

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Vinicius Clemente de Sousa Reis

**Aplicação de Redes Neurais Convolucionais na
Classificação de Amostras de Carne Suína por
Meio da Capacidade de Retenção de Água**

Uberlândia, Brasil

2021

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Vinicius Clemente de Sousa Reis

**Aplicação de Redes Neurais Convolucionais na
Classificação de Amostras de Carne Suína por Meio da
Capacidade de Retenção de Água**

Trabalho de conclusão de curso apresentado
à Faculdade de Computação da Universidade
Federal de Uberlândia, Minas Gerais, como
requisito exigido parcial à obtenção do grau
de Bacharel em Sistemas de Informação.

Orientador: André Ricardo Backes

Universidade Federal de Uberlândia – UFU

Faculdade de Ciência da Computação

Bacharelado em Sistemas de Informação

Uberlândia, Brasil

2021

Vinicius Clemente de Sousa Reis

Aplicação de Redes Neurais Convolucionais na Classificação de Amostras de Carne Suína por Meio da Capacidade de Retenção de Água

Trabalho de conclusão de curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, Minas Gerais, como requisito exigido parcial à obtenção do grau de Bacharel em Sistemas de Informação.

Trabalho aprovado. Uberlândia, Brasil, 17 de junho de 2021:

André Ricardo Backes
Orientador

Henrique Coelho Fernandes
Professor convidado

Jefferson Rodrigo de Souza
Professor convidado

Uberlândia, Brasil
2021

Agradecimentos

Primeiramente gostaria de agradecer a Deus.

Agradeço ao meu orientador André Ricardo Backes por aceitar conduzir o meu trabalho de pesquisa.

A todos os meus professores da Faculdade de Computação da Universidade Federal de Uberlândia, pela excelência da qualidade técnica de cada um.

À Isaura Maria Ferreira Doutora em Ciências Veterinárias pela Universidade Federal de Uberlândia, pela disponibilização das imagens utilizadas durante este trabalho.

À minha família que sempre estiveram ao meu lado me apoiando ao longo de toda a minha trajetória.

Resumo

Para analisar a qualidade de carne Suínas podemos utilizar várias técnicas, entre elas podemos destacar a capacidade de retenção de água (CRA). Normalmente, a CRA é estimada medindo o suco liberado após a compressão de amostras de carne. Entretanto, após gerar as marcações no filtro de papel, ainda é necessário a presença de profissional para analisar as amostras, esse trabalho sugere a utilização de redes neurais convolucionais para a segmentação automática das fotos dessas amostras. Para tanto utilizamos a arquitetura *Unet* e avaliamos duas abordagens: segmentação binária utilizando a função de ativação *sigmoide* e a segmentação multi classe utilizando a função de ativação *softmax*. Através de experimentos demonstramos a superioridade da segmentação multi classe que apresentou uma média de 97,78% utilizando a função de métrica *IoU*, além disso apresentaremos as vantagens e as limitações de se utilizar essa abordagem.

Palavras-chave: Capacidade de Retenção de Água, Carne Suína, Redes Neurais Convolucionais, Segmentação de imagens, *Unet*.

Lista de ilustrações

Figura 1 – CRA	9
Figura 2 – Perceptron	12
Figura 3 – Função Sigmoides	13
Figura 4 – Função ReLU	14
Figura 5 – Semantic segmentation	16
Figura 6 – Arquitetura da rede Unet	16
Figura 7 – Etapas do método de compressão	17
Figura 8 – IoU por épocas obtida pelas redes utilizando imagens 64×64	24
Figura 9 – IoU por épocas obtida pelas redes utilizando imagens 128×128	25
Figura 10 – IoU por épocas obtida pelas redes utilizando imagens 256×256	26
Figura 11 – Exemplo da dificuldade em encontrar a região de área externa	28
Figura 12 – Exemplo de resultado	30

Lista de tabelas

Tabela 1	–	Hiperparâmetros utilizados.	20
Tabela 2	–	<i>IoU</i> de todo o <i>dataset</i>	25
Tabela 3	–	<i>IoU</i> do <i>dataset</i> de treino.	26
Tabela 4	–	<i>IoU</i> do <i>dataset</i> de validação.	27
Tabela 5	–	<i>IoU</i> de todo o <i>dataset</i> utilizando imagens em seu tamanho original. . .	29
Tabela 6	–	<i>IoU</i> do <i>dataset</i> de treino utilizando imagens em seu tamanho original.	29
Tabela 7	–	<i>IoU</i> do <i>dataset</i> de validação utilizando imagens em seu tamanho original.	29

Sumário

1	INTRODUÇÃO	9
1.1	Objetivo geral	9
1.1.1	Objetivos específicos	9
1.2	Justificativa	10
2	REFERENCIAL TEÓRICO	11
2.1	Redes neurais artificiais	11
2.2	Aprendizado supervisionado	11
2.3	Perceptron	12
2.4	Função de ativação	12
2.4.1	Função sigmoide	13
2.4.2	Função ReLU	13
2.5	Redes neurais convolucionais	14
2.5.1	Camada convolucional	14
2.5.2	Camada de pooling	15
2.5.3	Camada totalmente conectada	15
2.6	Redes de segmentação semântica	15
3	TRABALHOS CORRELATOS	17
4	METODOLOGIA	19
4.1	TensorFlow	19
4.2	Arquitetura	19
4.3	Hiperparâmetros	19
4.3.1	Tamanho da imagem	20
4.3.2	Número de épocas	20
4.3.3	Tamanho do lote	20
4.3.4	Porcentagem de validação	21
4.3.5	Tamanho da rede <i>Unet</i>	21
4.3.6	Função de métrica	21
4.3.7	Função de ativação	21
4.4	Conjunto de imagens	22
5	RESULTADOS	23
5.1	Resultados obtidos considerando a saída original da rede	23
5.2	Resultados obtidos considerando a saída transformada da rede	27

6	CONCLUSÃO	31
	REFERÊNCIAS	32
	APÊNDICES	34
	ANEXOS	35

1 Introdução

De acordo com a ([ABPA, 2020](#)) em 2019 o Brasil produziu 3974 Mil toneladas de carne suína, ocupando assim a 4ª posição mundial. Para analisar a qualidade dessa carne podemos utilizar várias técnicas, entre elas podemos destacar a capacidade de retenção de água. De acordo com ([JOO, 2018](#)): “A capacidade de retenção de água (CRA) da musculatura suína pós morte é uma importante característica de qualidade. Suas variações precisam ser consideradas para avaliar e processar produtos suínos”.

Existem várias formas de medir o CRA da musculatura suína, neste trabalho focaremos no método compressão em papéis de filtro. Este método foi originalmente desenvolvido por Grau e Hamm ([ABDULLAH et al., 2013](#)), que consiste em remover uma pequena amostra de carne (0,3g), pressionar essa amostra em um papel de filtro por meios de pesos. Após cinco minutos a amostra é removida do papel e é marcada a posição da amostra no papel filtro.

Após gerar as marcações no filtro de papel, normalmente é necessário um profissional analisar as amostras. Esse trabalho sugere a utilização de redes neurais para a avaliação automática das fotos dessas amostras, sendo um exemplo de amostra mostrado na Figura 1.

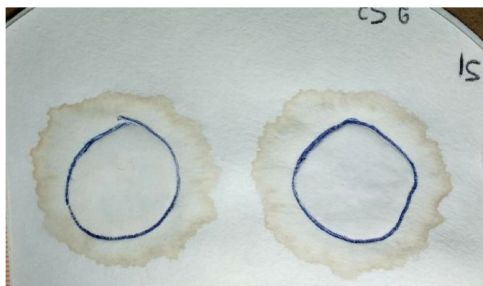


Figura 1 – Exemplo de coleta de CRA. Figura retirada de ([ANDRADE, 2019](#))

1.1 Objetivo geral

O principal objetivo deste trabalho é estudar e avaliar o desempenho de redes neurais convolucionais na segmentação de amostras de CRA de carne suína.

1.1.1 Objetivos específicos

- Quantificar o desempenho de redes neurais convolucionais *UNet*.

- Quantificar o desempenho de redes neurais convolucionais utilizando diferentes tamanhos de imagens.
- Quantificar o desempenho de redes neurais convolucionais utilizando segmentação binária e multi classe.

1.2 Justificativa

O Brasil é um dos maiores produtores de proteína animal do mundo. Para manter essa posição é necessário garantir que essa proteína tenha uma boa qualidade. Uma solução escalável é a utilização de redes neurais convolucionais para estimar e analisar a capacidade de retenção de água das amostras de carne, garantindo assim um processo rápido, fácil e automático, diminuindo a possibilidade de falhas.

2 Referencial teórico

2.1 Redes neurais artificiais

De acordo com (BRAGA ANTONIO; LUDEMIR, 2000) “ As Redes Neurais Artificiais (RNAs) são inspiradas no funcionamento do cérebro humano, ou seja, na rede neural biológica. Ela é definida como sendo uma forma de computação não algorítmica, por não ser baseada em regras ou programas, sendo uma alternativa à computação algorítmica convencional ”

As RNAs possuem neurônios artificiais que são interconectados um com os outros, essas conexões são normalmente realizadas por meio de um sistema de pesos e pela variação nesse sistema de pesos que realiza é o aprendizado de máquina ou seja o processo de aprendizagem é realizado ao tentar encontrar o número de pesos ótimos.

2.2 Aprendizado supervisionado

Podemos treinar uma RNA normalmente de 3 formas: aprendizado supervisionado, aprendizado semi-supervisionado, aprendizado não-supervisionado. Nesse trabalho utilizaremos o aprendizado supervisionado, nesse tipo de aprendizado deve existir um comportamento de referência para ensinar a rede, com isso podemos ajustar os pesos da rede de acordo com o erro entre o comportamento atual da rede e o comportamento referência.(OSÓRIO, 2000)

Existem vários desafios no aprendizado supervisionado, um dos principais é o tempo de treinamento da rede, para resolver esse desafio devemos olhar a taxa de atualização dos pesos, se essa taxa for muito baixa a rede consome muito tempo no treinamento, porem não devemos colocar essa taxa muito alta, pois apesar da rede convergir em um espaço de tempo menor, novas entradas na rede podem torná-la instável gerando assim uma baixa confiabilidade dos resultados.(Barca; Silveira; Magini, 2005)

Uma boa solução para resolver o desafio da taxa de atualização dos pesos é a técnica do *backpropagation*, nessa técnica o treinamento baseia-se na retro propagação dos ajustes dos pesos no sentido contrário, isto é, da camada de saída para as camadas intermediárias.(OSÓRIO, 2000)

2.3 Perceptron

Os Perceptrons foram um dos primeiros neurônios artificiais aplicadas nas RNAs, O Perceptron foi criado em 1958, por Frank Rosenblatt ([ROSENBLATT, 1958](#)) que criou o algoritmo com o objetivo de simular um neurônio biológico. Esse algoritmo (Figura 2) é descrito pelos passos:

1. Os sinais de entrada x_1, x_2, x_m são multiplicados pelos pesos w_{k1}, w_{k2}, w_{kp}
2. É realizado um somatório da multiplicação dos pesos e a soma de um bias (b_k)
3. Aplicamos a função de ativação φ ao resultado do somatório
4. O resultado da função de aplicação gera a saída binária y_k .

Podemos descrever um perceptron matematicamente como:

$$y_k = \varphi\left(\sum_{i=1}^m x_i \cdot w_{ki} + b_k\right) \quad (2.1)$$

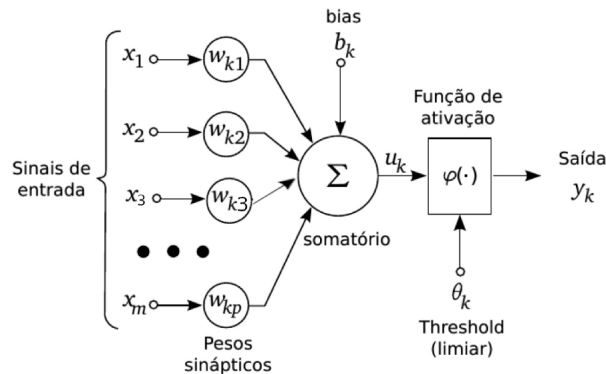


Figura 2 – Modelo de Neurônio Artificial, segundo McCulloch-Pitts. Figura adaptada de ([SILVA; SCHIMIDT, 2016](#))

2.4 Função de ativação

As funções de ativação calculam como vai ser a saída dos perceptron. A escolha da função de ativação é importante é por meio delas que introduzimos a não linearidade das RNAs, uma RNA com uma função de ativação mal escolhida pode ser tornar apenas um sistema de regressão linear.

Para exemplificar as funções de ativação citaremos as funções Sigmoid e ReLU.

2.4.1 Função sigmoide

A função Sigmoide (Figura 3) foi amplamente utilizada nas primeiras RNAs pois consegue assumir valores no intervalo $[0, 1]$, portanto é passível modelar matematicamente o comportamento dos neurônios biológicos que assumem valores ativado(0) e não ativado(1)

Podemos expressar matematicamente a função Sigmoide como:

$$\varphi(x) = \frac{1}{1 + e^{-x}} \quad (2.2)$$

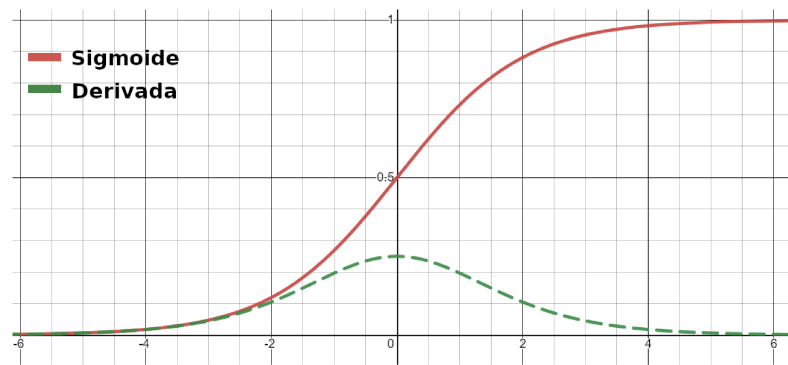


Figura 3 – Gráfico da função Sigmoide e sua derivada.

Apesar de amplamente utilizada nas primeiras RNAs seu uso em redes modernas é limitado pois a sua derivada que está no intervalo $[-5, 5]$ tem a tendência a zero. Dada a tendência a zero da derivada as funções sigmóides geram o problema conhecido como *vanishing gradient* isto é como utilizamos a derivada da função de ativação para calcular o ajuste nos pesos se essa derivada tende a zero os ajustes serão extremamente pequenos logo a rede para de aprender. O problema do *vanishing gradient* é facilmente encontrado em redes de *deep learning* dado que essas redes necessitam uma grande quantidade de camadas.

2.4.2 Função ReLU

A função ReLU (Figura 4) é parecida com a função identidade se diferenciarmos pois para dados menores ou iguais a zero o resultado é sempre zero. Portanto é uma função bem simples e fácil de otimizar. Apesar de simples as funções ReLU são bastante utilizadas como função de ativação pois é uma função não linear e a sua derivada é estável pois assume valor 1 para $x > 0$ e valor 0 para $x < 0$ portanto não sofre do problema de *vanishing gradient*.

A função ReLU ou ativação linear retificada é descrita matematicamente como:

$$ReLU(x) = \max(0, x) \quad (2.3)$$

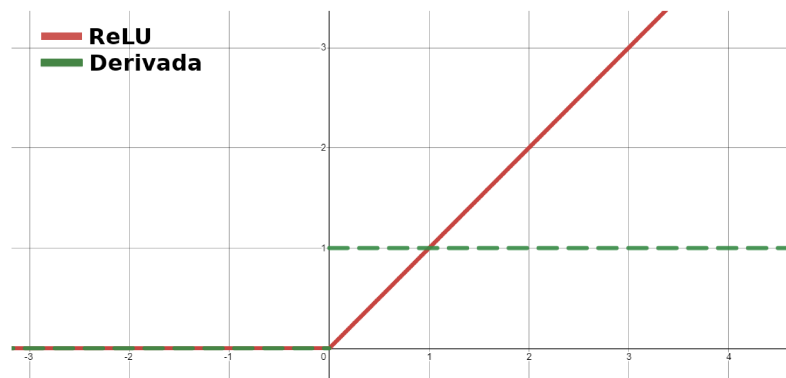


Figura 4 – Gráfico da função ReLU e sua derivada.

2.5 Redes neurais convolucionais

As redes neurais convolucionais (RNC) se diferenciam de uma rede perceptron multicamadas por englobar o conceito de campo receptivo, o qual permite extrair característica da imagem de modo similar ao sistema visual humano. Assim, os neurônios são treinados para encontrar um filtro que aplicado a imagem consiga ressaltar características relevantes ao problema. Pelo fato desta rede tentar encontrar filtros que são representadas como matrizes e não apenas pesos, existe um compartilhamento de cálculo entre os neurônios e, somado ao fato de que não é necessário que todos os neurônios estejam ligados pixel a pixel a imagem nem ao seus adjacentes, essa rede se torna mais fácil de ser treinada possibilitando assim a adição de mais camadas, cada qual representando um nível semântico, dando origem a técnica conhecida como *deep learning*. Algumas redes se tornam tão generalistas que podem ser aplicadas a problemas mais abrangentes do que seu objetivo inicial, portanto podemos não só treinar uma rede convolucional do zero como podemos também utilizar redes já treinando e avaliar o seu desempenho.

Podemos definir uma RNC como um tipo de RNA que é projetada para aprender características espaciais de atributos de forma automática e adaptativa por meio do método de *backpropagation* e utilizando as seguintes camadas: convolucional, pooling e camada totalmente conectada. (YAMASHITA et al., 2018)

2.5.1 Camada convolucional

A camada convolucional nas RNC são responsáveis por extrair as características das imagens. Para realizar tal objetivo realizamos a Convolução de uma parte da imagem com o filtro da camada. As redes convolucionais não analisam a imagem como um todo, faz parte da arquitetura da rede definir qual é a janela de cada camada convolucional isto é qual vai ser o tamanho do recorte da imagem que aplicaremos os filtros da camada. Uma das vantagens de se utilizar essa camada ao invés de uma camada totalmente conectada é a sua velocidade de treinamento, pois existe um compartilhamento de cálculo entre os

neurônios dessa camada tornando assim o treinamento mais rápido. (YAMASHITA et al., 2018)

2.5.2 Camada de pooling

A camada de pooling é responsável por simplificar a informação das camadas anteriores, esse passo é importante para evitar o *overfitting* que é quando a rede consegue prever muito bem os dados de treinamento mas não consegue uma boa precisão com dados reais. Para realizar essa simplificação da informação normalmente é utilizado o *max-pooling*, no qual apenas o maior número da unidade é passado para a saída. (YAMASHITA et al., 2018)

Assim como na camada convolucional, faz parte da arquitetura da rede definir qual vai ser a janela da camada de pooling, se essa janela for muito pequena podemos acabar não conseguindo simplificar a informação, mas se essa janela for muito grande podemos acabar perdendo muita informação deixando assim o treinamento inviável.

2.5.3 Camada totalmente conectada

A camada totalmente conectada é uma última camada das RNC essa camada pode ser uma camada convolucional com janela 1×1 ou podem ser uma camada totalmente conectada de *Perceptron*, seu objetivo é calcular o resultado final da classificação. Utilizamos esse tipo de camada no final pois essa camada apresenta uma alta precisão, como essa camada possui um treinamento mais lento é esperado que os dados que cheguem nessa camada sejam mais completos e em menor quantidade. (YAMASHITA et al., 2018)

2.6 Redes de segmentação semântica

As redes de segmentação semântica se diferenciam das outras redes, pois, tipicamente utilizamos as redes convolucionais para tarefas de classificação, onde o objetivo é gerar como saída um rótulo único para a imagem na totalidade. Entretanto, em algumas tarefas especialmente no processamento de imagens biomédicas, desejamos que a saída inclua localização, ou seja, devemos rotular não a imagem toda mas sim cada pixel (RONNEBERGER; FISCHER; BROX, 2015).

Portanto, ao contrário de redes como *lenet*, *alexnet*, *vgg*, *inception* que buscam classificar as imagem, as redes de segmentação semântica como *Unet*, *LinkNet*, *PSPNet* tem como objetivo fazer uma segmentação da imagem (Figura 5), gerando assim o limite exato de um determinado objeto na imagem.

Um dos exemplos de redes de segmentação semântica é a *Unet*. Para realizar a segmentação semântica a *Unet* realiza etapas de compressão em sequência até chegar em

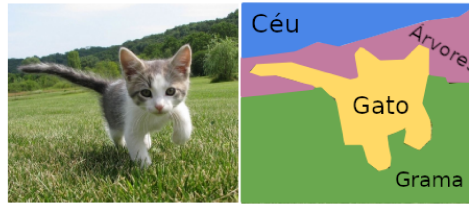


Figura 5 – Exemplo de semantic segmentation. Figura adaptada de (LI JUSTIN JOHNSON, 2017)

um ponto em que começa a realizar uma descompressão, e para fazer a descompressão é realizado também a concatenação de dados anteriores técnica que é importantíssima para a realização da segmentação. Uma curiosidade é que o nome *Unet* foi criado pois o desenho da sua arquitetura parece o desenho de uma letra U, como mostra a Figura 6.

Uma recomendação para o uso da rede *Unet* é a aplicação de *data augmentation*, técnicas que tem o objetivo de aumentar a diversidade do dataset, como, por exemplo, uso de rotações das imagem e introdução de retângulos em posições aleatórias (*Random Erasing*). Isso permite utilizar a *Unet* mesmo com um dataset reduzido se aplicarmos *data augmentation*.

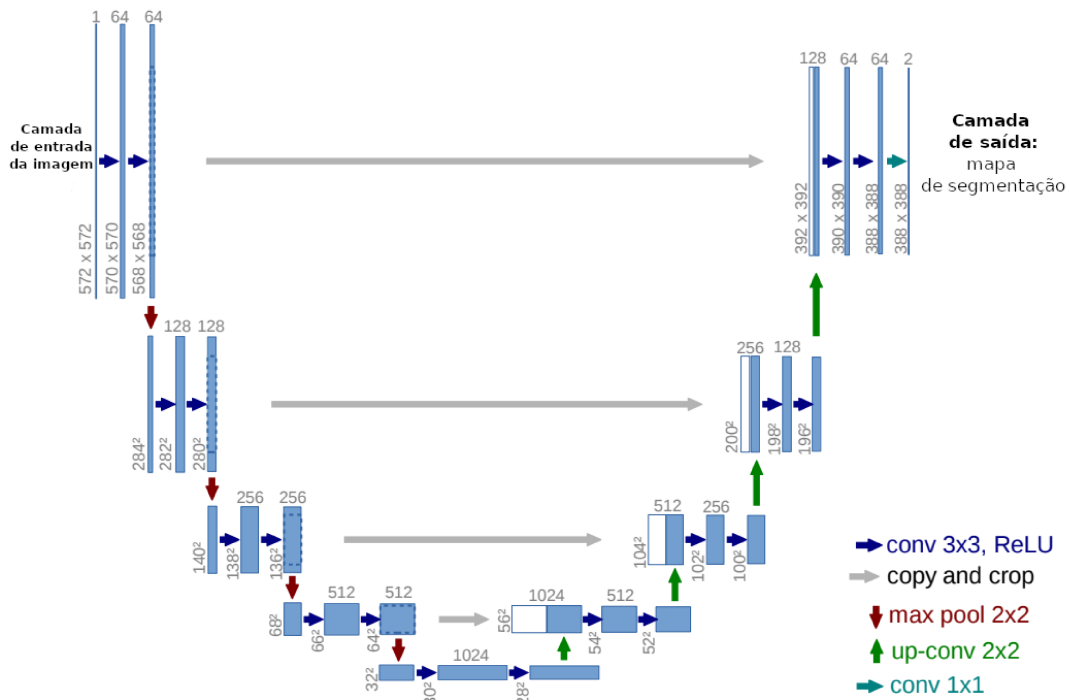


Figura 6 – Arquitetura da rede Unet. Figura adaptada de (RONNEBERGER; FISCHER; BROX, 2015)

3 Trabalhos correlatos

Nesta seção serão apresentados alguns trabalhos que abordam a utilização do CRA para analisar a qualidade da carne e também trabalhos que utilizam técnicas de Processamento digital de imagem (PDI) para automatizar o processo de análise e quantificação de CRA.

Não apresentamos nenhum trabalho de RNAs pois não foi encontrado nenhum trabalho que utiliza essa técnica para a segmentação de CRA de qualquer tipo de animal. Logo esse é o primeiro trabalho que utiliza RNAs para a segmentação de amostras de CRA de carne suína, demonstrando assim a importância desse trabalho.

O trabalho de ([ALVES, 2011](#)) tem o objetivo de avaliar o efeito do gene Halotano sobre a deposição de Gordura intramuscular e também investigar a qualidade final da carne suína a interferência dos minerais: Mg, Zn, Se e Cu adicionados à água de pocilga no pré abate, para investigar a qualidade final da carne foi utilizado a CRA da carne.

Para mensurar a CRA da carne a autora utiliza o método de compressão, demonstrando assim a viabilidade da utilização desse método, o qual está relacionado diretamente ao tema deste trabalho. Para a realização desse método foi utilizado aproximadamente 0.5g de amostras de cubo de carne, essas amostras foram pressionadas em papéis de filtro entre duas placas de acrílico com o auxílio de um peso de 10 kg sobre o material por um período de 5 minutos, após esse prazo a carne é contornada por uma caneta esferográfica e o suco liberado formam duas áreas sobrepostas, podemos observar as etapas desse método na Figura 7.

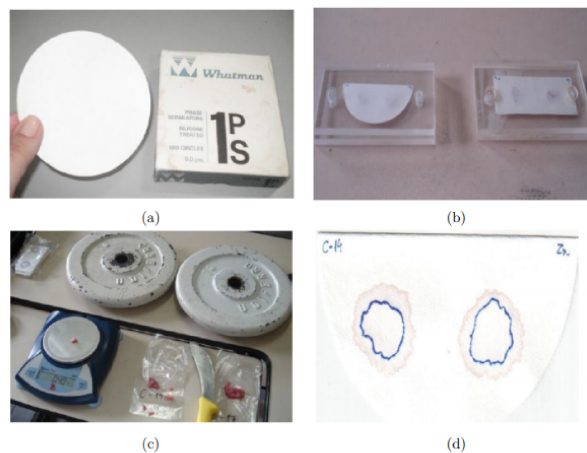


Figura 7 – Etapas do método de compressão. Imagem retirada de ([ALVES, 2011](#))

Ao término do trabalho é concluído que a utilização dos minerais: Mg e Se por um período de 12 horas melhorou a qualidade da carne em comparação às carnes que não

tiveram a utilização desses materiais.

Apesar deste trabalho utilizar o método de compressão, existem outros métodos que podem ser utilizados para a mensuração da CRA podemos citar o trabalho de (KATO, 1981) que utiliza um método não destrutivo, que captura as imagens das amostras de peito de frango por meio de uma câmera digital e avalia a relação da coloração das amostras com a capacidade de retenção de água, concluindo que amostras mais claras continham CRA menores do que amostras com coloração normal ou escura.

Como uma alternativa ao método de compressão também podemos citar o trabalho de (ELMASRY; SUN; ALLEN, 2011) que analisa a amostra de 27 touros de diferentes regiões utilizando imagens hiperespectrais em sequência com a finalidade de descobrir o CRA da amostra, segundo os autores esse método obteve uma taxa de acerto acima de 80% demonstrando assim ser uma alternativa, como um método não invasivo para predição de níveis de CRA em carnes bovinas.

Além de trabalhos que utilizam o CRA para analisar a qualidade da carne, citaremos também os trabalhos que utilizam técnicas de PDI para automatizar o processo de análise e quantificação de CRA.

No trabalho de (ANDRADE, 2019) é utilizado técnicas de análise de imagem para auxiliar na análise de carnes por CRA, apesar desse trabalho citado não utilizar nenhuma técnica de RNA podemos utilizar as técnicas de PDI para facilitar e generalizar o aprendizado da rede pois podemos aplicar essas técnicas ao dataset de treinamento e também podemos utilizar essas técnicas para comparar a eficiência da solução proposta por esse trabalho.

O artigo de (de Andrade et al., 2020) propõe uma nova técnica de *image enhancement* para amostras de CRA mensuradas pelo método de compressão, essa técnica facilita a identificação das diferentes regiões da imagem além de aplicar correções de iluminação de cores. Nesse artigo os autores comparam a técnica proposta com técnicas já utilizadas como: *Gray world*, *White patch*, *LLEA*, *CLAHE*, *LIME* e concluem que a técnica proposta apresentar uma superioridade em relação a essas outras técnicas.

4 Metodologia

4.1 TensorFlow

TensorFlow é um *framework* amplamente utilizado para a modelagem e execução de *machine learning*. Uma das vantagens é que a modelagem utiliza *dataflow graphs* que possibilita uma otimização do código durante a fase de modelagem. Outra possibilidade é a execução do treinamento utilizando outras arquiteturas de processamentos, atualmente o *TensorFlow* suporta: CPUs, GPUs, TPUs possibilitando assim uma flexibilidade para os desenvolvedores. O *TensorFlow* foi criado pela Google: “Vários serviços do Google usam o TensorFlow em produção, nós o lançamos como um projeto de código aberto e ele se tornou amplamente usado para pesquisas de aprendizado de máquina”(ABADI et al., 2016).

Para implementar nossa abordagem utilizamos a linguagem Python na versão 3.7.10 e o framework *Tensorflow* na versão 2.4.0 na variante otimizado para GPU conhecida como *Tensorflow-GPU*. Para a execução do código utilizamos uma máquina Intel i7 3770, 16GB de memória ram, placa de vídeo GTX 1650 4gb e um disco rígido de 1Terabytes.

4.2 Arquitetura

Neste trabalho iremos utilizar a arquitetura *Unet* escolhemos essa arquitetura pois a mesma apresenta bons resultados para dataset reduzidos e também porque já é uma arquitetura bastante utilizada na literatura para a segmentação de imagens.

4.3 Hiperparâmetros

Normalmente, na fase de treinamento da rede precisamos fornecer os hiperparâmetros. Esses parâmetros são parâmetros que controlam como a rede vai aprender e normalmente são definidos pelo programador. Para que a rede consiga gerar bons resultados o programador deve escolher esses parâmetros de forma inteligente.

Para realizar a fase de treinamento utilizamos os seguintes hiperparâmetros: tamanho da imagem, número de épocas, tamanho do lote, porcentagem de validação, tamanho da rede *Unet*, função de métrica, função de ativação. Os valores dos hiperparâmetros podem ser visualizados na Tabela 1.

Tabela 1 – Hiperparâmetros utilizados.

Nome	Valor
Tamanho da imagem	64×64 , 128×128 e 256×256
Número de épocas	30
Tamanho do lote	8 e 4
Porcentagem de validação	20%
Tamanho da rede <i>Unet</i>	64×64 , 128×128 e 256×256
Função de métrica	<i>Intersection-Over-Union</i>
Função de ativação	<i>softmax</i> ou <i>sigmoide</i>

4.3.1 Tamanho da imagem

Devemos escolher o tamanho da imagem para padronizar o dataset, mas principalmente porque não é possível utilizar as imagem em seu tamanho original. Atualmente é comum encontrar imagens em *full hd* e até mesmo em *4k*. Essas imagens possuem um grande número de pixels, necessitando assim a criação de redes *Unet* maiores.

Para treinar redes maiores é demandado mais tempo e é necessário placa de vídeos com mais memória. Para a avaliação da nossa abordagem estamos limitados a trabalhar com uma placa de vídeo de 4GB de memória, portanto escolhemos 3 tamanhos de imagens: 64×64 , 128×128 e 256×256 pixels.

4.3.2 Número de épocas

Para realizar a etapa de treinamento devemos definir por quantas épocas realizaremos esse treinamento. Ao utilizar um número de épocas maior, é esperado que a rede produza melhores resultados, porém, a etapa de treinamento consumirá mais tempo para ser concluída, além disso um número de épocas muito grande pode demorar muito tempo para treinar e acabar gerando *overfitting*.

Para a avaliação da nossa abordagem utilizamos 30 épocas para realizar o treinamento da rede.

4.3.3 Tamanho do lote

O tamanho do lote define quantas imagens devemos processar para atualizar os pesos da rede. Ao utilizar um tamanho de lote grande nosso treinamento será mais rápido, porém, precisará de um poder computacional maior.

Para implementar a nossa abordagem escolhemos o tamanho de lote 8 para as redes com tamanho de imagem 64×64 e 128×128 e tamanho de lote 4 para a rede com tamanho de imagem 256×256 . Tivemos que diminuir o tamanho do lote na rede 256×256

pois estamos utilizando a placa de vídeo com 4GB de memória e com essa placa não foi possível realizar o treinamento com tamanho de lote 8.

4.3.4 Porcentagem de validação

Para que a nossa rede não tenha um *overfitting* devemos separar uma parte do dataset para validação. Para a avaliação da nossa abordagem utilizamos 20% do dataset para validação.

4.3.5 Tamanho da rede *Unet*

O tamanho da rede *Unet* mais utilizada é a rede *Unet* com no máximo 256 filtros e 5 níveis de profundidade, é esperado que essa rede produza bons resultados com imagens de tamanho 256×256 e 512×512 . Porém, ao utilizar imagens menores é necessário diminuir o número máximos de filtros e a profundidade, pois a cada nível de profundidade a imagem é dividida na metade, logo imagens de tamanho 64×64 se forem utilizadas em redes com 5 níveis de profundidades serão trabalhadas como imagens 4×4 na camada mais profunda. Com esse tamanho a rede terá dificuldade em realizar a segmentação da imagem.

Para a avaliação da nossa abordagem implementamos redes com no máximo 256 filtros e profundidade 5 para imagens 256×256 , 128 filtros e profundidade 4 para imagens 128×128 e 64 filtros e profundidade 3 para imagens 64×64 .

4.3.6 Função de métrica

Para otimizar e monitorar o aprendizado da rede podemos utilizar uma função de métrica. As funções de métricas são utilizadas no momento de atualização dos pesos da rede para calcular o tanto que a rede está acertando. Para a avaliação da nossa solução inicialmente utilizamos a função de acurácia binária mas como a maior parte da imagem vai ser classificada como pixel de fundo caso a rede responda com todos os pixel como pixel de fundo a função de acurácia retornaria resultado próximo de 0,75 portanto a função foi trocada pela função *Intersection-Over-Union* essa função é calculada como:

$$IoU = \frac{true_positive}{true_positive + false_positive + false_negative} \quad (4.1)$$

Portanto utilizando a função *Intersection-Over-Union* podemos evitar que a rede tente segmentar toda a imagem como pixel de fundo.

4.3.7 Função de ativação

As funções de ativação calculam como vai ser a saída das camadas de rede. Apesar da rede possuir várias camadas, normalmente é necessário modificar apenas a função de ativação da última camada. Para a avaliação da nossa solução testamos duas abordagens.

A primeira abordagem utiliza a função de ativação *sigmoide*. Essa função é ideal para a realização da segmentação binária, com isso podemos criar uma rede que é especializada em segmentar a área externa da amostra e outra rede especializada em segmentar a área interna da amostra.

A outra abordagem é utilizar a função de ativação *softmax*. Com essa função a última camada pode ter n filtros. Com isso, podemos construir a última camada com 3 filtros: uma camada representando o fundo, outra a área externa e outra a área interna. Com isso podemos realizar a segmentação multi classe da imagem.

4.4 Conjunto de imagens

Para avaliar nossa abordagem, utilizamos um conjunto de dados composto por 659 imagens de CRA de tamanho variados. Todas as imagens foram disponibilizadas pela pesquisadora Isaura Maria Ferreira do Programa de Pós-Graduação em Ciências Veterinárias da Universidade Federal de Uberlândia.

Cada imagem corresponde a um papel de filtro contendo duas amostras de carne do mesmo animal. Os papéis de filtro são marcados (área de carne prensada) com uma caneta esferográfica no verso do papel. Após a secagem do papel, essas imagens foram fotografadas com um *smartphone*. Esse processo foi realizado em diferentes *smartphones* e por diferentes pessoas, resultando em imagens com diferentes orientações e condições ambientais (como iluminação e contraste). Devemos ressaltar que essas imagens foram adquiridas por pesquisadores da medicina veterinária que não estão familiarizados com os desafios nas tarefas de processamento de imagens. Como resultado, essas imagens não foram adquiridas em condições controladas, por exemplo do ponto de vista fixo, iluminação e a cor da caneta esferográfica. Foi aplicado uma operação de rotação em imagens orientadas verticalmente, para que todas as imagens possam ser vistas horizontalmente. Foi utilizado o software ImageJ® para cortar manualmente as imagens, selecionando assim apenas a região do papel de filtro contendo as amostras de carne e descartando todas as informações de fundo.

5 Resultados

Neste capítulo são apresentados os resultados da implementação da nossa abordagem. O objetivo da apresentação desses resultados é a comparação da eficiência das seguintes redes:

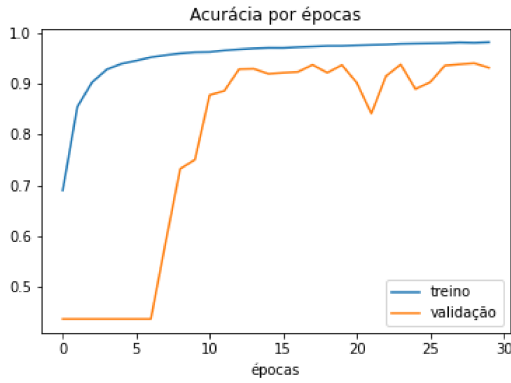
- Segmentação binária da região externa de imagens 64×64 ;
- Segmentação binária da região externa de imagens 128×128 ;
- Segmentação binária da região externa de imagens 256×256 ;
- Segmentação binária da região interna de imagens 64×64 ;
- Segmentação binária da região interna de imagens 128×128 ;
- Segmentação binária da região interna de imagens 256×256 ;
- Segmentação multi classe de imagens 64×64 ;
- Segmentação multi classe de imagens 128×128 ;
- Segmentação multi classe de imagens 256×256 .

A implementação do treinamento dessas redes pode ser encontrado no *Github* pelo link: <https://github.com/viniciusCSreis/ufu-tcc>.

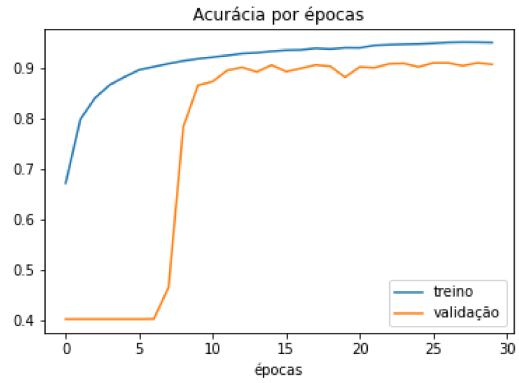
5.1 Resultados obtidos considerando a saída original da rede

As Figuras 8, 9 e 10 apresentam a evolução das redes em cada época. A Figura 8 representa as redes que utilizaram imagens 64×64 , a Figura 9 representa as redes que utilizaram imagens 128×128 , e a Figura 10 representa as redes que utilizaram imagens 256×256 . É possível observar que todas as redes conseguem evoluir rapidamente durante as primeiras 10 épocas. Porém, após esse período, a evolução é mais lenta. Também é possível observar que as redes multi classe possuem uma tendência instável de aprendizagem. Para corrigir esse problema utilizamos a ferramenta de *callbacks* do *TensorFlow* para considerar apenas a época que possuir o melhor resultado.

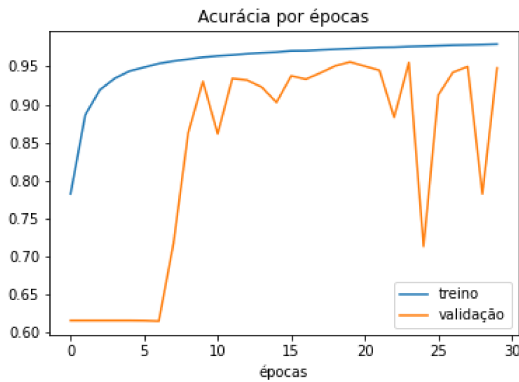
Após o treinamento geramos o resultado para todo o *dataset* e calculamos a métrica das redes utilizando a função *Intersection-Over-Union*. Para a realização de uma melhor avaliação dos resultados, a Tabela 2 apresenta o resultado *IoU* gerada pelas redes



(a) Segmentação binária da região interna.



(b) Segmentação binária da região externa.



(c) Segmentação multi classe.

Figura 8 – IoU por épocas obtida pelas redes utilizando imagens 64×64 .

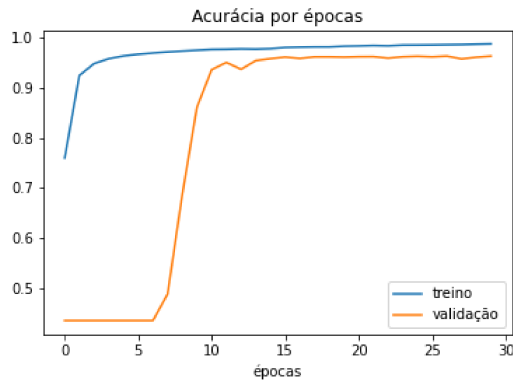
propostas considerando todo o dataset. Já a Tabela 3 considera apenas a parte de treino do *dataset*, e a Tabela 4 considera apenas a parte de validação do *dataset*.

É possível observar que as redes multi classes apresentaram um resultado mais consistente, pois se consideramos a coluna “mínimo”, que representa qual foi a pior segmentação gerada pela rede. As redes especializadas na região externa tiveram como melhor resultado 75,19% e as redes especializadas na região interna tiveram 85,34% enquanto as redes multi classes tiveram 95,34%.

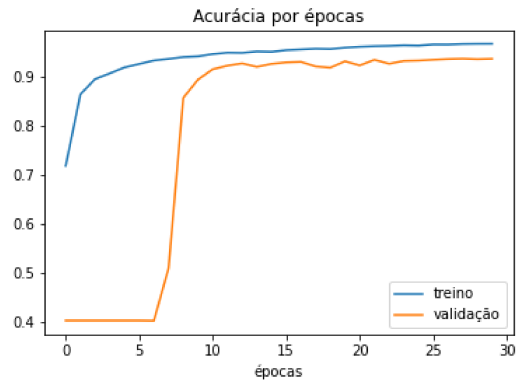
Considerando apenas a Tabela 4, que apresenta a IoU para o *dataset* de validação, também é possível constatar que as redes multi classes tiveram a melhor média e mediana no conjunto de validação.

Além disso, comparando as Tabelas 3 e 4 é fácil compreender que apesar das redes especializadas na região interna possuírem uma média e mediana parecida com as redes multi classe, existe uma variação muito grande na coluna “mínimo”, demonstrando assim que essa rede apresenta um *overfitting* maior, isto é, o conhecimento aprendido durante o treino não é genérico o suficiente para obter o mesmo desempenho na validação.

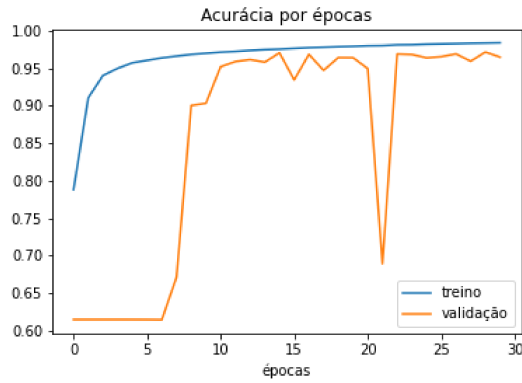
Também é possível afirmar que as redes especializadas na região externa tiveram



(a) Segmentação binária da região interna.



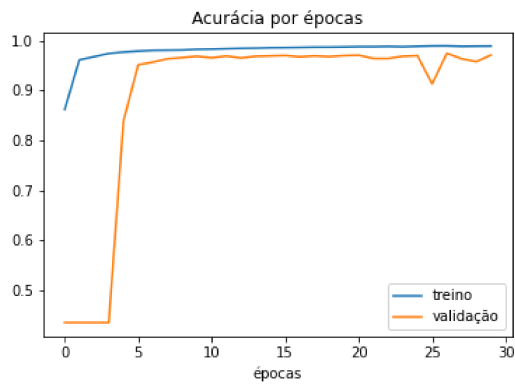
(b) Segmentação binária da região externa.



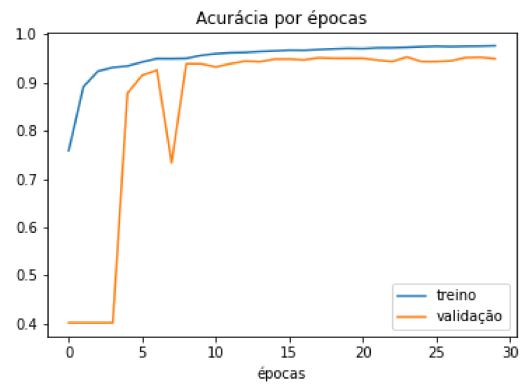
(c) Segmentação multi classe.

Figura 9 – IoU por épocas obtida pelas redes utilizando imagens 128×128 .Tabela 2 – IoU de todo o *dataset*.

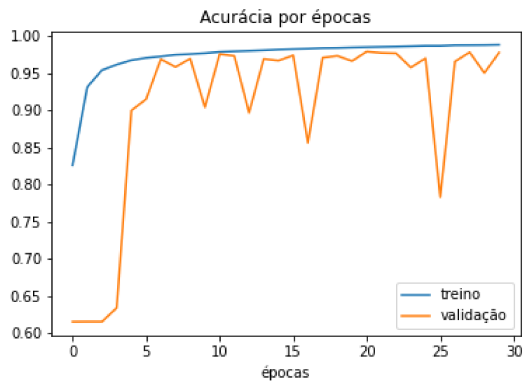
Região	Tamanho	Média	Mediana	Mínimo	Máximo
externa	64×64	93,30%	94,22%	67,23%	96,97%
externa	128×128	95,47%	96,15%	73,81%	97,67%
externa	256×256	96,21%	96,76%	71,45%	98,20%
interna	64×64	96,32%	96,95%	75,19%	98,61%
interna	128×128	97,81%	98,16%	85,17%	99,03%
interna	256×256	98,62%	98,90%	85,34%	99,42%
multi classe	64×64	96,79%	96,99%	71,00%	98,37%
multi classe	128×128	97,54%	97,70%	90,37%	98,58%
multi classe	256×256	98,43%	98,54%	95,34%	99,15%



(a) Segmentação binária da região interna.



(b) Segmentação binária da região externa.



(c) Segmentação multi classe.

Figura 10 – IoU por épocas obtida pelas redes utilizando imagens 256×256 .Tabela 3 – IoU do *dataset* de treino.

Região	Tamanho	Média	Mediana	Mínimo	Máximo
externa	64×64	93,89%	94,56%	73,21%	96,97%
externa	128×128	95,99%	96,38%	76,19%	97,67%
externa	256×256	96,55%	96,91%	81,98%	98,20%
interna	64×64	96,98%	97,10%	93,40%	98,61%
interna	128×128	98,22%	98,28%	96,07%	99,03%
interna	256×256	98,92%	98,97%	96,74%	99,42%
multi classe	64×64	97,08%	97,13%	91,70%	98,37%
multi classe	128×128	97,66%	97,75%	92,96%	98,58%
multi classe	256×256	98,56%	98,61%	97,16%	99,15%

Tabela 4 – *IoU* do *dataset* de validação.

Região	Tamanho	Média	Mediana	Mínimo	Máximo
externa	64×64	90,92%	91,88%	67,23%	94,97%
externa	128×128	93,39%	94,18%	73,81%	96,36%
externa	256×256	94,83%	95,69%	71,45%	97,46%
interna	64×64	93,67%	95,16%	75,19%	98,29%
interna	128×128	96,19%	96,79%	85,17%	98,52%
interna	256×256	97,39%	97,84%	85,34%	99,14%
multi classe	64×64	95,62%	96,16%	71,00%	97,58%
multi classe	128×128	97,08%	97,49%	90,37%	98,23%
multi classe	256×256	97,92%	98,11%	95,34%	98,66%

maior dificuldade no aprendizado. Um dos motivos para isso acontecer é que a marcação da região interna é feita a caneta, portanto é mais fácil identificar a suas bordas ao contrário da região externa, que pode ser facilmente confundida com o papel.

A Figura 11 exemplifica uma dificuldade da rede em encontrar a região externa, visto que a imagem de entrada apresenta uma região de área externa com um tamanho reduzido e uma cor parecida com a do papel.

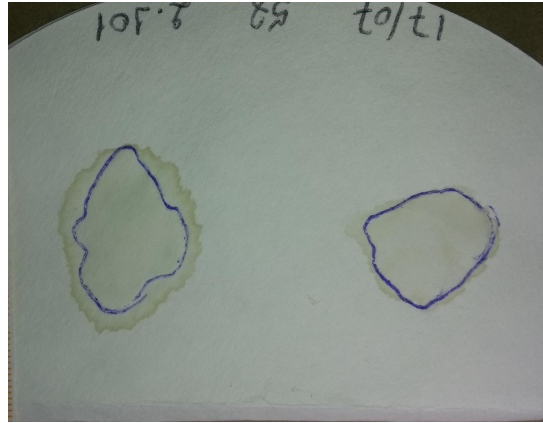
5.2 Resultados obtidos considerando a saída transformada da rede

As Tabelas 2, 3 e 4 apresentam a *IoU* das imagens que são geradas pelas redes sem nenhuma transformação. Porém é esperado que as redes gerem uma imagem em tamanho menor e que o desenvolvedor aumente as imagens resultantes para o seu tamanho original. Para transformar as imagens para seu tamanho original foi utilizada a função *resize*, da biblioteca *OpenCV*, com o parâmetro *interpolation* com o valor *INTER NEAREST*. Com isso ao transformar o tamanho da imagens não geramos novos tons de cor na imagem.

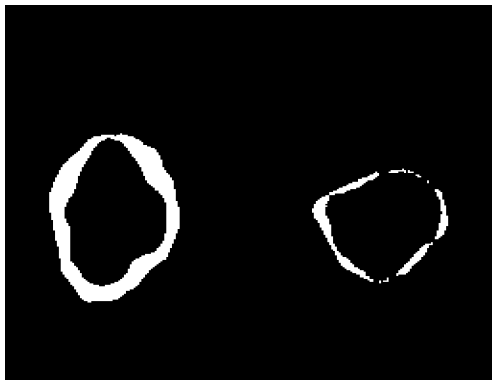
Após a transformação das imagens para seu tamanho original foi calculado a *IoU* das novas imagens gerando assim as Tabelas 5, 6 e 7. A Tabela 5 apresenta a *IoU* considerando todo o *dataset*, já a Tabela 6 considera apenas a parte de treino do *dataset*, e a Tabela 7 considera apenas a parte de validação do *dataset*.

Analizando as Tabelas 6 e 7 é possível afirmar que o seu padrão é o mesmo das Tabelas 3 e 4, logo a relação dos resultados do *dataset* de treino e validação se manteve a mesma.

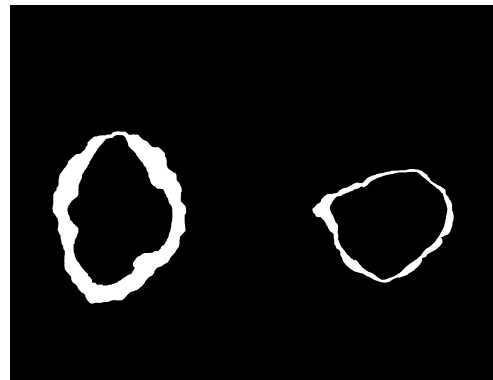
Porém, ao observar os novos valores de *IoU* é possível perceber que as redes que utilizam imagens menores apresentam uma variação maior na métrica *IoU* em relação a imagem reconstruída para seu tamanho original. Por exemplo, a região externa de tamanho 64×64 alterou sua média de 93,30% para 90,63% uma variação de 2,67%. Já



(a) Entrada



(b) Resultado gerado



(c) Resultado esperado

Figura 11 – Exemplo da dificuldade em encontrar a região de área externa na rede de segmentação binária da região externa de imagens 256×256 .

a região externa de tamanho 256×256 alterou sua média de 96,21% para 95,81%, uma variação de apenas 0,4%.

Esse comportamento é esperado visto que quanto menor a imagem mais se perde ao aumentar o seu tamanho para o original. Esse comportamento pode ser observado na Figura 12 que apresenta a evolução dos resultados das imagens em redes de com tamanhos de imagens diferentes.

Por meio da Figura 12 também é possível explicar porque as redes multi classes apresentam um melhor resultado. Como existe uma relação entre área externa e área interna, isto é, a área externa sempre vai estar após a borda da área interna, a rede consegue aprender com mais facilidade se analisar as duas áreas em conjunto.

Tabela 5 – *IoU* de todo o *dataset* utilizando imagens em seu tamanho original.

Região	Tamanho	Média	Mediana	Mínimo	Máximo
externa	64×64	90,63%	91,52%	66,82%	94,21%
externa	128×128	94,35%	95,02%	73,95%	96,70%
externa	256×256	95,81%	96,36%	71,30%	97,78%
interna	64×64	95,00%	95,59%	75,02%	96,87%
interna	128×128	97,25%	97,57%	84,93%	98,33%
interna	256×256	98,34%	98,60%	85,11%	99,03%
multi classe	64×64	95,61%	95,76%	70,36%	97,47%
multi classe	128×128	97,14%	97,31%	90,17%	98,31%
multi classe	256×256	98,24%	98,33%	95,31%	98,99%

Tabela 6 – *IoU* do *dataset* de treino utilizando imagens em seu tamanho original.

Região	Tamanho	Média	Mediana	Mínimo	Máximo
externa	64×64	91,02%	91,86%	68,79%	94,21%
externa	128×128	94,76%	95,26%	74,53%	96,70%
externa	256×256	96,12%	96,53%	80,72%	97,78%
interna	64×64	95,58%	95,72%	92,88%	96,87%
interna	128×128	97,61%	97,66%	95,78%	98,33%
interna	256×256	98,62%	98,66%	96,46%	99,03%
multi classe	64×64	95,84%	95,88%	90,68%	97,47%
multi classe	128×128	97,25%	97,34%	92,72%	98,31%
multi classe	256×256	98,36%	98,40%	97,00%	98,99%

Tabela 7 – *IoU* do *dataset* de validação utilizando imagens em seu tamanho original.

Região	Tamanho	Média	Mediana	Mínimo	Máximo
externa	64×64	89,08%	89,95%	66,82%	93,03%
externa	128×128	92,73%	93,47%	73,95%	95,57%
externa	256×256	94,56%	95,48%	71,30%	97,23%
interna	64×64	92,70%	94,21%	75,02%	96,55%
interna	128×128	95,82%	96,45%	84,93%	97,91%
interna	256×256	97,21%	97,68%	85,11%	98,90%
multi classe	64×64	94,71%	95,23%	70,36%	96,54%
multi classe	128×128	96,73%	97,16%	90,17%	97,87%
multi classe	256×256	97,78%	97,96%	95,31%	98,48%

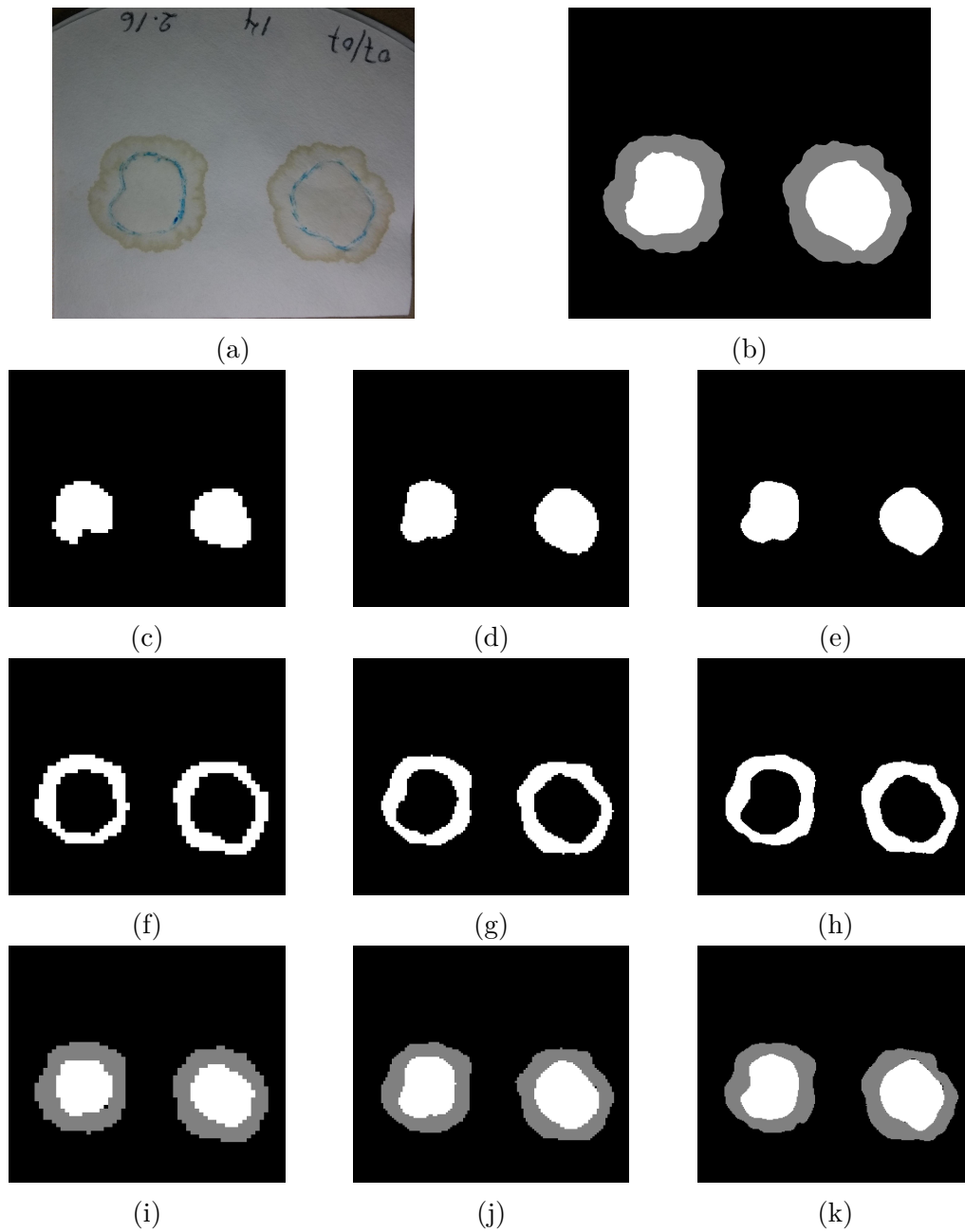


Figura 12 – Exemplo de resultado. a)Imagem Original b)Resultado esperado c)Segmentação binária da região interna de imagens 64×64 d)Segmentação binária da região interna de imagens 128×128 e)Segmentação binária da região interna de imagens 256×256 f)Segmentação binária da região externa de imagens 64×64 g)Segmentação binária da região externa de imagens 128×128 h)Segmentação binária da região externa de imagens 256×256 i)Segmentação multi classe de imagens de imagens 64×64 j)Segmentação multi classe de imagens de imagens 128×128 k)Segmentação multi classe de imagens 256×256

6 Conclusão

Nesse trabalho apresentamos a utilização de redes neurais convolucionais para a segmentação da área externa e interna de fotos de amostras da capacidade de retenção de água de carne suína. A nossa solução demonstrou ótimos resultados, mesmo sem a utilização de *data augmentation* ou a necessidade de placa de vídeos com mais de 4GB de memória.

Por meio dessa solução é possível inserir a imagem na rede sem a necessidade de pre-processamento além da redução do tamanho da imagem. Demostramos uma solução escalável que se beneficia com o aumento do *dataset*, mas mesmo com um *dataset* reduzido apresentou bons resultados.

Para trabalhos futuros sugerimos o treinamento da rede com *datasets* de amostras de carnes de outros animais além da comparação da arquitetura *Unet* com arquiteturas mais novas como *LinkNet* e *PSPNet*.

Referências

- ABADI, M. et al. Tensorflow: A system for large-scale machine learning. In: *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*. [S.l.: s.n.], 2016. p. 265–283. Citado na página 19.
- ABDULLAH, B. et al. Water-holding capacity assessment of meat using an electromagnetic sensing method. In: . [S.l.: s.n.], 2013. p. 11–16. ISBN 978-1-4799-2214-7. Citado na página 9.
- ABPA, A. B. de P. A. *Produção Brasileira de Carne Suína (Mil Ton)*. 2020. Disponível em: <<http://abpa-br.org/mercados/>>. Citado na página 9.
- ALVES, L. R. *Qualidade de carne suína. 1. efeito do gene halotano sobre a deposição de gordura intramuscular. 2. efeito da suplementação com minerais no pré-abate*. 2011. Disponível em: <<https://repositorio.ufu.br/handle/123456789/13017>>. Citado na página 17.
- ANDRADE, R. G. de. *Avaliação da capacidade de retenção de água em amostras de carne suína*. 2019. Disponível em: <<https://repositorio.ufu.br/handle/123456789/27552>>. Citado 2 vezes nas páginas 9 e 18.
- Barca, M. C. S.; Silveira, T. R. de S.; Magini, M. *TREINAMENTO DE REDES NEURAIS ARTIFICIAIS: O ALGORITMO BACKPROPAGATION*. 2005. Disponível em: <http://www.inicepg.univap.br/cd/INIC_2005/inic/IC1%20anais/IC1-17.pdf>. Citado na página 11.
- BRAGA ANTONIO; LUDEMIR, T. C. *Redes Neurais Artificiais - Teoria e Aplicações*. Rio de janeiro: LTC, 2000. Citado na página 11.
- de Andrade, R. G. et al. Image enhancement applied to water holding capacity images. In: *2020 International Conference on Systems, Signals and Image Processing (IWSSIP)*. [s.n.], 2020. p. 199–204. Disponível em: <<https://doi.org/10.1109/IWSSIP48289.2020.9145280>>. Citado na página 18.
- ELMASRY, G.; SUN, D.-W.; ALLEN, P. Non-destructive determination of water-holding capacity in fresh beef by using nir hyperspectral imaging. *Food Research International*, v. 44, n. 9, p. 2624 – 2633, 2011. ISSN 0963-9969. Disponível em: <<https://doi.org/10.1016/j.foodres.2011.05.001>>. Citado na página 18.
- JOO, S.-T. Determination of water-holding capacity of porcine musculature based on released water method using optimal load. *Korean journal for food science of animal resources*, The Korean Society for Food Science of Animal Resources, v. 38, n. 4, p. 823, 2018. Citado na página 9.
- KATOH, K. An improved method for determining water holding capacity of meat. *Nihon Chikusan Gakkaiho*, v. 52, n. 2, p. 139–141, 1981. Disponível em: <<https://doi.org/10.2508/chikusan.52.139>>. Citado na página 18.

- LI JUSTIN JOHNSON, S. Y. F.-F. *Lecture 11: Detection and Segmentation*. 2017. Disponível em: <http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture11.pdf>. Citado na página 16.
- OSÓRIO, J. R. B. F. *Sistemas Inteligentes baseados em Redes Neurais Artificiais aplicados ao Processamento de Imagens*. 2000. Disponível em: <<http://osorio.wait4.org/oldsite/wia-unisc/wia2000-mini.pdf>>. Citado na página 11.
- RONNEBERGER, O.; FISCHER, P.; BROX, T. U-net: Convolutional networks for biomedical image segmentation. In: SPRINGER. *International Conference on Medical image computing and computer-assisted intervention*. 2015. p. 234–241. Disponível em: <https://doi.org/10.1007/978-3-319-24574-4_28>. Citado 2 vezes nas páginas 15 e 16.
- ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, American Psychological Association, v. 65, n. 6, p. 386, 1958. Disponível em: <<https://doi.org/10.1037/h0042519>>. Citado na página 12.
- SILVA, S. R. e.; SCHIMIDT, F. REDUÇÃO DE VARIÁVEIS DE ENTRADA DE REDES NEURAS ARTIFICIAIS A PARTIR DE DADOS DE ANÁLISE DE COMPONENTES PRINCIPAIS NA MODELAGEM DE OXIGÊNIO DISSOLVIDO. *Química Nova*, scielo, v. 39, p. 273 – 278, 04 2016. ISSN 0100-4042. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0100-40422016000300273&nrm=iso>. Citado na página 12.
- YAMASHITA, R. et al. Convolutional neural networks: an overview and application in radiology. *Insights into imaging*, Springer, v. 9, n. 4, p. 611–629, 2018. Disponível em: <<https://doi.org/10.1007/s13244-018-0639-9>>. Citado 2 vezes nas páginas 14 e 15.

Apêndices

Anexos