao considerar o tamanho do filtro como “F”, passo “S” e *padding* “P”, pode-se utilizar as equações 5, 6 e 7 (LI; JOHNSON; YEUNG, 2017) para alcançar as dimensões da matriz de saída gerada por essa camada:

$$W_2 = \frac{W_1 - F + 2P}{S} + 1 \quad (5)$$

$$H_2 = \frac{H_1 - F + 2P}{S} + 1 \quad (6)$$

$$D_2 = D_1 \quad (7)$$

2.1.5 Camada totalmente conectada

A camada totalmente conectada é a última camada com pesos em uma arquitetura completa de uma CNN. Recebe esse nome em razão dessa camada funcionar da mesma forma que uma rede neural tradicional, cada neurônio de uma camada é conectado a todos os neurônios da camada anterior.

A quantidade de neurônios de sua entrada equivale ao número de elementos da camada anterior, recebendo como entrada todos os pixels existentes na camada anterior dispostos como um vetor. A quantidade de neurônios em sua saída corresponde ao número de classes que se deseja classificar.

A camada totalmente conectada recebe os mapas de características formados pela camada anterior e gera *scores* para cada neurônio em sua saída. Durante o processo de treinamento, os pesos da camada totalmente conectada são atualizados, de forma que para cada neurônio de saída, os *scores* gerados possuam um valor alto quando a imagem avaliada pertence à mesma classe que o neurônio representa e, valores baixos quando a imagem avaliada não pertence à mesma classe que o neurônio representa.

Para elucidar esse processo, é tomado como exemplo uma rede que possui 2 neurônios de saída, portanto 2 classes denominadas: A e B. Durante o processo de treinamento os pesos serão atualizados de forma que, quando a imagem avaliada pertença à classe A, o *score* gerado para o neurônio que representa a classe A terá um valor alto, enquanto o *score* gerado para o neurônio que representa a classe B terá um valor baixo.

Normalmente, nessa camada é adicionado *dropout* como forma de regularização. Em cada iteração do processo de treinamento da rede, neurônios individuais são desligados (apenas durante uma iteração) com uma probabilidade "p" de forma a impedir a coadaptação excessiva dos neurônios (SRIVASTAVA et al., 2014).

2.1.6 Camada de classificação

Essa camada é responsável pela classificação das imagens de entrada entre as diversas classes definidas através da escolha do número de neurônios na saída da camada totalmente conectada. A forma de classificação mais utilizada é o Classificador *Softmax* que utiliza a função *softmax*. Essa função recebe um vetor de valores arbitrários reais e o transforma em um vetor de valores entre 0 e 1, cuja soma vale 1. A equação 8 (LI; JOHNSON; YEUNG, 2017) apresenta a função *softmax*, na qual z_j simboliza o elemento de número j do vetor de valores a serem normalizados:

$$f_j(z) = \frac{e^{z_j}}{\sum_k e^{z_k}} \qquad (8)$$

A classificação é feita através da utilização da função denominada *cross entropy loss* (entropia cruzada). Essa função quantifica a diferença entre duas distribuições de probabilidade uma "verdadeira" e outra estimada.

A equação 9 (LI; JOHNSON; YEUNG, 2017) apresenta a entropia cruzada, na qual p é a distribuição "verdadeira", e q a distribuição estimada. O valor obtido pela função aumenta à medida que a probabilidade estimada diverge da "verdadeira":

$$H(p,q) = -\sum_x p(x)\, logq(x) \qquad (9)$$

Para a classificação na CNN, os valores da distribuição “verdadeira” são os valores de probabilidade dos dados utilizados no treinamento, assumindo o valor 1 para a classe correta e 0 para todas as outras. Enquanto a distribuição estimada equivale aos valores obtidos através da utilização da função *softmax* nos *scores* gerados na saída da camada totalmente conectada. O Classificador Softmax é então dado pela equação 10 (LI; JOHNSON; YEUNG, 2017), na qual f_{yi} simboliza o yi-ésimo elemento do vetor de *scores*:

$$L_i = -\log\left(\frac{e^{f_{yi}}}{\sum_j e^{f_j}}\right) \qquad (10)$$

A classificação é feita através da menor entropia cruzada entre a distribuição “verdadeira” e a distribuição estimada, de forma que cada elemento pertença a apenas uma classe.

2.2 TREINAMENTO

O processo de treinamento de uma rede neural convolucional possui duas fases: a primeira denominada *forwardpropagation* e a segunda *backpropagation*.

Durante a primeira fase (*forwardpropagation)* os exemplos de treinamento passam por todas as camadas descritas, anteriormente, neste capítulo, gerando um vetor de *scores* na saída da camada totalmente conectada, esse vetor é utilizado através da equação 10 para o cálculo de um erro, este erro é calculado através da média de Li sobre todos os exemplos de treinamento juntos.

Na segunda fase (*backpropagation*) é utilizado o algoritmo do gradiente descendente, em que cada peso é atualizado por uma redução escalar negativa da derivada do erro em relação a este peso, de forma a minimizar o erro. Este processo ocorre a partir do sentido final da CNN, em direção ao início da rede.

O principal hiperparâmetro a ser definido durante o treinamento é a taxa de aprendizagem, que indica o quão bruscamente os pesos serão alterados em cada atualização. Quanto maior for este fator, maior será a mudança nos pesos, aumentando a velocidade do aprendizado, o que pode levar a uma oscilação do modelo na superfície de erro.

Outros hiperparâmetos são *batch size* e número de épocas. O primeiro representa o número de subamostras entregues à rede após a atualização dos parâmetros, o segundo trata-se do número de vezes que todos os dados de treinamento são mostrados à rede durante o treinamento.

2.3 MÉTODOS DE ROTAÇÃO DE IMAGENS

2.3.1 Rotação por *software*

A rotação por *software* é feita através da utilização de algoritmos de interpolação, método que funciona usando dados conhecidos para estimar valores em pontos desconhecidos. Neste trabalho serão apresentados cinco métodos de interpolação*: Nearest Neighbor*, Linear, Cúbica de terceira ordem, cúbica *B-spline* e Lanczos.

2.3.1.1 *Nearest Neighbor*

Nearest Neighbor é o método mais básico e requer o menor tempo de processamento de todos os algoritmos de interpolação, o valor estimado é sempre igual ao pixel mais próximo, conforme a equação 11 (VIEIRA; GONZAGA, 2016):

$$NN(x) = \begin{cases} 1, & |x| \in [0\ 0,5] \\ 0, & caso\ contrário \end{cases} \quad (11)$$

2.3.1.2 Linear

Neste algoritmo, o valor é estimado com a utilização de uma função linear entre as intensidades de pixels, conforme a equação 12 (VIEIRA; GONZAGA, 2016):

$$LN(x) = \begin{cases} 1 - |x|, & |x| \in [0\ 1] \\ 0, & caso\ contrário \end{cases} \quad (12)$$

2.3.1.3 Cúbica de terceira ordem

Este algoritmo, primeiramente, introduzido por Keys (1981) utiliza um polinômio de terceira ordem, conforme a equação 13 (VIEIRA; GONZAGA, 2016):

$$CB(x) = \begin{cases} \frac{3}{2}|x|^3 - \frac{5}{2}|x|^2 + 1, \ |x| \in [0\ 1] \\ -\frac{1}{2}|x|^3 - \frac{5}{2}|x|^2 - 4|x| + 2, \ |x| \in [1\ 2] \\ 0, caso\ contrário \end{cases} \quad (13)$$

2.3.1.4 Cúbica *B-spline*

Este algoritmo trabalha com a utilização de um polinômio de terceira ordem introduzido em (HOU; ANDREWS, 1978), conforme a equação 14 (VIEIRA; GONZAGA, 2016):

$$BS(x) = \begin{cases} \frac{1}{2}|x|^3 - |x|^2 + \frac{4}{6}, \ |x| \in [0\ 1] \\ -\frac{1}{6}|x|^3 + |x|^2 - 2|x| + \frac{8}{6}, \ |x| \in [1\ 2] \\ 0, caso\ contrário \end{cases} \quad (14)$$

2.3.1.5 Lanczos

Este algoritmo utiliza a função *sinc* para a estimativa dos valores (LANCZOS, 1956), conforme a equação 15 (VIEIRA; GONZAGA, 2016):

$$LZ(x) = \begin{cases} sinc(x)sinc\left(\frac{x}{a}\right) \\ 0, caso\ contrário \end{cases}, x \in [-a\ \ a], \quad (15)$$

na qual, $sinc(x) = \frac{\sin(x)}{x}$, e $a \in$ N define o número de extremos na função *sinc*.

2.3.2 Rotação por *hardware*

Neste tipo de rotação não há “perda” de informação gerada por algoritmos computacionais. Porém, podem existir perdas causadas pelo ruído do sensor, já que as imagens serão amostradas em diversas orientações.

2.4 TRABALHOS RELACIONADOS

A classificação de textura tem uma longa história em visão computacional. Na última década, a utilização de técnicas de aprendizagem profunda, em geral, e de redes neurais convolucionais, em particular, permitiu uma melhoria drástica na precisão dos sistemas de reconhecimento de texturas. Nesta seção serão apresentados alguns trabalhos que envolvem a utilização de CNNs para problemas relacionados à classificação de textura.

2.4.1 Melhorando a classificação de textura baseada em CNN por balanceamento de Cores

Um problema que afeta o desempenho de CNNs na classificação de texturas é o fato de que, muitas vezes, as imagens de textura são encontradas com distribuições de cores que são diferentes em relação àquelas utilizadas durante o treinamento das redes.

Em seu trabalho os autores Bianco et al. (2017) avaliaram o efeito da utilização de modelos de balanceamento de cores no desempenho do reconhecimento de texturas para muitas arquiteturas de CNNs.

Para os experimentos foi utilizada a base de dados RawFooT (CUZANO; NAPOLETANO; SCHEINI, 2016) que foi especialmente projetada para investigar o desempenho de métodos de classificação de texturas coloridas sob várias condições de iluminação. A base de dados inclui imagens de 68 amostras diferentes de alimentos crus, cada uma adquirida através de 46 condições de iluminação diferentes (para um total de 68 × 46 = 3128 aquisições). A base de dados RawFooT, também, inclui um conjunto de aquisições de cores para as diferentes condições de iluminação (o verificador de cores Macbeth (MCCAMY; MARCUS; DAVIDSON, 1976). Os autores estudaram, em particular, os efeitos de mudanças na cor gerados pelo iluminante. Portanto, a análise foi feita utilizando amostras de 12 iluminantes que simulam condições de luz do dia, e de 6 que simulam iluminação interna.

As arquiteturas de CNN utilizadas foram variações pré-treinadas das redes AlexNet e VGG. Foram utilizados cinco métodos de balanceamento de cores: *device-raw, light-raw, dcraw-srgb, linear-srgb e rooted-srgb*.

O primeiro não realiza nenhuma alteração nas imagens, o segundo realiza a correção da cor do iluminante, enquanto os demais utilizam métodos que executam uma caracterização de cores completa. Todas as matrizes de correção utilizadas pelos algoritmos de balanceamento de cores foram obtidas utilizando o conjunto de aquisições do verificador de cores Macbeth.

Os experimentos foram realizados com a utilização de conjuntos de amostras nas 18 condições de iluminação descritas anteriormente e os métodos de balanceamento de cores foram utilizados como pré-processamento.

Os resultados obtidos mostram que balanceamento de cores como uma etapa de pré-processamento permitem uma melhoria significativa na acurácia para o reconhecimento de texturas coloridas na presença de condições de iluminação variáveis, para as arquiteturas de CNN utilizadas.

2.4.2 Reconhecimento de Espécies Florestais usando Redes Neurais Convolucionais Profundas

O reconhecimento de espécies florestais é uma tarefa importante em muitas áreas. É importante a certeza dos tipos de madeiras que estão sendo usadas para a fabricação dos materiais utilizados em uma determinada construção, ou em fabricação de mobílias, como mesas e cadeiras.

O reconhecimento de espécies florestais tem sido geralmente tratado como um problema de classificação de textura, devido à propriedade de que a superfície da seção transversal das árvores tem padrões diferentes em diferentes espécies (HARMS; GUNZER; AUS, 1986).

Em seu trabalho, os autores Hafemann, Oliveira e Cavalin (2014) fizeram um estudo sobre a utilização de redes neurais convolucionais para o reconhecimento de espécies florestais.

Os experimentos foram realizados com a utilização de duas bases de dados de espécies florestais brasileiras. A primeira base de dados contém imagens macroscópicas (FILHO, 2014): imagens de superfícies de seção transversal das árvores, obtidas usando uma câmera digital comum. Este conjunto de dados consiste em 41 classes, contendo mais de 50 imagens de alta resolução (3264 × 2448) para cada classe. A segunda base de dados contém imagens microscópicas (MARTINS et al., 2013) obtidas através de um procedimento realizado por laboratório, esta base de

dados é composta por 112 espécies, contendo 20 imagens de resolução 1024 × 768 para cada classe.

As imagens macroscópicas foram redimensionadas para 256 × 256 pixels, e as imagens microscópicas foram redimensionadas para 640 × 640 pixels.

Para o treinamento e teste da rede foram extraídas amostras de tamanho 48×48x3 (RGB) da base de dados macroscópica e amostras de tamanho e 64×64x1 da base de dados microscópica. Para a base microscópica, os experimentos foram feitos com amostras em escala de cinza, pois para esta base de dados as cores nas imagens não são naturais das espécies florestais, mas um resultado do procedimento realizado para produzir contraste nas imagens microscópicas.

A arquitetura utilizada pelos autores para a realização dos experimentos consiste nas seguintes camadas, com os seguintes parâmetros:

Camada de entrada: os parâmetros dependem da resolução da imagem e do número de canais do conjunto de dados (48×48x3 e 64×64x1 para as bases de dados macroscópica e microscópica respectivamente).

Duas camadas de convolução seguidas por camadas de *pooling*, camada de convolução com 64 filtros com tamanho 5 × 5 e passo 1, cada camada de *pooling* com filtros de tamanho 3 × 3 e passo 2.

Camada localmente conectada: 32 filtros de tamanho 3 × 3 e passo 1, camadas conectadas localmente só conectam os neurônios dentro de uma pequena janela da camada seguinte, similarmente às camadas convolucionais, porém não há compartilhamento de pesos.

Camada de saída totalmente conectada: depende do número de classes do conjunto de dados (41 e 112 neurônios para as bases de dados macroscópica e microscópica respectivamente).

Os resultados de acurácia obtidos pelos autores para a base de dados macroscópica (95,77%) superaram muitos métodos do estado da arte publicados para a mesma base de dados, porém, não superou os resultados obtidos por um classificador treinado com a utilização do descritor CLBP (96,22%) e pelo melhor método publicado que utiliza uma combinação de seis classificadores diferentes (97,77%). Já os resultados de acurácia obtidos pelos autores para a base de dados microscópica (97,32%) superaram todos os métodos do estado da arte publicados para a mesma base de dados.

2.4.3 Utilização de bancos de filtros em redes neurais convolucionais para classificação de texturas

Em seu trabalho, os autores Andrearczyk e Whelan (2016) propuseram uma nova arquitetura de redes neurais convolucionais para a classificação de texturas chamada T-CNN.

Os autores criaram uma nova camada denominada camada de energia. Essa camada é utilizada após a última camada de convolução e, antes da primeira camada totalmente conectada. Cada mapa de características da última camada de convolução é simplesmente agrupado, utilizando um *pooling* calculando a média de sua saída de ativação linear retificada. Isso resulta em um único valor por mapa de características, semelhante a uma resposta de energia a um banco de filtros em que são usadas características aprendidas de complexidade variável em vez de filtros fixos.

Uma vantagem do método proposto pelos autores (T-CNN) sobre uma CNN clássica, é que os tamanhos das imagens de entrada não devem ser necessariamente fixos, já que a camada de energia faz o *pooling* da resposta média dos mapas de características de qualquer tamanho.

Para os experimentos, os autores utilizaram 5 variações da arquitetura: de T-CNN-1 até T-CNN-5, contendo de uma a cinco camadas de convolução. Foi também utilizada a AlexNet original (KRIZHEVSKY; SUTSKEVER; HINTON, 2012) para a comparação dos resultados.

Os autores utilizaram um total de dez bases de dados, sete bases de dados de textura, e outras três bases de dados de objetos. As bases de dados estão descritas a seguir.

ImageNet (RUSSAKOVSKY et al.; 2014) contém 1000 classes. O conjunto de treinamento contém 1.281.167 imagens e o conjunto de validação (usado para teste) 50.000 imagens (50 imagens / classe) de tamanho 256x256.

Os autores criaram três subconjuntos do ImageNet. Cada subconjunto, contém 28 classes do ImageNet. Para cada classe selecionada, foram mantidas as mesmas imagens dos conjuntos de treinamento e teste do ImageNet. Portanto, cada base de dados contém 1400 imagens de teste e aproximadamente 36000 imagens, dependendo das classes selecionadas.

ImageNet-T é um subconjunto que mantém as classes de textura como “parede de pedra”, “telhado de telha” e “veludo”.

ImageNet-S1 é outro subconjunto com classes de objetos escolhidos, como “chihuahua”, “ambulância” e “martelo”.

ImageNet-S2, neste subconjunto as 28 classes foram escolhidas aleatoriamente.

kth-tips-2b (HAYMAN, 2004) contém 11 classes de 432 imagens de textura. Cada classe é composta por quatro amostras (108 imagens / amostra). Cada amostra é usada uma vez como um conjunto de treinamento, enquanto as três amostras restantes são usadas para teste. As imagens foram redimensionadas.

Kylberg (KYLBERG, 2011) é uma base de dados de textura contendo 28 classes de 160 imagens de tamanho 576x576. Para cada imagem foram retiradas quatro amostras, o que resulta em 28x160x4 = 17920 imagens de tamanho 288x288 que foram redimensionadas.

CUReT (DANA et al., 1999) é uma base de dados de texturas que contém 61 classes. 92 imagens foram usadas por classe com 46 para treinamento e 46 para teste. As imagens foram redimensionadas.

DTD (CIMPOI et al., 2014) que contém 47 classes cada qual com 120 imagens "na natureza". As imagens possuem tamanhos variáveis, e embora, seja possível a utilização de imagens de entrada de diversos tamanhos na arquitetura proposta pelos autores, as imagens foram redimensionadas para a comparação com a CNN AlexNet original.

Bases de dados **macroscópica** (FILHO, Pedro L. et al, 2014) e **microscópica** (MARTINS et al., 2013) de espécies florestais, foram utilizadas para o teste em imagens de textura com dimensões superiores, respectivamente 3264x2448 e 1024x768. As imagens de ambas bases de dados foram redimensionadas para 640x640. Para essas bases de dados não foi realizada a comparação com a AlexNet original, já que esta aceita apenas imagens de entrada de tamanho fixo.

Foram feitos experimentos com as redes treinadas a partir das bases de dados descritas anteriormente, e também, com as redes pré-treinadas na base de dados ImageNet.

Os experimentos realizados pelos autores mostram que, das variações propostas, a que obteve melhores resultados foi a T-CNN-3. Para as redes treinadas a partir das bases de dados descritas anteriormente, a T-CNN-3 superou a AlexNet

original nas bases Kylberg, DTD, kth-tips-2b, ImageNet-T, ImageNet-S1 e ImageNet-S2. Já para as redes pré-treinadas a T-CNN-3 superou a AlexNet original apenas nas bases CUReT e kth-tips-2b, além de obter a mesma acurácia para a base Kylberg. Porém, em ambos os casos a T-CNN não superou o estado da arte.

Os experimentos realizados pelos autores também mostraram que as redes pré-treinadas na base ImageNet obtiveram resultados superiores quando comparadas com às redes treinadas a partir das bases utilizadas para os testes.

Nos experimentos realizados com as bases de dados macroscópica e microscópica, a T-CNN-3 superou o estado da arte para a base microscópica.

2.4.4 Avaliação de Redes Neurais Convolucionais para Reconhecimento de Textura Rotacionada

Em seu trabalho, os autores Cavichiolli et al. (2018) avaliaram o desempenho de CNNs no reconhecimento de imagens rotacionadas.

A base de dados utilizada foi a *Kylberg Sintorn Rotation Dataset* (KYLBERG; SINTORN, 2014), que contém 25 classes de imagens de textura, cada classe contém 100 amostras de tamanho 122x122 pixels, rotacionadas em 9 orientações distintas: 0°, 40°, 80°, 120°, 160°, 200°, 240°, 280° e 320°.

Foram realizados experimentos utilizando três diferentes arquiteturas de CNNs: AlexNet, ResNet34 e ResNet50, além dos descritores CLMP e SMLP.

Foram utilizadas amostras rotacionadas via *hardware*, e via *software*, através dos métodos de interpolação B-*spline* e *Lanczos.* As amostras passaram por redimensionamento e recorte para se adequarem as entradas das CNNs.

Os experimentos realizados pelos autores mostraram que, com a utilização de amostras apenas no ângulo 0° no conjunto de treinamento, o desempenho das CNNs em reconhecer imagens em outros ângulos de rotação (para todos os métodos de rotação) é bastante inferior ao desempenho dos descritores SMLP e CLMP; este último obtendo as melhores acurácias para todos os diferentes ângulos utilizados nos testes.

Os experimentos ainda mostram que com a adição de imagens rotacionadas na orientação 200° no conjunto de treino o desempenho das CNNs melhora. Neste caso, a rede ResNet34 obteve as melhores acurácias para todos os ângulos de teste.

Porém a adição de amostras rotacionadas em mais orientações piora a acurácia média (de todas as orientações) obtida pelas CNNs.

3 MATERIAIS E MÉTODOS

3.1 FERRAMENTAS UTILIZADAS

Para o desenvolvimento deste trabalho foi utilizado o *software* MATLAB R2018a juntamente com as Toolboxes: *Deep Learning Toolbox*, *Deep Learning Toolbox Model for VGG-16 Network*, *Deep Learning Toolbox Model for AlexNet Network,* rodando em um microcomputador i5-7300HQ com 8 Gb de RAM e uma GPU NVIDIA GEFORCE GTX 1050.

3.2 SELEÇÃO DA BASE DE DADOS

Na literatura são encontradas poucas opções de bases com imagens de textura rotacionadas, as três bases mais utilizadas atualmente são, Outex (OJALA et al., 2002), Mondial Marmi (BIANCONI et al., 2012) e *Kylberg Sintorn Rotation dataset* (KYLBERG; SINTORN, 2015). Estas bases possuem respectivamente 9, 12 e 25 classes de imagens de textura rotacionadas em 9, 10 e 9 diferentes orientações.

Com a necessidade de uma base com maior quantidade de classes e maior diversidade de ângulos de rotação, Vieira (2017) apresentou uma base criada a partir do Álbum de Brodatz (BRODATZ, 1966), denominada Álbum de Brodatz Rotacionado[1]. Sendo essa a base escolhida para a realização dos experimentos propostos neste trabalho.

3.2.1 Álbum de Brodatz Rotacionado

A base foi formada a partir da escolha de 32 classes, aleatoriamente, dentre as 111 imagens de textura do Álbum de Brodatz, e para cada uma dessas classes foram geradas as imagens rotacionadas para 21 ângulos diferentes de rotação, sendo eles: 0°, 10°, 20°, 30°, 40°, 45°, 50°, 60°, 70°, 80°, 90°, 100°, 110°, 120°, 130°, 135°, 140°, 150°, 160°, 170°, 180°. Estes ângulos foram alcançados via *hardware* e via *software*, utilizando métodos de interpolação *Nearest Neighbor*, Linear, Cúbica de terceira ordem, cúbica B-*spline* e Lanczos.

[1] http://imagem.sel.eesc.usp.br/base/

A Figura 3 exibe as 32 imagens do Álbum de Brodatz, escolhidas para a formação das classes do Álbum de Brodatz rotacionado.

Figura 3: 32 imagens escolhidas do Álbum de Brodatz

Fonte: (VIEIRA, 2017) p. 71.

Para a rotação via *hardware* as imagens foram capturadas em dias diferentes sem nenhum tipo de controle de iluminação. As texturas foram digitalizadas na resolução VGA, resultando em uma imagem de tamanho 640 x 480 pixels e formato JPEG. Para cada classe foi realizada a captura de imagens nos 21 ângulos de rotação descritos anteriormente. A Figura 4 mostra um exemplo de uma textura digitalizada nos 21 ângulos de rotação, compondo uma das 32 classes para o método de rotação via *hardware*.

Figura 4 :Exemplo de digitalização de textura em 21 ângulos de rotação.

Fonte: (VIEIRA, 2017), p. 72.

A rotação por *software* foi conduzida com o auxílio do toolbox DIPimage[2] utilizando os métodos de interpolação: *Nearest Neighbor*, Linear, Cúbica de terceira ordem, cúbica B-*spline* e Lanczos.

O Álbum de Brodatz rotacionado é, portanto, dividido em 6 métodos de rotação. Todos os métodos de rotação possuem as mesmas 32 classes, e cada uma dessas classes contém 21 imagens de textura, que correspondem a 21 ângulos de rotação diferentes. Essas imagens, apesar de conterem apenas tons de cinza, foram salvas no formato RGB, possuindo assim, três canais.

[2] http://www.diplib.org/

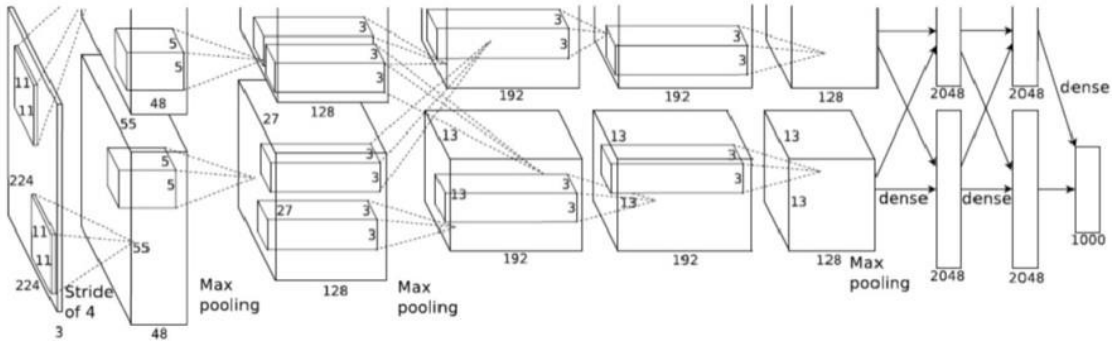
3.3 ARQUITETURA DAS CNNs UTILIZADAS

Para a realização dos experimentos foram utilizadas duas arquiteturas de CNNs: a rede AlexNet desenvolvida por Krizhevsky, Sutskever e Hinton (2012) e a rede VGG-16 desenvolvida por Simonyan e Zisserman (2014). Nesta seção serão abordadas e detalhadas as arquiteturas de ambas as redes.

3.3.1 Alexnet

A Alexnet cuja estrutura básica é exibida na Figura 5, possui 8 camadas com pesos, sendo 5 camadas de convolução e 3 camadas totalmente conectadas. Esta rede ainda possui 1 camada de entrada, 3 camadas de *pooling*, 7 camadas de ativação, 2 camadas de normalização, 2 camadas de regularização (*dropout*), 1 camada de classificação e 1 camada de saída. As camadas da CNN AlexNet estão detalhadas na Tabela 1.

Figura 5: Estrutura da rede Alexnet



Fonte: (KRIZHEVSKY; SUTSKEVER; HINTON, 2012)

Tabela 1: Arquitetura da rede AlexNet

Camada 1	Entrada: Imagens de dimensão 227x227x3 normalizadas com média 0 e variância 1
Camada 2	Convolução1: 96 filtros de dimensão 11x11x3, passo [4 4], e *padding* [0 0 0 0]
Camada 3	Ativação1: função ReLU

Camada 4	Normalização1: normalização de canal cruzado com 5 canais por elemento
Camada 5	*Pooling1*: Max *pooling* de dimensão 3x3, passo [2 2] e *padding* [0 0 0 0]
Camada 6	Convolução2: 256 filtros de dimensão 5x5x48, passo [1 1], e *padding* [2 2 2 2]
Camada 7	Ativação2: função ReLU
Camada 8	Normalização2: normalização de canal cruzado com 5 canais por elemento
Camada 9	Pooling2: Max pooling de dimensão 3x3, passo [2 2] e *padding* [0 0 0 0]
Camada 10	Convolução3: 384 filtros de dimensão 3x3x256, passo [1 1] e *padding* [1 1 1 1]
Camada 11	Ativação3: Função ReLU
Camada 12	Convolução4: 384 filtros de dimensão 3x3x192, passo [1 1] e *padding* [1 1 1 1]
Camada 13	Ativação4: Função ReLU
Camada 14	Convolução5: 256 filtros de dimensão 3x3x192, passo [1 1] e *padding* [1 1 1 1]
Camada 15	Ativação5: Função ReLU
Camada 16	Pooling3: Max p*ooling* de dimensão, passo [2 2] e *padding* [0 0 0 0]
Camada 17	Totalmente conectada1: 4096 neurônios
Camada 18	Ativação5: Função ReLU
Camada 19	Regularização1: *Dropout* com probabilidade 50%
Camada 20	Totalmente conectada2: 4096 neurônios
Camada 21	Ativação6: Função ReLU
Camada 22	Regularização2: *Dropout* com probabilidade 50%
Camada 23	Totalmente conectada3: 1000 neurônios
Camada 24	Classificação: Função Softmax
Camada 25	Saída: para 1000 classes

Fonte: Adaptado de *Deep Learning Toolbox Model for AlexNet Network*[3]

É importante observar que todos os *paddings* são feitos com zeros. Na representação "*padding* [S I E D]" as letras S e I referem-se aos números de linhas adicionadas respectivamente nas bordas superior e inferior, já as letras E

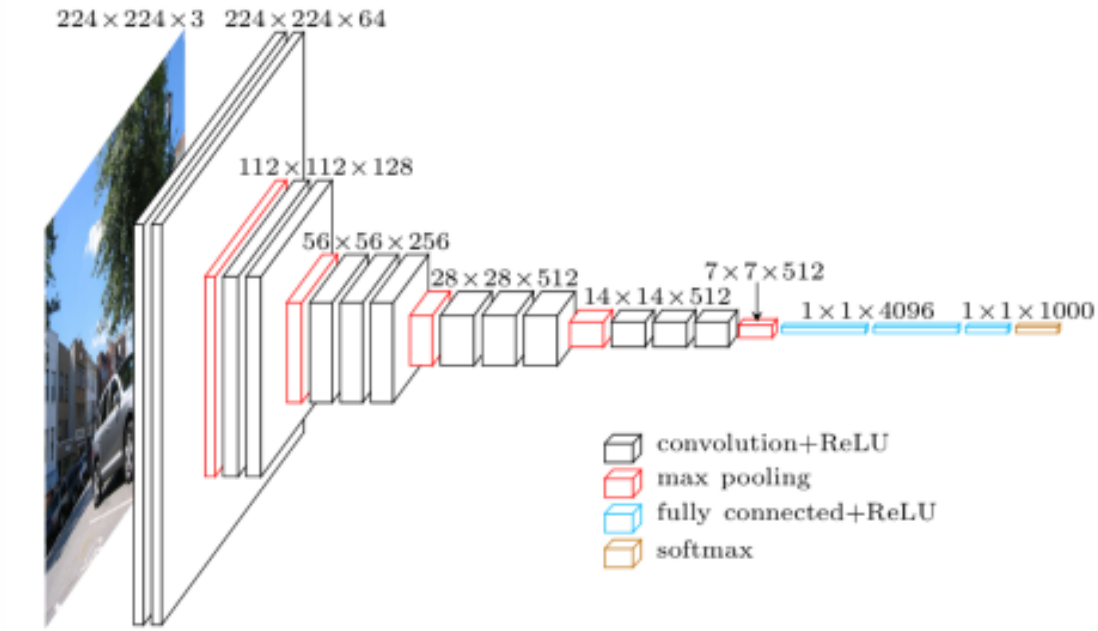
[3] https://www.mathworks.com/matlabcentral/fileexchange/59133-deep-learning-toolbox-model-for-alexnet-network Acesso: Out 2018

e D representam o número de colunas adicionadas nas bordas esquerda e direita.

3.3.2 VGG-16

A rede VGG-16, cuja estrutura básica é exibida na Figura 6, possui uma arquitetura mais profunda quando comparada com a Alexnet. É constituída de 16 camadas com pesos, sendo 13 camadas de convolução e 3 camadas totalmente conectadas. Possui ainda 1 camada de entrada, 5 camadas de *pooling*, 15 camadas de ativação, 2 camadas de regularização (*dropout*), 1 camada de classificação e 1 camada de saída. As camadas da CNN VGG-16 estão detalhadas na Tabela 2.

Figura 6: Estrutura da rede VGG-16



Fonte: (MARTIGNY, 2018)

Tabela 2: Arquitetura da rede VGG-16

Camada 1	Entrada: Imagens de dimensão 224x224x3 normalizadas com média 0 e variância 1
Camada 2	Convolução1: 64 filtros de dimensão 3x3x3, passo [1 1] e *padding* [1 1 1 1]
Camada3	Ativação1: função ReLU
Camada 4	Convolução2: 64 filtros de dimensão 3x3x64, passo [1 1] e *padding* [1 1 1 1]

Camada 5	Ativação2: função ReLU
Camada 6	Pooling1: Max *pooling* de dimensão 2x2, passo [2 2] e *padding* [0 0 0 0]
Camada 7	Convolução3: 128 filtros de dimensão 3x3x64, passo [1 1] e *padding* [1 1 1 1]
Camada 8	Ativação3: função ReLU
Camada 9	Convolução4: 128 filtros de dimensão 3x3x128, passo [1 1] e *padding* [1 1 1 1]
Camada 10	Ativação4: função ReLU
Camada 11	Pooling2: Max *pooling* de dimensão 2x2, passo [2 2] e padding [0 0 0 0]
Camada 12	Convolução5: 256 filtros de dimensão 3x3x128, passo [1 1] e *padding* [1 1 1 1]
Camada 13	Ativação5: função ReLU
Camada 14	Convolução6: 256 filtros de dimensão 3x3x256, passo [1 1] e *padding* [1 1 1 1]
Camada 15	Ativação6: função ReLU
Camada 16	Convolução7: 256 filtros de dimensão 3x3x256, passo [1 1] e padding [1 1 1 1]
Camada 17	Ativação7: função ReLU
Camada 18	Pooling3: Max *pooling* de dimensão 2x2, passo [2 2] e *padding* [0 0 0 0]
Camada 19	Convolução8: 512 filtros de dimensão 3x3x256, passo [1 1] e *padding* [1 1 1 1]
Camada 20	Ativação8: função ReLU
Camada 21	Convolução9: 512 filtros de dimensão 3x3x512, passo [1 1] e *padding* [1 1 1 1]
Camada 22	Ativação9: função ReLU
Camada 23	Convolução10: 512 filtros de dimensão 3x3x512, passo [1 1] e *padding* [1 1 1 1]
Camada 24	Camada 24 – Ativação10: função ReLU
Camada 25	Pooling4: Max *pooling* de dimensão 2x2, passo [2 2] e *padding* [0 0 0 0]
Camada 26	Convolução11: 512 filtros de dimensão 3x3x512, passo [1 1] e *padding* [1 1 1 1]

Camada 27	Ativação11: função ReLU
Camada 28	Convolução12: 512 filtros de dimensão 3x3x512, passo [1 1] e *padding* [1 1 1 1]
Camada 29	Ativação12: função ReLU
Camada 30	Convolução13: 512 filtros de dimensão 3x3x512, passo [1 1] e *padding* [1 1 1 1]
Camada 31	Camada 31 – Ativação13: função ReLU
Camada 32	Pooling5: Max *pooling* de dimensão 2x2, passo [2 2] e *padding* [0 0 0 0]
Camada 33	Totalmente conectada1: 4096 neurônios
Camada 34	Ativação14: função ReLU
Camada 35	Regularização1: *Dropout* com probabilidade 50%
Camada 36	Totalmente conectada2: 4096 neurônios
Camada 37	Ativação15: função ReLU
Camada 38	Regularização2: *Dropout* com probabilidade 50%
Camada 39	Totalmente conectada3: 1000 neurônios
Camada 40	Classificação: Função Softmax
Camada 41	Saída: para 1000 classes

Fonte: Adaptado de *Deep Learning Toolbox Model for VGG-16 Network*[4]

3.3.3 Alterações nas arquiteturas

As três últimas camadas dos dois modelos de CNNs (camada totalmente conectada, função *softmax* e classificador) utilizadas são, originalmente, configuradas para a classificação de 1000 diferentes classes. Para a realização dos experimentos neste trabalho, estas camadas foram substituídas para realizarem a classificação das 32 classes do Álbum de Brodatz Rotacionado. Em outras palavras, as três últimas camadas de ambas as redes, Alexnet e VGG-16, foram alteradas para:

- Camada n-2 – Totalmente conectada: 32 neurônios
- Camada n-1 – Classificação: Função Softmax
- Camada n – Saída: para 32 classes

tal que “n” vale 25 para a AlexNet e 41 para a VGG-16.

[4] https://www.mathworks.com/matlabcentral/fileexchange/61733-deep-learning-toolbox-model-for-vgg-16-network Acesso: Out 2018

3.4 CONJUNTOS GERADOS PARA TREINO E TESTE

Como descrito, previamente, para a realização dos experimentos foram utilizados modelos pré-treinados das arquiteturas VGG-16 e AlexNet. Estas redes aceitam, como entrada, imagens com dimensões fixas de 224x224x3 e 227x227x3 respectivamente. Como descrito na seção **3.2.1**, a base de dados empregada possui imagens de dimensões muito superiores.

Para a realização dos treinamentos e validações das CNNs foram gerados conjuntos de imagens criados a partir do recorte (sem sobreposição) das imagens originais do Álbum de Brodatz rotacionado, em seguida foram selecionadas 2500 amostras para cada conjunto, mantendo-se a proporção entre classes, resultando na criação de seis conjuntos a serem utilizados para cada rede:

- Conjunto HW, formado por 2500 imagens de textura divididas em 32 classes rotacionadas via hardware.
- Conjunto CB, formado por 2500 imagens de textura divididas em 32 classes rotacionadas via software por interpolação Cúbica de terceira ordem.
- Conjunto BS, formado por 2500 imagens de textura divididas em 32 classes rotacionadas via software por interpolação cúbica B-*spline.*
- Conjunto NN formado por 2500 imagens de textura divididas em 32 classes rotacionadas via software por interpolação *Nearest Neighbor.*
- Conjunto LN formado por 2500 imagens de textura divididas em 32 classes rotacionadas via software por interpolação Linear.
- Conjunto LZ formado por 2500 imagens de textura divididas em 32 classes rotacionadas via software por interpolação Lanczos.

As imagens contidas nos conjuntos usados para as redes VGG-16 e AlexNet possuem dimensões de 224x224x3 e de 227x227x3, respectivamente.

Para os testes foram selecionadas de forma aleatória 15% das imagens de cada conjunto, mantendo-se a proporção entre classes. Das imagens restantes 85% foram utilizadas para treinamento e 15% para validação.

3.5 PROCESSO DE TREINAMENTO

As redes não foram treinadas com a inicialização de forma aleatória dos pesos, foram utilizados modelos pré-treinados em mais de um milhão de imagens da base de dados ImageNet (RUSSAKOVSKY et al., 2014). Estes modelos foram adquiridos com a utilização das Toolboxes descritas na seção 3.1.

Para o treinamento das redes, foi utilizado o método de *transfer learning*. Este método consiste em utilizar uma rede pré-treinada como ponto de partida para aprender uma nova tarefa. Para uma base de dados pequena, a utilização de *transfer learning* obtém performance superior à inicialização dos pesos de forma aleatória (HUANG; PAN; LEI, 2017).

Para ambas as CNNs, foi utilizado um *batch size* de 10 amostras e um total de 10 épocas de treinamento. As redes foram treinadas através de um gradiente descendente estocástico com taxa de aprendizagem global de 0.0001. Por não ter sido pré-treinada, a nova camada totalmente conectada, descrita na seção **3.3.3,** foi configurada com um fator de taxa de aprendizado de 0.2 para pesos e *biases*, ou seja, esta camada foi treinada com uma taxa de aprendizado 5 vezes menor do que a taxa global.

3.6 EXPERIMENTOS

3.6.1 Considerações iniciais

As modificações realizadas para cada rede nos detalhes de aprendizagem e nos conjuntos gerados para treino e teste foram descritas anteriormente.

Todos os experimentos foram realizados para as duas redes, com a utilização da rotina base no Apêndice A, com as devidas alterações para cada rede e conjunto de dados utilizados.

O experimento 1 foi elaborado para analisar a robustez das CNNs na classificação de imagens de textura quando a rede é treinada e testada com amostras rotacionadas através do mesmo método.

O experimento 2 foi elaborado para analisar a robustez das CNNs na classificação de imagens de textura quando a rede é treinada com amostras rotacionadas através de apenas um método, e testada com amostras rotacionadas

por outros métodos, individualmente, e com amostras rotacionadas através de todos os métodos utilizados neste projeto.

O experimento 3 foi elaborado para analisar a robustez das CNNs na classificação de imagens de textura quando a rede é treinada com amostras rotacionadas através de todos os diversos métodos utilizados no desenvolvimento desse projeto, porém mantendo-se o mesmo tamanho do conjunto de treinamento utilizado no experimento 1.

O experimento 4 foi elaborado para analisar a robustez das CNNs na classificação de imagens de textura quando a rede é treinada com a utilização de *data augmentation* através da adição de imagens rotacionadas por diversos métodos no conjunto de treinamento.

3.6.2 Experimento 1

Neste experimento o treino e teste das redes foi realizado para cada conjunto de método de rotação. Dessa maneira, foram realizadas 6 simulações:

- Treinamento e teste com o conjunto HW.
- Treinamento e teste com o conjunto CB
- Treinamento e teste com o conjunto BS
- Treinamento e teste com o conjunto NN
- Treinamento e teste com o conjunto LN
- Treinamento e teste com o conjunto LZ

3.6.3 Experimento 2

Neste experimento, para cada rede treinada no experimento 1 pelos conjuntos: HW, CB, BS, NN, LN e LZ foi realizado o teste com utilização de cada conjunto não utilizado no treinamento, e de um conjunto geral formado pela junção de todos os conjuntos de testes dos seis métodos. Sendo assim, foram estudadas as seguintes situações:

- Rede treinada com conjunto HW e testes com conjuntos: CB, BS, NN, LN, LZ e Todos os 6
- Rede treinada com conjunto CB e testes com conjuntos: HW, BS, NN, LN, LZ e Todos os 6
- Rede treinada com conjunto BS e testes com conjuntos: HW, CB, NN, LN, LZ e Todos os 6
- Rede treinada com conjunto NN e testes com conjuntos: HW, CB, BS, LN, LZ e Todos os 6
- Rede treinada com conjunto LN e testes com conjuntos: HW, CB, BS, NN, LZ e Todos os 6
- Rede treinada com conjunto LZ e testes com conjunto: HW, CB, BS, NN, LN e Todos os 6

3.6.4 Experimento 3

Neste experimento, o treinamento foi realizado com a utilização de um conjunto geral composto por amostras rotacionadas através dos seis métodos, enquanto o teste foi realizado, individualmente, para cada um dos conjuntos: HW, CB, BS, NN, LN e LZ, e de um conjunto geral formado pela junção de todos os conjuntos de testes dos seis métodos. Assim, foram estudadas 7 situações:

- Treinamento com o conjunto geral e teste com conjunto HW
- Treinamento com o conjunto geral e teste com conjunto CB
- Treinamento com o conjunto geral e teste com conjunto BS
- Treinamento com o conjunto geral e teste com conjunto NN
- Treinamento com o conjunto geral e teste com conjunto LN
- Treinamento com o conjunto geral e teste com conjunto LZ
- Treinamento com o conjunto geral e teste com Todos os 6

Neste experimento foi formado um conjunto geral que contém todas as imagens dos conjuntos de treinamento: HW, CB, BS NN, LN e LZ. Para o treinamento foram selecionadas de forma aleatória 16,67% das imagens desse conjunto geral, mantendo-se a proporção entre as classes e conjuntos.

3.6.5 Experimento 4

Neste experimento, o treinamento das redes foi feito com a utilização de 100% das imagens dos conjuntos HW, CB, BS NN, LN e LZ enquanto o teste foi realizado individualmente para cada um dos conjuntos: HW, CB, BS, NN, LN e LZ, e de um conjunto geral formado pela junção de todos os conjuntos de testes dos seis métodos.

4 RESULTADOS

4.1 RESULTADO DO EXPERIMENTO 1

Neste experimento as redes foram treinadas e testadas, individualmente, para cada método de rotação. A Tabela 3 exibe os resultados obtidos para a rede AlexNet.

Tabela 3: Acurácia obtida pela rede AlexNet treinada e testada com imagens rotacionadas através do mesmo método

Método de rotação no treinamento e teste	*Hardware*	B-*Spline*	Cúbica	Lanczos	Linear	*Nearest neighbor*
Acurácia(%)	100	93,86	96,80	97,60	97,33	95,73

Fonte: Elaborada pelo autor

Apesar de todos os métodos de rotação se mostrarem eficazes no reconhecimento de texturas rotacionadas, pode-se observar uma variação na acurácia obtida pelos diferentes métodos.

Neste cenário, o melhor resultado foi obtido através da rede treinada e testada com amostras rotacionadas via *hardware,* apresentando uma acurácia de 100%, ou seja, classificando corretamente todas as amostras no conjunto de testes.

Este resultado não foi obtido para nenhum dos métodos de interpolação utilizados para rotações via *software*. Isso pode, de alguma forma, ser explicado pelo fato de que, na rotação via *hardware* não existe perda de informação para os diferentes ângulos de rotação, já que não são feitas estimativas para a localização de pixels com a utilização de algoritmos. Isto demonstra que a rotação computacional acarreta em perdas para a classificação de texturas rotacionadas utilizando CNNs.

O melhor resultado para rotação via *software* foi obtido pela interpolação *Lanczos,* com acurácia de 97,60%, uma variação de 2,4% para a rotação via *hardware.* Enquanto que, o pior resultado obtido foi para o método de interpolação *B-spline* com acurácia de 93,86%, uma variação de 6,14% para a rotação via *hardware*.

O experimento foi reproduzido utilizando a CNN VGG-16. A Tabela 4 exibe os resultados obtidos para a rede VGG-16.

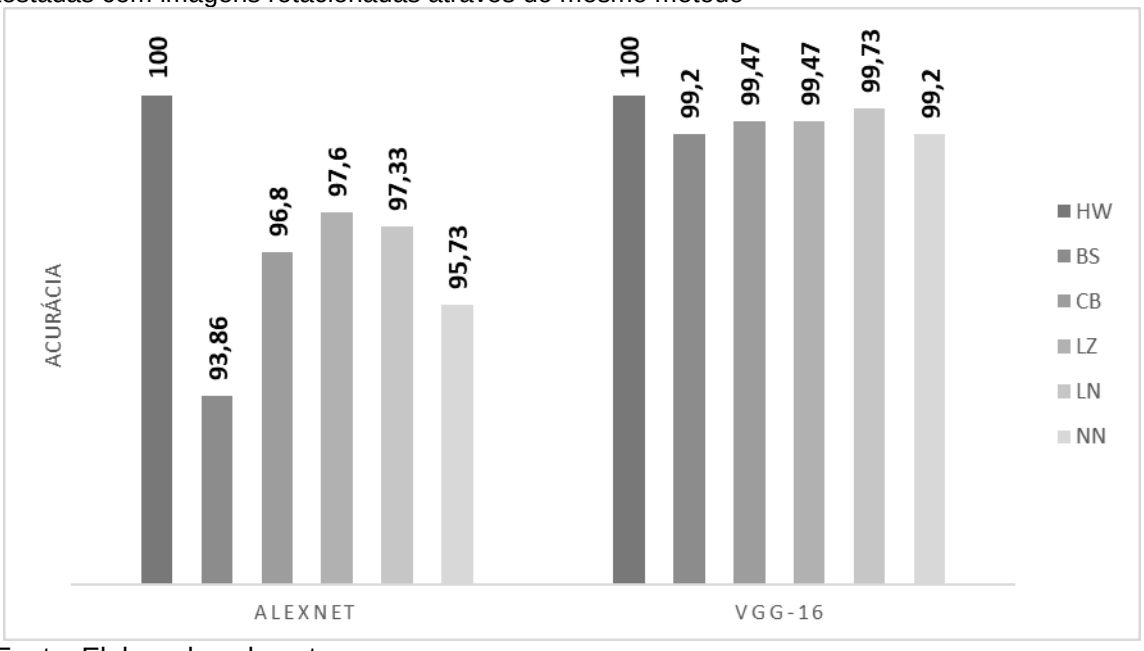
Tabela 4: Acurácia obtida pela rede VGG-16 treinada e testada com imagens rotacionadas através do mesmo método

Método de rotação no treinamento e teste	*Hardware*	B-*Spline*	Cúbica	Lanczos	Linear	*Nearest neighbor*
Acurácia(%)	100	99,20	99,47	99,47	99,73	99,20

Fonte: Elaborada pelo autor

Da mesma forma que o ocorrido com a utilização da AlexNet, o melhor resultado obtido pela VGG-16 foi pela rede treinada e testada com amostras rotacionadas via *hardware* apresentando a mesma acurácia de 100%. Novamente este resultado não foi obtido para nenhum dos métodos de interpolação utilizados para rotações via *software*, porém, para esta rede o pior resultado foi obtido pelas interpolações foi 99,20% para B-*spline* e *Nearest neighbor*, com uma variação de apenas 0,8% para a rotação via *hardware*. A Figura 7 exibe a comparação entre os resultados obtidos pelas duas redes para cada método de rotação.

Figura 7: Comparação entre acurácias obtidas pela AlexNet e VGG-16, para as redes treinadas e testadas com imagens rotacionadas através do mesmo método



Fonte: Elaborada pelo autor

Nota-se que, apesar da rotação via *hardware* apresentar um resultado superior quando comparada com a feita via *software* para ambas as redes treinadas, observa-se que CNNs mais profundas são capazes de diminuir a variação entre estes métodos, já que obtêm melhores acurácias para as rotações via *software*. No entanto, apresentam um maior custo computacional.

Vieira (2017) propôs os descritores CLMP e SLMP_M para o reconhecimento robusto ao método de rotação de texturas rotacionadas. A Tabela 5 exibe os resultados obtidos pela configuração CLMP_S/M(24,3) que obteve os melhores resultados de acurácia para 50 amostras de tamanho 130 x 130 pixels da base de dados *Brodatz Texture Rotation Database* nos testes realizados pela autora.

Tabela 5: Acurácia média obtida pelo descritor CLMP_S/M(24,3) (VIEIRA,2017) testado com 50 amostras de tamanho 130 x 130 pixels da base de dados Brodatz Texture Rotation Database

Método de rotação no teste	Hardware	B-*Spline*	Cúbica	Lanczos	Linear	*Nearest neighbor*
Acurácia média(%)	92,40	91,90	91,80	91,80	89,50	89,20

Fonte: adaptado de (VIEIRA,2017) p. 149

É importante ressaltar que, apesar dos resultados em todos os casos serem obtidos a partir da mesma base de dados, as CNNs foram treinadas com amostras geradas por recortes de tamanho 227x227 e 22x224 pixels para AlexNet e VGG-16 respectivamente, e com a utilização de amostras rotacionadas no conjunto de treinamento. Já o descritor CLMP foi treinado por Vieira (2017) com amostras apenas na orientação 0°, e testado com 50 amostras de tamanho 130x130 pixels.

Fazer uma comparação justa entre o desempenho de CNNs e descritores de textura para classificação de texturas rotacionadas torna-se uma tarefa complicada, já que a acurácia obtida e o custo computacional das CNNs podem variar de acordo com a arquitetura, com o tamanho da base de dados, além das opções de treinamento como taxa de aprendizado global. Em seu trabalho, Vieira (2017) demonstra que a quantidade e tamanho de amostras altera a acurácia e o custo computacional de diversos descritores do estado da arte. Consequentemente, existem parâmetros que afetam o desempenho de CNNs e descritores que não são comuns entre os dois métodos.

4.2 RESULTADO DO EXPERIMENTO 2

Neste experimento, as redes foram treinadas para cada método de rotação e testadas com imagens rotacionadas pelos outros métodos individualmente e, por todos os seis métodos. As Tabelas 6, 7, 8, 9, 10 e 11 exibem os resultados obtidos para a rede AlexNet no decorrer do experimento 2.

Tabela 6: Acurácia obtida pela rede AlexNet treinada com imagens rotacionadas via hardware e testada com imagens rotacionadas através de cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos.

Método de rotação no teste	*B-Spline*	Cúbica	Lanczos	Linear	*Nearest neighbor*	Todos os seis
Acurácia(%)	72,53	72,80	77,60	75,20	68,80	77,82

Fonte: Elaborada pelo autor

Tabela 7: Acurácia obtida pela rede AlexNet treinada com imagens rotacionadas por interpolação *B-spline* e testada com imagens rotacionadas através de cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos.

Método de rotação no teste	*Hardware*	Cúbica	Lanczos	Linear	*Nearest neighbor*	Todos os seis
Acurácia(%)	97,60	96,00	95,20	93,60	91,89	94,69

Fonte: Elaborada pelo autor

Tabela 8: Acurácia obtida pela rede AlexNet treinada com imagens rotacionadas por interpolação Cúbica e testada com imagens rotacionadas através de cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos.

Método de rotação no teste	*Hardware*	B-*Spline*	Lanczos	Linear	*Nearest neighbor*	Todos os seis
Acurácia(%)	99,47	95,20	98,93	99,20	95,20	97,47

Fonte: Elaborada pelo autor

Tabela 9: Acurácia obtida pela rede AlexNet treinada com imagens rotacionadas por interpolação Lanczos e testada com imagens rotacionadas através de cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos.

Método de rotação no teste	*Hardware*	B-*Spline*	Cúbica	Linear	*Nearest neighbor*	Todos os seis
Acurácia(%)	99,73	96,00	98,93	99,20	96,80	98,04

Fonte: Elaborada pelo autor

Tabela 10: Acurácia obtida pela rede AlexNet treinada com imagens rotacionadas por interpolação Linear e testada com imagens rotacionadas através de cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos.

Método de rotação no teste	*Hardware*	B-*Spline*	Cúbica	Lanczos	*Nearest neighbor*	Todos os seis
Acurácia(%)	99,47	96,53	99,20	98,93	94,13	97,60

Fonte: Elaborada pelo autor

Tabela 11: Acurácia obtida pela rede AlexNet treinada com imagens rotacionadas por interpolação Nearest Neighbor e testada com imagens rotacionadas através de cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos

Método de rotação no teste	*Hardware*	B-*Spline*	Cúbica	Lanczos	Linear	Todos os seis
Acurácia(%)	99,47	95,73	97,33	97,87	97,01	97,19

Fonte: Elaborada pelo autor

Para a análise dos resultados obtidos neste experimento é inevitável a comparação com o experimento 1. Surpreendentemente para as redes treinadas com os métodos de rotação via *software*, os melhores resultados não são obtidos quando a rede é testada com amostras rotacionadas pelo mesmo método (93,86% para BS; 96,80% para CB; 97,60% para LZ; 97,33% para LN e 95,73% para NN) (ver tabela 3). Os melhores resultados de acurácia são obtidos para os testes realizados com amostras rotacionadas via *hardware* (97,60% para BS; 99,47% para CB; 99,73% para LZ; 99,47% para LN e 99,47% para NN).

Nota-se que para os métodos de rotação via *software,* quando a rede é treinada por apenas um método individual, existe um leve aumento de acurácia quando a rede é testada por amostras rotacionadas por todos os seis diferentes métodos, em comparação com o teste realizado apenas com amostras rotacionadas através do mesmo método (aumento de: 0,86% para BS; 0,67% para CB; 0,44% para LZ; 0,27% para LN e 1,46% para NN). Os resultados deixam claro que para uma rede treinada com amostras rotacionadas por um dos métodos de rotação via *software* é capaz de reconhecer imagens rotacionadas por todos os métodos.

Uma variação significativa ocorre para o método de rotação via *hardware*, pois quando a rede é treinada e testada com amostras desse método (experimento 1) apresenta uma acurácia de 100%; todavia quando o teste é realizado com amostra rotacionadas por outros métodos o desempenho da rede diminui, sendo que o teste realizado com todos os seis métodos apresenta uma acurácia de

77,82%, resultando em uma variação de 22,18%. A explicação se dá, pelo fato de que, no segundo caso a rede sofre *overfitting*. Quando a rede é treinada apenas com imagens rotacionadas via *hardware* o modelo se encaixa muito bem para o conjunto de treinamento, e dessa forma, durante o treinamento não existe variação significativa nos pesos da rede após cada iteração. Como visto no experimento 1, a rotação computacional acarreta em perdas para a classificação de texturas rotacionadas utilizando CNNs, ou seja, a rede não aprende a reconhecer os "defeitos" gerados pelas estimativas dos diferentes algoritmos de rotação computacional.

O experimento foi reproduzido utilizando a CNN VGG-16. As Tabelas 12, 13, 14, 15, 16 e 17 exibem os resultados obtidos para a rede VGG-16.

Tabela 12: Acurácia obtida pela rede VGG-16 treinada com imagens rotacionadas via hardware e testada com imagens rotacionadas através de cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos.

Método de rotação no teste	B-*Spline*	Cúbica	Lanczos	Linear	*Nearest neighbor*	Todos os seis
Acurácia(%)	89,33	89,60	90,40	89,87	86,40	90,93

Fonte: Elaborada pelo autor

Tabela 13:Acurácia obtida pela rede VGG-16 treinada com imagens rotacionadas por interpolação *B-spline* e testada com imagens rotacionadas através de cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos.

Método de rotação no teste	*Hardware*	Cúbica	Lanczos	Linear	*Nearest neighbor*	Todos os seis
Acurácia(%)	100	99,73	99,73	98,93	98,93	99,42

Fonte: Elaborada pelo autor

Tabela 14: Acurácia obtida pela rede VGG-16 treinada com imagens rotacionadas por interpolação Cúbica e testada com imagens rotacionadas através de cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos.

Método de rotação no teste	*Hardware*	B-*Spline*	Lanczos	Linear	*Nearest neighbor*	Todos os seis
Acurácia(%)	100	99,47	99,73	100	99,47	99,69

Fonte: Elaborada pelo autor

Tabela 15: Acurácia obtida pela rede VGG-16 treinada com imagens rotacionadas por interpolação Lanczos e testada com imagens rotacionadas através de cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos.

Método de rotação no teste	*Hardware*	B-*Spline*	Cúbica	Linear	*Nearest neighbor*	Todos os seis
Acurácia(%)	100	99,20	99,47	99,73	99,73	99,60

Fonte: Elaborada pelo autor

*T*abela 16: Acurácia obtida pela rede VGG-16 treinada com imagens rotacionadas por interpolação Linear e testada com imagens rotacionadas através de cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos.

Método de rotação no teste	*Hardware*	B-*Spline*	Cúbica	Lanczos	*Nearest neighbor*	Todos os seis
Acurácia(%)	100	99,20	99,47	99,47	98,93	99,47

Fonte: Elaborada pelo autor

Tabela 17: Acurácia obtida pela rede VGG-16 treinada com imagens rotacionadas por interpolação *Nearest Neighbor* e testada com imagens rotacionadas através de cada um dos outros cinco métodos e de um conjunto geral formado por imagens rotacionadas através dos 6 métodos

Método de rotação no teste	*Hardware*	B-*Spline*	Cúbica	Lanczos	Linear	Todos os seis
Acurácia(%)	100	98,67	99,47	99,47	98,93	99,29

Fonte: Elaborada pelo autor

Observa-se que, como ocorrido para a rede AlexNet, a comparação dos resultados obtidos pela rede VGG-16 no experimento 2 com os resultados do experimento 1, mostra que para as redes treinadas com os métodos de rotação via software os melhores resultados de acurácia não são obtidos quando o teste é realizado com amostras rotacionadas pelo mesmo método utilizado no treinamento (99,20% para BS; 99,47% para CB; 99,47% para LZ; 99,73% para LN e 99,20% para NN) (ver tabela 4); os melhores resultados de acurácia são obtidos para os testes realizados com amostras rotacionadas via *hardware* atingindo uma acurácia de 100% para todas essas redes.

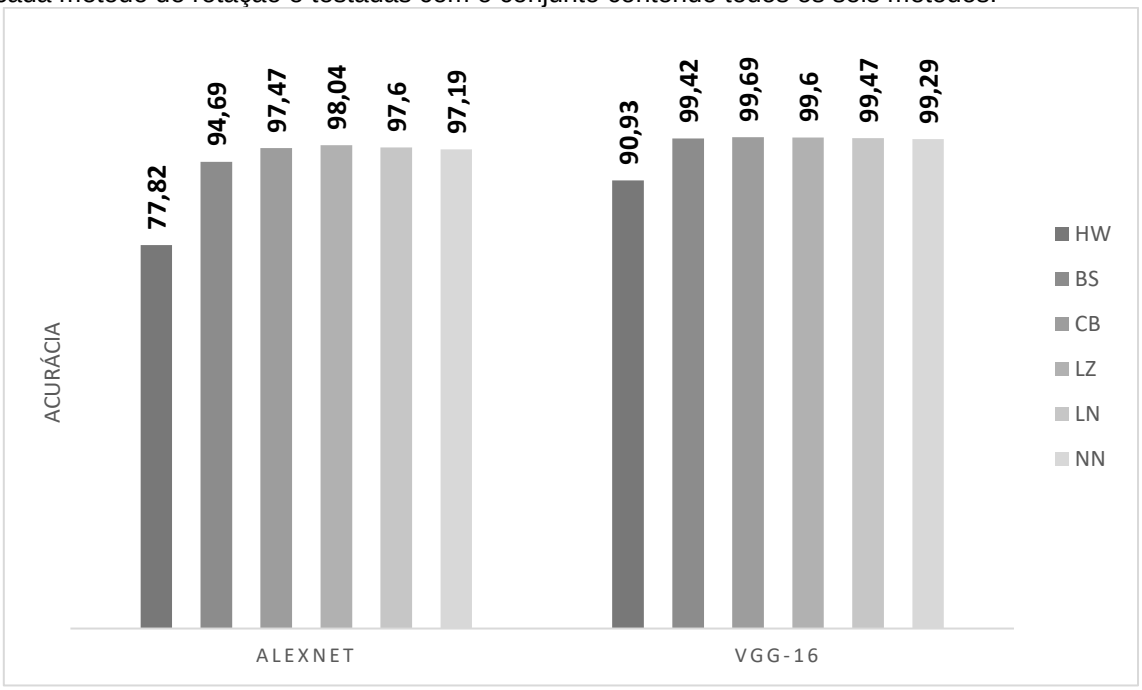
Nota-se que para quatro dos cinco métodos de rotação via *software,* para a rede treinada por um método individual, existe um leve aumento de acurácia quando a rede é testada por amostras rotacionadas por todos os seis diferentes métodos, em comparação ao teste realizado apenas com amostras rotacionadas através do mesmo método (aumento de: 0,22% para BS; 0,22% para CB; 0,13% para LZ; 0,27 e 0,09% para NN). Porém ocorre um leve decrescimento de 0,26% para o método de interpolação Linear.

Novamente, uma variação significativa ocorre para o método de rotação via *hardware*, pois quando a rede é treinada e testada com amostras deste método (experimento 1) apresenta uma acurácia de 100%; todavia, quando o teste é realizado com amostra rotacionadas por outros métodos o desempenho da rede diminui, sendo que o teste realizado com todos os seis métodos apresenta uma

acurácia de 90,93%, resultando em uma variação de 9,07%; demonstrando, novamente, a ocorrência de *overfitting*.

A Figura 8 exibe a comparação entre os resultados obtidos pelas duas redes treinadas por cada método de rotação e testadas com o conjunto contendo todos os seis métodos.

Figura 8: Comparação entre os resultados obtidos pelas CNNs AlexNet e VGG-16 treinadas por cada método de rotação e testadas com o conjunto contendo todos os seis métodos.



Fonte: Elaborada pelo autor

Nota-se que, apesar de também sofrer *overfitting* quando treinada com imagens rotacionadas via *hardware,* a rede mais profunda obtém melhores resultados para todos os casos.

4.3 RESULTADO DO EXPERIMENTO 3

Neste experimento, as redes foram treinadas com 16,67% das imagens de todos os conjuntos de treinamento: HW, BS, CB, LZ, LN e NN e testadas individualmente para cada método e para um conjunto geral composto por amostras de todos os métodos. A Tabela 18 exibe os resultados obtidos para a rede AlexNet.

Tabela 18: Acurácia obtida pela rede AlexNet treinada com imagens rotacionadas através dos 6 métodos (16,67%). E teste realizado com imagens rotacionadas por cada método individualmente, e com um conjunto geral composto por amostras rotacionadas pelos 6 métodos.

Método de rotação no teste	Hardware	B-*Spline*	Cúbica	Lanczos	Linear	*Nearest neighbor*	Todos os 6
Acurácia(%)	100	93,29	96,76	98,09	97,02	96,58	96,96

Fonte: Elaborada pelo autor

Para a análise dos resultados obtidos neste experimento é inevitável a comparação com o experimento 1. No experimento 1 a rede foi treinada e testada para cada método de rotação individualmente, já no experimento 3 a rede foi treinada com amostras de todos os seis métodos e testada com amostras de cada método, individualmente.

Observa-se que para ambos os experimentos, os métodos de rotação utilizados no teste apresentam a mesma ordem de acurácia do melhor para o pior: *Hardware*, Lanczos, Linear, Cúbica, *Nearest Neighbor*, B-*spline*.

Nos dois experimentos o teste com amostras rotacionadas via *Hardware* apresentam 100% de acurácia, já as interpolações B-spline, Cúbica, e Linear apresentam melhores resultados no experimento 1, enquanto as interpolações *Lanczos* e *Nearest neighbor* apresentam melhores resultados no experimento 3.

A maior variação obtida foi para o teste por interpolação *Nearest Neighbor* que obteve acurácia de 95,73% no experimento 1 e 96,58% no experimento 3, apresentando variação de 0,85%.

Portanto, não há melhora ou piora significativa entre os experimentos 1 e 3, demonstrando que uma CNN treinada com imagens de textura rotacionadas por todos os seguintes métodos: *Hardware*, Lanczos, Linear, Cúbica, *Nearest neighbor*, B-*spline*, apresenta grande robustez na classificação de imagens de textura rotacionadas independentemente do método de rotação utilizado no teste.

O experimento foi reproduzido utilizando a CNN VGG-16, e a Tabela 19 exibe os resultados obtidos.

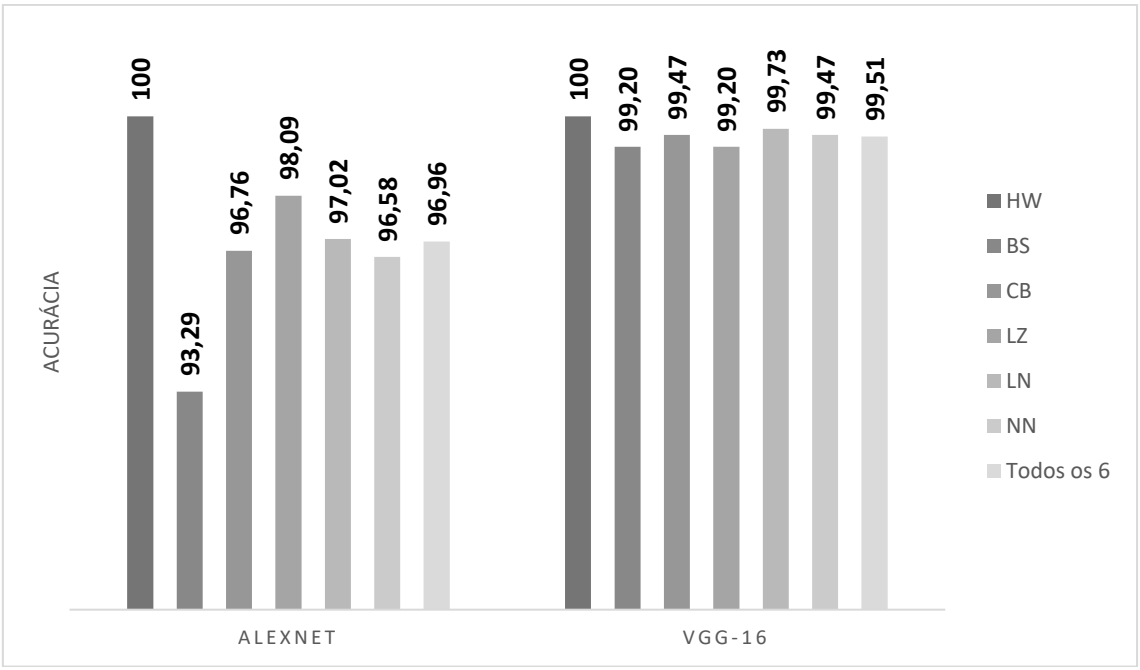
Tabela 19: Acurácia obtida pela rede VGG-16 treinada com imagens rotacionadas através dos 6 métodos (16,67%). E teste realizado com imagens rotacionadas por cada método individualmente, e com um conjunto geral composto por amostras rotacionadas pelos 6 métodos.

Método de rotação no teste	*Hardware*	B-*Spline*	Cúbica	Lanczos	Linear	*Nearest neighbor*	Todos os 6
Acurácia(%)	100	99,20	99,47	99,20	99,73	99,47	99,51

Fonte: Elaborada pelo autor

Como esperado para uma rede mais profunda, a VGG-16 obteve melhores resultados em todos os casos em comparação com a rede AlexNet, esse resultado pode ser observado na Figura 9 que apresenta um gráfico comparando os resultados obtidos pelas duas redes neste experimento.

Figura 9: Comparação entre acurácias obtidas pelas duas redes treinadas com imagens rotacionadas através dos 6 métodos (16,66%). E teste realizado com imagens rotacionadas por cada método individualmente, e com um conjunto geral composto por amostras rotacionadas pelos 6 métodos.



Fonte: Elaborada pelo autor

4.4 RESULTADO DO EXPERIMENTO 4

Neste experimento, as redes foram treinadas com 100% das imagens de todos os conjuntos de treinamento: HW, BS, CB, LZ, LN e NN e testadas individualmente para cada método e para um conjunto geral composto por amostras de todos os métodos. As Tabelas 20 e 21 exibem os resultados obtidos para as duas redes.

Tabela 20: Acurácia obtida pela rede AlexNet treinada com imagens rotacionadas através dos 6 métodos (100%). E teste realizado com imagens rotacionadas por cada método individualmente, e com um conjunto geral composto por amostras rotacionadas pelos 6 métodos.

Método de rotação no teste	*Hardware*	B-*Spline*	Cúbica	Lanczos	Linear	*Nearest neighbor*	Todos os 6
Acurácia(%)	100	99,73	100	100	100	100	99,96

Fonte: Elaborada pelo autor

Tabela 21: Acurácia obtida pela rede VGG-16 treinada com imagens rotacionadas através dos 6 métodos (100%). E teste realizado com imagens rotacionadas por cada método individualmente, e com um conjunto geral composto por amostras rotacionadas pelos 6 métodos.

Método de rotação no teste	*Hardware*	B-*Spline*	Cúbica	Lanczos	Linear	*Nearest neighbor*	Todos os 6
Acurácia(%)	100	100	100	100	100	100	100

Fonte: Elaborada pelo autor

Para a análise dos resultados obtidos neste experimento é feita a comparação com os resultados do experimento 1. No experimento 1 as redes foram treinadas e testadas individualmente com amostras rotacionadas por cada método, porém no experimento 4 o treinamento foi realizado com a utilização de 100% das imagens de todos os conjuntos de treinamento HW, BS, CB, LZ, LN e NN.

No experimento 1 foi observado que a acurácia se altera para cada método de rotação utilizado. Essa variação foi de 93,86%(B-spline) até 100%(*hardware*) para a AlexNet e de 99,2(B-*spline* e *Nearest Neighbor*) até 100% (*hardware*) para a VGG-16.

Já no experimento 4, para a rede AlexNet o único método que não obteve 100% de acurácia no teste foi a interpolação B-*spline*, que obteve acurácia de 99,73% valor que corresponde à classificação incorreta de apenas uma amostra no conjunto de teste. A rede VGG-16 obteve acurácia de 100% para todos os métodos de rotação utilizados nos testes. Portanto a utilização de *data augmentation* no conjunto de treinamento, através da adição de imagens rotacionadas por todos os métodos estudados nesse trabalho, melhora o desempenho das CNNs em classificar imagens de texturas rotacionadas, independentemente do método de interpolação utilizado.

5 CONCLUSÃO

Este trabalho realizou uma análise da robustez de Redes Neurais Convolucionais para o reconhecimento de imagens de textura rotacionadas através de diversos métodos. Foram avaliadas amostras rotacionadas via *hardware* e via *software*, estas rotacionadas computacionalmente através de cinco métodos de interpolação: *Nearest Neighbor*, Linear, Cúbica de terceira ordem, cúbica B-*spline* e *Lanczos*.

Os resultados obtidos mostram que, uma rede treinada e testada com amostras rotacionadas através de apenas um método apresenta bom desempenho na classificação. A rotação via *hardware* apresenta o melhor desempenho obtendo 100% de acurácia para ambas as redes (AlexNet e VGG-16) utilizadas neste trabalho. Já os algoritmos de interpolação usados para rotação via *software* acarretam em perdas para a classificação de texturas rotacionadas utilizando CNNs.

Os experimentos também mostram que para CNNs treinadas com imagens rotacionadas via software, o melhor método de rotação para o teste é com imagens rotacionadas por *hardware*. No entanto, quando a rede é treinada apenas com amostras rotacionadas via *hardware* e testada com amostras rotacionadas via *software* o desempenho cai, drasticamente, por conta da ocorrência de *overfitting*.

Quando são utilizadas imagens rotacionadas por diversos métodos no treinamento as CNNs apresentam grande robustez na classificação de imagens de textura rotacionadas independentemente do método de rotação utilizado no teste. Além disso, foi demonstrado que a utilização de *data augmentation* no conjunto de treinamento, com a adição de amostras rotacionadas através de diversos métodos melhora o desempenho das CNNs em classificar imagens de texturas rotacionadas, independentemente do método de interpolação utilizado.

Por fim foi demonstrado que arquiteturas mais profundas conseguem obter melhor desempenho no reconhecimento de imagens de textura rotacionadas em troca de um maior custo computacional para o treinamento.

REFERÊNCIAS BIBLIOGRÁFICAS

ANDREARCZYK, Vincent; WHELAN, Paul F. Using filter banks in Convolutional Neural Networks for texture classification. **ScienceDirect.** ELSEVIER, v. 84, p. 63-69, 2016.

BIANCO, Simone et al. Improving CNN-Based Texture Classification by Color Balancing. **Journal of Imaging,** v. 3, n. 3, 20017.

BIANCONI, Francesco et al. Automatic classification of granite tiles through colour and texture features. **Expert Systems with Application**, v.39, n. 12, p. 11212-11218, 2012.

BRODATZ, P. Textures: A Photographic Album for Artists & Designers. New York: Dover Publications, 1966.

CAVALIN, P.; OLIVEIRA, L. S. **A Review of Texture Classification Methods and Databases**. 2017 30th SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T). Niterói: IEEE. 2017.

CAVICHIOLLI, Adriane et al. Evaluation of Convolutional Neural Networks for Rotated Texture Recognition. In: **XIV Workshop de Visão Computacional.** Ilhéus, 2018.

CIMPOI, M. et al. Describing textures in the wild. In: **2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. IEEE, p. 3606–3613, 2014.

CUZANO, Claudio; NAPOLETANO, Paolo; SCHEINI, Raimondo. Evaluating color texture descriptors under large variations of controlled lighting conditions. **Journal of the Optical Society of America A,** v. 33, n. 1, p. 17-30, 2016.

DANA, K. J. et al. Reflectance and texture of real-world surfaces. **ACM Transactions on Graphics (TOG)**, v. 18, n. 1, p. 1-34, 1999.

DAS, Paramitkyla et al. A texture based approach for automatic identification of benign and malignant tumor from FNAC images. In: **Recent Trends in Information Systems(RetIS), 2015 IEEE 2nd International Conference on.** IEEE, 2015. p. 249-254.

FERRAZ, Carolina Toledo; GONZAGA, Adilson. Object classification using a local texture descriptor and a support vector machine. **Multimedia Tools and Applications,** p. 1-33, 2016.

FILHO, P. P. L. et al. **Forest species recognition using color-based features**, in Proceedings of the 20th International Conference on Pattern Recognition. 20th International Conference on Pattern Recognition. Instanbul: IEEE. 2010. p. 4178–4181.

FILHO, P. P. L. et al. Forest species recognition using macroscopic images. **Machine Vision and Applications,** v. 25, n. 4, p. 1019-1031, 2014.

GUO, Zhenhua; ZHANG, Lei; ZHANG, David. A completed modeling of local binary pattern operator for texture classification. **IEEE Transactions on Image Processing,** v.19, n.6, p.1657-1663, 2010.

HADID, A. et al. **Computer Vision Using Local Binary Patterns**. London: Springer, v. 40, 2011.

HAFEMANN, Luiz G.; OLIVEIRA, Luiz S.; CAVALIN, Paulo. Forest Species Recognition using DeepConvolutional Neural Networks. In: **22nd International Conference on Pattern Recognition (ICPR),** v. 22, p. 1103 - 1107, 2014.

HARMS, H.; GUNZER, U.; AUS, H. M. Combined local color and texture analysis of stained cells. **Computer Vision, Graphics, and Image Processing,** V. 33, n. 3, P. 364-376, 1986.

HAYMAN, E. et al. On the significance of real-world conditions for material classification. In: **Computer Vision-ECCV 2004**, p. 253–266. Springer, 2004.

HE, K. et al. **Deep Residual Learning for Image Recognition**. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas: IEEE. 2016.

HOU, Hisieh; ANDREWS, H. Cubic splines for image interpolation and digital filtering. I**EEE TRANSACTIONS ON ACOUSTICS, SPEECH, AND SIGNAL PROCESSING,** v. 26, n. 6, p. 508-517, 1978.

HUANG, Zhongling; PAN, Zongxu; LEI, Bin. Transfer Learning with Deep Convolutional Neural Network for SAR Target Classification with Limited Labeled Data. **Remote sensing.** MDPI, v. 9, n. 9, 2017

HUBEL, D. H.; WIESEL, T. N. Receptive fields and functional architecture of monkey striate cortex. **The Journal of physiology**, v. 195, n.1, p. 215-243, 1968.

KASHYAP, Rangasami L.; KHOTANZAD, Alireza. A model-based method for rotation invariant texture classification. **IEEE Transactions on Pattern Analysis and Machine Intelligence,** n.4, p. 472-481, 1986.

KEYS, Robert. Cubic convolution interpolation for digital image processing. I**EEE TRANSACTIONS ON ACOUSTICS, SPEECH, AND SIGNAL PROCESSING,** v. 29, n. 1, p**.** 1153–1160, 1981.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. **ImageNet Classification with Deep Convolutional Neural Networks**. International Conference on Neural Information Processing Systems. Lake Tahoe: Curran Associates. 2012. p. 1097-1105.

KYLBERG, Gusaf. The kylberg texture dataset v. 1.0. External report (Blue series) 35, Centre for Image Analysis, Swedish University of Agricultural Sciences and Uppsala University, Uppsala, Sweden, September 2011.
KYLBERG, Gusaf; SINTORN, Ida-Maria. Kylberg Sintorn Rotation dataset (2015). <http://www.cb.uu.se/~gustaf/KylbergSintornRotation/>. Acesso em 14 Outubro 2018

KYLBERG, Gusaf; SINTORN, Ida-Maria. On the influence of interpolation method on rotation invariance in texture recognition. **EURASIP Journal on Image and Video Processing**, v. 2016, n.1, p.17, 2016.

LANCZOS, Cornelius. **Applied Analysis**. Prentice-Hallmathematics series. New Jersey: Prentice-Hall, 1956.

LATIF, M. H. et al. Texture descriptors based affective states reconition-frontal face thermal image. In: **Biomedical Engineering and Sciences (IECBES), 2016 IEEE EMBS Conference on.** IEEE, 2016. P. 80-85.

LECUN, Y. et al. Gradient-Based Learning Applied to Document Recognition. **Proceedings of the IEEE**, v. 86, n. 11, p. 2278-2324, 1998.

LECUN, Y.; CORTES, C.; BURGES, C. J. C. THE MNIST DATABASE of handwritten digits, 1998. Disponivel em: <http://yann.lecun.com/exdb/mnist/>. Acesso em: 14 Outubro 2018.

LI, Fei-Fei; JOHNSON, Justin; YEUNG, Serena; **Convolutional Neural Networks for Visual Recognition**, Stanford University, 2017. Notas de Aula. Slides.

LUENGO, Cris. DIPimage, a Matlab toolbox for scientific image processing and analysis. Disponível em: <http://diplib.org/>. Acesso em: 14 Outubro 2018.

MARTIGNY, Peter. Transfer Learning in NLP. **Feedlyblog,** 2018 Disponivel em: <https://blog.feedly.com/transfer-learning-in-nlp/>. Acesso em: 14 Outubro 2018.

MARTINS, J. et al. A database for automatic classification of forest species. **Machine Vision and Applications,** v. 24, n. 3, p. 567-578, 2013.

MCCAMY, C.S.; MARCUS H.; DAVIDSON, J. A color-rendition chart. **J. App. Photog. Eng.**, v. 2, n. 3, p. 95-99, 1976.

OJALA, Timo et al. Outex-new framework for empirical evaluation of texture analysis algorithms. In: **Pattern Recognition, 2002. Proceedings. 16th International Conference on.** IEEE, 2002. P. 701-706.

OJALA, Timo; PIEKTIKAINEN, Matti; MAENPAA, Topi. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. **IEEE Transactions on pattern analysis and machine intelligence,** v.24, n.7, p.971-987, 2002.

PACHECO, A. Redes Neurais Convolutivas – CNN. **Computação Inteligente**, 2017. Disponivel em: <http://www.computacaointeligente.com.br/artigos/redes-neurais-convolutivas-cnn/>. Acesso em: 14 Outubro 2018.

RUSSAKOVSKY, Olga et al. Imagenet large scale visual recognition challenge. **International Journal of Computer Vision,** p. 1–42, 2014.

SCHERER, Dominik; MÜLLER, Andreas; BEHNKE, Sven. Evaluation of pooling operations in convolutional architectures for object recognition. **ARTIFICIAL NEURAL NETWORKS–ICANN 2010**. Springer, Berlin, Heidelberg, 2010. p. 92-101.

SIMONYAN, K.; ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. **arXiv preprint arXiv:1409.1556**, 2014.

SRIVASTAVA, Nitish et al. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. **Journal of Machine Learning Research**. v.15, n.1, p.1929-1958.

SZEGEDY, C. et al. **Going deeper with convolutions**. THE IEEE CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION. Boston: IEEE. 2015.

VIEIRA, Raissa Tavares. Brodatz Texture Rotation Dataset. Disponível em: <http://imagem.sel.eesc.usp.br/base/>. Acesso em: 14 Outubro 2018.

VIEIRA, Raissa Tavares. **Descritores robustos à rotação de texturas baseados na abordagem LMP com acréscimo da informação de Magnitude e Sinal**.

2017. Tese (Doutorado em Processamento de Sinais de Instrumentação) - Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2017. doi:10.11606/T.18.2017.tde-04102017-110333. Acesso em: 2018-10-14.

VIEIRA, Raissa; GONZAGA, Adilson. Evaluation of robustness against interpolation methods of texture descriptors in image classification. **XII WORKSHOP DE VISÃO COMPUTACIONAL.** Campo Grande, p.33-38, 2016.

ZHENG, Liang et al. Good Practice in CNN Feature Transfer. **arXiv:1604.00133v1**, 2016.

APÊNDICE A - ROTINA BASE PARA TREINAR E TESTAR CNN

```
%Carregar base de dados para treino.
imdsTrain =
imageDatastore('hardwarealexnetTrain','IncludeSubfolders',true,'LabelSource','fol
dernames');


%Divisão do conjunto de treino em conjunto de treino e validação
[imdsTrain,imdsValidation] = splitEachLabel(imds1,0.85,'randomized');


%Carregar base de dados para teste.
imdsTest =
imageDatastore('hardwarealexnetTest','IncludeSubfolders',true,'LabelSource','fol
dernames');


%Carregar modelo pré-treinado da CNN
net = alexnet;


%Alterações na arquitetura da CNN
layersTransfer = net.Layers(1:end-3);
numClasses = numel(categories(imdsTrain.Labels))
layers = [...
    layersTransfer

fullyConnectedLayer(numClasses,'WeightLearnRateFactor',20,'BiasLearnRateFa
ctor',20)
    softmaxLayer
    classificationLayer];


%Opções de treinamento
options = trainingOptions('sgdm', ...
    'MiniBatchSize',10, ...
```

```
    'MaxEpochs',10, ...
    'Shuffle','every-epoch', ...
    'InitialLearnRate',1e-4, ...
    'ValidationData',imdsValidation, ...
    'ValidationPatience', inf, ...
    'ValidationFrequency',50, ...
    'Verbose',false, ...
    'Plots','training-progress');


%Treinamento
netTransfer = trainNetwork(imdsTrain,layers,options);


%Teste e cálculo da acurácia
[YPred,probs] = classify(netTransfer,imdsTest);
accuracy = mean(YPred == imdsTest.Labels)
```