Controle de Qualidade de *Citrus limon* Utilizando Aprendizado por Transferência

BRYAN MAURICIO DE SOUZA FRANCISCO[1]; JOÃO PEDRO AGUILAR VICARI[1]; JOÃO RICARDO FAVAN[2]

[1] Discentes em Big Data no Agronegócio na FATEC Pompeia “Shunji Nishimura”, Pompeia-SP.
[2] Docente em Big Data no Agronegócio na FATEC Pompeia “Shunji Nishimura”, Pompeia-SP.

RESUMO

O trabalho visa explicitar o problema do desperdício na Agricultura, com atenção voltada para verificar seu efeito para pequeno e médio agricultor, e como o problema pode ser remediado por meio da utilização de procedimentos de Controle de Qualidade da Produção. Visando elaborar parte de uma solução, o trabalho discorre sobre a utilização de tecnologias de aprendizado de máquina e técnicas de aprendizado por transferência para a criação de um modelo capaz de identificar frutos de *Citrus limon* que apresentem algum tipo de defeito ou doença. Por fim, o trabalho constata, através de diversas métricas de avaliação, diversos métodos de avaliação e de teste estatístico, a viabilidade da utilização de ambas tecnologias para o controle de qualidade no cultivo do limão siciliano, apresentando assertividade entre 99,1% e 99,7% na versão final do modelo.

Palavras-chave: Limão. Controle de Qualidade. Aprendizado de Máquina. Aprendizado por Transferência.

INTRODUÇÃO

No ambiente do agronegócio consta como um problema grave o desperdício de culturas. Segundo Quintão (2021), são desperdiçados no Brasil em média 27 milhões de toneladas de culturas por ano. O país é o segundo maior exportador da citricultura no mundo, sendo responsável por 32,8% da produção mundial de citros, de acordo com Santos *et al* (2022), onde 56 mil hectares correspondem à produção de limão (CNA, 2021). No mercado dos citros, parte do desperdício é resultante da falta do Controle da Qualidade do produto final, visto ser um processo custoso. Isto acarreta um impacto maior para quem é pequeno produtor ou que cultiva para o consumo próprio.

Diversos tipos de tecnologias têm sido utilizados no âmbito agrícola visando, entre outros objetivos, mitigar os impactos causados pelo desperdício. Uma dessas tecnologias é o Aprendizado de Máquina, mais conhecido como *Machine Learning,* que já se demonstrou e ainda se demonstra muito impactante, como descrito na

revisão feita por Liakos *et al* (2018) a respeito de tecnologias desse tipo empregadas na área da Agricultura, além de estimular e servir de base para inovação e criação de novas tecnologias futuras, como por exemplo o *Plantix* (PEAT GmbH, 2015), um software desenvolvido pela empresa PEAT GmbH (*Progressive Enviromental & Agricultural Technologies*) que utiliza uma série de modelos de *Machine Learning* para identificar doenças em diversas plantas.

Aliado ao Aprendizado de Máquina, o Aprendizado por Transferência é uma forma de aprimorar e facilitar o processo de criação de um modelo de *Machine Learning*, podendo ser descrito como a utilização de um modelo de *deep learning* que foi treinado para um propósito, por exemplo: classificar imagens de carros, e utilizá-lo para realizar outra tarefa similar, mas para qual o mesmo não fora previamente treinado, como utilizar um modelo de classificação de imagens de carros para classificar imagens de caminhonetes (PAN e YANG, 2010). Isso se provou muito útil, conforme demonstrado pelos trabalhos de Zhang *et al* (2015) e Modarrares *et al* (2017), pois o custo computacional para treinar um modelo com alta assertividade se torna muito menor e seu processo de desenvolvimento se torna muito mais rápido. Tal método foi escolhido devido ao seu sucesso comprovado em classificar imagens *in situ* da hibridização da mosca da fruta (ZHANG *et al*, 2015); reconhecimento de imagens produzidas por um microscópio eletrônico de varredura (MODARRES *et al*, 2017); classificação de padrões de condrócitos em tomografias computadorizadas (ABIDIN *et al*, 2018) e; discriminação de imagens de patologias (KHOSRAVI *et al*, 2018).

Com base nesses dois conceitos, o presente trabalho visa utilizar as técnicas do Aprendizado de Máquina, aliado ao processo de Aprendizado por Transferência, para apresentar parte de uma solução para o desperdício de safras, visando auxiliar o pequeno e médio agricultor na tarefa de Controle de Qualidade da cultura de *Citrus limon*, conhecido popularmente como limão siciliano, ou ainda, limão verdadeiro.

MATERIAL E MÉTODOS

Linguagens, softwares e bibliotecas utilizadas

Os processos de importação de imagens, extração de parâmetros, criação dos modelos de *Machine Learning* e validação dos mesmos foram desenvolvidos de forma visual através do *software* Orange (DEMSAR *et al*, 2013) na versão 3.32.0. Para realizar tarefas de pré-processamento de imagens e utilizar métodos de validação não

suportados de forma nativa ou na forma de *add-on* (extensão) pelo *software*, foi criado um *script* de análise de dados foi desenvolvido utilizando a linguagem Python na versão 3.10.6 (VAN ROSSUM e DRAKE, 2009). Para manipulação e visualização de imagens, foram utilizadas as bibliotecas Numpy (HARRIS *et al*, 2020) e Pillow (CLARK *et al*, 2011), nas versões 0.21.0 e 9.2.0, respectivamente. Para criação dos modelos de *Machine Learning* e validação dos mesmos, foram utilizadas as bibliotecas Scikit-Learn (PEDREGOSA *et al*, 2011) na versão 1.1.2 e MLxtend (RASCHKA *et al*, 2018) na versão 0.21.3. Os gráficos gerados foram criados utilizando a biblioteca Matplotlib (HUNTER, 2007) na versão 3.5.3.

Para o desenvolvimento do *script* foi utilizado o IDE (*Integrated Development Enviroment* – Ambiente de Desenvolvimento Integrado) Visual Studio Code (MICROSOFT *et al*, 2015) na versão 1.71.0, juntamente com Jupyter Notebooks (KLUYVER *et al*, 2016).

Seleção de datasets

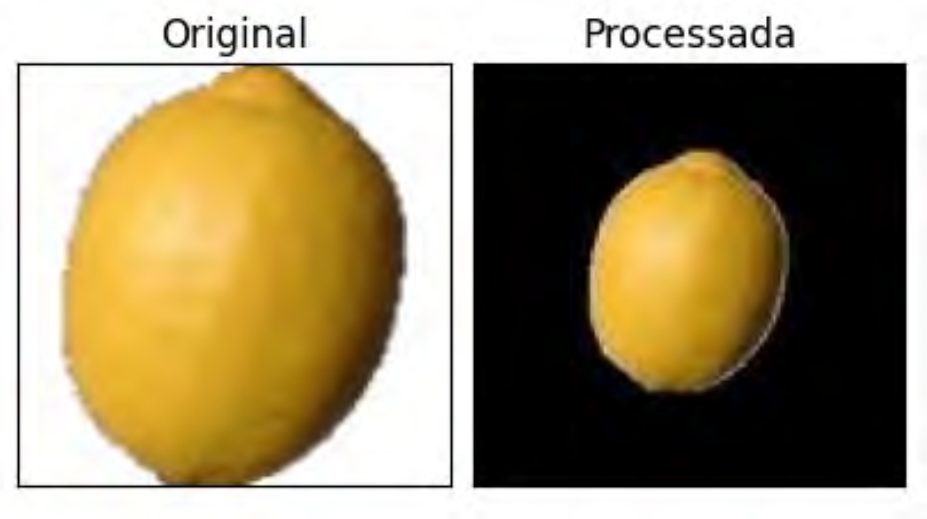
O *dataset* descrito por Adamiak (2020) foi selecionado devido a alta qualidade de imagens e a ausência de características indesejadas. Com base nas anotações propostas pelo autor, reconstruiu-se o conjunto de dados, agora separando as imagens em duas classes: *"healthy"* e "*unhealthy"* ("saudável" e "doente" em tradução livre). Devido a enorme discrepância na quantidade de imagens para cada classe, foi encontrado outro *dataset* similar, dado que o desequilíbrio na quantidade de elementos pode deteriorar a performance de classificação do modelo (CHAWLA *et al*, 2004). O conjunto de dados disponibilizado por Oltean (2020) foi escolhido para complementar o *dataset* final, visto a presença de amostras consideradas "saudáveis".

Pré-processamento

Foi realizado o pré-processamento dos dados objetivando a padronização entre as amostras utilizadas para treinamento, dado que ele apresentava uma grande diferença de resolução em comparação com as imagens do conjunto principal. Esse pré-processamento consistiu em redimensionar as imagens do *dataset* auxiliar, deixando-as com o mesmo tamanho em *pixels* que as imagens do *dataset* principal, visando reduzir a quantidade de parâmetros a serem otimizados e, por consequência, reduzir o risco de sobreajuste no modelo (apud BATTITI, 1994, SABOTTKE; SPIELER, 2020). Após isso, foi feita uma substituição de tonalidades brancas

presentes no plano de fundo das imagens originais por uma tonalidade sólida preta, com o objetivo de evitar *overfiting* (sobreajuste) e, com isso, evitar que os modelos de redes neurais passassem a memorizar os dados ao invés de aprender com eles (YING, 2019). A Figura 1 mostra um exemplo de uma imagem de *Citrus limon* retirada do *dataset* auxiliar e o resultado da mesma após realizado o pré-processamento.

Figura 1 – Comparação entre a imagem original e a imagem pré-processada.



Fonte: Autores (2022)

Comparação entre Redes Neurais Convolucionais Profundas

Foi realizada a comparação entre diferentes modelos de Redes Neurais Convolucionais Profundas (DCNN – *Deep Convolutional Neural Network*) pré-treinados no *dataset* ImageNet (DENG *et al*, 2009). Buscou-se replicar o processo de aprendizado por transferência descrito por Kiela e Bottou (2014), bem como por Godec *et al* (2019).

Foram selecionadas as arquiteturas de DCNN VGG-16 e VGG-19 (SIMONYAN e ZISSERMAN, 2015), Inception versão 3 (SZEGEDY *et al*, 2015) e SqueezeNet (IANDOLA *et al*, 2017), sendo o critério de seleção o propósito de resolução de problemas de classificação de imagens encontrado em todas as arquiteturas. Para realizar a comparação, foi utilizada uma rede neural do tipo Perceptron Multicamadas (*Multilayer Perceptron* – MLP) para realizar o processo de aprendizado dos

parâmetros extraídos de cada uma das arquiteturas DCNN. A Tabela 1 apresenta os hiperparâmetros utilizados no processo de aprendizagem.

Tabela 1 – Hiperparâmetros utilizados pelo modelo de Perceptron Multicamadas

Hiperparâmetro	**Valor**
Nº de camadas na camada oculta	1
Nº de neurônios em cada camada na camada oculta	100
Função de Ativação	ReLU
Método de Otimização	Adam
Valor alfa	0,0001
Número máximo de épocas	200

Fonte: Autores (2022)

Otimização

Após os testes iniciais, foi realizado um processo de otimização do modelo, buscando encontrar um equilíbrio entre assertividade e complexidade computacional do modelo. Buscou-se reduzir a quantidade de neurônios presentes na camada oculta (*hidden layer*) sem comprometer a eficiência e eficácia do modelo. Os testes foram realizados de forma empírica, baseando-se fortemente no Método Elbow (THORNDIKE, 1953). Na primeira fase do processo, utilizou-se de diversas quantidades de neurônios, variando de 10 a 100 neurônios. Dados os resultados encontrados nessa fase de otimização, realizou-se uma segunda fase, agora utilizando de 1 a 10 neurônios apenas.

Validação

Para validar modelos de *machine learning*, é necessário escolher um método de validação e quais métricas devem ser levadas em consideração para analisar a

performance do modelo. Para este trabalho, inicialmente foi utilizado o método de amostragem aleatória (*random sampling*), que consiste em:

> [...] dividir aleatoriamente os dados em conjuntos de treinamento e teste na proporção indicada (por exemplo 70:30); todo o procedimento é repetido por um número especificado de vezes. (DEMSAR *et al*, 2013, tradução nossa).

Também foram utilizados outros métodos para comprovar a eficiência e eficácia do modelo final e descartar a hipótese de sobreajuste, como a utilização de dados exclusivos para teste (HASTIE *et al*, 2017, p. 222), que consiste em separar uma parte do *dataset* antes de realizar o processo de treinamento de validação, e utilizá-la para testar o modelo e extrair suas métricas, sendo assim uma forma de testar a eficácia do modelo utilizando dados nunca vistos por ele. Para o presente trabalho, foram separados 25% do *dataset* original exclusivamente para esse fim, e os demais 75% para treinamento e validação do modelo. Também foi utilizado o método K-fold (HASTIE *et al*, 2017, p. 241 – 245), definido como "k = 10" dadas as recomendações de McLachlan *et al* (2004, p. 214). Para Hastie *et al* (2017), tal método consiste em separar o conjunto de dados em K partes aproximadamente iguais (também conhecidas como "partições"), selecionar uma dessas partes para servir como conjunto de validação e utilizar as demais para realizar o treinamento do modelo, repetindo o processo até que todas as partes tenham sido utilizadas exatamente uma vez como conjunto de validação, combinando as K estimativas de erro ao final do processo.

Por fim, também foram utilizados os estimadores .632 (EFFRON, 1979, p. 10 – 11) e .632+ (EFFRON e TIBSHIRANI, 1997) como métodos de validação utilizando *bootstrapping*. O *bootstrap* é um método estatístico no qual várias amostras menores são retiradas de uma amostra de uma população. Em *Machine Learning*, essas pequenas amostras são utilizadas para realizar o treinamento e validação do modelo, e juntamente com um método de avaliação, são utilizadas para gerar uma determinada métrica a respeito do modelo. Os estimadores .632 e .632+ foram utilizados como métodos de avaliação, dando ênfase para o último, visto que o mesmo

leva o cenário de sobreajuste em consideração no cálculo (EFFRON e TIBSHIRANI, 1997, p. 552).

Como métricas de avaliação, foram selecionadas “AUC” (*Area Under the ROC Curve* – Área abaixo da curva ROC), que a partir da análise ROC (*Receiver Operating Characteristic* – Curva de Característica de Operação do Receptor) calcula a área abaixo da curva gerada, indicando o desempenho do modelo em todos os limites de classificação possíveis; “CA” (*Classification Accuracy* – Acurácia de Classificação), que calcula a razão entre a quantidade de previsões feitas corretamente e a quantidade total de previsões realizadas; “F1-Score”, que calcula a média harmônica entre precisão e revocação; e “Precisão” e “Revocação” (*recall*), que calculam a razão entre os verdadeiros positivos e a soma de verdadeiros positivos e falsos positivos, e a razão entre os verdadeiros positivos e a soma de verdadeiros positivos e falsos negativos, respectivamente.

Teste Estatístico

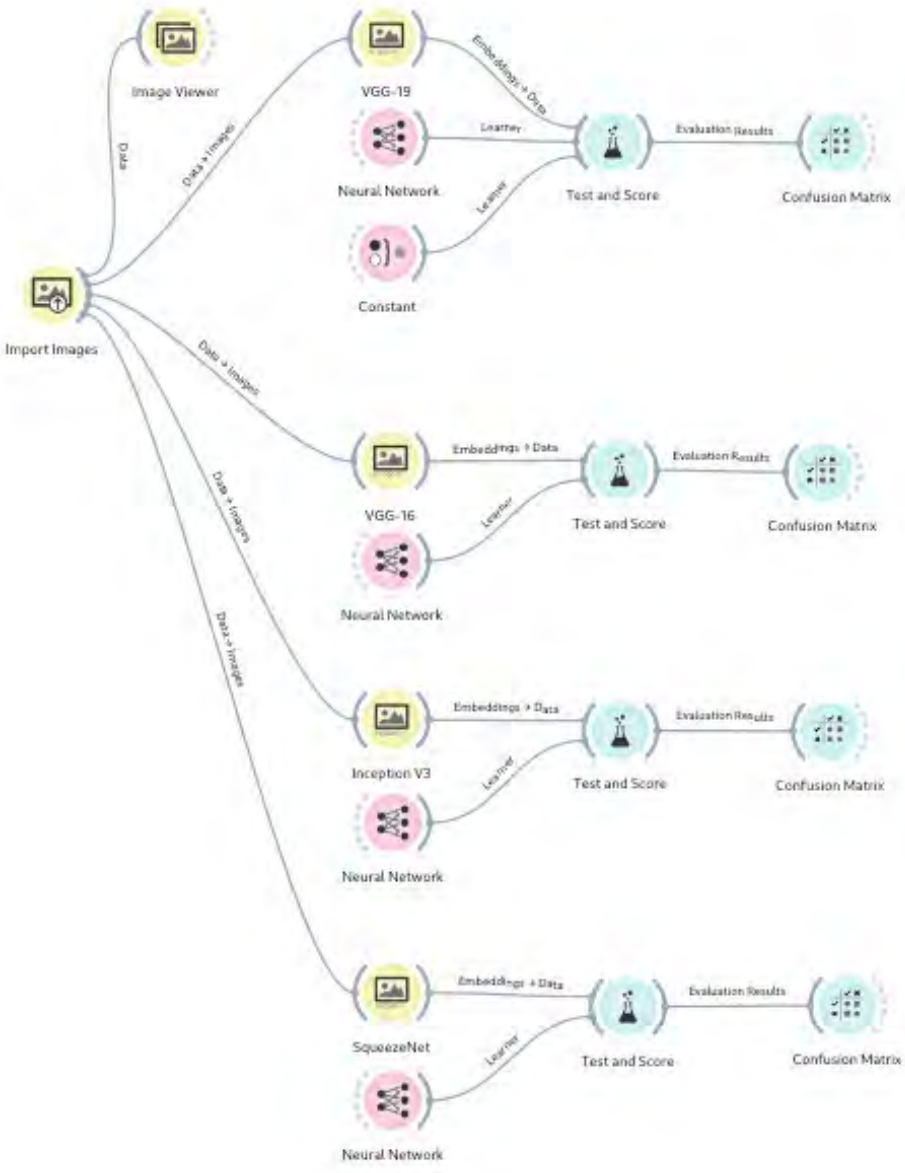
Devido a alta assertividade do modelo, surgiu a hipótese de que o modelo estivesse sobre ajustado. Realizou-se o teste não-paramétrico de Mann-Whitney, com 6 atributos do dataset, escolhidos aleatoriamente, a fim de identificar a existência da homogeneidade do conjunto de dados final.

RESULTADOS E DISCUSSÃO

Workflows desenvolvidos

Nessa seção, destacamos os processos realizados de forma visual utilizando o *software* Orange. A Figura 2 destaca o processo realizado para comparar as diferentes arquiteturas de DCNN escolhidas para o trabalho.

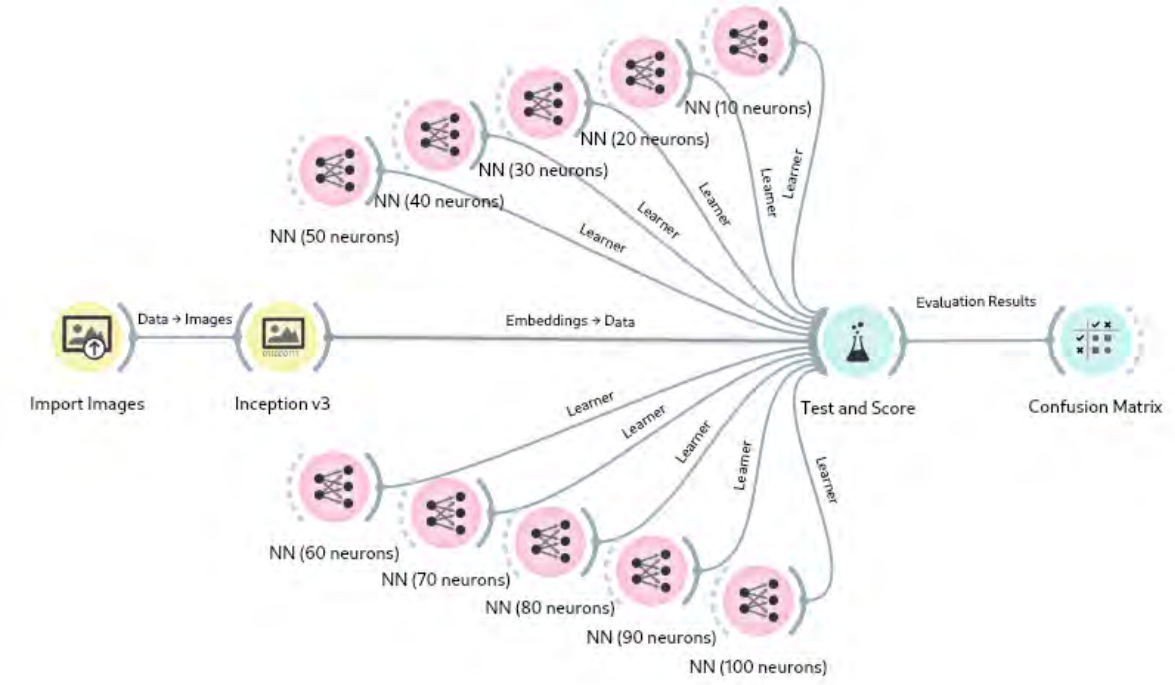
Figura 2 – *Workflow* utilizado para realizar a comparação entre as arquiteturas de DCNN, feito através do software orange-canvas



Fonte: Autores (2022)

A Figura 3 mostra o *workflow* utilizado na primeira fase de otimização do modelo de perceptron multicamadas, utilizando de 10 a 100 neurônios na camada oculta para realização de testes empíricos.

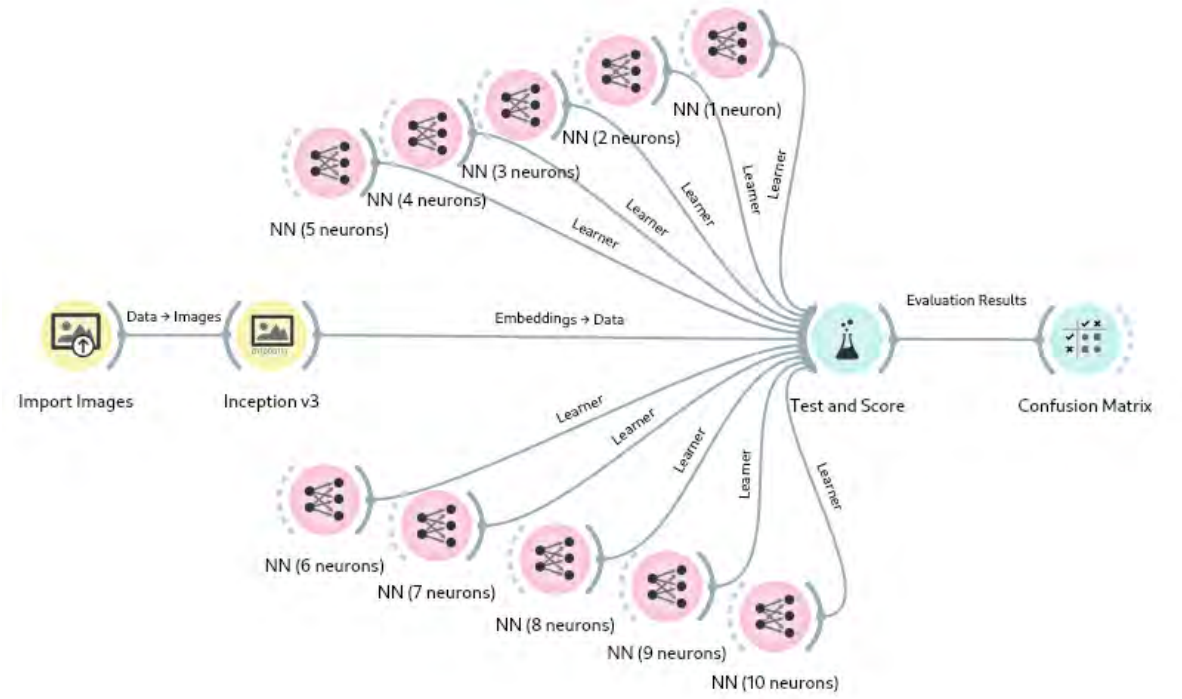
Figura 3 – *Workflow* desenvolvido utilizando o *software* orange-canvas para realização de testes empíricos utilizando diversas quantidades de neurônios nas camadas ocultas.



Fonte: Autores (2022)

A Figura 4 mostra o *workflow* desenvolvido no *software* orange-canvas e utilizado durante a segunda fase de otimização, realizando testes empíricos com modelos de perceptron multicamadas contendo 1 a 10 neurônios na camada oculta.

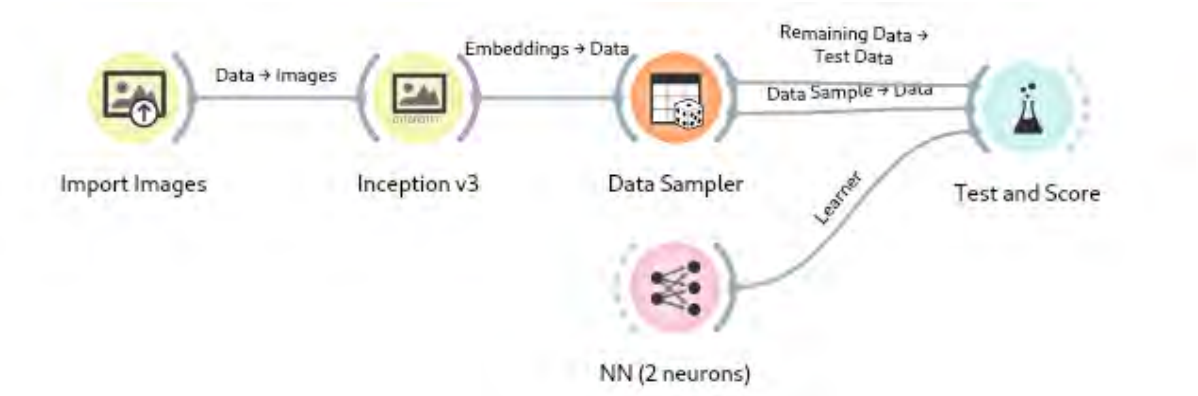
Figura 4 – *Workflow* desenvolvido para realização de testes empíricos utilizando entre 1 e 10 neurônios na camada oculta.



Fonte: Autores (2022)

A Figura 5 mostra o *workflow* desenvolvido para realizar validação do modelo final utilizando dados exclusivos para teste.

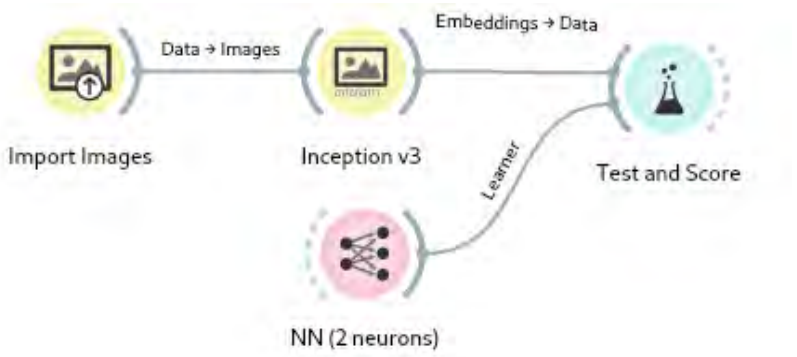
Figura 5 – *Workflow* desenvolvido no software orange-canvas para realizar a validação do modelo final utilizando dados exclusivos para teste



Fonte: Autores (2022)

A Figura 6 descreve o *workflow* utilizado no *software* orange-canvas para o processo de validação cruzada, utilizando o método de K-Fold para 10 partições.

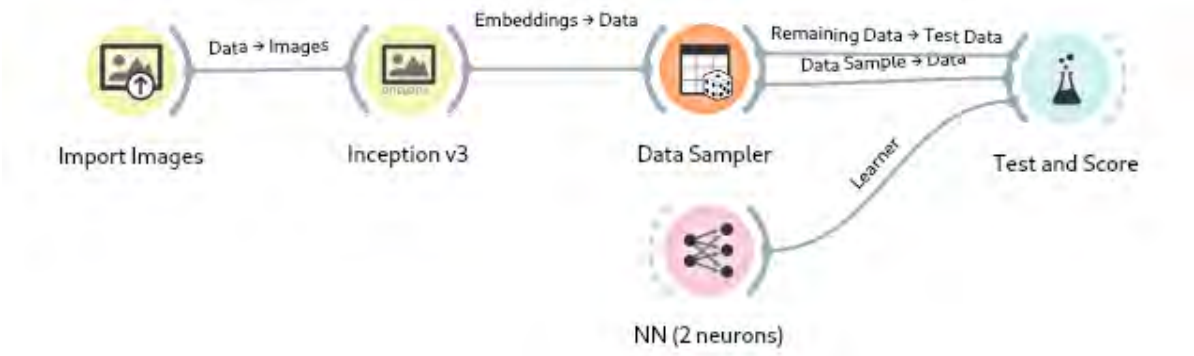
Figura 6 – *Workflow* utilizado para realizar validação cruzada com o metodo K-Fold para 10 partições



Fonte: Autores (2022)

A Figura 7 contém o *workflow* criado utilizando o *software* orange-canvas para o realizar validação cruzada do modelo final utilizando *bootstrapping*, sendo utilizado para esse fim o estimador .632.

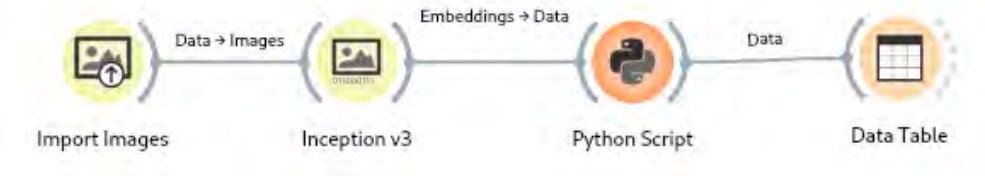
Figura 7 – *Workflow* desenvolvido para realizar validação cruzada do modelo final utilizando *bootstrapping* com o estimador .632



Fonte: Autores (2022)

Por fim, a figura 8 mostra o processo de validação com *bootstrapping*, utilizando o estimador .632+. Dado o fato de o *software* usado não possuir uma implementação nativa do estimador, foi necessário utilizar o componente “Python Script”, juntamente com a biblioteca MLxtend para alcançar tal feito.

Figura 8 – *Workflow* desenvolvido utilizando o *software* orange-canvas para validação do modelo final utilizando *bootstrapping* com o estimador .632+



Fonte: Autores (2022)

Resultados obtidos

Com base nos resultados obtidos da comparação entre as diferentes arquiteturas de DCNN escolhidas para o presente trabalho, foi escolhida a arquitetura Inception v3 por ter apresentado maior F1-Score entre todas as demais. A Tabela 2 apresenta as métricas obtidas durante a fase de comparação entre as diferentes arquiteturas de DCNN escolhidas, juntamente com a comparação entre as mesmas e um modelo de previsão baseada na classe mais frequente do *dataset*, identificado como “dummy”.

Tabela 2 – Métricas de cada arquitetura, ordenadas por F1-Score em ordem decrescente, na qual o modelo Inception v3 se demonstrou o mais preciso por apresentar maior valor de F1-Score.

Modelo	AUC	CA	F1	Precisão	Revocação
Inception v3	0,997	0,995	0,995	0,995	0,995
VGG-16	0,997	0,994	0,994	0,994	0,994
VGG-19	0,997	0,994	0,994	0,994	0,994
SqueezeNet	0,997	0,992	0,992	0,992	0,992
Constant (Dummy)	0,500	0,500	0,479	0,500	0,500

Fonte: Autores (2022)

Na Tabela 3 constam as métricas de desempenho de cada modelo de redes neurais utilizados durante a primeira fase de otimização. É possível observar que a redução no número de neurônios presentes na camada oculta causa uma redução irrisória na assertividade do modelo e, em certos casos há uma pequena melhoria,

atestando que é possível reduzir a complexidade computacional do modelo sem comprometer a assertividade dele.

Tabela 3 – Métricas de cada modelo de redes neurais utilizando entre 10 e 100 neurônios na camada oculta, indicando a possibilidade de reduzir a quantidade de neurônios sem comprometer a assertividade dos modelos

Nº de Neurônios	AUC	CA	F1	Precisão	Revocação
10	0,997	0,996	0,996	0,997	0,996
30	0,998	0,996	0,996	0,996	0,996
20	0,997	0,996	0,996	0,996	0,996
50	0,997	0,995	0,995	0,995	0,995
90	0,997	0,995	0,995	0,995	0,995
70	0,997	0,995	0,995	0,995	0,995
40	0,997	0,995	0,995	0,995	0,995
80	0,997	0,995	0,995	0,995	0,995
100	0,997	0,995	0,995	0,995	0,995
60	0,997	0,994	0,994	0,994	0,994

Fonte: Autores (2022)

Na Tabela 4 são apresentados os resultados da segunda fase de otimização do modelo. Através dos resultados é possível visualizar e atestar que o modelo pode ter sua complexidade computacional reduzida mais ainda sem comprometer sua assertividade. Foi eleito como “final” o modelo que contém 2 neurônios em sua camada oculta por ser aquele que traz a melhor razão entre custo computacional e assertividade entre todos os demais experimentados até aqui.

Tabela 4 – Métricas de cada modelo de redes neurais, na qual o modelo com 2 neurônios demonstra baixo custo computacional e alta assertividade.

Nº de Neurônios	AUC	CA	F1	Precisão	Revocação
1	0,996	0,500	0,333	0,250	0,500
2	0,997	0,995	0,995	0,995	0,995
3	0,997	0,995	0,995	0,995	0,995
4	0,997	0,997	0,997	0,997	0,997
5	0,997	0,996	0,996	0,996	0,996
6	0,998	0,997	0,997	0,997	0,997
7	0,997	0,995	0,995	0,995	0,995
8	0,997	0,995	0,995	0,995	0,995
9	0,997	0,995	0,995	0,995	0,995
10	0,997	0,996	0,996	0,996	0,996

Fonte: Autores (2022)

Na Tabela 5 constam as métricas obtidas e os diferentes processos de validação do modelo utilizados nessa fase. Como é possível observar, o modelo final apresentou consistência entre os diferentes métodos de validação utilizados, tendo variação menor que 1% em todas as métricas avaliadas. Também é possível observar que o modelo não apresenta sobreajuste, dado sua alta performance utilizando o estimador .632+.

Tabela 5 – Métodos de validação utilizados e suas respectivas métricas, indicando consistência de assertividade independentemente do método usado.

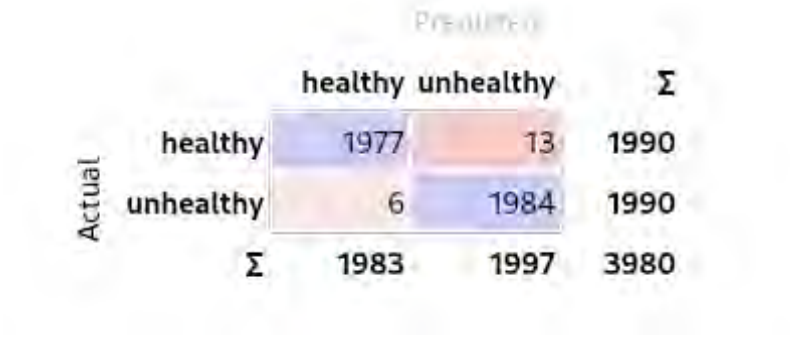
Método	AUC	CA	F1	Precisão	Revocação
Dados Exclusivos	0,994	0,997	0,997	0,997	0,997
K-fold	0,997	0,994	0,994	0,994	0,994
.632	0,998	0,998	0,998	0,998	0,998
.632+	0,996	0,994	0,995	0,991	0,997

Fonte: Autores (2022)

Matriz de Confusão

A fim de visualizar a quantidade de previsões corretas e erradas geradas pelo modelo de MLP, elaborou-se uma matriz de confusão, apresentada na Figura 9.

Figura 9 – Matriz de confusão para o modelo de MLP com 2 neurônios na camada oculta



Fonte: Autores (2022)

Teste U de Mann-Whitney

A Tabela 6 mostra os resultados coletados a partir do Teste U de Mann-Whitney realizado com o *dataset* final.

Tabela 6 – Resultados do teste não paramétrico de Mann-Whitney para amostras independentes, indicando heterogeneidade entre amostras do *dataset*.

Amostra	Score	P
n0	77.259	< 0,001
n1	102.699	< 0,001
n993	171.447	< 0,001
n991	178.002	< 0,001
n817	83.549	< 0,001
n648	87.445	< 0,001

Fonte: Autores (2022)

Discussão

O presente trabalho alcançou resultados similares aos encontrados por Khosravi *et al* (2018, p. 1), que também utiliza o aprendizado por transferência para realizar classificação de imagens. No trabalho citado, o intervalo de precisão dos

modelos foi entre 92% e 100%, muito similares aos 99% encontrados no presente trabalho.

O trabalho de Modarrares *et al* (2017) utiliza técnicas de aprendizado por transferência para classificar imagens produzidas por um microscópio eletrônico de varredura, apresentando resultados entre 82% e 87% de assertividade. Os resultados apresentados neste trabalho foram superiores aos encontrados pelo trabalho citado, com um aumento na precisão de aproximadamente 10%.

CONCLUSÕES

O presente trabalho visou apresentar, de forma parcial, uma solução que pudesse ser aplicada no controle de qualidade da cultura de *Citrus limon*. A partir da metodologia utilizada e dos resultados apresentados é possível concluir que a partir das evidências apresentadas tanto na forma de métricas como F1-Score, precisão e revocação, bem como na forma de matriz de confusão, que é possível classificar imagens de *Citrus limon* utilizando um modelo de *machine learning* em conjunto com o aprendizado por transferência. A evidência coletada a partir do processo de diferentes formas de validação demonstra que o modelo final apresenta assertividade acima de 90% independentemente do método utilizado para tal fim, indicando que há uma baixa possibilidade de o modelo apresentar sobreajuste. Conclui-se ainda, a partir do fato comprovado pelo teste de Mann-Whitney, que as imagens de cada classe possuem conjuntos de atributos heterogêneos entre si, sendo de fácil diferenciação mesmo ao olho humano, corroborando para a alta assertividade do modelo.

Por fim, é possível destacar a abertura para trabalhos futuros visando realizar testes com modelos de Aprendizado de Máquina computacionalmente mais simples do que os modelos de perceptron multicamadas utilizados no presente trabalho, otimizando ainda mais a razão custo *versus* assertividade.

REFERÊNCIAS

ABIDIN, A. Z. *et al*. **Deep transfer learning for characterizing chondrocyte patterns in phase contrast X-Ray computed tomography images of the human patellar cartilage**. Computers in Biology and Medicine, 2018. vol 95. p. 24-33.

ADAMIAK, M.; **Lemons Quality Control Dataset**. SoftwareMill, 2020. Disponível em: https://github.com/softwaremill/lemon-dataset. Acesso em: 30. ago. 2022

BATTITI, R.; **Using mutual information for selecting features in supervised neural net learning**. IEEE Transactions on Neural Networks. IEEE, 1994. vol. 5. p. 537-550.

CHAWLA, N. V.; JAPKOWICZ, N.; KOLCZ, A.; **Editorial: Special Issue on Learning from Imbalanced Data**. Sigkdd Explorations, 2004. vol 6. p 1-6. Disponível em: https://dl.acm.org/doi/pdf/10.1145/1007730.1007733. Acesso em: 04 out. 2022.

CNA. **Dia do Citricultor**: Produtores de frutas cítricas garantem alimento saudável no Brasil e no mundo. CNA, 2021. Disponivel em: https://cnabrasil.org.br/noticias/dia-do-citricultor-produtores-de-frutas-citricas-garantem-alimento-saudavel-no-brasil-e-no-mundo. Acesso em: 30 out. 2022.

CLARK, A. *et al*. **Pillow**. 2010. Disponível em: https://pillow.readthedocs.io/en/stable/index.html. Acesso em: 20 set. 2022.

DEMSAR, J. *et al*. **Orange**: Data Mining Toolbox in Python. Journal of Machine Learning Research, 2013. vol 14, p. 2349-2353.

DENG, J. *et al*. **ImageNet**: A large-scale hierarchical image database. IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2009. p. 248-255.

DUFOURNAUD, Y.; SCHMID, C.; HORAUD, R.; **Image Matching with Scale Adjustment**. Computer Vision and Image Understanding, 2004. vol 93. p 175-194.

EFFRON, B. **Bootstrap Methods**: Another Look at the Jacknife. The Annals of Statistics, 1979. vol 7. Nº 1. p. 1-26.

EFFRON, B.; TIBSHIRANI, R. **Improvements on Cross-Validation**: The .632+ Bootstrap Method. American Statistical Association, 1997. vol 92. Nº 438. p. 548-560.

GODEC, P. *et al*. **Democratized image analytics by visual programming through integration of deep models and small-scale machine learning**. Nature Communications, 2019. ed. 10. Nº 4551. Disponível em: https://www.nature.com/articles/s41467-019-12397-x. Acesso em: 07. out 2022.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. MIT Press, 2016. 1 ed.

HARRIS, C. R. *et al*. **Array Programming with NumPy**. Nature, 2020. vol 585. p. 357-362. Disponível em: https://www.nature.com/articles/s41586-020-2649-2. Acesso em: 02 out. 2022.

HASTIE, F.; TIBSHIRANI, R.; FRIEDMAN, J. **The Elements of Statistical Learning**: Data Mining, Inference, and Prediction. 2. ed. Springer, 2017. Disponível em: https://hastie.su.domains/ElemStatLearn/printings/ESLII_print12_toc.pdf. Acesso em 02 out. 2022

HAYKIN, S. **Neural Networks** – A Comprehensible Foundation. Pearson Prentice Hall, 1994. 2 ed.

HUNTER, J. D.; **Matplotlib**: A 2D Graphics Enviroment. Computing in Science & Engineering. IEEE Computer Society, 2007. vol 9. Nº 3. p. 90-95.

IANDOLA, F. N. *et al*. **SqueezeNet**: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size. International Conference on Learning Representations, 2017.

KHOSRAVI, P. *et al*. **Deep Convolutional Neural Networks Enable Discrimination of Heterogeneous Digital Pathology Images**. eBioMedicine, 2018. vol 27. p. 317-328.

KIELA, D.; BOTTOU, L.; **Learning Image Embeddings using Convolutional Neural Networks for Improved Multi-Modal Semantics**. Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2014. Doha, Qatar. p. 36-45.

KINGMA, D. P.; BA, J. L.; **Adam**: A Method for Stochastic Optimization. International Conference on Learning Representations, 2015. Disponível em: https://arxiv.org/pdf/1412.6980.pdf. Acesso em: 08 out. 2022.

KLUYVER, T. *et al*. **Jupyter Notebooks**: a publishing format for reproducible computational workflows. Positioning and Power in Academic Publishing: Players, Agents and Agendas. IOS Press, 2016. p. 87-90.

LIAKOS, K. G. *et al*. **Machine Learning in Agriculture**: A Review. MDPI, 2018. Disponível em: https://www.mdpi.com/1424-8220/18/8/2674/pdf. Acesso em: 23 out. 2022.

MANN, H. B.; WHITNEY, D. R.; **On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other**. Annals of Mathematical Statistics, 1947. vol 18. Nº 1. p. 50-60. Disponível em: https://projecteuclid.org/journals/annals-of-mathematical-statistics/volume-18/issue-1/On-a-Test-of-Whether-one-of-Two-Random-Variables/10.1214/aoms/1177730491.full. Acesso em: 15 out. 2022.

MICROSOFT *et al*. **Documentation for Visual Studio Code**. Seattle. 2015. Disponível em: https://code.visualstudio.com/docs. Acesso em: 29 ago. 2022.

MODARRES, M. H. *et al*. **Neural Network for Nanoscience Scanning Electron Microscope Image Recognition**. Nature Scientific Reports, 2017. Disponível em: https://www.nature.com/articles/s41598-017-13565-z. Acesso em: 07 out. 2022.

OLTEAN, M.; **Fruits 360**: A dataset with 90380 images of 131 fruits and vegetables. Kaggle, 2020. Disponível em: https://www.kaggle.com/datasets/moltean/fruits. Acesso em: 03 set. 2022.

PEAT GmbH. **Plantix**. *S. I.*, 2015. Disponivel em: https://play.google.com/store/apps/details?id=com.peat.GartenBank&hl=pt. Acesso em: 23 out. 2022.

PEDREGOSA, F. *et al*. **Scikit-Learn**: Machine Learning in Python. Journal of Machine Learning Research, 2011. vol 12. p. 2825-2830.

QUINTÃO, L. C. **Estamos atolados em uma cultura do desperdício.** ONG Banco de Alimentos, 2021. Disponível em: https://bancodealimentos.org.br/estamos-atolados-em-uma-cultura-do-desperdicio/. Acesso em: 30 out. 2022.

RASCHKA, S. **MLxtend**: Providing machine learning and data science utilities and extensions to Python's scientific computing stack. Journal of Open Source Software, 2018. vol 3. Disponível em: http://joss.theoj.org/papers/10.21105/joss.00638. Acesso em: 02 out. 2022.

SANTOS, A. T.; PINTO, C. C.; FREITAS, T. S.; **Produção Brasileira de Citros**. ILSA Brasil, 2022. Disponível em: https://ilsabrasil.com.br/producao-brasileira-de-citros/. Acesso em: 30 out. 2022.

SIMONYAN, K; ZISSERMAN, A. **Very Deep Convolutional Networks for Large-Scale Image Recognition**. International Conference on Learning Representations, 2015.

SZEGEDY, C. *et al*. **Going Deeper with Convolutions**. Computer Vision and Pattern Recognition Conference, 2015.

THORNDIKE, R. L.; **Who belongs in the family?**. Psychometrika, 1953. ed. 18. p. 267-276.

VAN ROSSUM, G.; DRAKE, F. L.; **Python 3 Reference Manual**. Scotts Valley. 2009.
Disponível em: https://docs.python.org/3/reference/. Acesso em: 29 ago. 2022.

ZHANG, W. *et al*. **Deep model based transfer and multi-task learning for biological image analysis**. International Conference on Knowledge Discovery and Data Mining, 2015. p. 1475-1484.