UNIVERSIDADE FEDERAL DO PAMPA

JEAN DA SILVA BRAGAMONTE

MENSURAÇÃO AUTOMÁTICA DE ESPESSURA DE GORDURA SUBCUTÂNEA A PARTIR DE IMAGENS ULTRASSONOGRÁFICAS DE BOVINOS UTILIZANDO DEEP LEARNING

Bagé
2019

JEAN DA SILVA BRAGAMONTE

MENSURAÇÃO AUTOMÁTICA DE ESPESSURA DE GORDURA SUBCUTÂNEA A PARTIR DE IMAGENS ULTRASSONOGRÁFICAS DE BOVINOS UTILIZANDO DEEP LEARNING

Trabalho de Conclusão de Curso apresentado ao curso de Bacharelado em Engenharia de Computação como requisito parcial para a obtenção do grau de Bacharel em Engenharia de Computação.

Orientador: Sandro da Silva Camargo
Coorientador: Leandro Lunardini Cardoso

Bagé
2019

Ficha catalográfica elaborada automaticamente com os dados fornecidos
pelo(a) autor(a) através do Módulo de Biblioteca do
Sistema GURI (Gestão Unificada de Recursos Institucionais) .

B813m Bragamonte, Jean da Silva
MENSURAÇÃO AUTOMÁTICA DEESPESSURA DE GORDURASUBCUTÂNEA A PARTIR DE IMAGENSULTRASSONOGRÁFICAS DE BOVINOSUTILIZANDO DEEP LEARNING / Jean da Silva Bragamonte.
70 p.

Trabalho de Conclusão de Curso(Graduação)-- Universidade Federal do Pampa, ENGENHARIA DE COMPUTAÇÃO, 2019.
"Orientação: Sandro da Silva Camargo".

1. Redes neurais convolucionais. 2. Aprendizado de máquina. 3. Carcaça bovina. 4. Classificação de gordura. I. Título.

JEAN DA SILVA BRAGAMONTE

MENSURAÇÃO AUTOMÁTICA DE ESPESSURA DE GORDURA SUBCUTÂNEA A PARTIR DE IMAGENS ULTRASSONOGRÁFICAS DE BOVINOS UTILIZANDO DEEP LEARNING

Projeto de Trabalho de Conclusão de Curso apresentado ao curso de Bacharelado em Engenharia de Computação como requisito parcial para a obtenção do grau de Bacharel em Engenharia de Computação.

Projeto de Trabalho de Conclusão de Curso defendido e aprovado em: 29 de Junho de 2019.

Banca examinadora:

Prof. Dr. Sandro da Silva Camargo
Orientador

Prof. Dr. Milton Roberto Heinen
Universidade Federal do Pampa

Prof. Dr. Sandra Dutra Piovesan
Universidade Federal do Pampa

AGRADECIMENTO

Agradeço a Universidade Federal do Pampa, direção e administração por todas as oportunidades de crescimento pessoal e profissional que tive dentro da instituição.

Ao meu orientador, por toda confiança e empenho dedicado na elaboração deste trabalho.

Ao meu co-orientador, a empresa Meat Science e a Empresa Brasileira de Pesquisa Agropecuária, por tornarem possíveis a execução deste trabalho.

A todos os docente por me proporcionar o conhecimento não apenas racional, mas a manifestação do caráter e afetividade da educação no processo de formação profissional. A todos meu eterno agradecimento.

A minha mãe Luciane da Silva Bragamonte, que me deu todo apoio e incentivo nas horas mais difíceis.

A toda minha família, sem ela nada seria possível.

A minha namorada, por estar sempre ao meu lado, me ajudando e dando todo o apoio necessário.

A todos que direta ou indiretamente fizeram parte da minha formação, o meu muito obrigado.

“A morte do homem começa no instante em que ele desiste de aprender.”
— Albino Teixeira

RESUMO

Atualmente, a pecuária de corte é uma das principais atividades econômicas no Brasil, tendo uma grande participação nas exportações. Devido às exigências dos mercados internacionais, a busca pelo incremento da qualidade da carne tem se tornado uma preocupação preponderante em toda a cadeia produtiva. Neste sentido, diferentes estratégias de melhoramento genético têm sido utilizadas, dentre elas está a busca por bovinos com melhor acabamento de gordura corporal. A avaliação de gordura corporal é realizada em diferentes momentos da cadeia produtiva, na maioria das vezes de forma não invasiva, com o animal vivo e com o uso de ultrassom. A partir das imagens de ultrassom obtidas, um especialista realiza manualmente a classificação de acabamento de gordura, gerando uma demora no processo que acaba prejudicando a veracidade dos resultados obtidos. Este trabalho tem como objetivo desenvolver uma abordagem automática de avaliação de acabamento de gordura em bovinos, baseada em redes neurais convolucionais, a partir de imagens ultrassonográficas. As imagens para construção dos modelos de regressão foram fornecidas pela Associação Brasileira de *Hereford* e *Braford* e foram previamente classificadas por um especialista. Das 7951 imagens utilizadas no projeto, 6758 foram destinadas ao treinamento da rede e 1193 para testes. A capacidade de generalização dos classificadores, foi avaliada pela correlação dos resultados obtidos com os valores previamente classificados que foram fornecidos. De acordo com especialista, a correlação das classificações e das previsões deve estar acima de 0.85. A correlação dos resultados da rede com os resultados reais foi de 0.97. Essa correlação dos resultados finais, comprovam a eficácia do uso de Redes Neurais Convolucionais para automatizar o processo de classificação de gordura subcutânea em imagens ultrassonográficas de bovinos.

Palavras-chave: Redes neurais convolucionais. Aprendizado de máquina. Carcaça bovina. Classificação de gordura.

ABSTRACT

Currently, livestock is one of the main economic activities in Brazil and has a large share of the country's exports. Due to the demands of international markets, the search for increased meat quality has become a preponderant concern in the whole production chain. In this sense, different strategies of genetic improvement have been used, among them, is a search for cattle with better body fat finishing. The evaluation of body fat is performed at different moments in the production chain, most of the time noninvasively, with live animals and with the use of ultrasound. From the obtained ultrasound images, a specialist manually performs the fat finishing classification, generating a delay in the process, that ends up harming the veracity of the results obtained. This study aims to develop an automatic approach to evaluate the finished fat in cattle, based on convolutional neural networks, from ultrasonographic images. The images for construction of the regression models were provided by the Brazilian Association of Hereford and Braford and were previously classified by a specialist. Of the 7,951 images used in the project, 6,758 were separated for network training and 1,193 for testing. The generalizability of the classifiers was evaluated by the correlation of the results obtained with the previously classified values that were provided. According to a specialist, the correlation of the classifications and the predictions should be at 0.85 or greater. The correlation of the results of the network with the actual results was 0.97. This correlation of the final results prove the effectiveness of the use of Convolutional Neural Networks to automate the process of subcutaneous fat classification in ultrasound images of bovine.

Keywords: Convolution neural network. Machine learning. Bovine carcass. Fat classification.

LISTA DE FIGURAS

Figura 1 Equipamento utilizado para obter as imagens ultrassonográficas 19
Figura 2 Representação de sítios anatômicos 20
Figura 3 Área de medida da EGS e AOL.....21
Figura 4 Área de medida da EGP.....21
Figura 5 Exemplo de camadas em uma rede neural.....24
Figura 6 Exemplo de camadas de uma rede neural convolucional 25
Figura 7 Filtro da camada convolutiva dentro de um quadro 3x3.....26
Figura 8 Filtro da camada convolutiva após completar todos os *pixels* da imagem de entrada.....27
Figura 9 Filtro da camada de *pooling* 28
Figura 10 Utilização da camada *dropout* em uma rede neural.....29
Figura 11 Fluxograma da metodologia utilizada 38
Figura 12 Estrutura das planilhas disponibilizadas.....41
Figura 13 Estrutura da pasta onde estão as imagens para treinamento.....42
Figura 14 Corte feito nas imagens para diminuir o tempo de processamento do treinamento 43
Figura 15 Arquitetura da rede neural convolucional utilizada nos primeiros testes 44
Figura 16 Arquitetura de camadas do modelo *Inception V3*.....46
Figura 17 Gráfico de perda do treinamento de um modelo inicial.....49
Figura 18 Gráfico de comparação do tempo de execução de treinamento com CPU e GPU.....50
Figura 19 Ultrassons muito escuros ou com ruídos, defeitos causados na hora de capturar a imagem.....51
Figura 20 Modelo 1.....52
Figura 21 Modelo 2.....53
Figura 22 Modelo 3.....54
Figura 23 Modelo 4.....55
Figura 24 Modelo 5.....56
Figura 25 Gráfico do erro médio quadrático durante o treinamento com diferentes arquiteturas de camadas 57
Figura 26 Gráfico do erro médio absoluto para os dados de teste com diferentes arquiteturas de camadas 58
Figura 27 Gráfico do erro médio quadrático para os dados de treino, com diferentes otimizadores.....59
Figura 28 Gráfico do erro médio absoluto para os dados de teste, com diferentes otimizadores.....60
Figura 29 Gráfico do erro médio quadrático para os dados de treino, com diferentes funções de ativação 61
Figura 30 Gráfico do erro médio absoluto para os dados de teste, com diferentes funções de ativação 61
Figura 31 Gráfico de perda do treinamento final do projeto 63
Figura 32 Gráfico de dispersão entre os valores reais e previstos.....64
Figura 33 Tabela com valores de EGS previstos, gerado pelo algoritmo de predição....65

LISTA DE TABELAS

Tabela 1 Arquitetura inicial da rede neural convolucional.....47
Tabela 2 Arquitetura da rede neural convolucional após modificações 48
Tabela 3 Arquitetura da rede neural convolucional utilizada nos testes finais.....62

LISTA DE ABREVIATURAS E SIGLAS

AOL	Área de olho de lombo
API	Interface de programação de aplicativos
CNTK	Kit de ferramentas cognitivas da Microsoft
CPU	Unidade central de processamento
DEP	Diferença esperada na progênie
EGP	Espessura de gordura na picanha
EGS	Espessura de gordura subcutânea
ELU	Unidade linear exponencial
GPU	Unidade de processamento gráfico
IA	Inteligência artificial
MLP	Perceptron multicamadas
RELU	Unidade linear retificada
RNA	Rede neural artificial
RNC	Rede neural convolucional
SGD	Descida do gradiente estocástico
TANH	Tangente hiperbólica
TCC	Trabalho de conclusão de curso
TPU	Unidade de processamento tensorial

SUMÁRIO

1 INTRODUÇÃO 12
1.1 Objetivos 13
1.2 Organização deste trabalho 14
2 FUNDAMENTAÇÃO TEÓRICA 15
2.1 Cadeia produtiva da carne bovina no Brasil 15
2.1.1 Recursos genéticos e estratégias de melhoramento 15
2.1.2 Critérios de seleção 17
2.1.3 Avaliação genética: DEPS 17
2.2 Ultrassom 18
2.2.1 Ultrassom de carcaça 19
2.3 Inteligência artificial 22
2.3.1 Aprendizado de máquina 22
2.3.2 Aprendizado profundo 23
2.3.3 Redes neurais 24
2.3.4 Redes neurais *Feed-Forward* 25
2.3.5 Redes neurais convolucionais 25
2.4 Métodos de otimização 29
2.4.1 Retropropagação de erro 29
2.4.2 Funções de ativação 30
2.4.3 Otimizadores 32
2.5 Validação cruzada 33
2.6 Ambiente de desenvolvimento 34
2.6.1 *Anaconda navigator* 34
2.6.2 *Jupyter notebook* 35
2.6.3 *Tensorflow* 35
2.6.4 *Keras* 35
2.7 Trabalhos correlatos 36
3 METODOLOGIA 38
3.1 Classificação do trabalho 39
3.2 Escolha das ferramentas 39
3.3 Estudo das técnicas 40
3.4 Obtenção das imagens 41
3.4.1 Tratamento dos dados 42
3.5 Desenvolvimento da abordagem 43
3.5.1 Teste de diferente arquiteturas 45
3.5.2 Refinamento do modelo 45
3.5.3 Validação do modelo 45
4 RESULTADOS 46
5 CONSIDERAÇÕES FINAIS 66
5.1 Trabalhos Futuros 66
REFERÊNCIAS 68

1 INTRODUÇÃO

Hoje em dia, a pecuária de corte é uma das principais fontes econômicas no Brasil, que é um dos maiores exportadores mundiais de carne bovina. Para que o país continue nesse patamar de produção é necessário que a genética bovina produzida seja de alta qualidade (SILVA; CESARIO; CAVALCANTI, 2017). Nessa busca pela melhor genética, são usados vários métodos de acompanhamento e melhoramento na criação dos bovinos, dentre os quais está a ultrassonografia, que é utilizada para avaliar a carcaça bovina, buscando mensurar seu nível de gordura (YOKOO *et al.*, 2015).

A utilização de ultrassonografias tem sido de fundamental importância para esse melhoramento na genética. Particularmente para acompanhar o ganho de peso, é utilizado em bovinos ao sobreano, idade na qual ocorre a maior taxa de crescimento no desenvolvimento corporal. As informações obtidas nas ultrassonografias auxiliam na seleção de animais e/ou linhagens de maior potencial para produção de carne e possibilitam grande ganho genético em um menor tempo quando comparado ao modelo convencional (YOKOO *et al.*, 2015).

Atualmente, imagens ultrassonográficas são usadas para obter informações sobre os níveis de gordura bovina. No protocolo proposto por YOKOO *et al.* (2017), por meio de um aparelho de ultrassonografia são coletadas duas amostras por animal, contendo informações da espessura de gordura subcutânea (EGS), área de olho de lombo (AOL) e a espessura de gordura da picanha (EGP). Após o especialista observar em um software as imagens coletadas, ele destaca os pontos onde estão a EGS, EGP e AOL. Com esses pontos marcados, o software faz os cálculos necessários para gerar as medidas, salvando em um arquivo (.xls). Com o acompanhamento dessas classificações ao longo da vida do bovino, pode-se avaliar a sua genética e definir o momento de abate.

Como são geradas centenas de imagens ultrassonográficas, o processo de classificação pelo especialista humano acaba se tornando lento, demandando, assim, um método com melhor performance de processamento com mais eficiência e que tenha um grau de confiabilidade tão alto quanto ao do processo de classificação humano (YOKOO *et al.*, 2015). Técnicas e métodos de aprendizado de máquina, tais como as Redes Neurais Convolucionais (RNC), são opções para automatização desses tipos de processos (VILHARVA *et al.*, 2017).

As Redes Neurais Convolucionais têm um grande potencial nos processos de aprendizado de padrões em imagens, consistindo em uma ferramenta extremamente po-

derosa e eficiente. Esse modelo de rede possui uma importante habilidade de trabalhar com variações nos dados de entrada devido a estrutura de várias camadas, com isso se tornam capazes de processar um grande conjunto de imagens com vários fatores usados para obter padrões, conseguindo resultar em saídas consistentes (MARCOMINI, 2013; NETO, 2017; PEREIRA *et al.*, 2017).

Contudo, o treinamento de uma Rede Neural Convolucional demanda muito tempo de processamento e um grande custo computacional. Para solucionar esse problema, têm-se utilizado unidades de processamento gráfico (*Graphics Processing Unit* - GPU) para o processamento. Esse método de processar o treinamento em uma GPU tem sido utilizado para obter uma melhora significativa no tempo de treinamento de Redes Neurais Convolucionais se comparados ao uso de uma unidade central de processamento (*Central Processing Unit* - CPU) (ZACCONE; KARIM; MENSHAWY, 2017).

Neste contexto, o problema de pesquisa deste trabalho é investigar se a utilização de Redes Neurais Convolucionais em imagens ultrassonográficas de carcaças bovinas permite predizer o nível de acabamento de gordura.

1.1 Objetivos

O presente trabalho tem como objetivo geral desenvolver uma solução baseada em Redes Neurais Convolucionais para automatizar o processo de mensuração da espessura de gordura subcutânea utilizando imagens ultrassonográficas de carcaças de bovinos com um ambiente de alto desempenho.

Este trabalho será realizado a partir dos seguintes objetivos específicos:

1. Padronizar um banco de imagens ultrassonográficas para servir como base de treinamento/teste/validação para aplicação de métodos de aprendizado de máquina.
2. Desenvolver um modelo de redes neurais convolucionais para mensuração da espessura de gordura subcutânea
3. Apresentar uma proposta de ambiente para uso da solução desenvolvida.

1.2 Organização deste trabalho

Este trabalho está organizado da seguinte forma: No capítulo 2, são apresentados os conceitos fundamentais para o entendimento deste trabalho, dentre os quais se destacam a problemática da mensuração de acabamento de gordura corporal em bovinos, a discussão da utilização de imagens ultrassonográficas e os aspectos relativos ao aprendizado de máquina e redes neurais convolucionais. No capítulo 3, é apresentada a metodologia utilizada no desenvolvimento do trabalho, descrevendo as decisões e ações seguidas ou planejadas para realização deste trabalho. Os resultados obtidos são discutidos no capítulo 4. No capítulo 5 discute-se as conclusões e trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo irá abordar os conceitos básicos da cadeia de carne produtiva do Brasil, dos critérios de seleção da carne e do melhoramento genético. Também serão apresentados conceitos de ultrassons em carcaças bovinas e sobre como é feita a classificação da gordura bovina utilizando as imagens ultrassonográficas, além disso, serão abordados os conceitos de aprendizado de máquina, redes neurais artificiais e convolucionais. Por fim, será feita uma discussão sobre o uso de redes neurais convolucionais em imagens ultrassonográficas.

2.1 Cadeia produtiva da carne bovina no Brasil

A inserção da pecuária brasileira no mercado internacional ocorreu recentemente. Apesar de ser uma conquista recente, o Brasil vem alternando entre o primeiro e o segundo lugar na lista dos maiores países exportadores de carne (OLIVEIRA *et al.*, 2017). Para esta carne ser de boa qualidade, são necessários diversos cuidados, a todo o tempo, desde o nascimento do bovino. Se forem executados os devidos cuidados, será produzida uma carne com a melhor qualidade possível (OLIVEIRA *et al.*, 2017).

Segundo Oliveira *et al.* (2017), a carne pode ser classificada da seguinte maneira:

1. Qualidade visual: são os aspectos visuais que atraem ou repelem o consumidor;
2. Qualidade gustativa: são atributos referentes ao aspecto e gosto da carne, que irá fazer com que o consumidor volte a adquirir o produto ou não;
3. Qualidade nutricional: são os nutrientes para passar uma imagem favorável ou desfavorável, quanto a carne ser um alimento compatível com as exigências do consumidor para uma vida saudável;
4. Segurança: são aspectos higiênico-sanitários e a presença ou não de contaminantes químicos, como resíduos de pesticidas na carne.

2.1.1 Recursos genéticos e estratégias de melhoramento

A qualidade da carne é um dos elementos avaliados pelos países importadores. A melhoria de qualidade é frequentemente buscada através do melhoramento genético animal, que também conduz a um aumento na produção. Para que o melhoramento gené-

tico seja atingido, algumas características dos animais devem ser observadas, tais como: adaptabilidade, eficiência reprodutiva, viabilidade, pesos corporais, taxas de crescimento, qualidade da carcaça e da carne (ROMEIRO; MENEZES, 2013).

A expressão mensurável das características da carne, conhecida por fenótipo (P), é o resultado dependente de alguns fatores:

1. Genótipo do animal (G), que, no momento da fecundação, é herdado em porções iguais da mãe e do pai;
2. Ambiente (E) no qual o animal é criado;
3. Interação genótipo x ambiente (GxE), que representa as expressões dos genótipos quando expostos a diferentes condições ambientais. Todos esses fatores compõem a equação básica do melhoramento genético animal para o fenótipo:

$$P = G + E + GxE$$

(ROMEIRO; MENEZES, 2013).

Existem alguns outros elementos que também devem ser considerados, que são: a temperatura do ambiente no qual o animal é criado, a radiação solar a que o animal está exposto, a altitude do ambiente de criação e o índice pluviométrico. Após avaliar todos estes elementos, é possível começar a busca pela raça mais adaptada para este ambiente. A adaptação da raça ao ambiente tende a ser inversamente proporcional aos gastos adicionais ao processo produtivo, o que influi nos custos de produção (ROMEIRO; MENEZES, 2013).

Dentro de um rebanho podem existir animais que não tem as características desejáveis, ou seja, o seu fenótipo não é o ideal para o ambiente. Com isso, esses animais devem ser retirados do rebanho e substituídos por outros que possuem o fenótipo mais adequado, para não serem gerados terneiros com as características indesejáveis. Desta forma, busca-se gerar um conjunto de terneiros que herdem as características dos melhores pais, produzindo uma nova geração geneticamente melhor que a anterior (ROMEIRO; MENEZES, 2013).

2.1.2 Critérios de seleção

Como falado anteriormente, os animais que são gerados são melhores que os anteriores, sendo assim, se adaptam mais facilmente ao ambiente. Além de depender dessa produção para o melhoramento, também tem que ser avaliado o mercado, quais são as necessidades solicitadas (NIETO; ALENCAR; ROSA, 2013).

Os critérios de seleção ocorrem pela escolha de uma ou mais características do animal. Estes critérios são divididos em 4 grupos, que são eles: reprodução, produção, produto e biótipo (NIETO; ALENCAR; ROSA, 2013).

As características ligadas à reprodução bovina que podem ser citadas dentre outras são: idade ao primeiro parto, dias para criar, período de gestação, idade à puberdade, perímetro escrotal e viabilidade (sobrevivência ou porcentagem de bezerros produzidos). Pesos corporais, taxas de crescimento, altura, eficiência alimentar e tamanho adulto são relacionados à produção, enquanto peso e qualidade da carcaça, conformação frigorífica, área de olho de lombo, espessura de gordura subcutânea e maciez de carne são associadas especificamente à qualidade do produto. Quanto ao biótipo e outras características de natureza morfológica e de comportamento, nessas são envolvidos aspectos raciais, sexuais e de funcionalidade, incluem-se aprumos, comprimento de pelos, cor da pele e da pelagem, velocidade de fuga e outros (NIETO; ALENCAR; ROSA, 2013).

Para uma melhor criação de animais, as características do grupo de reprodução são as principais, pois, quanto melhor este grupo for, maior vai ser a quantidade de animais que nascerão com boas características. Já o grupo de produção beneficia a qualidade do produto bem como a produtividade. E por fim, o grupo biótipo tem relação aos custos para a sobrevivência dos animais (NIETO; ALENCAR; ROSA, 2013).

2.1.3 Avaliação genética: DEPS

Para serem reconhecidos os animais com uma genética superior, é necessária a avaliação genética. Utilizando essa avaliação genética, fará com que os filhotes que nascerem posteriormente tenham a capacidade de nascerem com uma genética cada vez melhor (JUNIOS *et al.*, 2013).

Um programa de melhoramento é utilizado para que a taxa de ganhos em cima do produto, que no caso é a carne bovina, seja mais elevada. Para que ocorra esse melhoramento devem ser feitas seleções, quanto mais seleções houverem e quanto mais mudanças

existirem na genética do animal, melhor será a taxa de ganho (JUNIOS *et al.*, 2013).

DEP significa Diferença Esperada na Progênie. A melhor maneira de definir DEP é a seguinte:

> *"A DEP é uma medida da diferença entre o desempenho médio da progênie de um dado touro e o desempenho médio da progênie de um grupo de touros referência, quando acasalados com fêmeas geneticamente semelhantes"* (JUNIOS *et al.*, 2013, p. 151).

Ela não é um número constante e sim uma estimativa que surge via uma equação matemática. Mesmo sendo resultado de uma equação é um valor incerto, como se fosse uma dedução, mas que normalmente está correta (JUNIOS *et al.*, 2013).

2.2 Ultrassom

Ultrassom é um som que possui uma alta frequência, incapaz de ser ouvido pelo ser humano. Na natureza, ele é utilizado pelos animais para auxiliar no encontro das suas presas. O ultrassom é utilizado em diversas aplicações como, medicina, tratamentos, soldas, etc. Um dos exemplos de utilização, tanto na medicina quanto na medicina veterinária, é a ultrassonografia (ALVES *et al.*, 2017).

A ultrassonografia é um procedimento utilizado para produzir análises em tempo real, que são estabelecidos através do eco. As ondas de alta frequência que geram os ecos são denominadas ondas ultrassônicas, essas ondas são geradas pelo equipamento de ultrassonografia mostrado na Figura 1. Esse eco observa as reflexões que são produzidas pelos órgãos ou estruturas de um certo organismo e após o processamento, geram imagens. É um exame de certa forma, que é executado rapidamente, em tempo real, e que não utiliza radiação (ALVES *et al.*, 2017).

Os aparelhos de ultrassonografia possuem transdutores que transformam os ecos em sinais. Os cristais do aparelho recebem os ecos gerando uma diferença de potencial elétrico e através de um computador estes sinais são interpretados e geram as imagens internas de algum organismo (ALVES *et al.*, 2017).

Para a imagem ser formada são observados qual é o tempo da transmissão e recepção do eco, verifica-se qual a distância em que foi formado o eco e qual a sua intensidade. Quanto melhor e mais precisos forem os dados melhor será a imagem gerada pelo aparelho (ALVES *et al.*, 2017).

2.2.1 Ultrassom de carcaça

O ultrassom é usado na pecuária para melhorar a seleção da carcaça. Com essa melhora ocorrem avanços no rendimento, acabamento e marmoreio, beneficiando os lucros por haver um aumento na qualidade da carne, gerando uma melhor produtividade (YOKOO *et al.*, 2015).

Na Figura 1 pode-se observar o equipamento utilizado na obtenção das imagens ultrassonográficas. Esse equipamento é instalado perto de onde o animal é preso para que o técnico tenha condições de realizar o procedimento de captação das imagens ultrassonográficas.

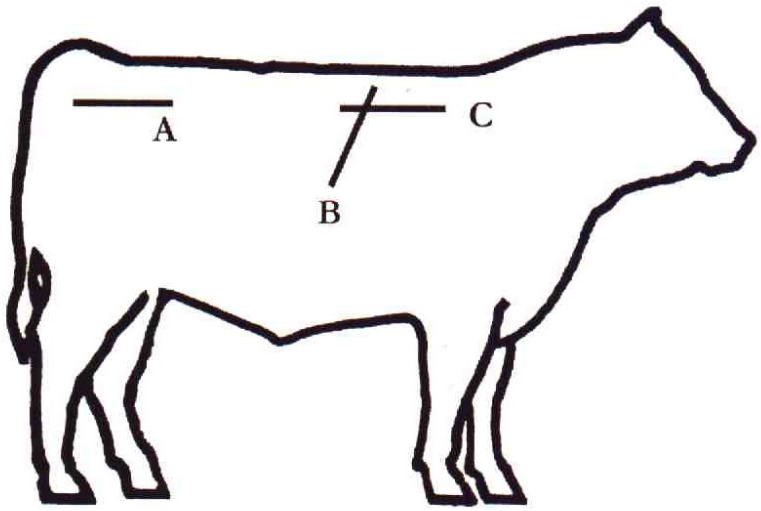
Figura 1 – Equipamento utilizado para obter as imagens ultrassonográficas

Fonte: (Empresa Brasileira de Pesquisa Agropecuária, 2018)

O técnico responsável coloca um gel no local onde será feito o ultrassom, após aplicar o gel ele coloca o transdutor posicionado conforme a Figura 2. O técnico pode acompanhar visualmente na tela do equipamento de ultrassonografia se a imagem captada está em bom estado de visualização, assim que obtiver uma boa visualização efetua o salvamento da imagem.

O ultrassom é feito em 3 partes do bovino, conforme mostrado na Figura 2. Em cada uma dessas partes é gerada uma imagem que prediz características diferentes da carcaça.

Figura 2 – Representação de sítios anatômicos



Fonte: (Empresa Brasileira de Pesquisa Agropecuária, 2018)

Legenda da Figura 2:

A: Espessura de gordura na garupa e profundidade do músculo *Gluteus medius*

B: Área do músculo *longissimus* e espessura de gordura subcutânea

C: Gordura intramuscular e marmoreio

As características da carcaça que são avaliadas no ultrassom são: Área de olho de lombo (AOL), Espessura de gordura subcutânea na costela (EGS) e espessura de gordura na picanha (EGP) (YOKOO *et al.*, 2015).

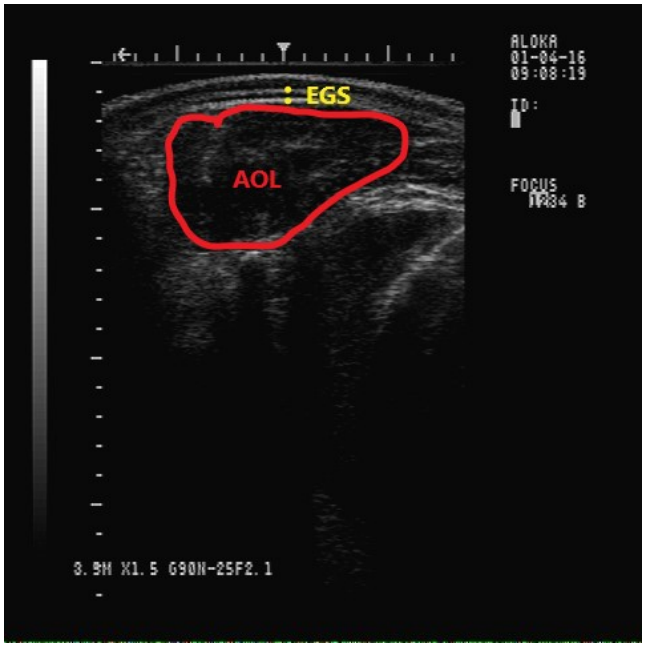
AOL - é o aspecto que define a qualidade da carcaça e a musculosidade.

EGS - é o aspecto que define o grau de acabamento da carcaça.

EGP - é o aspecto que junto com o EGS fornece o grau de acabamento da carcaça.

A imagem abordada na Figura 3 mostra as características obtidas através do ultrassom na posição B da Figura 2. Após ser selecionada a área de AOL e os pontos de EGS no *software* pelo técnico, é gerado um arquivo de texto com as coordenadas selecionadas, no mesmo *software* as coordenadas são processadas gerando os valores de AOL e EGS em uma planilha no formato do Microsoft Excel.

Figura 3 – Área de medida da EGS e AOL



Fonte: Adaptado de: (Empresa Brasileira de Pesquisa Agropecuária, 2018)

A imagem abordada na Figura 4 mostra as características que são obtidas através do ultrassom na posição A da Figura 2. Após serem selecionados os pontos de EGP, ocorre o mesmo processamento feito para AOL e EGS.

Figura 4 – Área de medida da EGP

Fonte: Adaptado de: (Empresa Brasileira de Pesquisa Agropecuária, 2018)

Com os valores obtidos de AOL, EGS e EGP, pode-se fazer algumas observações sobre o estado atual do bovino. Como, por exemplo, se o animal está no momento ideal para o abate ou fazer o acompanhamento desses dados durante algum tempo, afim de analisar se o animal está tendo uma melhora no seu desenvolvimento (YOKOO *et al.*, 2015).

2.3 Inteligência artificial

A inteligência artificial é um mecanismo utilizado para ensinar a máquina a pensar, de uma forma semelhante ao cérebro humano. Através de vários dados, imagens ou vídeos, a máquina recebe estes dados e melhora as suas chances de entendimento do assunto desejado (Norvig, P., & Russell, 2017).

Atualmente existem métodos desenvolvidos que permitem fazer com que a inteligência artificial seja tão capaz quanto o ser humano para entender leituras. Hoje em dia a inteligência artificial é mais eficiente para a resolução de diversos problemas, como classificações de vários tipos de dados (NASCIMENTO, 2001).

Desde o primeiro dia de vida, os seres humanos são colocados em um ambiente social onde eles devem aprender a se desenvolver e se comunicar com os seus semelhantes. Tendo em vista esse meio de aprendizagem, a inteligência artificial também supõe que o treinamento seja feito em um ambiente capaz de proporcionar o aprendizado para a máquina (GUDWIN, 2005).

2.3.1 Aprendizado de máquina

O aprendizado de máquina é uma área de inteligência artificial que tem como base analisar dados para deixar a construção de modelos analíticos feitos de forma automática. Este tipo de aprendizado consegue aprender através de cálculos, fazendo com que sejam produzidas as suas decisões e resultados. Esse mecanismo têm sido muito utilizado, principalmente em sugestões de anúncios de empresas e produtos (DIAS; PASCUTTI; Da Silva, 2017).

Com um aumento significativo nas quantidades e tamanho de dados existentes, era necessário um método que obtivesse uma melhor eficácia para classificar esses dados, com o aprendizado de máquina é possível que isso seja feito. Atualmente, existem vários

métodos de aprendizado de máquina que são capazes de classificar uma grande quantidade de dados de uma forma automática e rápida, como desejado (DIAS; PASCUTTI; Da Silva, 2017).

Dentre os tipos de aprendizado de máquina, existem: aprendizado supervisionado, não supervisionado e semi-supervisionado.

- Aprendizado supervisionado:
  Este aprendizado recebe exemplos de dados, como se já existisse uma pré-classificação, com isso o algoritmo descobre informações a partir dos dados, os quais não eram ainda conhecidos. Exemplos de aprendizado supervisionado são: algoritmos de redes neurais, algoritmos de árvore de decisão, entre outros (DIAS; PASCUTTI; Da Silva, 2017).
- Aprendizado não supervisionado:
  É um aprendizado que não há quaisquer informações de relação entre os dados de entrada e saída, o algoritmo aprende a encontrar as classes por si só com base nas informações obtidas pelo treinamento através dos dados de entrada (DIAS; PASCUTTI; Da Silva, 2017).
- Aprendizado semi-supervisionado:
  Aprendizados semi-supervisionados, são modelos onde não se tem uma grande quantidade de dados rotulados, com isso é necessário a utilização de métodos para encontrar alguma correlação entre as variáveis para que os dados sem rótulos possam ser rotulados. Com isso, mesmo os dados que não tinham rótulos inicialmente são utilizados para o treinamento (DIAS; PASCUTTI; Da Silva, 2017).

2.3.2 Aprendizado profundo

Nos últimos anos, técnicas de aprendizado profundo (do inglês *Deep Learning*) têm sido cada vez mais usadas em várias áreas para o tratamento de dados. A justificativa para esse crescente uso são grandes quantidades de dados disponíveis hoje e a melhoria de máquinas para processamento (ZACCONE; KARIM; MENSHAWY, 2017).

Com o avanço da tecnologia tornou-se viável a utilização de métodos como *Deep Learning*, pois, o treinamento de uma rede neural convolucional tem um alto custo computacional, sendo necessário um alto poder de processamento para que se torne um processo utilizável (BEZERRA, 2016).

Um dos principais marcos que atraiu a atenção para o aprendizado profundo foi a publicação de um artigo e código fonte que implementa a rede neural convolucional *AlexNet*. Os resultados mostravam uma indiscutível performance estatística para realizar a classificação de imagens e, a partir desse ponto, o aprendizado profundo passou a ser aplicado em diversas áreas, como visão computacional, processamento de imagens, computação gráfica, entre outros (Moacir A. Ponti; COSTA, 2017).

2.3.3 Redes neurais

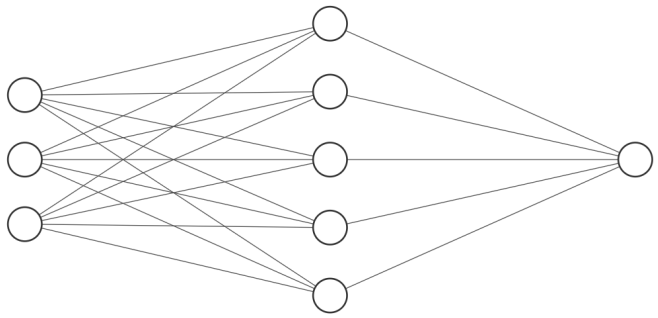
As redes neurais têm como objetivo processar informações com base na lógica de um cérebro humano, que tem um processamento completamente diferente de qualquer computador digital convencional (HAYKIN, 2008).

O cérebro tem a capacidade de organizar a sua estrutura, os neurônios, de modo que realize o processamento das informações com um paralelismo superior ao que os computadores são capazes atualmente (NIELSEN, 2013).

No conceito de redes neurais, as camadas têm o trabalho dos neurônios, no qual são montadas por meio de algoritmos de aprendizagem, formando camadas interligadas que filtram as informações de entrada com objetivo de aprender uma possível saída (HAYKIN, 2001).

Na Figura 5 pode-se observar um exemplo de uma Rede Neural, os dados são passados para uma camada de entrada que distribuem esses dados em camadas ocultas onde são aplicados os filtros de pesos e as funções de ativação, após isso é gerada a saída.

Figura 5 – Exemplo de camadas em uma rede neural



Fonte: Autor (2019)

Por terem como base o funcionamento do cérebro humano, as redes neurais possuem um processamento maciçamente paralelo. Tornando-as capazes de processar uma

grande quantidade de dados.

2.3.4 Redes neurais *Feed-Forward*

Esse é o tipo de rede mais comum de redes neurais em aplicações práticas. Sua arquitetura é formada pela primeira camada que é a entrada e a última camada que é a saída. Se houver mais de uma camada oculta, são chamadas de redes neurais profundas (ou *Deep Learning*). São calculadas uma série de transformações para atualização dos pesos na rede (NIELSEN, 2013).

Os neurônios tem atividades não lineares em cima das atividades da camada anterior (HAYKIN, 2008).

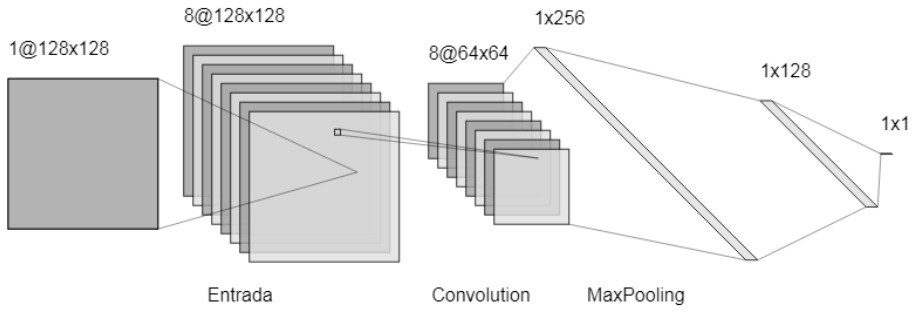
2.3.5 Redes neurais convolucionais

Inspiradas no modelo biológico da visão, as redes neurais convolucionais são redes neurais artificiais em que se aplica a operação de convolução em pelo menos uma das suas camadas (HAYKIN, 2008).

As redes neurais convolucionais se tornaram o novo padrão em visão computacional e são de fácil treinamento quando se têm uma grande quantidade de amostras rotuladas com diferentes níveis de classificação (CAROLINE *et al.*, 2017).

Dada uma entrada, são gerados os mapas característicos que são passados para as camadas subsequentes até gerar uma saída, conforme mostra a Figura 6.

Figura 6 – Exemplo de camadas de uma rede neural convolucional



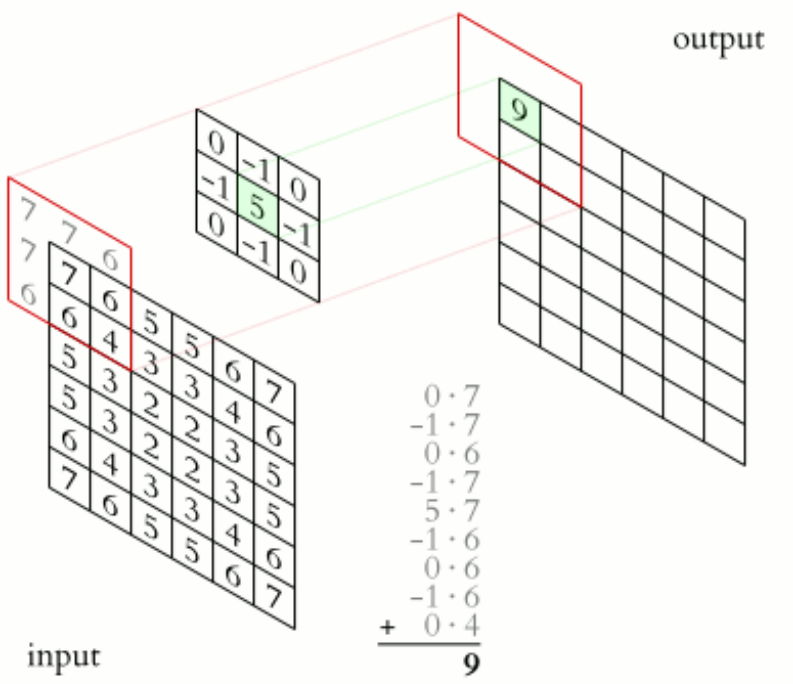
Fonte: Autor (2019)

Na Figura 6 pode-se observar que a rede tem como entrada uma imagem 128x128 *pixels*, após serem aplicadas as camadas de convolução, *pooling* e densa, a rede então gera uma única saída.

- Camada convolutiva:

  É comum aplicar inicialmente sobre os dados de entrada camadas de convolução, que são compostas de neurônios. Cada neurônio é conectado a uma parte da imagem de entrada, como se ligasse os neurônios a *pixels* da imagem e a cada conexão se atribui pesos. Essa combinação produz uma saída que é enviada para a camada seguinte(CAROLINE *et al.*, 2017). Esses pesos são representados como uma matriz que representam o filtro da camada conforme mostra as Figuras 7 e 8.

Figura 7 – Filtro da camada convolutiva dentro de um quadro 3x3



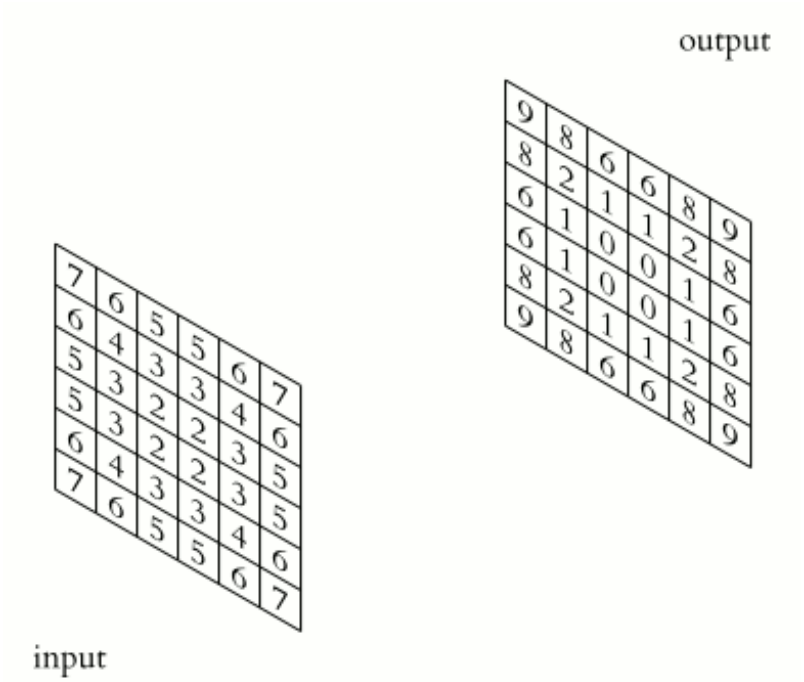
Fonte: (BRUCKNER; ROSEN; SPARKS, 2013)

Na Figura 7 há uma matriz 3x3 que aparece transparente, essa matriz é denominada filtro de características ou filtro convolucional. O tamanho e como esse filtro irá percorrer os *pixels* da imagem de entrada podem ser definidos quando se aplica a camada convolucional. Quando aplicada a camada convolucional irá gerar um *feature map*, ou mapa de característica. A quantidade de mapas de características que serão gerados em cada camada convolucional também pode ser definido na hora de criar o modelo. Cada um desses mapas gerados terão valores de *pixels*

diferentes, extraindo da imagem, características diferentes para cada filtro aplicado. Com o aprendizado da rede, será definido quais mapas de características e quais filtros tem maior importância na saída da rede (CAROLINE *et al.*, 2017).

A Figura 8 mostra o mapa de característica gerado após a aplicação do filtro convolucional que aparece na Figura 7. Cada mapa de característica que é gerado na camada convolucional, serve como entrada da próxima camada da rede neural convolucional (CAROLINE *et al.*, 2017).

Figura 8 – Filtro da camada convolutiva após completar todos os *pixels* da imagem de entrada



Fonte: (BRUCKNER; ROSEN; SPARKS, 2013)

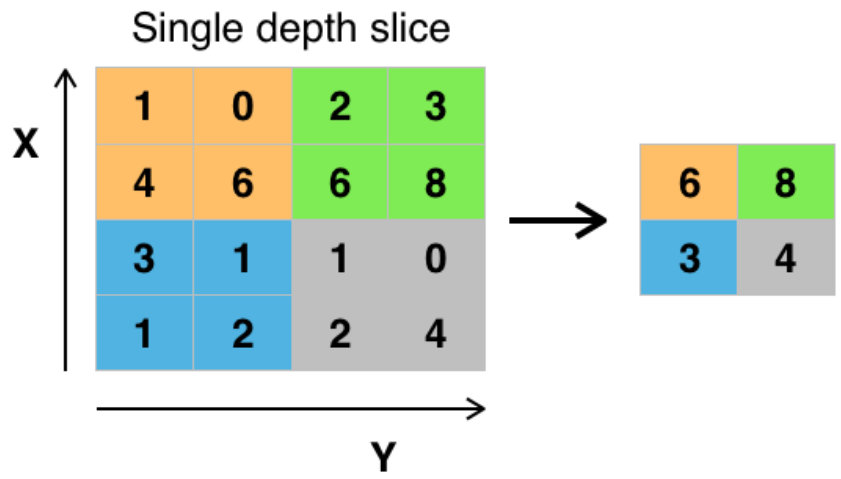
- Camada de *pooling*:

  A camada de agrupamento (*pooling*) tem como objetivo diminuir a dimensionalidade de dados na rede, importante para o quesito de eficiência e custo do treinamento e para criar uma invariância espacial (CAROLINE *et al.*, 2017). A camada funciona agrupando um conjunto de dados utilizando vários tipos de funções, a função de máximo é a principal utilizada.

  Na saída de uma camada convolucional, pode ser aplicado uma camada de *pooling* com tamanho 2x2 como mostra na figura 9. A função escolhe um valor para representar, no exemplo da Figura 9 é utilizado um *MaxPooling*, onde é escolhido o

maior valor nos *pixels* que foram aplicados o filtro de *pooling*. A camada de *pooling* não reduz a profundidade, reduz a altura e largura apenas (CAROLINE *et al.*, 2017).

Figura 9 – Filtro da camada de *pooling*

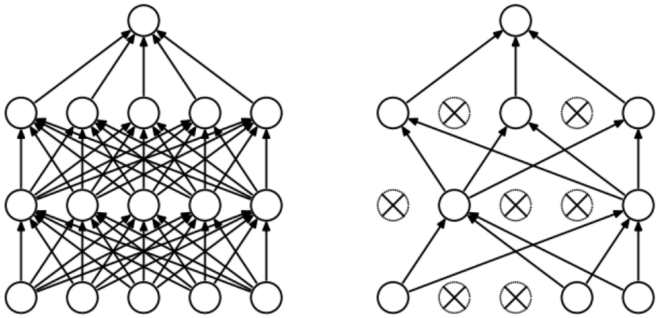


Fonte: (BRUCKNER; ROSEN; SPARKS, 2013)

O uso das camadas de *pooling* extraem das imagens as informações com maior importância, dependendo dos mapas de características em que são aplicadas. Com isso, reduzem as dimensões dos dados para a camada seguinte.

- Camada de *Dropout*:
  Método utilizado para redução de *overfitting*. Quando utilizado, ele faz um descarte aleatório de neurônios entre as camadas da rede conforme Figura 10. Esse processo tende a dificultar a co-adaptação de parâmetros. Com a interrupção aleatória na conexão entre as camadas, a co-adaptação dos parâmetros acaba se tornando mais difícil, resultado em um aumento na capacidade de generalização da rede (MAZZA, 2017).

Figura 10 – Utilização da camada *dropout* em uma rede neural



Fonte: (MAZZA, 2017)

- Camada *flatten*:

  Após aplicar as camadas convolutivas e camadas de *pooling*, as imagens de saída da última camada são transformadas pela camada *flatten* em um vetor de características. A partir desse ponto, a rede neural convolucional se comporta como uma neural tradicional, por exemplo, uma *multilayer perceptron* (HAYKIN, 2008).

- Camada densa:

  As camadas densas ou totalmente conectadas, são conectas por pesos. Por fim, a camada de saída da rede poderá ser uma previsão ou classificação. Se for uma classificação, poderá ser um valor indicando a probabilidade de presença de algum objeto. Na previsão, há apenas uma saída, com o valor previsto pela rede (HAYKIN, 2008).

2.4 Métodos de otimização

Nesse subcapítulo será abordado alguns métodos que foram utilizados para aprimorar a rede neural, como o gradiente descendente, retropropagação de erro, funções de ativação e otimizadores.

2.4.1 Retropropagação de erro

O primeiro passo nesse tipo de algoritmo é a inicialização dos pesos da rede. Se os pesos não forem setados manualmente, eles são inicializados aleatoriamente (HAYKIN, 2008).

O algoritmo é utilizado no treinamento de redes neurais multicamadas. Pode ser definido nos seguintes passos:

1. Inicialização: inicializa os pesos aleatoriamente ou usando algum método;
2. Processamento direto: com os pesos inicializado, então o algoritmo computa as ativações de todos os neurônios e calcula o erro;
3. Passo reverso: calcula os novos pesos para cada neurônio da rede, fazer o processo de modo retroativo, isto é, da camada de saída até a camada de entrada;
4. Teste de parada: testa se o critério para parada foi satisfeito, se sim, encerra o treinamento;
5. Repetição: se o critério de parada não foi satisfeito, volta ao passo 2.

(NIELSEN, 2013)

2.4.2 Funções de ativação

A função de ativação é utilizada para fazer a transformação não linear ao longo do sinal de entrada. Após essa transformação, a saída é enviada como entrada para a próxima camada de neurônios. Em uma rede sem função de ativação, a transformação dos pesos e bias se torna linear. Equações lineares são simples de resolver, mas são limitadas para resolução de problemas com maior complexidade. Essa rede sem função de ativação servirá apenas para modelos de regressão linear(HAYKIN, 2008).

As funções de ativação só tornam a propagação posterior possível se os gradientes sejam fornecidos juntamente com o erro para atualizar, os pesos e bias. Essas funções são componentes matemáticos incluídos na estrutura das redes neurais (NIELSEN, 2013).

Existem vários tipos de funções de ativação, dentre eles estão:

- *Step function*:

  A função de etapa binária é extremamente simples e mais utilizada para classificadores binários. Quando é necessário dizer sim ou não para uma classe somente. É definida pela Equação 1:

$$f(x) = \begin{cases} 1 & \text{se } x \geq 0 \\ 0 & \text{se } x < 0 \end{cases} \tag{1}$$

  (NIELSEN, 2013).

- Função linear:

  A derivada de uma função linear é constante, independente do valor de entrada x. Significa que toda vez que ocorrer o *backpropagation*, o gradiente seria o mesmo. É definida pela Equação 2.

$$f(x) = ax \tag{2}$$

  (NIELSEN, 2013).

- Sigmóide:

  A função sigmóide é amplamente utilizada atualmente. Varia de 0 a 1. É definida pela Equação 3:

$$f(x) = \frac{1}{1+e^{-x}} \tag{3}$$

  Sua maior vantagem sobre as funções anteriores, é por ser uma função não linear (NIELSEN, 2013).

- Tangente hiperbólica:

  A função tangente hiperbólica ou tanh, é semelhante a função sigmóide. Sua variação é de -1 a 1. É definida pela Equação 4:

$$f(x) = \frac{2}{(1+e^{(}-2x)) - 1} \tag{4}$$

  (NIELSEN, 2013).

- ReLU:

  ReLU é a função de ativação mais utilizada atualmente ao projetar redes neurais. É definida pela Equação 5:

$$f(x) = max(0,x) \tag{5}$$

  É uma função não linear. A principal vantagem da função é que ela não ativa todos os neurônios ao mesmo tempo. Isso significa que ela pode tornar a rede esparsa, eficiente e fácil para computação (NIELSEN, 2013).

- LeakyReLU:

  A função LeakyReLU foi proposta para solucionar um problema da ReLU, dando uma pequena inclinação α para a função na parte negativa do seu domínio. Sua função é definida na Equação 6:

$$f(x,\alpha) = max(\alpha x, x) \tag{6}$$

(NIELSEN, 2013).

- ELU:

  A função ELU, unidade linear exponencial, é dada pela Equação 7:

$$f(x,\alpha) = \begin{cases} x & \text{se } x \geq 0 \\ \alpha(e^x - 1) & \text{c.c.} \end{cases} \tag{7}$$

  Assim como a LeakyReLU, a ativação ELU resolve o problema da ReLU. Ela não satura na parte negativa do seu domínio. Na prática, as ELUs são extremamente eficazes (NIELSEN, 2013).

- Softmax:

  A função softmax também é um tipo de função sigmóide, mas sua utilidade principal é para problemas de classificação. 8:

$$\delta(z)_j = \frac{e^{(z_j)}}{\sum_{k=1}^{K} e^{(z_k)}} for j = 1, ..., K \tag{8}$$

  Ela retorna um vetor com probabilidades para cada classe de saída da rede neural (NIELSEN, 2013).

2.4.3 Otimizadores

Otimizadores são técnicas utilizadas para atualização dos pesos em uma rede, alguns deles são:

- Descida do gradiente:

  O gradiente descendente é um algoritmo utilizado para otimização, a sua função é encontrar o ponto com maior inclinação onde há a menor perda nos resultados da rede neural.

  O processo começa pelo cálculo da derivada $dF(x)/dx$ da função objetivo em relação à variável independente x. O próximo passo é atualizar o valor da variável x de acordo com a Equação 9:

$$x_i = x_{i-1} - \alpha_k \frac{dF(x_{i-1})}{dx} \tag{9}$$

  O processo é repetido até que a convergência seja atingida (RODRÍGUEZ, 2011).

- Adagrad:

  No otimizador Adagrad, as taxas de aprendizado são adaptadas em relação à frequência com que um parâmetro é atualizado durante o treinamento. Quanto mais atualizações esse parâmetro receber, menor será a taxa de aprendizado (DUCHI; HAZAN; SINGER, 2011).

- Adam:

  Algoritmo baseado em gradiente de primeira ordem de funções objetivas estocásticas. Os parâmetros são definidos quando a função é implementada no código. O método é simples de implementar, computacionalmente eficiente, com baixa requisição de memória e é adequado para problemas que são grandes em termos de dados e/ou parâmetros (KINGMA; BA, 2015).

- Descida do gradiente estocástica (SGD):

  O método é chamado de estocástico porque usa amostras selecionadas aleatoriamente para avaliar os gradientes. O método é considerado como uma aproximação da otimização de gradiente descendente (RODRÍGUEZ, 2011).

2.5 Validação cruzada

Encontrar o ponto ideal no treinamento de uma rede neural, é uma das etapas mais complicadas. Esse problema ocorre devido ao erro inicial do treinamento começar com um valor alto e cair rapidamente. Após essa queda rápida, o erro tende a cair lentamente, com tendência a atingir um mínimo local na superfície de erro (HAYKIN, 2008). Para que seja feita a identificação do ponto de parada do aprendizado, pode ser utilizada a regra de parada antecipada, com base na validação cruzada.

O método de validação cruzada é utilizado para validar o modelo, utilizando um conjunto de dados de teste separados do treinamento. Esse conjunto de dados para teste é separado para que possa estimar os parâmetros durante o treinamento da rede. Com base nos resultados obtidos com os dados de teste, o treinamento pode ser interrompido quando a curva de validação decresce a um erro mínimo, antes que a curva volte a crescer. (HAYKIN, 2008)

Existem vários métodos de validação cruzada, dentre eles estão:

- Método *holdout*:

  Este método consiste em dividir os dados em dois subconjuntos, um conjunto que

será utilizado para o treinamento e outro utilizado para teste. Usualmente são utilizados 70% para treinamento e 30% para teste (NIELSEN, 2013).

- Método *k-fold*:
  No método *k-fold*, os dados são divididos em k subconjuntos do mesmo tamanho. Um subconjunto é utilizado para teste e os outros k-1 subconjuntos são utilizados para estimação dos parâmetros. O processo é realizado k vezes, alternando o subconjunto de teste. Ao final do processo calcula-se a acurácia sobre os erros encontrados (CAMARGO, 2010).
- Método *leave-one-out*:
  Esse método é um caso específico do *k-fold*, onde o k é igual a quantidade de dados N. No processo são realizados N cálculos de erros, um para cada lado. Esse método tem um alto custo computacional, indicado para situações onde há uma menor quantidade de dados (CAMARGO, 2010).

2.6 Ambiente de desenvolvimento

Nesse subcapítulo será abordado alguns aspectos sobre a interface gráfica *Anaconda Navigator*, o ambiente de desenvolvimento *Jupyter Notebook*, as bibliotecas *Tensorflow Keras*. Além de uma breve descrição sobre CPU, GPU e TPU.

2.6.1 *Anaconda navigator*

O *Anaconda Navigator*[1] é uma interface gráfica de usuário (GUI) para *desktop*, incluída na distribuição *Anaconda*, que permite lançar aplicativos e gerenciar facilmente pacotes, ambientes e canais conda sem usar linha de comando. O *Navigator* pode procurar por pacotes no *Anaconda Cloud* ou em um Repositório *Anaconda local*. Está disponível para *Windows*, *macOS* e *Linux*.

[1] https://anaconda.org/anaconda/anaconda-navigator

2.6.2 *Jupyter notebook*

O *Jupyter Notebook*[2] é um aplicativo da *Web* de código aberto que permite criar e compartilhar documentos que contêm código ativo, equações, visualizações e texto narrativo. Os usos incluem: limpeza e transformação de dados, simulação numérica, modelagem estatística, visualização de dados, aprendizado de máquina, dentre outros.

2.6.3 *Tensorflow*

TensorFlow[3] é uma biblioteca de *software* de código aberto para computação numérica de alto desempenho. Sua arquitetura flexível permite a implantação de computação em várias plataformas (CPUs, GPUs, TPUs) e de *desktops* a *clusters* de servidores para dispositivos móveis e periféricos. Originalmente desenvolvido por pesquisadores e engenheiros da equipe *Google Brain* dentro da organização de inteligência artificial do *Google*, ele oferece suporte para aprendizado de máquina e aprendizado profundo, e o núcleo flexível de computação numérica é usado em outros domínios científicos.

CPU são unidades de processamento unitário, GPU são unidades de processamento gráfico e TPU são unidades de processamento tensorial. TPUs foram criados pela *Google*, tendo como finalidade o processamento com maior eficiência de redes neurais que utilizam a biblioteca *TensorFlow*.

2.6.4 *Keras*

Keras[4] é uma *Application Programming Interface* (API) de alto nível para redes neurais, escrito em *Python* e capaz de ser utilizada com a biblioteca *TensorFlow* em *backend*, *CNTK*, ou *Theano*. Foi desenvolvido com foco em permitir a experimentação rápida. Ser capaz de ir da ideia ao resultado com o menor atraso possível é a chave para fazer uma boa pesquisa.

Dentre as principais características da biblioteca, estão:

1. Permite a criação de protótipos fácil e rapidamente, aproveitando características como facilidade de uso, modularidade e extensibilidade;

[2] http://jupyter.org/
[3] https://www.tensorflow.org/
[4] https://keras.io/

2. Suporta redes convolucionais e redes recorrentes, bem como combinações das duas;
3. Possibilidade de processamento em CPU e GPU.

2.7 Trabalhos correlatos

O aumento na quantidade de dados para serem processados tem sido uma área de constante pesquisa. O *deep learning* é uma forma bastante utilizada para esse processamento (BEZERRA, 2016). Algumas destas pesquisas se destacam, dentre elas:

Neto (2017) criou um sistema para detectar precocemente tumores cerebrais utilizando RNC. A rede em questão obteve 94% de precisão para classificar imagens sadias e de 62% para imagens com a presença da doença. Para validação foi utilizada como base a média de acerto de 5 conjuntos utilizados na validação cruzada. A estrutura da rede possuía 3 camadas convolucionais. A primeira camada convolucional era constituída de 32 neurônios com uma janela de convolução 2D 3x3, com 2 camadas de *max-pooling* 2x2 na sequência. A segunda e terceira camadas eram semelhantes à primeira, diferindo apenas no número de neurônios, tendo sido utilizados 16 e 8 neurônios, respectivamente.

Marcomini (2013) criou uma aplicação para a classificação de nódulos na mama a partir de imagens de ultrassonografia, utilizando Redes Neurais Artificiais (RNA). A acurácia do modelo proposto por Marcomini foi de 80,92%. Para a classificação, utilizou-se a RNA *Multilayer Perceptron* (MLP), cuja interface possibilita a seleção do número de camadas intermediárias, número de neurônios, taxa de aprendizagem, tipo de validação, critério de parada e quantidade máxima de ciclos executados. Foi utilizada a MLP com retropropagação do erro e validação cruzada para validar a generalização do modelo.

Pereira *et al.* (2017) possuíam uma ideia semelhante à de Marcomini (2013), tendo feito a análise de ultrassonografias mamárias, porém utilizando RNC profundas. A maior acurácia do modelo proposto ocorreu para imagens que não possuíam lesões. Utilizou-se as arquiteturas *AlexNet* e *GoogleLeNet* na abordagem do problema. Os únicos parâmetros alterados das arquiteturas, foram o número de iterações fixado em 90 para ambas as redes, a taxa de aprendizado em 0,01 e reduzida a cada 30 interações a uma taxa gama de 0,1 e ambas as redes foram treinadas utilizando gradiente descendente estocástico (*Stochastic gradient descent* - SGD). Os resultados foram obtidos com base nas métricas de especificidade, sensibilidade e acurácia. A especificidade mostra quão bem o método é bom em reconhecer tecidos que não possuem massa, a sensibilidade mostra a relação da acurácia do método para acerto em massa e a acurácia demonstra a relação global de acerto. O

trabalho obteve os melhores resultados com a utilização da rede neural AlexNet sem a aplicação de filtros sobre a base de dados.

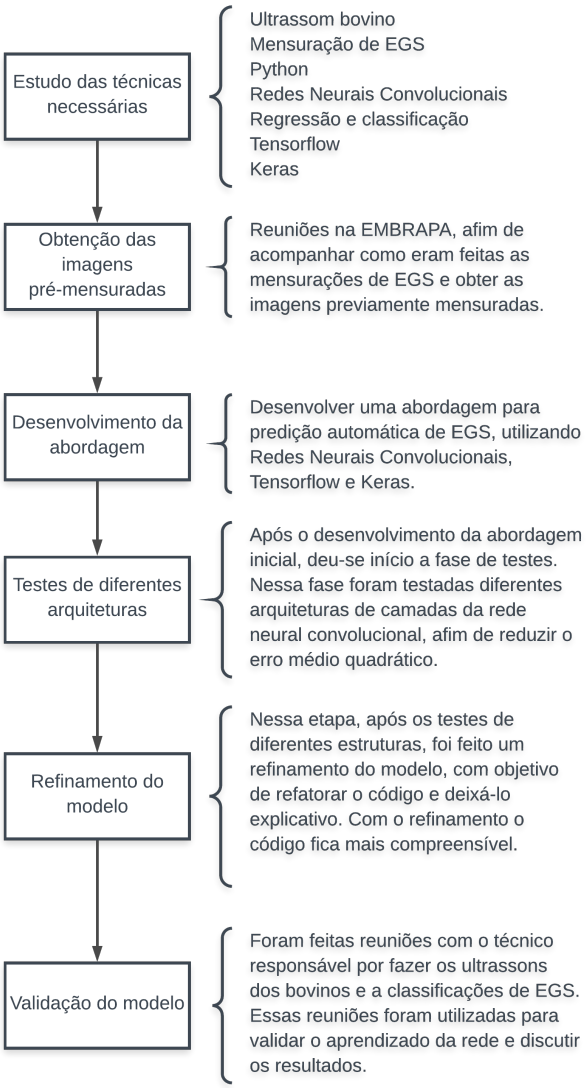
Carmen (2017) propôs um método automático para classificação de acabamento de gordura bovina em imagens digitais baseada no uso de RNA utilizando um modelo MLP com treinamento de retropropagação do erro. Foram utilizados os seguintes parâmetros para treinamento da rede: uma taxa de aprendizado de 0,4 com 10.000 iterações. A estrutura da rede ficou com uma camada de entrada com 384 nerônios, seguida por 3 camadas, com 44, 32 e 12 neurônios, respectivamente. Tanto nas camadas intermediárias como na camada de saída foi utilizada a função de ativação sigmóide. As métricas utilizadas para avaliação são as mesmas utilizadas no trabalho feito por Pereira *et al.* (2017). A rede obteve um resultado com 86% de acurácia na classificação do acabamento de gordura.

Analisando os trabalho correlatos encontrados, pode-se observar a importância do aprendizado de máquina para automatizar processos humanos. A utilização de redes neurais tendem a trazer bons resultados quando se tem acesso a um banco com grande quantidade de imagens rotuladas, como mostram os resultados dos trabalhos destacados. O uso de redes neurais convolucionais tendem a seguir esse aspecto, como no trabalho de Pereira *et al.* (2017) que utilizou a técnica para analisar ultrassonografias mamárias.

3 METODOLOGIA

Este capítulo apresenta a classificação desta pesquisa e detalha os procedimentos metodológicos que foram realizados ao longo do trabalho. A metodologia científica consiste no conjunto de procedimentos intelectuais e técnicos adotados para que se atinja o conhecimento (GIL, 2008). As etapas da metodologia podem ser vistas na Figura 11, que abrange os processos de estudo das técnicas necessárias, obtenção das imagens pré-classificadas, desenvolvimento de uma abordagem, testes de diferentes estruturas, refinamento do modelo e validação do modelo. Esses processos serão abordados no decorrer do texto.

Figura 11 – Fluxograma da metodologia utilizada



Fonte: Autor (2019)

3.1 Classificação do trabalho

De acordo com Gil (2008), as pesquisas podem ser classificadas quanto à finalidade, objetivos, abordagem, método e aos procedimentos técnicos. Quanto à finalidade, esta é uma pesquisa aplicada, pois o principal objetivo é construir uma abordagem capaz de resolver um problema real através da automatização, com utilização de técnicas de aprendizado profundo, de um modelo de classificação já existente e baseado em trabalho humano.

Quanto aos objetivos, trata-se de uma pesquisa explicativa pois tem o objetivo de fazer um estudo aprofundado sobre o funcionamento das classificações de acabamento de gordura subcutânea bovina e propor uma solução automática para esse processo, simulando a decisão do especialista humano para esta atividade.

Quanto à abordagem, este trabalho é quantitativo pois visa a construção de modelos computacionais que possam definir numericamente o nível de acabamento de gordura de bovinos com base na análise das imagens ultrassonográficas, com um nível de acurácia associado, permitindo inferir a capacidade de generalização dos modelos criados.

Quanto ao método, este trabalho pode ser classificado como hipotético-dedutivo, pois formulou-se a hipótese de que as Redes Neurais Convolucionais podem simular a decisão humana de classificação de imagens. Serão então definidos, construídos e testados os modelos de aprendizado de máquina para realizar esta atividade.

Quanto aos procedimentos técnicos, pode ser avaliada como uma pesquisa bibliográfica, pois foram estudadas técnicas e métodos em livros, artigos e trabalhos já publicados. Também pode ser classificada como um estudo de caso e uma pesquisa de campo, ela tem como objetivo estudar um problema isolado e procurar um método para solucioná-lo, para isso, foi necessário ir a campo para avaliar como funciona a coleta de dados e sua classificação, para que pudesse, então, propor uma abordagem para o problema.

3.2 Escolha das ferramentas

Primeiramente foi feito um estudo de quais ferramentas, métodos e técnicas seriam utilizadas no desenvolvimento do projeto. Após um levantamento referencial bibliográfico foi possível obter uma visão mais abrangente sobre o problema e quais poderiam ser os melhores métodos para que, então, pudesse ser proposta uma solução para a mensuração de acabamento de gordura subcutânea em bovinos. Levando em consideração

o estudo dos trabalhos correlatos, pode-se observar a vantagem na utilização de redes neurais convolucionais no uso de trabalho com imagens.

Optou-se por trabalhar com as bibliotecas *Tensorflow* e *Keras* utilizando a linguagem de programação *Python*, pois atualmente são as ferramentas mais utilizadas em redes neurais convolucionais (CARNEIRO; SILVA, 2017). Para facilitar o uso das ferramentas, foi utilizado o ambiente *Anaconda Navigator*. A finalidade na escolha das ferramentas foi a possibilidade de construir uma aplicação de alta performance sendo possível fazer o processamento com a utilização de uma GPU, obtendo uma melhora significava no tempo de treinamento.

3.3 Estudo das técnicas

Após serem analisadas quais ferramentas seriam utilizadas, foi feita uma pesquisa para estudo das mesmas.

Para a linguagem de programação *Python*, que não é abordada durante a graduação de Engenharia de Computação na Universidade Federal do Pampa, foi necessário fazer um curso online para aprender os conceitos da linguagem. Também foi necessário um estudo sobre as bibliotecas e o ambiente *Jupyter Notebook*.

Em paralelo ao aprendizado da linguagem de programação, também foram feitos dois cursos sobre *Deep Learning e Tensorflow*. Um curso com 8 módulos abordando conceitos de *Deep Learning* e outro curso com 22 módulos que aborda diferente métodos de redes neurais, utilizando as bibliotecas *Scikit Learn*, *Keras* e *Tensorflow*. Para a biblioteca *Tensorflow* foram abordados exemplos em *low-level* e *high-level*.

Foi necessário um estudo sobre os conceitos e técnicas de obtenção e classificação das imagens ultrassonográficas, esse estudo foi feito com base em leituras de artigos e reuniões com o coorientador para o esclarecimento de dúvidas. Também foram feitas algumas visitas à Empresa Brasileira de Pesquisa Agropecuária, para o acompanhamento da obtenção e classificação dos ultrassons, afim de melhorar o aprendizado sobre como era o processo de mensuração de EGS.

3.4 Obtenção das imagens

As imagens para o treinamento da rede neural foram disponibilizadas pelo Dr. Leandro Lunardini Cardoso, técnico de laboratório credenciado pela *Ultrasound Guidelines Council* - Estados Unidos, desde Julho de 2011. As imagens foram obtidas por ele através de ultrassonografias executadas nos bovinos da Empresa Brasileira de Pesquisa Agropecuária, e de vários outros criadouros, totalizando em torno de 8 mil imagens previamente avaliadas. Essas imagens estavam armazenadas em pastas separadas, de acordo com cada local onde foram realizados os ultrassons.

Junto com as imagens, foram disponibilizadas as planilhas, onde constam a identificação de cada animal e os valores de EGS, EGP e AOL, conforme Figura 12, que foram medidos pelo especialista.

Figura 12 – Estrutura das planilhas disponibilizadas

	A	B	C	
1	ID	AOL	EGS	EGP
2	9branco	53.8	2.4	6.7
3	8branco	56.0	1.4	2.4
4	7branco	54.4	2.8	5.7
5	6branco	46.8	3.9	5.0
6	4branco	49.1	3.3	4.8
7	3branco	59.2	4.7	7.7
8	31verde	51.8	2.6	3.3
9	2branco	65.4	3.0	3.8
10	29verde	55.1	4.8	9.6
11	24laranja	62.3	2.8	5.7
12	16laranja	57.4	5.0	9.3
13	14laranja	60.2	3.8	4.4
14	12branco	54.4	4.5	3.8
15	10branco	49.3	1.9	2.2

Fonte: Autor (2019)

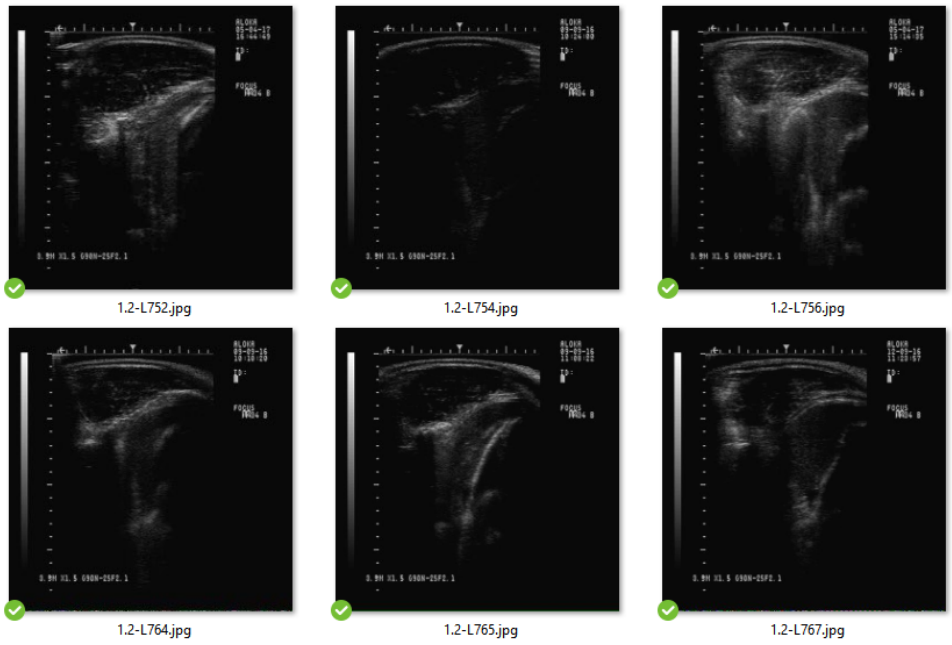
Os valores de EGS utilizados no treinamento/teste foram obtidos a partir das planilhas.

3.4.1 Tratamento dos dados

Primeiramente foi feito um algoritmo em *python*, para que as imagens fossem colocadas em uma única pasta. Nesse mesmo algoritmo, o nome das imagens era concatenado com o valor de EGS referente à imagem que constava na planilha, conforme Figura 13, onde o valor antes do hífen é referente ao EGS da imagem, e o nome após o hífen é a identificação do bovino em que foi realizado o ultrassom. As imagens ficaram estruturadas conforme a Figura 13.

O valor de EGS foi concatenado junto ao nome da imagem, para que no momento de utilizar as imagens no treinamento, já seja possível utilizar o valor de EGS referente a cada imagem.

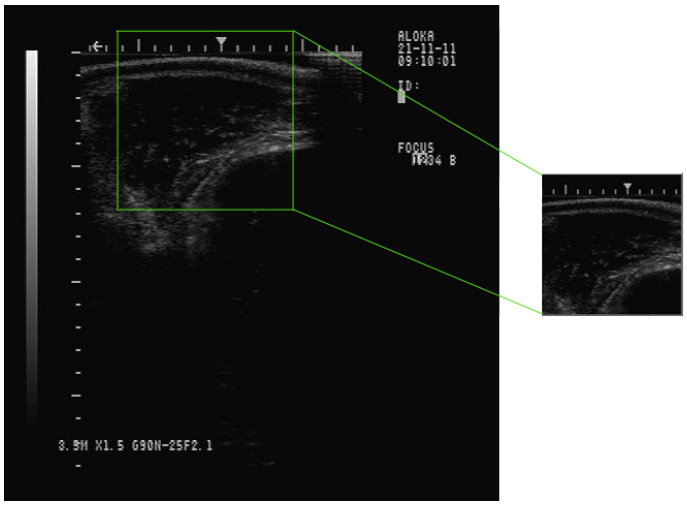
Figura 13 – Estrutura da pasta onde estão as imagens para treinamento



Fonte: Autor (2019)

Também foi utilizado um método para recortar partes da imagem que não eram importantes. Esse recorte nas imagens foi feito apenas para melhorar o desempenho de treinamento da rede, tendo como entradas imagens 180x180 *pixels* ao invés de ter imagens 480x480. Esse recorte nas imagens é exemplificado pela Figura 14.

Figura 14 – Corte feito nas imagens para diminuir o tempo de processamento do treinamento



Fonte: Autor (2019)

Os testes que foram feitos utilizando imagens com tamanho 480x480 *pixels* obtiveram os mesmo resultados que os testes utilizando as imagens cortadas. O recorte foi utilizado apenas para reduzir a quantidade de neurônios na primeira camada convolucional, resultando em uma melhora no tempo de treinamento da rede.

O recorte nas imagens foi feito, após alguns testes visuais, focando a área que abrangesse o local onde é feita a medida de EGS pelo especialista, para que não houvesse perca de informações importantes para a rede. Esse recorte foi feito automaticamente no algoritmo de treinamento, utilizando uma técnica da biblioteca OpenCV. O recorte foi feito em uma área de 180x180 *pixels*, pois em algumas imagens o local de mensuração do EGS era diferente de outras, dependendo de como o equipamento de obtenção do ultrassom foi posicionado.

3.5 Desenvolvimento da abordagem

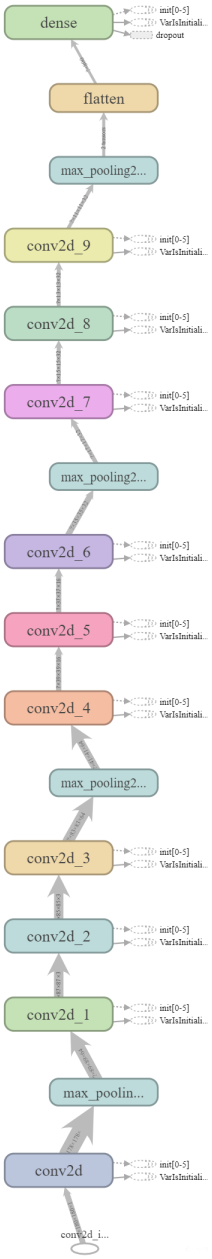
A primeira abordagem construída foi utilizando um modelo de classificação. Esse modelo foi construído para fazer classificações de 3 classes de níveis de EGS, de 0 até 2, 2 até 4 e para valores maiores que 4. Essa divisão foi feita, para que houvesse um equilíbrio na quantidade de imagens para cada classe.

O modelo de classificação não foi satisfatório para o projeto. Apesar da acurácia não ser tão baixa, a classe que era prevista tinha uma faixa de valores muito alta. A

opção de criar mais classes também ficaria inviável por se tratar de poucas imagens para alguns níveis de EGS. Após chegar a conclusão que o modelo não solucionaria o problema proposto, então deu-se a construção de um modelo de regressão.

O primeiro modelo de regressão foi construído utilizando apenas a biblioteca *Tensorflow*, sem utilização da biblioteca *Keras*. Nesse modelo, foram criadas as camadas convolucionais, camadas de *pooling* e as camadas densas. A arquitetura da rede utilizada nos primeiros testes é apresentada na Figura 15.

Figura 15 – Arquitetura da rede neural convolucional utilizada nos primeiros testes



Fonte: Autor (2019)

No algoritmo de treinamento foram utilizados métodos do *Tensorboard* para salvar as etapas de treinamento e os diagramas da arquitetura utilizada.

Após alguns testes utilizando a abordagem desenvolvida com *Tensorflow*. Então foi construído um modelo utilizando *Keras*, que é uma biblioteca de alto nível para construção de modelos utilizando *Tensorflow*, mas torna a arquitetura mais intuitiva para fazer os testes.

3.5.1 Teste de diferente arquiteturas

Após terminar a construção das primeiras arquiteturas, foram feitos diferentes testes, os quais geraram arquivos de *log* para documentar a arquitetura, os resultados e salvar o modelo final.

Os testes de diferentes arquiteturas, foram feitos para analisar os resultados e, então, poder escolher qual a arquitetura melhor se enquadra para o projeto, obtendo um menor erro médio quadrático possível dentro dos recursos disponíveis.

Foram testadas diferentes quantidades de camadas, neurônios, funções de ativação, número de épocas, *batch size*, entre outros parâmetros da rede.

3.5.2 Refinamento do modelo

Após terminar os testes, o modelo passou por um refinamento. Foi feito uma refatoração do código, afim de melhorar a estrutura interna do código sem alterar seu comportamento, tornando-o mais intuitivo e explicativo.

3.5.3 Validação do modelo

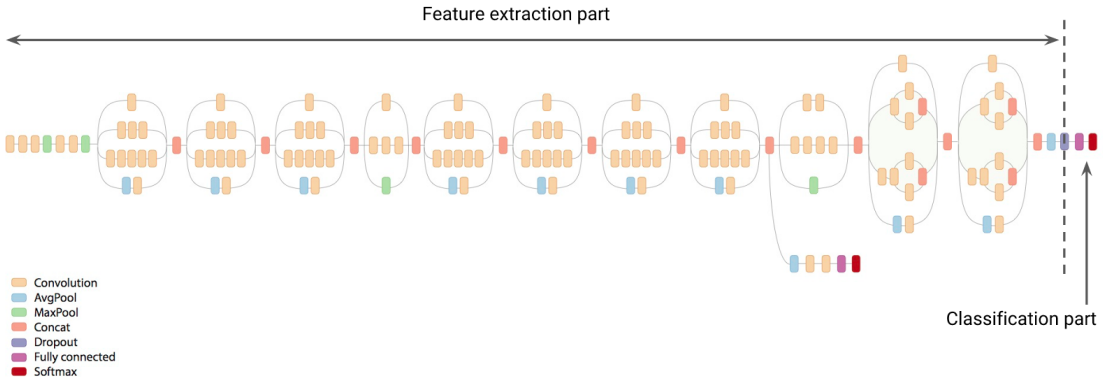
Para validação, foi utilizado o método de validação cruzada, *holdout*. As imagens foram divididas em conjuntos de treinamento e teste, como há uma grande quantidade de imagens, foi utilizado 80% para treinamento e 20% para teste. O conjunto de treinamento é utilizado para o treino da rede neural convolucional e o conjunto de teste é utilizado para obter as métricas da rede e validar o modelo.

Para validar os resultados também foram feitas reuniões com o especialista responsável pelos ultrassons, para discutir sobre os resultados da rede.

4 RESULTADOS

Após um período de estudos sobre arquiteturas de classificação de imagens com o uso de *TensorFlow* e tentativas de implementação, foi utilizado um modelo de classificação de imagens com a utilização da arquitetura *Inception V3*, construída com a biblioteca *Tensorflow*, e um conjunto de imagens com 8630 amostras. A arquitetura foi escolhida, por se tratar de uma das principais arquiteturas utilizadas para classificação (SZEGEDY *et al.*, 2016). Na figura 16 pode-se observar a arquitetura de camadas que o modelo *Inception V3* é composto.

Figura 16 – Arquitetura de camadas do modelo *Inception V3*



Fonte: https://codelabs.developers.google.com/codelabs/cpb102-txf-learning/

Com a utilização desse modelo, pode-se obter uma acurácia de 70,9%, que não seria o ideal para o trabalho. A classificação trata-se de um modelo, em que as imagens são divididas em 3 rótulos. Nesse teste foram utilizados rótulos "0-2", que contempla imagens que tem um nível de EGS de 0 até imagens com níveis menores que 2, "2-4"contempla imagens com níveis de EGS de 2 até níveis menores que 4, e "4+"que contempla imagens que tem níveis de EGS maiores ou iguais a 4. Essa divisão foi feita com base na quantidade de imagens que estava disponível no banco de imagens, tornando as classes equilibradas. Também foi necessário dividir os dados porque o algoritmo utilizado, trata-se de um modelo de classificação.

Como a classificação não solucionou o problema do projeto, a melhor opção foi aplicar um modelo de regressão. Após o estudo de diferentes arquiteturas de regressão e seu funcionamento, deu-se a construção de um modelo utilizando a biblioteca *Tensorflow*. Os primeiros testes utilizando regressão não obtiveram bons resultados, com erro médio absoluto maior que 10, tornando inviável a utilização do modelo treinado, pois um

erro médio dessa proporção não poderia ser utilizado para predizer níveis de EGS, pois não teria nenhuma precisão. Os detalhes da arquitetura utilizada nesse primeiro modelo podem ser vistos na Tabela 1.

Tabela 1 – Arquitetura inicial da rede neural convolucional

Camadas	Filtros / Kernel	Ativação	Entrada	Retorno	Batch_size	Strides
Entrada	-	-	480x480x1	480x480x1	8	-
convolucional	5x5	Relu	480x480x1	480x480x32	8	2
Pooling	2x2	-	480x480x32	240x240x32	8	2
convolucional	5x5	Relu	240x240x32	240x240x32	8	2
Pooling	2x2	-	240x240x32	120x120x32	8	2
convolucional	5x5	Relu	120x120x32	120x120x32	8	2
Pooling	2x2	-	120x120x32	60x60x32	8	2
convolucional	5x5	Relu	60x60x32	60x60x32	8	2
Pooling	2x2	-	60x60x32	30x30x32	8	2
convolucional	5x5	Relu	30x30x32	30x30x32	8	2
Pooling	2x2	-	30x30x32	15x15x32	8	2
Flattening	-	-	15x15x32	7200	8	-
Densa	-	Relu	7200	2048	8	-
Dropout 0,2	-	-	2048	2048	8	-
Densa	-	Relu	2048	1024	8	-
Dropout 0,2	-	-	1024	1024	8	-
Densa	-	Relu	1024	512	8	-
Dropout 0,2	-	-	512	512	8	-
Densa	-	Relu	512	1	8	-

Fonte: Autor(2019)

As escolhas dos parâmetros foram utilizadas para testes, o uso do *kernel* 5x5 na camada convolucional é o mais utilizado e com melhores resultados segundo (PEREIRA *et al.*, 2017). Na camada de *pooling* foi utilizado um *kernel* 2x2 e 2 de *stride*, por se tratar de imagens que não tem uma grande dimensão, um *kernel* maior acaba se tornando inviável. A quantidade de camadas densas e o número de neurônios, foram escolhidos para testes, afim de encontrar melhores soluções. O *dropout* de 0.2 foi usado para evitar

que houvesse um *overfitting* no treinamento.

Com a utilização do otimizador *AdamOptimizer* e taxa de aprendizado de 0.001. O treinamento, para 10 mil *steps* utilizando essa arquitetura, obteve um erro médio absoluto nos dados de teste de 10.233851, com um tempo médio de 1.4s para cada *step*.

Para a regressão foram utilizadas 6904 imagens no treinamento da rede neural convolucional e 1726 imagens para testes e validação da rede.

Após algumas alterações no modelo inicial, o dados de teste obtiveram um erro médio absoluto menor. Um dos problemas no teste inicial foi utilizar a função de ativação *ReLu* na camada de saída, sendo que para regressão a função de ativação deveria ser linear (NIELSEN, 2013). Com as alterações feitas, a arquitetura ficou conforme a Tabela 2.

Tabela 2 – Arquitetura da rede neural convolucional após modificações

Camadas	Filtros / Kernel	Ativação	Entrada	Retorno	Batch_size	Strides
Entrada	-	-	480x480x1	480x480x1	8	-
convolucional	5x5	Relu	480x480x1	480x480x32	8	2
Pooling	2x2	-	480x480x32	240x240x32	8	2
convolucional	5x5	Relu	240x240x32	240x240x32	8	2
Pooling	2x2	-	240x240x32	120x120x32	8	2
convolucional	5x5	Relu	120x120x32	120x120x32	8	2
Pooling	2x2	-	120x120x32	60x60x32	8	2
convolucional	5x5	Relu	60x60x32	60x60x32	8	2
Pooling	2x2	-	60x60x32	30x30x32	8	2
convolucional	5x5	Relu	30x30x32	30x30x32	8	2
Pooling	2x2	-	30x30x32	15x15x32	8	2
convolucional	5x5	Relu	15x15x32	15x15x32	8	2
Pooling	2x2	-	15x15x32	7x7x32	8	2
Flattening	-	-	7x7x32	1568	8	-
Densa	-	Relu	1568	1024	8	-
Dropout 0,2	-	-	1024	1024	8	-
Densa	-	Linear	1024	1	8	-

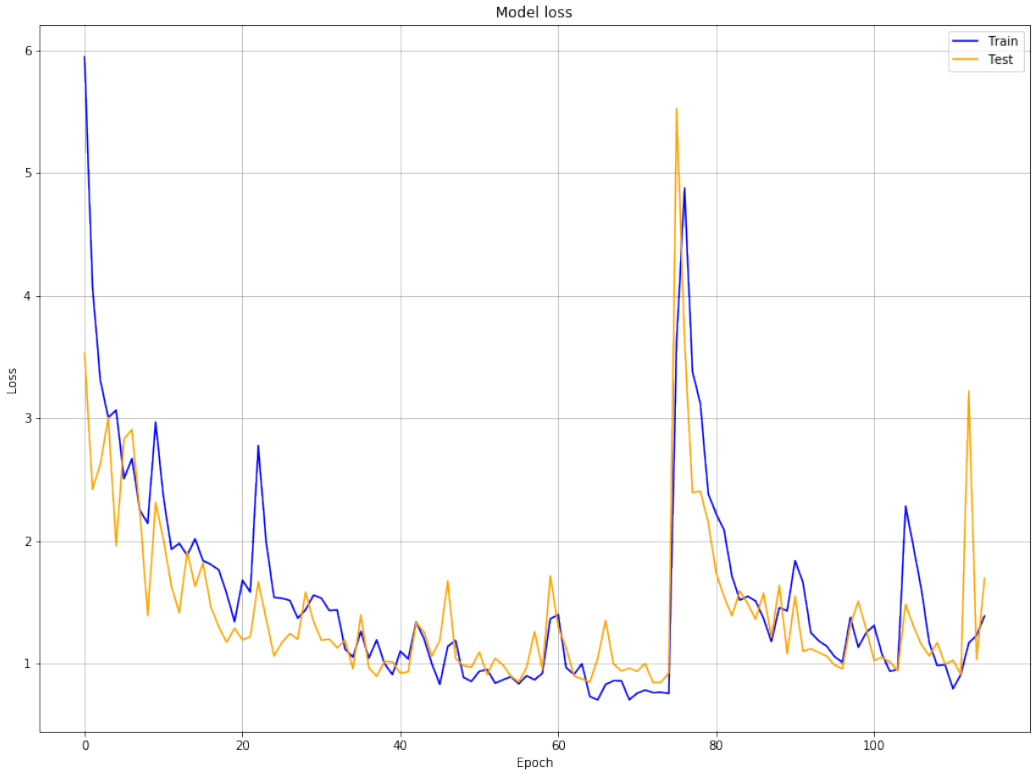
Fonte: Autor(2019)

Para esse modelo também foi utilizado o otimizador *AdamOptimizer* com taxa de

aprendizado de 0.001. O treinamento para 12 mil *steps* obteve um erro médio absoluto de 1.5459172 com um tempo médio de 1.7s para cada *step*.

O gráfico de erro do treinamento pode ser observado na Figura 17.

Figura 17 – Gráfico de perda do treinamento de um modelo inicial



Fonte: Autor (2019)

Alguns picos de erro ocorrem no gráfico, entre as épocas 70 e 80 da Figura 17, devido às imagens que são enviadas para treinamento. Em cada situação que as imagens são enviadas para execução do treinamento, é enviado um pacote, *batch size*, de 32 imagens aleatórias do conjunto de treinamento, nessas imagens podem ocorrer erros maiores, o quê resulta em picos mais altos de erro no gráfico em diferentes etapas do treinamento. Outra justificativa para esses elevados pontos do gráfico, é quando a atualização de pesos é feita com finalidade de não estagnar em mínimos locais.

Os dois resultados apresentados na Tabela 1 e Tabela 2, são partes do processo de testes iniciais do projeto. Através desses testes foi possível um melhor entendimento sobre o funcionamento dos algoritmos de treinamento utilizados no projeto.

Com base nos testes e no conhecimento obtido, pode-se fazer algumas alterações no algoritmo, para que os treinamentos retornassem resultados com erros cada vez menores, tornando viável sua utilização em processos reais. Satisfazendo o objetivo proposto

do trabalho de pesquisa.

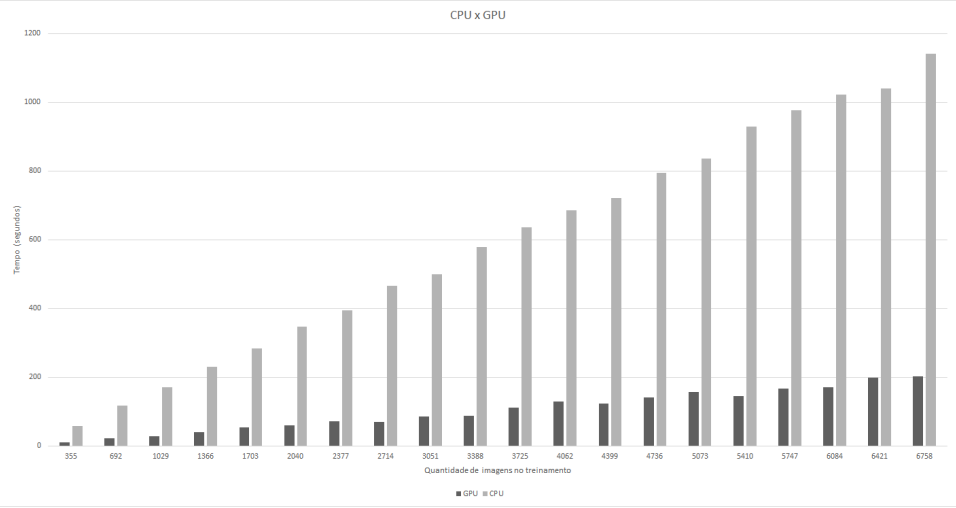
Para obter um melhor desempenho no treinamento da Rede Neural Convolucional, foi testado e analisado o uso de imagens de ultrassom recortadas, transformando as imagens iniciais que eram 480x480 *pixels* em imagens 180x180 *pixels*. Esse recorte foi feito com foco no quadro da imagem onde é feita a medida de EGS pelo especialista conforme a Figura 14.

Nessa etapa de testes também foi construído um modelo utilizando *Keras*, que torna mais eficiente a alteração da arquitetura de rede. Essa biblioteca também trás uma grande disponibilidade de ferramentas como o *Tensorboard*, que é utilizado para visualização do aprendizado da rede. A construção do modelo com *Keras* foi feito pensando na elaboração dos testes.

Com a utilização de imagens recortadas, pode-se observar que o aprendizado da máquina não teve alterações. Porém o tempo de processamento caiu drasticamente, deixando o processo de treinamento mais eficiente e tornando possível uma carga maior de testes.

Além dos métodos citados para o aperfeiçoamento do treino da rede, foi utilizado um ambiente de alto desempenho com uso de uma GPU *NVidia GeForce* 910M com 2GB de memória dedicada. Foram feitos alguns testes para comparar o uso de CPU e GPU no treinamento da rede, conforme a Figura 18. A comparação da GPU foi feita com um processador *Intel Core* i5-5200U 2.20GHz.

Figura 18 – Gráfico de comparação do tempo de execução de treinamento com CPU e GPU

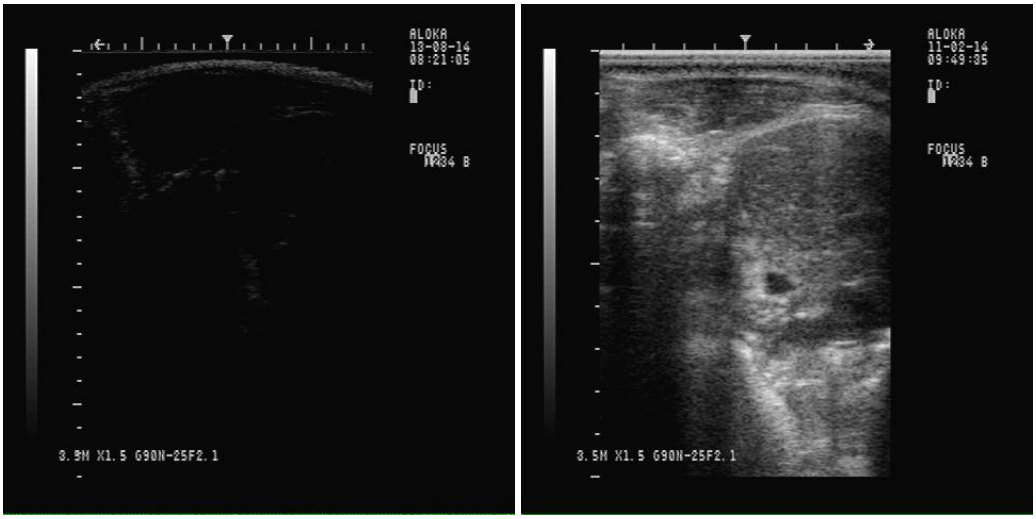


Fonte: Autor (2019)

As modificações na estrutura da rede, para que melhores resultados pudessem ser obtidos, foram feitas com alterações na quantidade e estrutura de camadas, tipo de otimizador utilizado, funções de ativação e taxa de aprendizado, afim de chegar nas menores médias de erro absoluto nos dados de teste.

Outra melhoria feita para o treinamento, foi na separação de algumas imagens com erro. Após uma reunião com o especialista, foi discutido sobre algumas imagens que estavam dando os erros mais altos na predição da rede neural convolucional. Após análise do especialista, foi aconselhado retirar essas imagens do banco utilizado no projeto, pois se tratavam de imagens com algum erro na classificação manual, ou então imagens defeituosas, com algum tipo de ruído. Exemplos dessas imagens podem ser observados na Figura 19. Após a retirada dessas imagens defeituosas, ficaram disponíveis 7951 imagens para utilização no projeto.

Figura 19 – Ultrassons muito escuros ou com ruídos, defeitos causados na hora de capturar a imagem

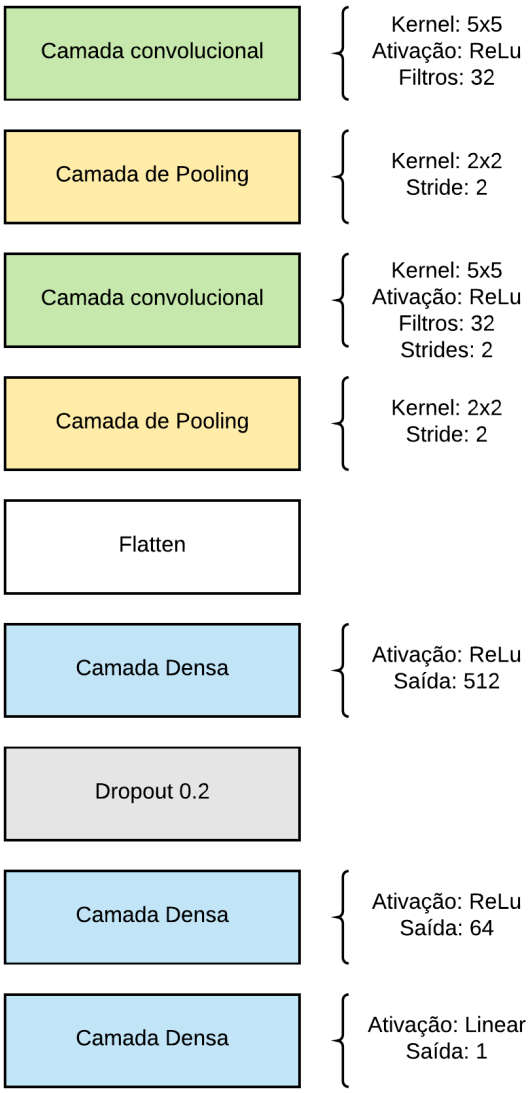


Fonte: (Empresa Brasileira de Pesquisa Agropecuária, 2018)

Diferentes modelos de arquitetura foram construídos para testes. Desses modelos construídos, 5 obtiveram melhor desempenho.

O modelo 1 é composto por 2 camadas convolucionais com *kernel* 5x5, 2 camadas de *pooling* com *kernel* 2x2, 2 camadas densas ocultas, com 512 e 64 neurônios, respectivamente, e 1 camada densa de saída. O modelo pode ser visto na Figura 20

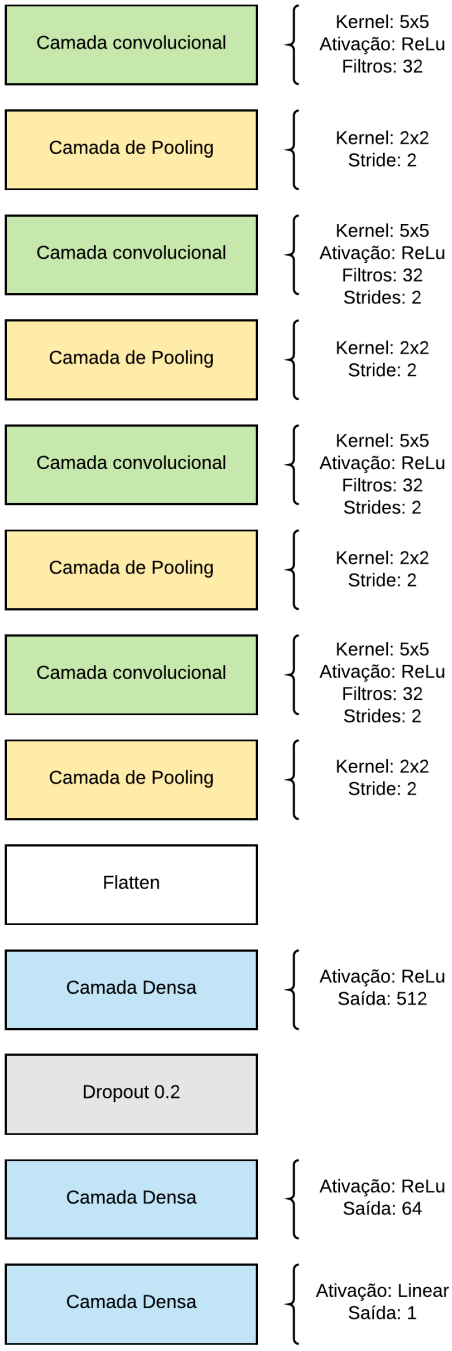
Figura 20 – Modelo 1



Fonte: Autor (2019)

O modelo 2 é composto por 4 camadas convolucionais com *kernel* 5x5, 4 camadas de *pooling* com *kernel* 2x2, 2 camadas densas ocultas, com 512 e 64 neurônios, respectivamente, e 1 camada densa de saída. O modelo pode ser visto na Figura 21

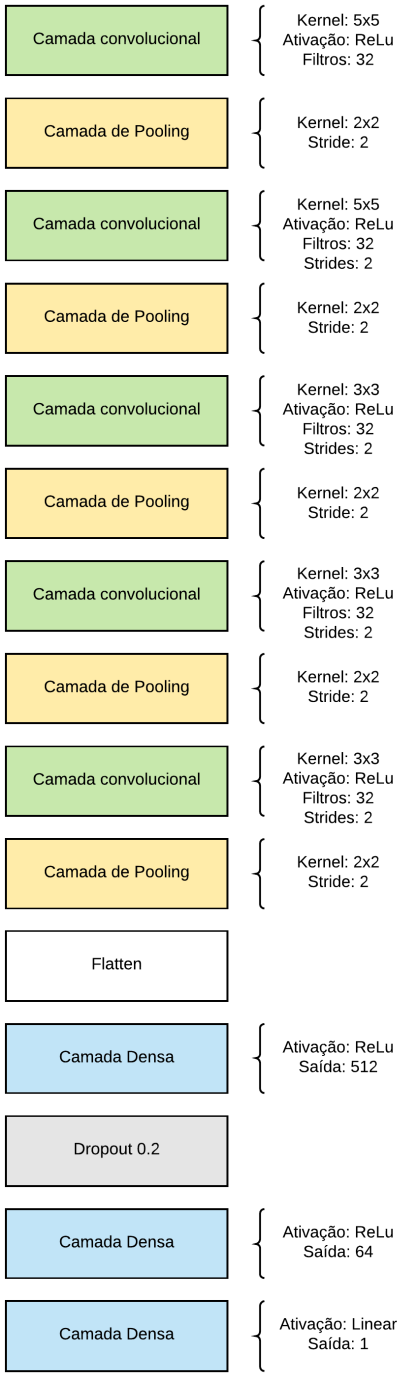
Figura 21 – Modelo 2



Fonte: Autor (2019)

O modelo 3 é composto por 5 camadas convolucionais com 2 camadas convolucionais utilizando *kernel* 5x5 e 3 camadas convolucionais utilizando *kernel* 3x3, 5 camadas de *pooling* com *kernel* 2x2, 2 camadas densas ocultas, com 512 e 64 neurônios, respectivamente, e 1 camada densa de saída. O modelo pode ser visto na Figura 22

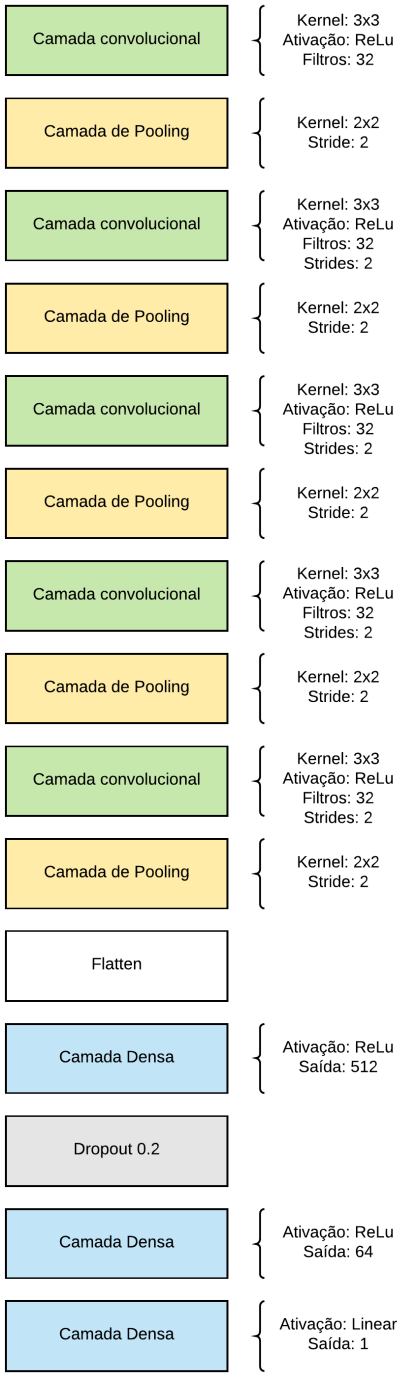
Figura 22 – Modelo 3



Fonte: Autor (2019)

O modelo 4 é composto por 5 camadas convolucionais com *kernel* 3x3, 5 camadas de *pooling* com *kernel* 2x2, 2 camadas densas ocultas, com 512 e 64 neurônios, respectivamente, e 1 camada densa de saída. O modelo pode ser visto na Figura 23

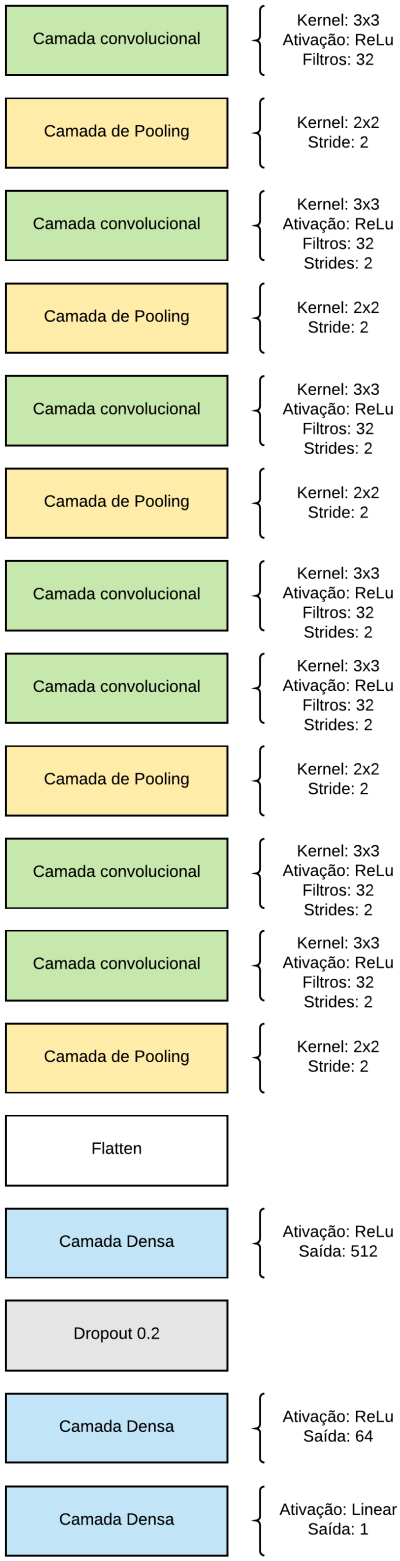
Figura 23 – Modelo 4



Fonte: Autor (2019)

O modelo 5 é composto por 7 camadas convolucionais com *kernel* 3x3, 5 camadas de *pooling* com *kernel* 2x2, 2 camadas densas ocultas, com 512 e 64 neurônios, respectivamente, e 1 camada densa de saída. O modelo pode ser visto na Figura 24

Figura 24 – Modelo 5

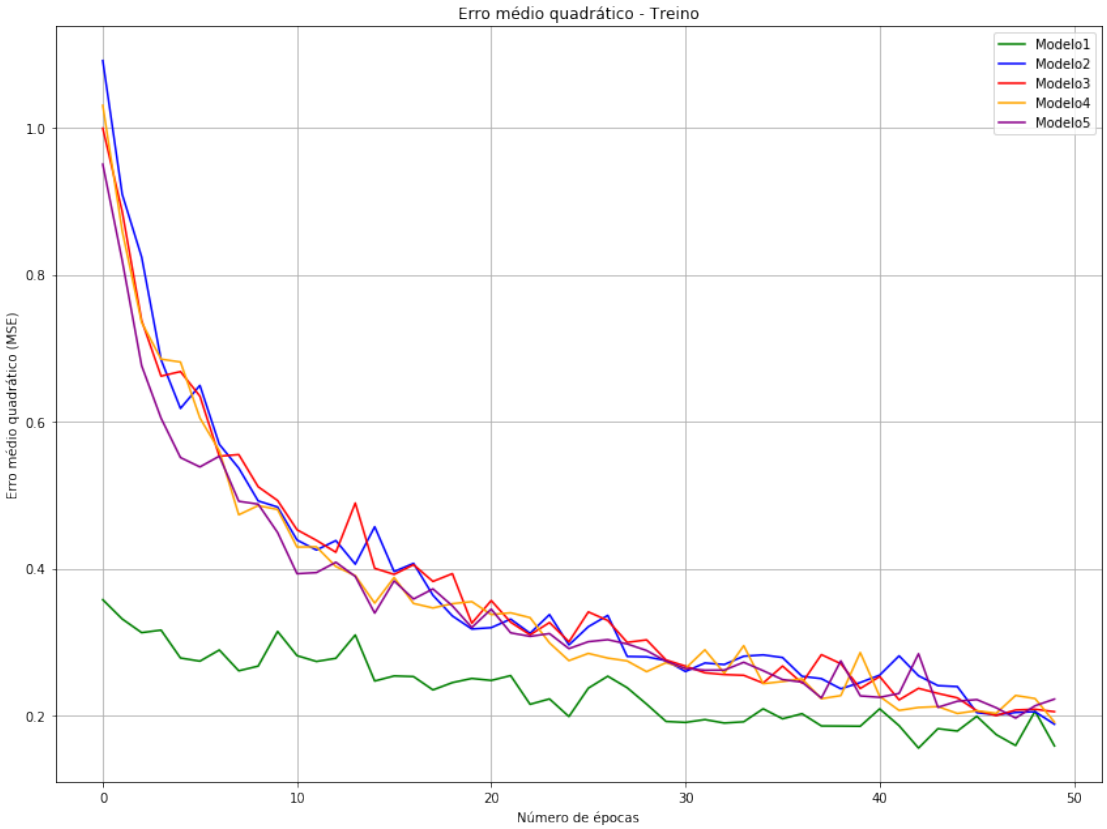


Fonte: Autor (2019)

Os resultados obtidos nos 5 modelos podem ser vistos nos gráficos de erro médio

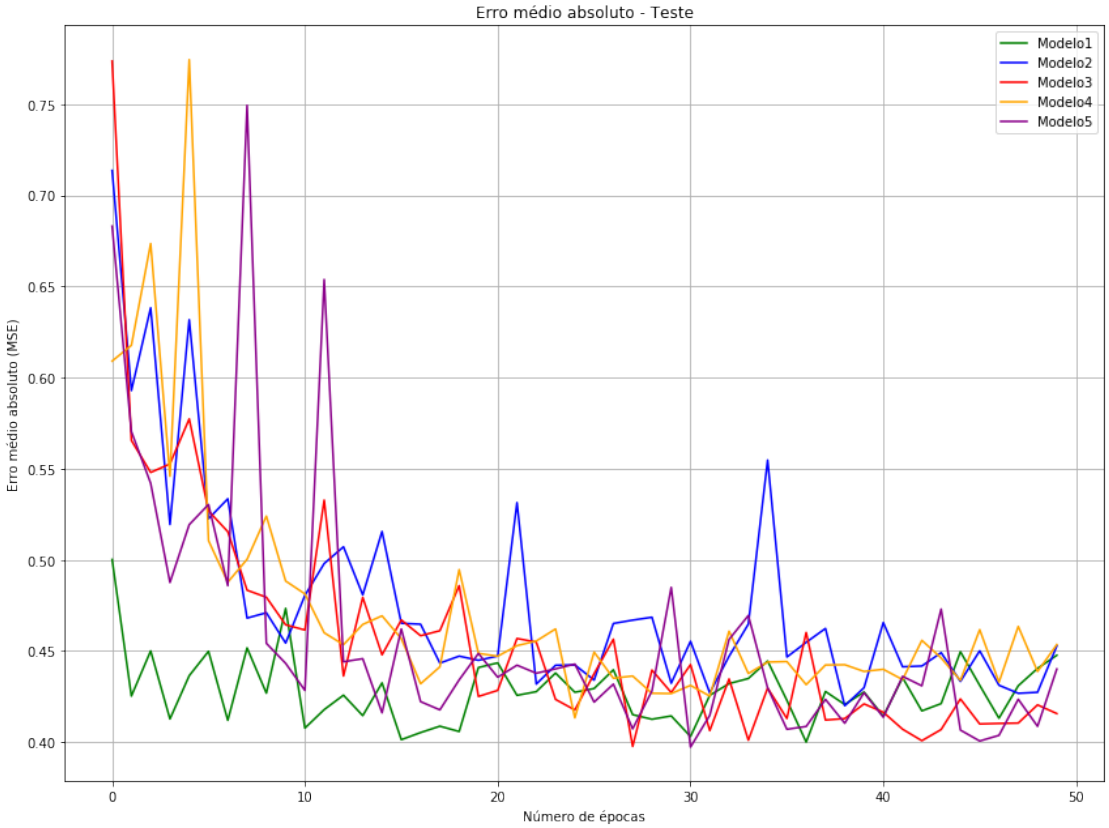
quadrático dos dados de treinamento e erro médio absoluto dos dados de teste, Figura 25 e Figura 26, respectivamente. Erro médio quadrático é a função utilizada para calcular a perda de predição obtida nos dados de treinamento da rede neural convolucional, calculada pela média de erro da predição de EGS elevada ao quadrado. Com base no erro médio quadrático a rede neural convolucional aplica as alterações de pesos com uso de retropropagação do erro. O erro médio absoluto, é a média de erros na predição dos valores de EGS obtidos, utilizando os dados de teste.

Figura 25 – Gráfico do erro médio quadrático durante o treinamento com diferentes arquiteturas de camadas



Fonte: Autor (2019)

Figura 26 – Gráfico do erro médio absoluto para os dados de teste com diferentes arquiteturas de camadas



Fonte: Autor (2019)

Após os testes, com a análise dos resultados, verificou-se que os melhores resultados obtidos no treinamento foi dos modelos 3 e 5, com erro médio absoluto de 0.397 e 0.389, respectivamente, para os dados de teste, os outros modelos começaram a decorar os dados de treino, perdendo assim, a capacidade de generalização.

Diferentes testes foram feitos entre os modelos 3 e 5, com o modelo 5 sempre obtendo uma melhor eficiência. Após concluir que o modelo 5 foi o que melhor se enquadrou para solucionar o projeto, então os testes passaram a ser feitos com foco no modelo 5.

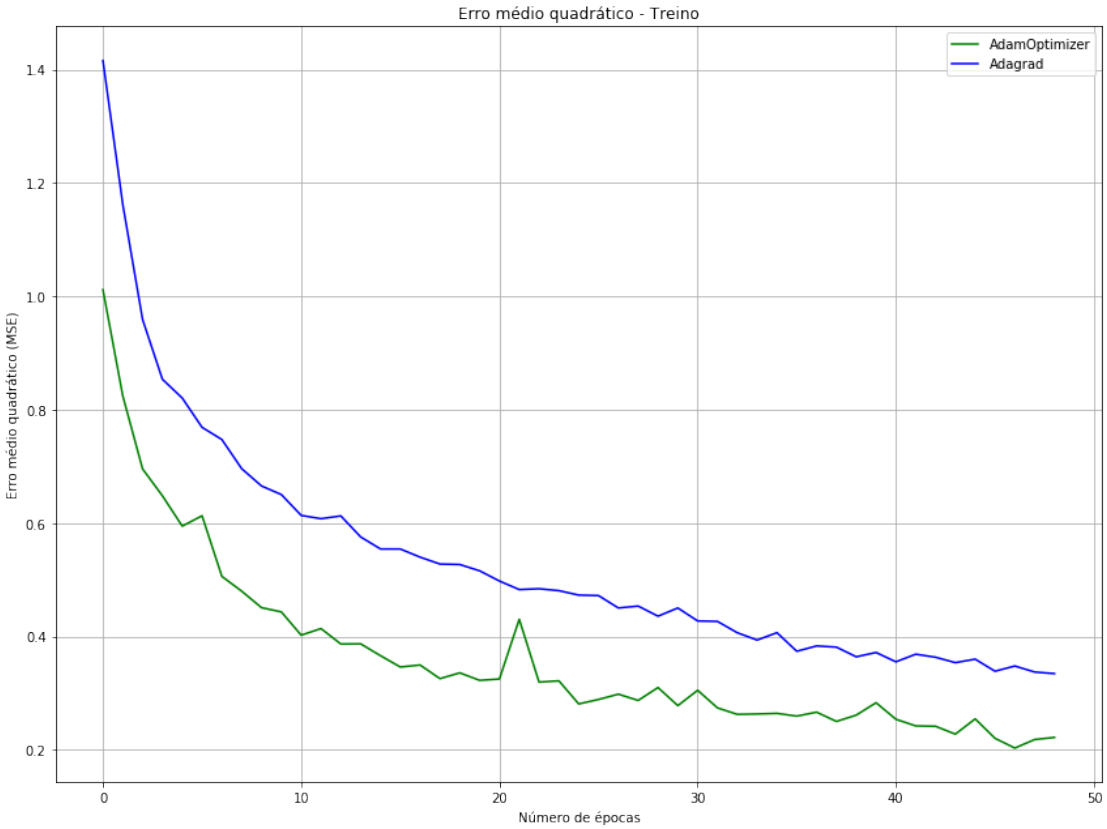
Algumas alterações foram feitas, utilizando a arquitetura do modelo 5. Essas alterações tornaram possível obter um resultado final satisfatório para o projeto com a utilização do modelo 5. O erro médio absoluto encontrado após as modificações foi de 0.38, com uma correlação de 0.97 com os dados reais.

A correlação utilizada no projeto, é o coeficiente de correlação de Pearson ou coeficiente de correlação produto-momento. Esse coeficiente mede o grau da correlação

linear entre duas variáveis quantitativas. Seu índice adimensional pode variar entre -1.0 a 1.0, que especifica a intensidade de correlação entre dois conjuntos de dados. Um coeficiente de correlação de 1.0 significa uma correlação perfeita positiva entre as duas variáveis, -1.0 significa uma correlação perfeita negativa. um coeficiente de correlação 0 significa que as variáveis não tem nenhum tipo de correlação linear (RODGERS; NICEWANDER, 2006).

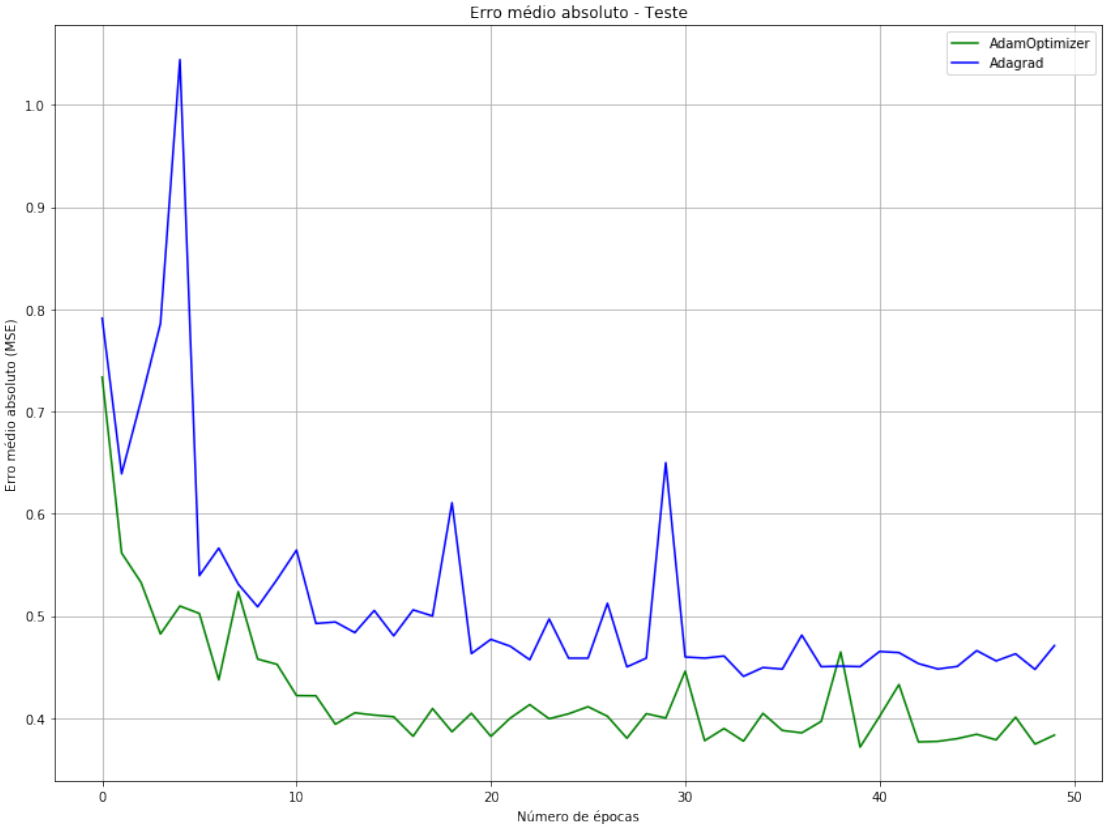
Para escolha do otimizador, foi feito uma comparação entre os 2 que melhor destacaram-se nos testes, o *AdamOptimizer* e o *Adagrad*. Nos gráficos da Figura 27 e Figura 28, pode-se observar o comportamento de cada otimizador durante o treinamento da rede neural convolucional.

Figura 27 – Gráfico do erro médio quadrático para os dados de treino, com diferentes otimizadores



Fonte: Autor (2019)

Figura 28 – Gráfico do erro médio absoluto para os dados de teste, com diferentes otimizadores

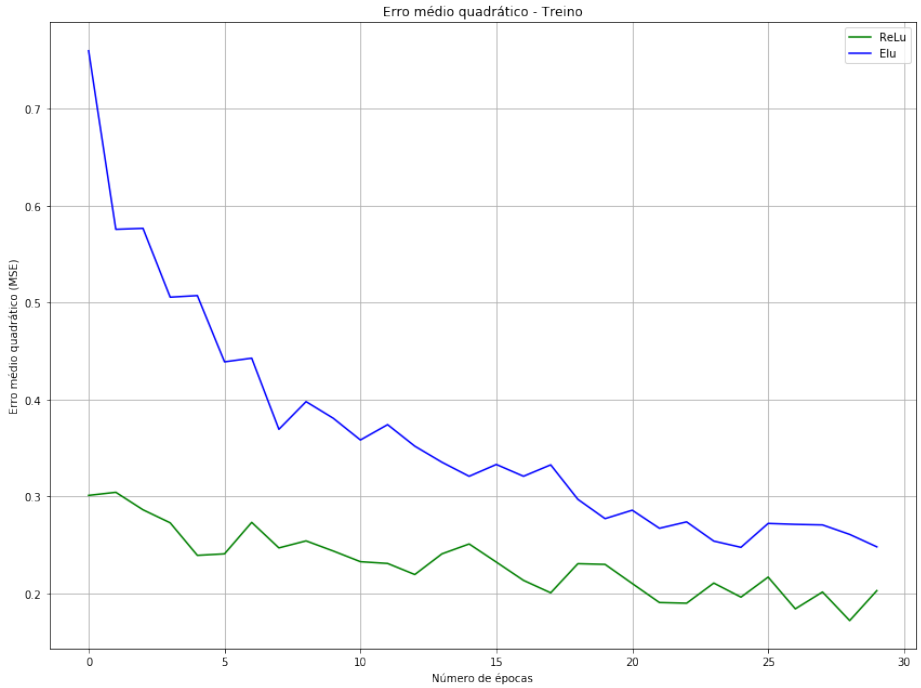


Fonte: Autor (2019)

Com a análise dos teste, o otimizador escolhido para o modelo final foi o *AdamOptimizer*, obtendo um melhor resultado entre os otimizadores testados.

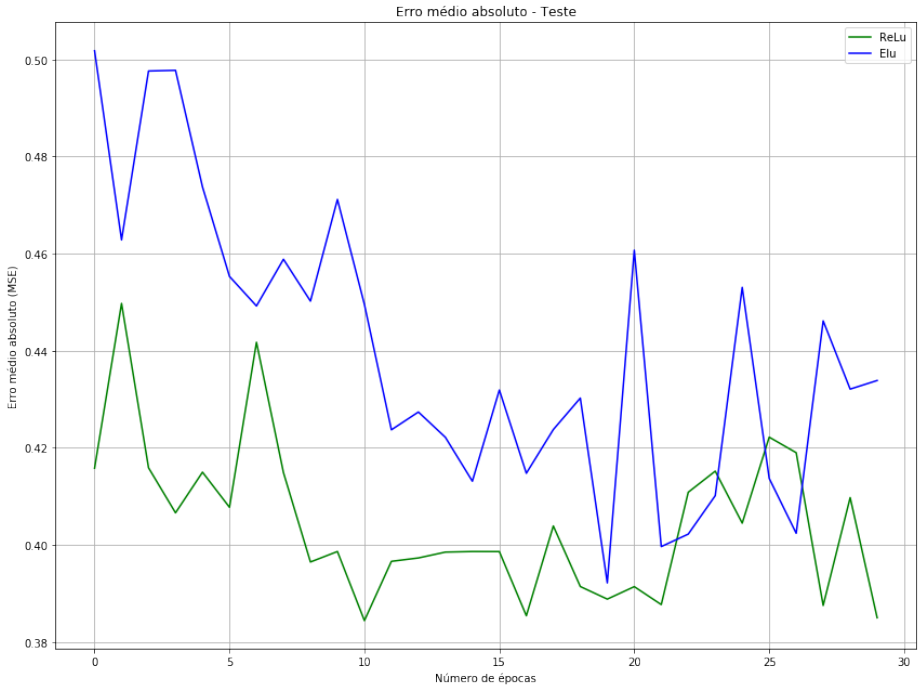
Com o uso do otimizador e modelo escolhidos, foram feitos testes com as 2 funções de ativação que melhor destacaram-se nos testes gerais. A função de ativação *ReLu* e a função de ativação *Elu*. Na Figura 29 e Figura 30, pode-se observar que a melhor eficiência foi com o uso da função *ReLu*.

Figura 29 – Gráfico do erro médio quadrático para os dados de treino, com diferentes funções de ativação



Fonte: Autor (2019)

Figura 30 – Gráfico do erro médio absoluto para os dados de teste, com diferentes funções de ativação



Fonte: Autor (2019)

Para validar o treinamento foi utilizado o método de validação cruzada *holdout*. Onde os dados fornecidos foram divididos em 2 conjuntos: um conjunto com 20% dos dados foi utilizado para teste e o outro conjunto com 80% dos dados foi utilizado para o treinamento.

A diferença no tamanho do *batch size* utilizado entre os modelos iniciais e o modelo final, é justificado pela capacidade de processamento da máquina utilizada para o treinamento. Nos modelos iniciais, como eram imagens 480x480 *pixels*, se tornou inviável a utilização de um *batch size* maior, ocasionando erro no treinamento por falta de memória. No modelo final, com a utilização de imagens 180x180 *pixels*, um *batch size* maior pôde ser utilizado.

A arquitetura utilizada no modelo final, conforme Tabela 3, foi escolhida com base em testes. Alguns parâmetros da rede foram configurados com base em informações obtidas pelo site da biblioteca *Tensorflow*, onde consta configurações como tamanho da matriz do *kernel* utilizado nas camadas da rede, *strides*, dentre outros. Mesmo assim, ainda foram executados testes com arquiteturas diferentes das indicadas, afim de verificar os melhores resultados.

Tabela 3 – Arquitetura da rede neural convolucional utilizada nos testes finais

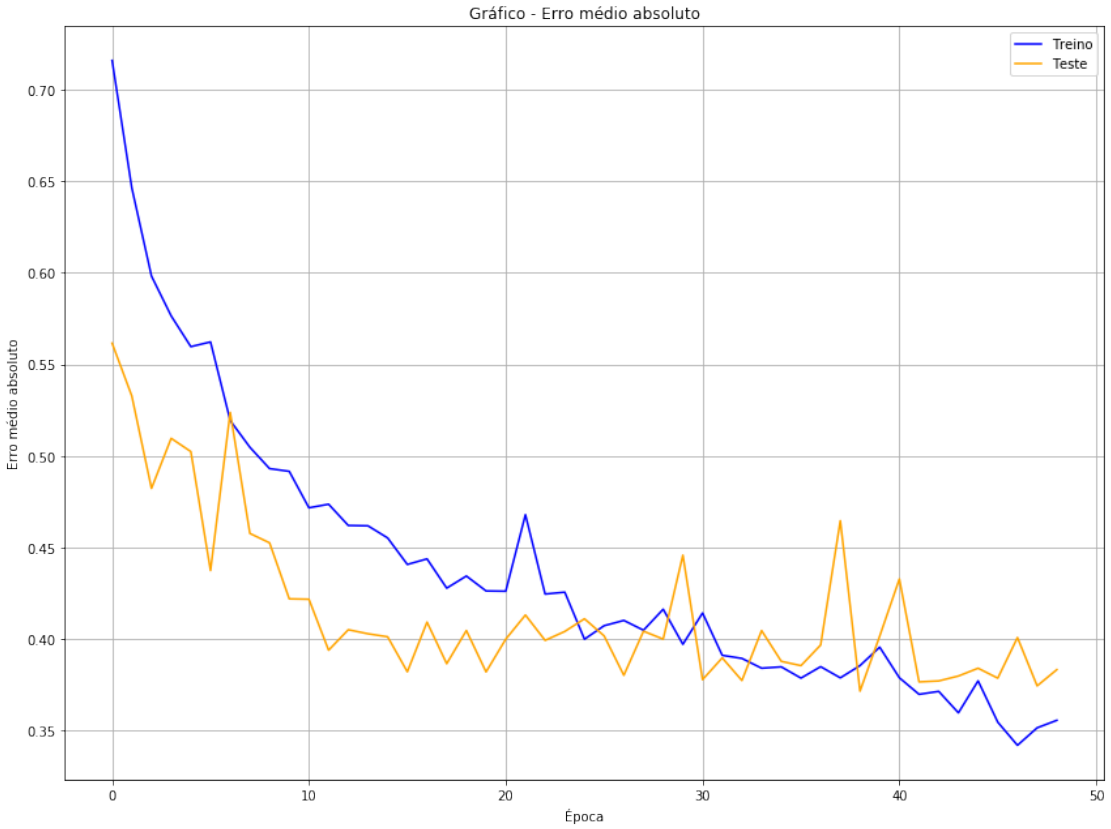
Camadas	Filtros / Kernel	Ativação	Entrada	Retorno	Batch_size	Strides
Entrada	-	-	180x180x1	180x180x1	32	-
convolucional	3x3	Relu	180x180x1	180x180x32	32	2
Pooling	2x2	-	180x180x32	90x90x32	32	2
convolucional	3x3	Relu	90x90x32	90x90x32	32	2
Pooling	2x2	-	90x90x32	45x45x32	32	2
convolucional	3x3	Relu	45x45x32	45x45x32	32	2
Pooling	2x2	-	45x45x32	22x22x32	32	2
convolucional	3x3	Relu	22x22x32	22x22x32	32	2
convolucional	3x3	Relu	22x22x32	22x22x32	32	2
Pooling	2x2	-	22x22x32	11x11x32	32	2
convolucional	3x3	Relu	11x11x32	11x11x32	32	2
convolucional	3x3	Relu	11x11x32	11x11x32	32	2
Pooling	2x2	-	11x11x32	5x5x32	32	2
Flattening	-	-	5x5x32	800	32	-

Densa	-	Relu	800	512	32	-
Dropout 0,2	-	-	512	512	32	-
Densa	-	Relu	512	64	32	-
Dropout 0,2	-	-	64	64	32	-
Densa	-	Linear	64	1	32	-

Fonte: Autor(2019)

No treinamento do modelo final, foram utilizados o otimizador *AdamOptimizer* e uma taxa de aprendizado de 0.001. Para função de perda do treinamento foi utilizado o erro médio quadrático, e como métrica de validação foi utilizado o erro médio absoluto. O treinamento foi executado durante 50 épocas, conforme a Figura 31.

Figura 31 – Gráfico de perda do treinamento final do projeto



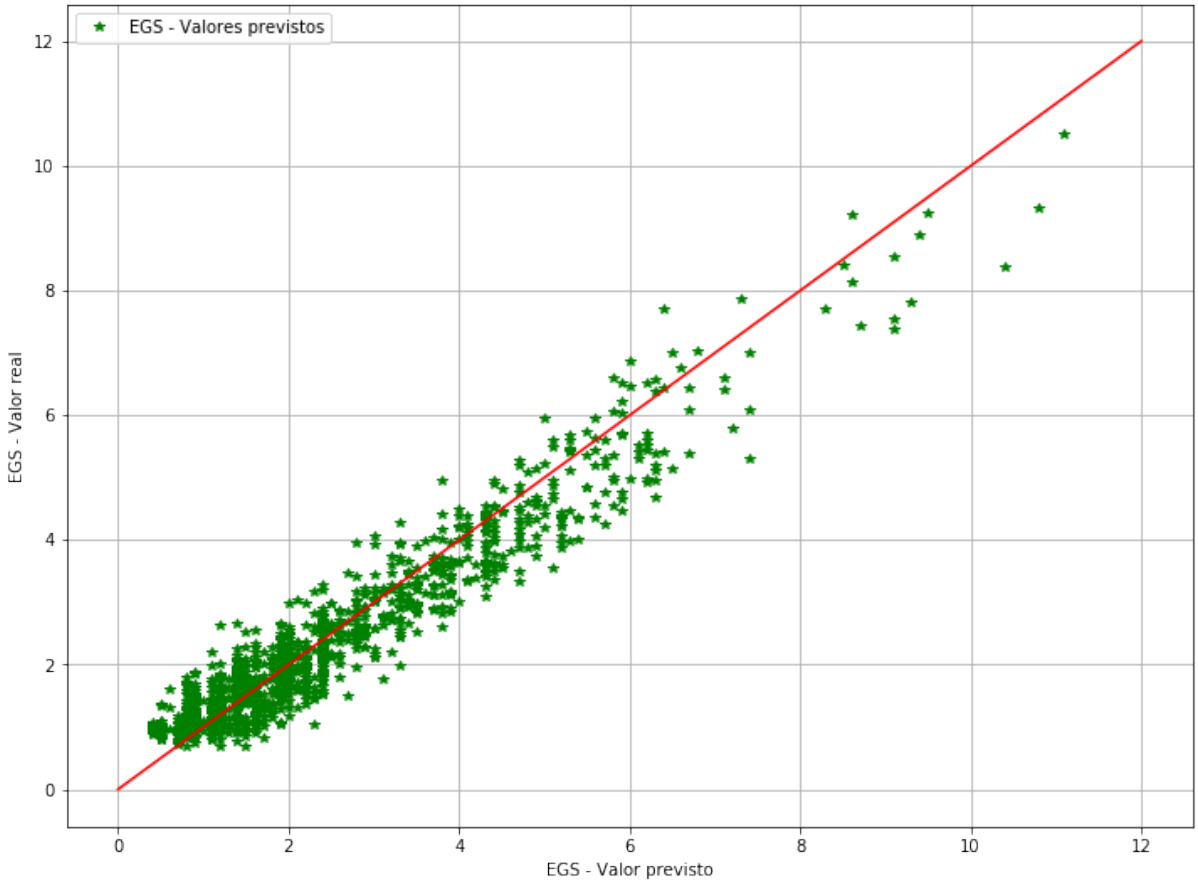
Fonte: Autor (2019)

O treinamento final da rede neural convolucional demorou cerca de 4 horas para

ser finalizado, sendo executado em um sistema operacional *Windows* 10, com 8GB de memória e uma GPU *NVIDIA GeForce* 910M com 2GB de memória dedicada.

O gráfico de dispersão da Figura 32, mostra uma reta traçada na diagonal entre os valores previstos e os valores reais. Os pontos destacados em verde são os valores previstos. A reta traçada serve para exemplificar os valores reais. Quanto mais longe da reta, maior o erro do valor previsto pela rede. Pode-se observar que os valores que foram mensurados pela rede estão com um alto nível de acerto.

Figura 32 – Gráfico de dispersão entre os valores reais e previstos



Fonte: Autor (2019)

Após a conclusão dos testes, foi construído um algoritmo para predição de valores de EGS, utilizando um método de carregamento de um arquivo .h5, que é salvo durante o treinamento, onde ficam salvas as configurações do treinamento feito com o modelo construído no projeto. Ao executar esse algoritmo, é solicitado que selecione a pasta onde estão as imagens para serem mensuradas, após o processamento, o algoritmo gera uma planilha, onde consta o nome da imagem, com o valor de EGS previsto conforme

Figura 33.

Figura 33 – Tabela com valores de EGS previstos, gerado pelo algoritmo de predição

	A	B
1	ID	EGS
2	2182015	[0.8823328]
3	659	[0.6919203]
4	7349	[0.9602729]
5	L752	[0.92019933]
6	24az	[1.8645344]
7	2308	[2.3598044]
8	82534	[4.3573675]
9	2055	[5.6515255]
10	82571	[5.609479]

Fonte: Autor (2019)

O algoritmo para predição construído, ainda precisa passar por testes de utilização. Sua função é gerar uma planilha com valores de EGS, semelhante ao software utilizado atualmente pelo especialista. A diferença é que, o software utilizado pelo especialista, necessita que cada ultrassom tenha a área de EGS marcada, para que ele possa fazer o cálculo e gerar a planilha com os valores de EGS. O algoritmo construído, apenas necessita que a pasta, cujas imagens ultrassonográficas estão armazenadas, seja selecionada pelo usuário. Após isso, o algoritmo gera automaticamente a planilha com os valores de EGS.

5 CONSIDERAÇÕES FINAIS

O presente estudo, possibilitou analisar a importância das redes neurais convolucionais na automatização de processos em diversas áreas. Além disso, também permitiu um estudo a campo para um melhor entendimento sobre o processo de classificação da espessura de gordura subcutânea (EGS) em imagens ultrassonográficas de bovinos e sua importância dentro da pecuária de corte.

Ao acompanhar o processo de obtenção das imagens ultrassonográficas, e a classificação de EGS, que é feita manualmente, verificou-se que o processo pode se tornar demorado, tornando ainda mais importante o uso de um método automatizado e eficiente.

Dada essa importância do assunto, o projeto teve como objetivo construir esse método de automatização, para que tornasse o processo mais eficiente, podendo auxiliar em tomadas de decisões. Através de estudos, a ferramenta escolhida para construção do automatizador foi a utilização de redes neurais convolucionais.

O modelo criado para automatização da mensuração de EGS, obteve um erro médio absoluto de 0.38, com uma correlação de 0.97 com os valores reais de EGS. Sendo que, para a comprovação de medidas de EGS, é necessário uma correlação de 0.85 entre os valores previstos e valores reais de EGS. Com base nessa informação, o objetivo do trabalho foi cumprido, com a construção de um modelo baseado em redes neurais convolucionais.

Com os resultados apresentados no projeto, torna-se viável a utilização do método apresentado. Através de um software de predição, o especialista poderá ter acesso às informações de EGS logo após o processo de obtenção das imagens ultrassonográficas.

5.1 Trabalhos Futuros

Seguindo a linha de pesquisa do projeto, uma melhor interface de predição de EGS poderá ser construída. Utilizando talvez algum método mais eficiente para uso a campo, podendo ser utilizado na beira de bretes, onde é feita a obtenção das imagens ultrassonográficas. Podendo ser utilizado sem ser necessário o uso de um computador de mesa ou notebook para uso do algoritmo de predição, uma solução seria implementar o algoritmo de predição em um *Raspberry*.

Outro trabalho futuro, é a construção de modelos para predição de EGP e AOL. Como comprovado pelo presente trabalho, o uso de redes neurais convolucionais pode ser

utilizado para essa finalidade.

REFERÊNCIAS

ALVES, L. d. L. *et al.* O ultrassom no amaciamento de carnes. **Ciência Rural**, v. 43, n. 8, p. 1522–1528, 2017. ISSN 0103-8478. Acesso em: 03 outubro 2018. Disponível em: http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0103-84782013000800029&lng=pt&nrm=iso&tlng=en.

BEZERRA, E. Introdução à Aprendizagem Profunda. In: **Tópicos em Gerenciamento de Dados e Informações**. [S.l.: s.n.], 2016. ISBN 9788576693444.

BRUCKNER, D.; ROSEN, J.; SPARKS, E. R. deepViz: Visualizing Convolutional Neural Networks for Image Classification. **PhD Dissertations**, 2013.

CAMARGO, S. **Um Modelo Neural de Aprimoramento Progressivo para Redução de Dimensionalidade**. 107 p. Tese (Doutorado), 2010. Acesso em: 08 dezembro 2018. Disponível em: http://www.lume.ufrgs.br/handle/10183/26500.

CARMEN, D. Classificação Automática do Acabamento de Gordura em Imagens Digitais de Carcaças Bovinas. 2017.

CARNEIRO, A. C.; SILVA, R. R. V. Redes Neurais Convolucionais com Tensorflow: Teoria e Prática. **III Escola Regional de Informática do Piauí**, p. 382–406, 2017.

CAROLINE, A. *et al.* Um Estudo sobre Redes Neurais Convolucionais e sua Aplicação em Detecção de Pedestres. p. 4, 2017. Acesso em: 24 outrubro 2018. Disponível em: http://sibgrapi.sid.inpe.br/col/sid.inpe.br/sibgrapi/2016/09.12.15.44/doc/um-estudo-sobre.pdf.

DIAS, M. F. R.; PASCUTTI, P. G.; Da Silva, M. L. Aprendizado De Máquina E Suas Aplicações Em Bioinformática. **Semioses**, v. 10, n. 1, p. 23–37, 2017. ISSN 1981-996X. Acesso em: 03 setembro 2018. Disponível em: http://apl.unisuam.edu.br/revistas/index.php/Semioses/article/view/1070.

DUCHI, J.; HAZAN, E.; SINGER, Y. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. **Jmlr**, v. 12, p. 1–40, 2011.

Empresa Brasileira de Pesquisa Agropecuária. **EMBRAPA**. 2018.

GIL, A. C. **Métodos e técnicas de pesquisa social**. [s.n.], 2008. 199 p. Acesso em: 11 setembro 2018. ISBN 9788522451425. Disponível em: https://ayanrafael.files.wordpress.com/2011/08/gil-a-c-mc3a9todos-e-tc3a9cnicas-de-pesquisa-social.pdf.

GUDWIN, R. R. Novas Fronteiras na Inteligência Artificial e na Robótica. **4º Congresso Temático de Dinâmica, Controle e Aplicações**, p. 18, 2005.

HAYKIN, S. Redes neurais: princípios e prática. **Bookman**, p. 900, 2001.

HAYKIN, S. **Neural Networks and Learning Machines**. [s.n.], 2008. 889 p. Acesso em: 15 setembro 2018. ISSN 14337851. ISBN 9780131471399. Disponível em: https://arxiv.org/pdf/1312.6199v4.pdf.

JUNIOS, R. A. d. A. T. *et al.* Melhoramento Animal na Era das DEPS. In: **Melhoramento genético aplicado em gado de corte**. [S.l.: s.n.], 2013. p. 216.

KINGMA, D. P.; BA, J. Adam: A Method for Stochastic Optimization. **ICLR 2015**, p. 15, 2015. Acesso em: 10 março 2019. Disponível em: http://arxiv.org/abs/1412.6980.

MARCOMINI, K. D. **Aplicação de modelos de redes neurais artificiais na segmentação e classificação de nódulos em imagens de ultrassonografia de mama**. 129 p. Tese (Doutorado), 2013.

MAZZA, L. O. **Aplicacao De Redes Neurais Convolucionais Densamente Conectadas No Processamento**. 88 p. Tese (Doutorado), 2017.

Moacir A. Ponti; COSTA, G. B. P. da. **Como funciona o Deep Learning**. [S.l.: s.n.], 2017. 63–93 p. ISBN 9788576694007.

NASCIMENTO, P. C. Inteligência artificial. **Editora Blucher**, p. 2001, 2001. Acesso em: 11 setembro 2018. Disponível em: http://www.unicamp.br/unicamp/unicamp_hoje/ju/jornalPDF/ju170_p04.pdf.

NETO, S. A. d. L. H. C. **Reconhecimento de Tumores Cerebrais Utilizando Redes Neurais Convolucionais**. 66 p. Tese (Doutorado), 2017.

NIELSEN, M. **Neural Networks and Deep Learning**. [S.l.: s.n.], 2013. 216 p.

NIETO, L. M.; ALENCAR, M. M. D.; ROSA, A. d. N. Critérios de Seleção. In: **Melhoramento genético aplicado em gado de corte**. [S.l.: s.n.], 2013. p. 256.

Norvig, P., & Russell, S. **Inteligencia artificial**. [S.l.: s.n.], 2017. v. 1. 1021 p. ISSN 08858977. ISBN 9688870757.

OLIVEIRA, F. *et al.* Cadeia produtiva da carne bovina no Brasil. **Interação Interdisciplinar**, v. 1, n. 1, p. 229–244, 2017.

PEREIRA, R. M. P. *et al.* Abordagem Deep Learning para Classificacao de Lesoes Mamarias. p. 2597–2600, 2017.

RODGERS, J. L.; NICEWANDER, W. A. Thirteen Ways to Look at the Correlation Coefficient. **The American Statistician**, v. 42, n. 1, p. 59, 2006. ISSN 00031305.

RODRÍGUEZ, R. A. M. **Teoria e Aplicação do Classificador baseado em Segmentos de Reta em Problemas de Multiclassificação**. 35 p. Tese (Doutorado), 2011.

ROMEIRO, G.; MENEZES, D. O. Recursos Genéticos e Estratégias de Melhoramento. In: **Melhoramento genético aplicado em gado de corte**. [S.l.: s.n.], 2013. p. 256.

SILVA, N. M. G. da; CESARIO, A. V.; CAVALCANTI, I. R. Relevância do agronegócio para economia brasileira atual. In: **X Encontro de Iniciação à Docência**. [S.l.: s.n.], 2017. p. 5.

SZEGEDY, C. *et al.* Rethinking the Inception Architecture for Computer Vision. **Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition**, v. 2016-Decem, p. 9, 2016. ISSN 10636919.

VILHARVA, G. *et al.* Tipificação de Carcaça Bovina usando Redes Neurais Convolucionais. p. 12, 2017.

YOKOO, M. J.-i. *et al.* Avaliação de Carcaça por Ultrassom e sua Aplicação Prática. 2015.

ZACCONE, G.; KARIM, M. R.; MENSHAWY, A. **Deep Learning with TensorFlow**. [S.l.: s.n.], 2017. 300 p. ISBN 9781786469786.