Classificação de Vinhas a partir de Redes Neuronais Convolucionais

Gabriel Antonio Pereira dos Santos Carneiro

I. Introdução

Um dos motivos de Portugal ser conhecido internacionalmente é pelo seu vinho, seja pela variedade, ou pela qualidade de tal produto. Dentre os fatores de maior relevância na cadeia de produção de vinho, encontra-se a casta da uva utilizada, como ela influencia diretamente na autenticidade e classificação do vinho, conseguir identificar os diferentes tipos de vinha é fundamental para a regulamentação da produção [1].

Uma das maneiras de se identificar as castas de uva é através da análise visual, abordagem chamada de Ampelografia. Entretanto, como qualquer tarefa de análise visual, a ampelografia acaba dependendo fortemente da expertise de quem a estar realizando, tornando o processo demasiadamente subjetivo e exposto a interferências de condições de ambiente, de cultura e de genética, o que pode introduzir uma incerteza no processo de identificação [1, 2, 3].

Uma abordagem comum a tarefas de classificação que se baseiam em inspeção de imagens é o *Deep Learning*. Trabalhos que utilizam Redes Neuronais Convolucionais Profundos vêm figurado, desde 2012, o estado-da-arte na classificação de imagens dos mais diversos tipos.

Levando tais fatores em consideração, nesse trabalho apresentamos o desenvolvimento de um modelo convolucional para classificação de 12 castas de uva presentes no Vale do Douro em Portugal. Nossa metodologia foi baseada em aplicar a técnica de *transfer learning*, utilizando o conhecimento da rede neuronal Xception [4] obtido no ImageNet. Como o *dataset* utilizado para treino, teste e validação não era balanceado, utilizamos a função de *loss Focal Loss* [5] para treinamento. Nosso modelo alcançou um escore F1 de 0.92 no conjunto de teste e para validar o resultado, apresentamos explicações de classificação obtidas através da técnica Local Interpretable Model-Agnostic Explanations [6].

O texto está organizado da seguinte maneira: na Seção II detalhamos a metodologia que empregamos; em Seção III os resultados que obtivemos com o nosso modelo; enquanto que em Seção IV nossas conclusões acerca do desenvolvimento e resultados.

II. Metodologia

Nesta seção descreveremos a metodologia empregada para classificação das castas de uvas. Partindo de uma breve descrição do *dataset*, até concluirmos com a abordagem que utilizamos para depurar o modelo convolucional testado.

A. Dataset

O *dataset* utilizado é composto por por uma coleção de imagens de folhas de uva, sendo que as uvas pertenciam a 12 castas diferentes: Codega (72), Malvasia Fina (66), Malvasia Preta (66), Malvasia Rei (69), Moscatel Galego (62), Mourisco Tinto (68), Rabigato (64), Tinta Amarela (68), Tinta Barroca (70), Tinta Roriz(74), Tinto Cao (67) e Touriga Nacional (69). O número ao lado de cada casta representa quantidade de imagens para cada uma delas.

Além disso, as imagens do *dataset* foram adquirias com uma câmera Cannon EOS 600D, equipada com uma lente 50 mm f/1.4 e posteriormente passaram por um processo de recorte e redimensionamento, para que as folhas ficassem dentro do aspecto de um quadrado, resultando em imagens quadradas de resolução 300x300. Um exemplar para cada diferente classe pode ser observada na Figura 1.

Figura 1. Exemplos de imagens presentes no *dataset*.

1) Data augmentation: Visando aumentar a quantidade de dados disponíveis para treinamento, aplicamos a técnica de *data augmentation* nas imagens de treino. Antes de iniciarmos o treinamento, dividimos o dataset, aleatoriamente, em teste, treino e validação - sendo que as quantidades utilizadas foram 10%, 70% e 20% para cada divisão, respectivamente -, e aplicamos o *data augmentation* gerando 10 imagens de cada imagem presente no conjunto de treino. No final obtivemos 6718 imagens de treino, 132 de validação e 72 imagens de teste. As imagens de teste e validação não passaram por nenhum tipo de *data augmentation* e, além disso, a quantidade de cada um desses conjuntos foi calculada de acordo com a Eq. 1 de maneira que eles fossem balanceados. A quantidade de imagens Q_{imagem} é obtida multiplicando o $percentual$ requerido para o conjunto pelo valor mínimo do conjunto de quantidade de imagens pertencentes a cada classe A.

$$Q_{imagem} = percentual * min(A) \tag{1}$$

B. Modelo Convolucional

Inicialmente buscamos construir modelos convolucionais sequenciais ao invés de de utilizar *transfer learning*. Entretanto, as variações dos modelos testados não alcançaram acurácia e F1 Escore maior que 60%. O código de um dos modelos testados pode ser observado no GitHub[1].

Tendo em consideração a quantidade dados que tínhamos, e o desempenho ruim encontrado em testes iniciais com modelos *handcraft*, optamos por aplicar a técnica de *transfer learning*. Observando os resultados obtidos Adão et al. [1], utilizamos o modelo convolucional Xception [4], pois foi o modelo o que neste trabalho alcançou o melhor resultado na tarefa de classificação de castas de uvas com imagens com fundo branco.

Entretanto, diferente de Adão et al. [1] utilizamos os pesos do ImageNet, não retreinando todo o modelo convolucional, apenas substituindo o classificador no topo da rede.

Assim como o modelo Xception padrão, utilizamos o *Average Global Pooling* e para o classificador optamos por um camada densa com 40 neurônios, com função de ativação *Rectified Linear Units (ReLU)*, seguida por um Dropout de 25%, buscando evitar *overfitting* e por último uma camada Dense, com 12 neurônios, por termos buscado classificar 12 castas de uvas, com função de ativação *SoftMax*.

O tamanho da imagem utilizado para treinar a rede foi de 300x300 pixeis, não requerendo nenhum redimensionamento no *dataset* utilizado. Adicionalmente, o valor dos pixeis das imagens foram normalizados entre -1 e 1, por conta da ferramenta que optamos por utilizar, o Keras, ter treinado o modelo Xception com tais configurações.

C. Hiperparametrização de Treinamento

Nesta subseção detalharemos o otimizador, função de *loss* e números de épocas utilizados para treino e, além disso, discutiremos as métricas utilizadas para mensurar o desempenho dos classificadores resultantes.

a) Otimizador: Testamos dois otimizadores diferentes, o Adam [7] e o *Sthocastic Gradient Descent* (SGD), com decaimento de taxa de aprendizagem. Para o Adam utilizamos testamos as taxas de aprendizagem de 0.01, 0.001 e 0.00001. E para o SGD testamos a taxa de aprendizagem 0.01, entretanto variando a forma de decaimento da taxa de aprendizado em decaimento exponencial e decaimento por passos.

Para o decaimento exponencial (vide Eq. 2) e decaimento por passos (vide Eq. 3), utilizamos o equacionamento proposto por Adão et al. [1]. Para garantir a coerência no decaimento, as variáveis dependentes foram isoladas de maneira $DropRate$ e k, foram modeladas nas Equações 4 e 5, respectivamente.

$$LR = LRInitial * \mathrm{e}^{-k*epoch} \tag{2}$$

$$LR = LRInitial * \mathrm{DropRate}^{floor(\frac{epoch}{epochs\_drop})} \tag{3}$$

$$DropRate = \mathrm{LRInitial}^{\frac{flatten\_factor}{epochs\_drop}} \tag{4}$$

[1]https://github.com/gabri14el/dl_utad_projeto/blob/main/projeto_DL_handcraft.ipynb

$$k = \frac{\log \frac{LRInitial^{flatten\_factor}}{LRInitial}}{n\_epochs * \log \mathrm{e}} \tag{5}$$

Para ambos os decaimentos um $flatten\_factor$ de $LRInitial^{2.25}$ foi utilizado para conduzir a um platô de taxa de aprendizagem leve [1].

b) Função de Loss: Diferente de Adão et al. [1], o *dataset* utilizado para treino não foi balanceado e portanto, utilizar puramente a função de entropia cruzada pode levar a um pior desempenho do classificador [5, 8]. Então nesse trabalho optamos por utilizar a *Focal Loss*, função de *Loss* proposta por Lin et al. [5], que busca resolver o problema das classes não balanceadas durante o treinamento, fazendo uma modificação na função de entropia cruzada. A implementação de Focal Loss utilizada foi feita por Zhang [9] e está disponível em seu GitHub[2]. Um dos hiperparâmetros da *Focal Loss* é o *alpha*, seguindo Zhang [9] em nossos experimentos utilizamos $alpha = 1$.

c) Métricas: Por estarmos lidando com um *dataset* não balanceado durante o treinamento, optamos por utilizar como padrão o Escore F1 para medir o desempenho dos modelos. Além disso, no decorrer do texto exibiremos a acurácia, precisão e revocação, também calculadas para cada um dos modelos.

d) Batch, Épocas e Early Stop: Optamos por utilizar 100 épocas para o treinamento, assim como Adão et al. [1], entretanto buscando agilizar o processo de treinamento, evitando o *overfitting*, utilizamos a métrica de *loss* da validação para decidir quando é hora de parar, aplicando a técnica de *Early Stop* com paciência de 10 épocas. Isso significa que se em 10 épocas a *loss* da validação não diminuir, o treinamento é finalizado. O Tamanho de *batch* utilizado foi 12, por limitações do hardware utilizado para treinamento dos modelos neuronais.

D. Validação dos Resultados

Buscando explicar os resultados obtidos pelo classificador que alcançou o melhor desempenho, utilizamos o método Local Interpretable Model-Agnostic Explanations (LIME) [6] para mostrar regiões de algumas imagens que contribuíram para a classificação. Nesse trabalho nos concentramos em utilizar o LIME para explicar onde o classificador errou, entretanto o caminho inverso também pode ser seguido, utilizando a técnica para percebermos o porquê de confiar nas classificações obtidas.

E. Ferramentas

Utilizamos o Framework Keras com *backend* TensorFlow para desenvolvimento. Não tivemos controle do hardware utilizado, pois utilizamos a ferramenta Google Colaboratory para executar os experimentos. Todo o código utilizado nos experimentos está hospedado no repositório GitHub *https://github.com/gabri14el/dl_utad_projeto*.

[2]https://github.com/Tony607/Focal_Loss_Keras

III. Resultados e Discussões

Nesta seção descreveremos os resultados dos modelos que testamos para a classificação das castas de uva. A Tabela I resume os resultados alcançados pelas duas estratégias de decaimento de taxa de aprendizado testadas.

Decay Strategy	F1 Score	Precision	Recall	Accuracy
Step Decay	0.92	0.94	0.92	0.92
Exponential Decay	0.90	0.93	0.90	0.90

Tabela I
Desempenho atingido pelas duas estratégias de decaimento de taxa de aprendizado testadas.

Figura 2. A esquerda um exemplar de Codega, no meio um exemplar da Malsavia Rei e a direita um exemplar de Malsavia Preta.

a) Decaimento por passos: O modelo treinado com decaimento por passos alcançou o melhor resultado no *dataset* de teste, obtendo 0.92 de Escore F1. Nas Figuras 3, 4, 5 podemos ver, respectivamente, os gráfico de *loss*, acurácia e Escore F1 para treinamento e validação do modelo utilizando tal estratégia de decaimento. Enquanto na Figura 6 podemos observar a matriz de confusão obtida através do conjunto de teste.

Observando o gráfico de acurácia e escore F1, Figs. 4 e 10, pode-se notar que ainda há possibilidade de crescimento de ambas as medidas. Portanto, utilizar o *Early Stop* com paciência de 10 não uma boa estratégia. Utilizar uma paciência maior, ou não utilizar o *Early Stop* pode fazer com que o modelo tenha mais tempo para aprender mais.

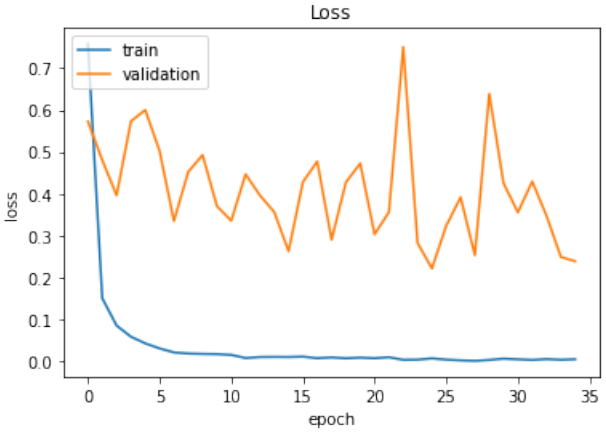

Figura 3. Loss do treinamento com decaimento por passos.

Observando a matriz de confusão verifica-se que a classes com mais erros de classificação foram Codega e Mourisco Tinto. Na Figura 2 podemos observar um dos exemplares de Codega mal classificados ao lado de exemplares de Malsavia

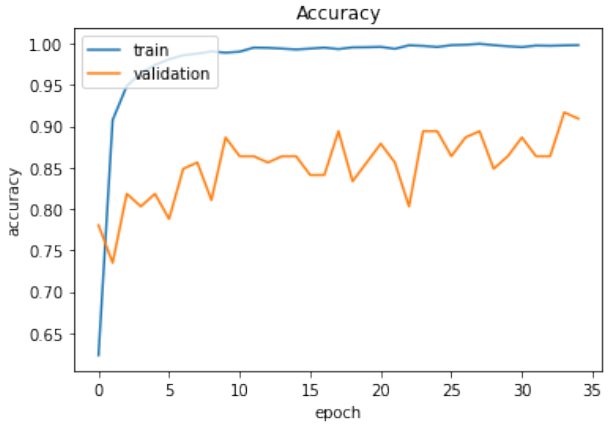

Figura 4. Acurácia do treinamento com decaimento por passos..

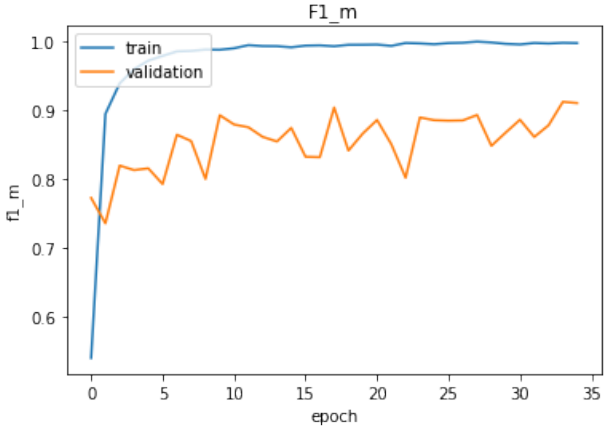

Figura 5. Escore f1 do treinamento com decaimento por passos.

Rei e Malsavia preta, note a similaridade entre as imagens. Note também que Codega e Mourisco Tinto não estão entre as classes com menos imagens para o treinamento, sendo que Codega é a classe com mais exemplares.

A fim de verificar as regiões que mais contribuíram para as classificações errôneas, exibimos na Fig. 7 explicações de classificações incorretas de 5 imagens, obtidas com o LIME, para o modelo treinado utilizando o decaimento por passos de taxa de aprendizagem. A organização da imagem está detalhada em sua legenda.

Note que para a primeira imagem, na classificação que deveria ter sido feita, o classificador está considerando uma porção do solo como uma região que contribui negativamente. Considerando que o solo não contribui para a classificação das uvas na vida real, esse é um problema que deve ser tratado em trabalho futuro. Mesmo em imagens onde o classificador acerta, tem de se verificar se porções pertencentes ao solo estão sendo utilizados na classificação. De forma similar, na terceira imagem a forma de uma sombra é considerada como região importante a favor da classificação da região errada e a mesma região contribuiu contra a classificação correta da imagem. Entretanto diferente do solo, o aspecto da região sombreada se assemelha ao contorno de uma folha.

Observando as explicações da quarta imagem, o caule é

interpretado em duas partes diferentes durante a classificação. A ramificação cortada foi uma região considerada a favor da classificação da classe errada, enquanto o resto do caule foi considerado um região contra a classificação da classe correta. É interessante notar que regiões do caule estão também sendo consideradas na hora da classificação. Um outro aspecto que vale apena observar é que quase toda a forma da folha contribui positivamente para a classificação da classe errada.

Na quinta imagem, pedaços do *background* de solo desfocado são considerados superpixeis levados em consideração durante a classificação, o que não deveria acontecer.

b) Decaimento Exponencial: Comparado com o modelo treinado com o decaimento por passos da taxa de aprendizado o modelo treinado com decaimento exponencial teve um desempenho pior. Os gráficos de *loss*, acurácia e escore F1 podem ser observados nas Figuras 8, 9, 10, respectivamente, enquanto a matriz de confusão gerada a partir do dataset de teste pode ser visto na Figura 11.

É interessante notar nos gráficos de treinamento (Figs. 8, 9, 10) que ele foi menor se comparado ao treinamento utilizando o decaimento por passos (por passos: 35 épocas; exponencial: 16 épocas). Isso aconteceu porque a função de *loss* teve um comportamento mais estável, e atingiu o mínimo, com uma paciência de 10 épocas, mais rapidamente. É interessante notar também que a *loss* atinge valores menores (por passos: 0.2216; exponencial: 0.1364).

c) Adam: Testamos também o otimizador Adam [7], em substituição ao SGD. Em nossos testes o modelo não convergiu utilizando os parâmetros padrões. Além disso, tentamos também modificar a taxa de aprendizado para 0.01 e 0.0001, e em nenhum dos casos houve convergência.

d) Trabalhos Futuros: Diante dos resultados apresentados aqui, fica claro que muito pode ser feito para melhorar o modelo em questão. Hiperparâmetros em redes neuronais é um mundo a se explorar, e nesse trabalho, só testamos escalonadores para o decaimento da taxa de aprendizado. A paramtrização para a *Focal Loss*, utilização de outros classificadores como *Random Forests* ou *Support Vector Machine* no topo do modelo convolucional e aplicar a técnica de *fine tuning*, retreinando toda a rede neuronal são exemplos de abordagens que podem ser testadas em busca de melhores resultados.

IV. Conclusão

Portugal é um país conhecido por seus vinhos, seja pela qualidade, seja pela variedade disponível. A casta da uva é um fator fundamental na produção de um bom vinho, sendo que a ciência que mapeia essas castas é a ampelografia. Visando automatizar e tornar o processo do reconhecimento de castas de uva menos subjetivo, nesse trabalho detalhamos o desenvolvimento de um modelo neuronal convolucional para a classificação de 12 castas de uvas.

O modelo neuronal foi baseado na Xception, utilizando a técnica de *transfer learning* a partir de pesos do ImageNet e alcançou F1 Score de 0.92. Além disso, empregamos técnicas como taxa de aprendizado adaptativa e utilizamos a *Focal Loss* como função de *loss*. Como meio de validação do modelo, entregamos explicações obtidas através da técnica de Local Interpretable Model-Agnostic Explanations para 5 imagens de teste onde o classificador errou.

A maior mais valia desse trabalho está no fato de a classificação ter sido feita utilizando imagens com *background* natural, possibilitando que futuramente possa-se capturar imagens utilizando o telemóvel, e uma classificação de castas de uva possa ser realizada em questão de segundos por qualquer pessoa.

Para trabalho futuro fica a possibilidade de testar outros classificadores no topo da rede, aplicar a técnica de *fine tuning*, testar novos parâmetros para a *Focal Loss*, além de variar vários parâmetros que os modelos neuronais permitem.

Referências

[1] T. Adão, T. M. Pinho, A. Ferreira, A. Sousa, L. Pádua, J. Sousa, J. J. Sousa, E. Peres, and R. Morais, "Digital ampelographer: A cnn based preliminary approach," in *Progress in Artificial Intelligence*, P. Moura Oliveira, P. Novais, and L. P. Reis, Eds. Cham: Springer International Publishing, 2019, pp. 258–271.

[2] S. Garcia-Muñoz, G. Muñoz-Organero, M. T. de Andrés, and F. Cabello, "Ampelography - an old technique with future uses: the case of minor varieties of amp;lt;emamp;gt;vitis viniferaamp;lt;/emamp;gt; l. from the balearic islands," *OENO One*, vol. 45, no. 3, p. 125–137, Sep. 2011. [Online]. Available: https://oeno-one.eu/article/view/1497

[3] L. Tassie, "Vine identification–knowing what you have," *Grape and wine research and development corporation â€"Australian Government. GW RDC Innovators Network, Greenhill Road Wayville*, 2010.

[4] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 1800–1807.

[5] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *CoRR*, vol. abs/1708.02002, 2017. [Online]. Available: http://arxiv.org/abs/1708.02002

[6] M. Ribeiro, S. Singh, and C. Guestrin, ""why should i trust you?": Explaining the predictions of any classifier," 02 2016, pp. 97–101.

[7] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.

[8] K. Pasupa, S. Vatathanavaro, and S. Tungjitnob, "Convolutional neural networks based focal loss for class imbalance problem: a case study of canine red blood cells morphology classification," *Journal of Ambient Intelligence and Humanized Computing*, Feb 2020. [Online]. Available: http://dx.doi.org/10.1007/s12652-020-01773-x

[9] C. Zhang, "Multi-class classification with focal loss for imbalanced datasets," https://www.dlology.com/blog/multi-class-classification-with-focal-loss-for-imbalanced-datasets/, (Accessed on 01/14/2021).

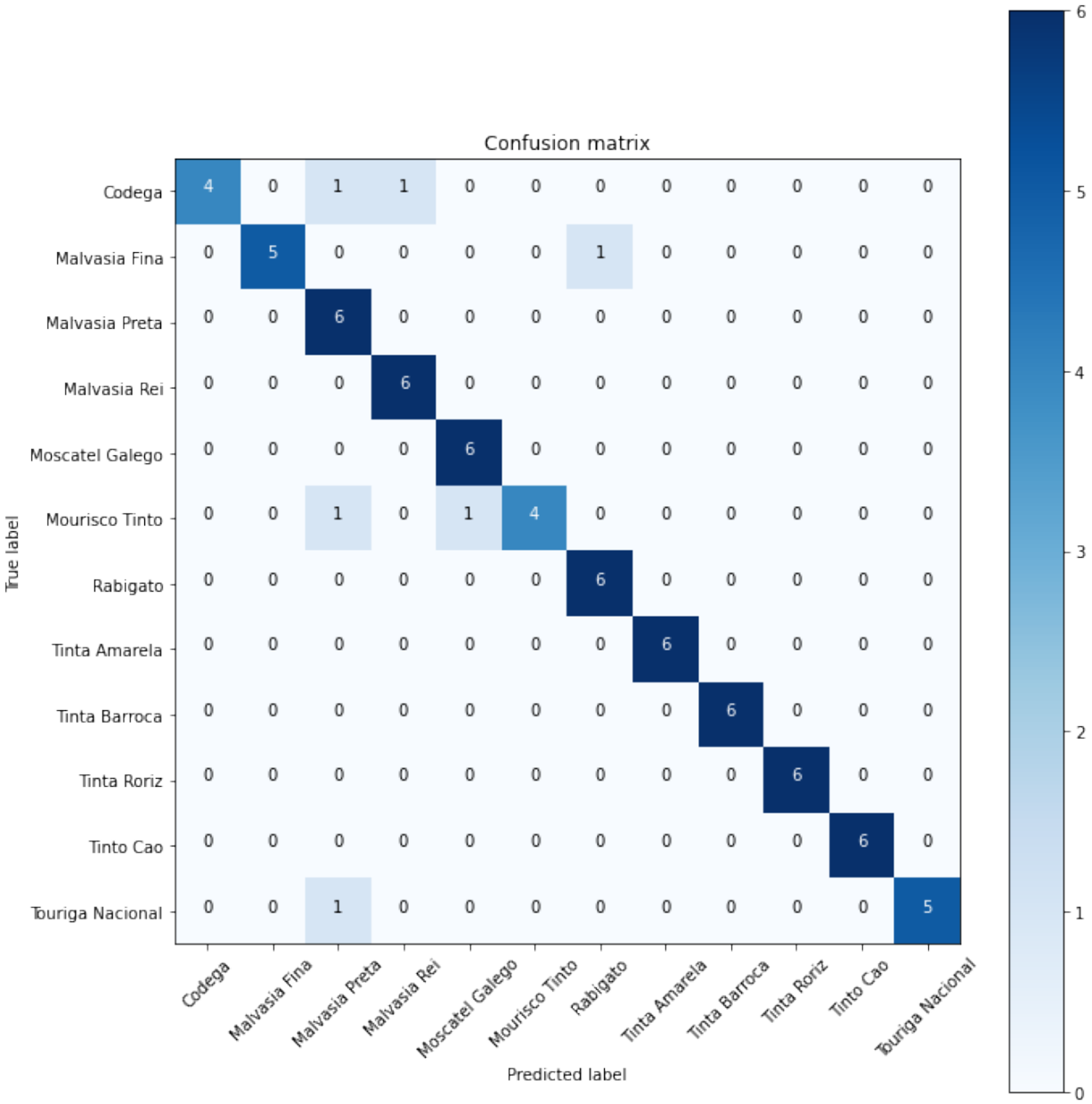

Figura 6. Matriz de Confusão do treinamento com decaimento por passos.

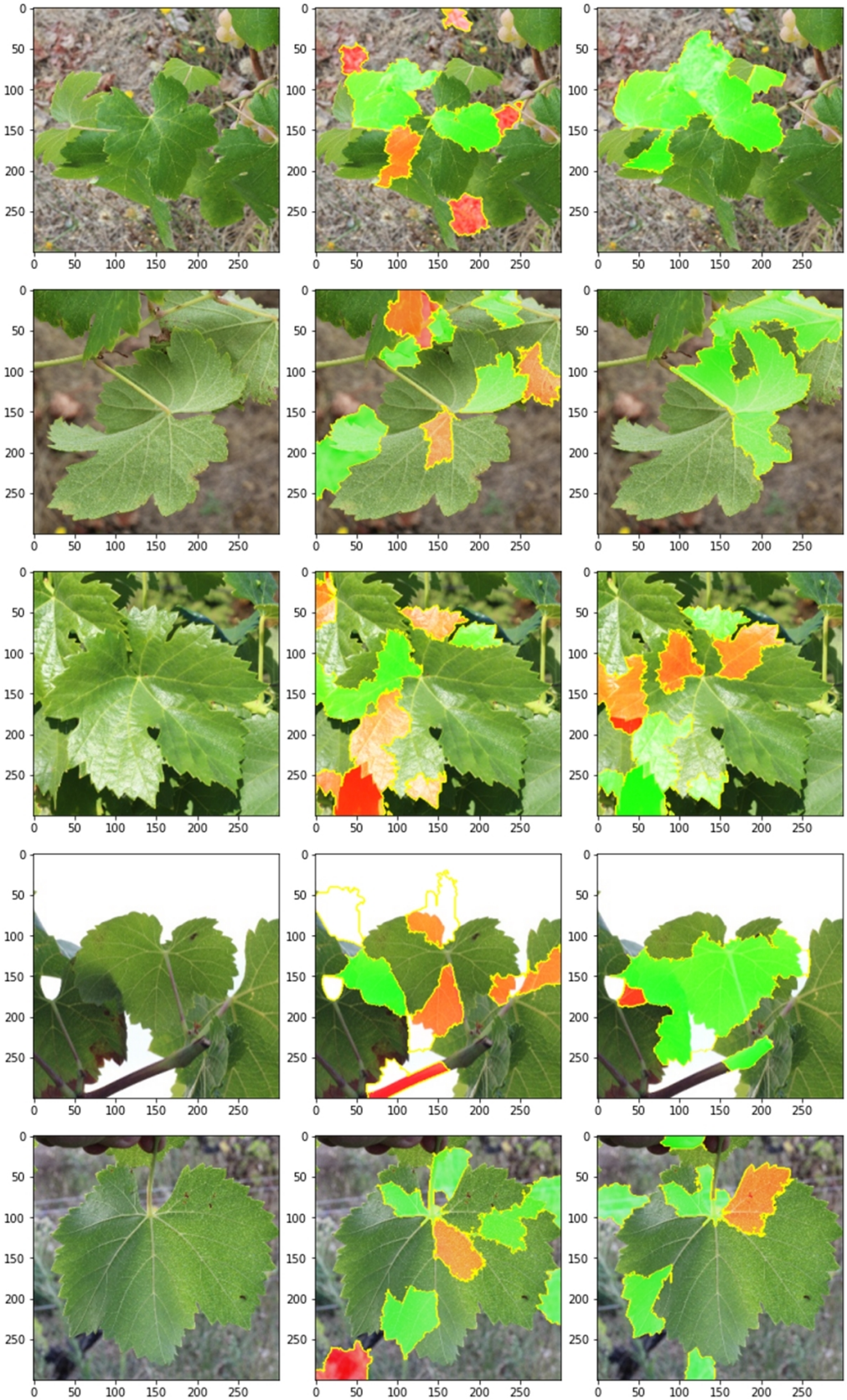

Figura 7. Explicações da classificação para o modelo treinado com decaimento de taca de aprendizagem por passos utilizando o LIME. Cada linha representa a classificação de uma imagem. Na primeira coluna, da esquerda para a direita, se encontram as imagens originais, na segunda coluna, da esquerda para a direita, a explicação para a classe que deveria ter sido prevista e na última coluna, da esquerda para a direita a explicação para a classe prevista. As regiões em tons de verde são superpixeis (vide [6]) que contribuíram para a classificação da classe em questão, enquanto que as vermelhas contribuíram contra tal classificação.

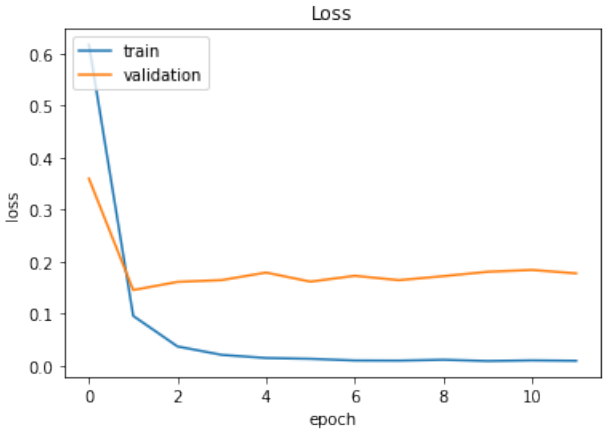

Figura 8. Loss do treinamento com decaimento exponencial.

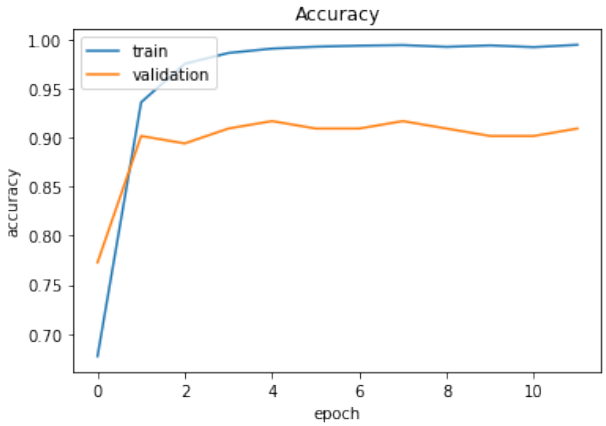

Figura 9. Acurácia do treinamento com decaimento exponencial.

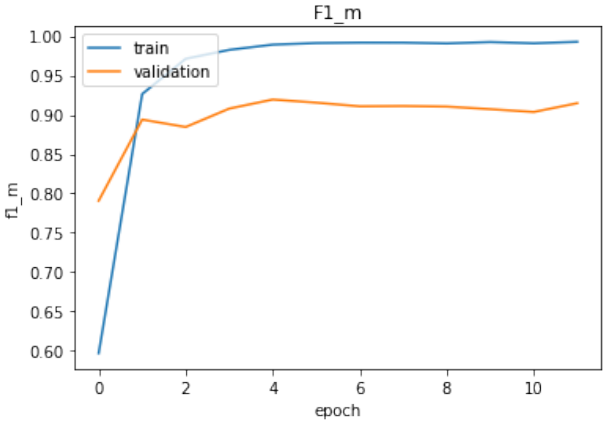

Figura 10. Escore f1 do treinamento com decaimento exponencial.

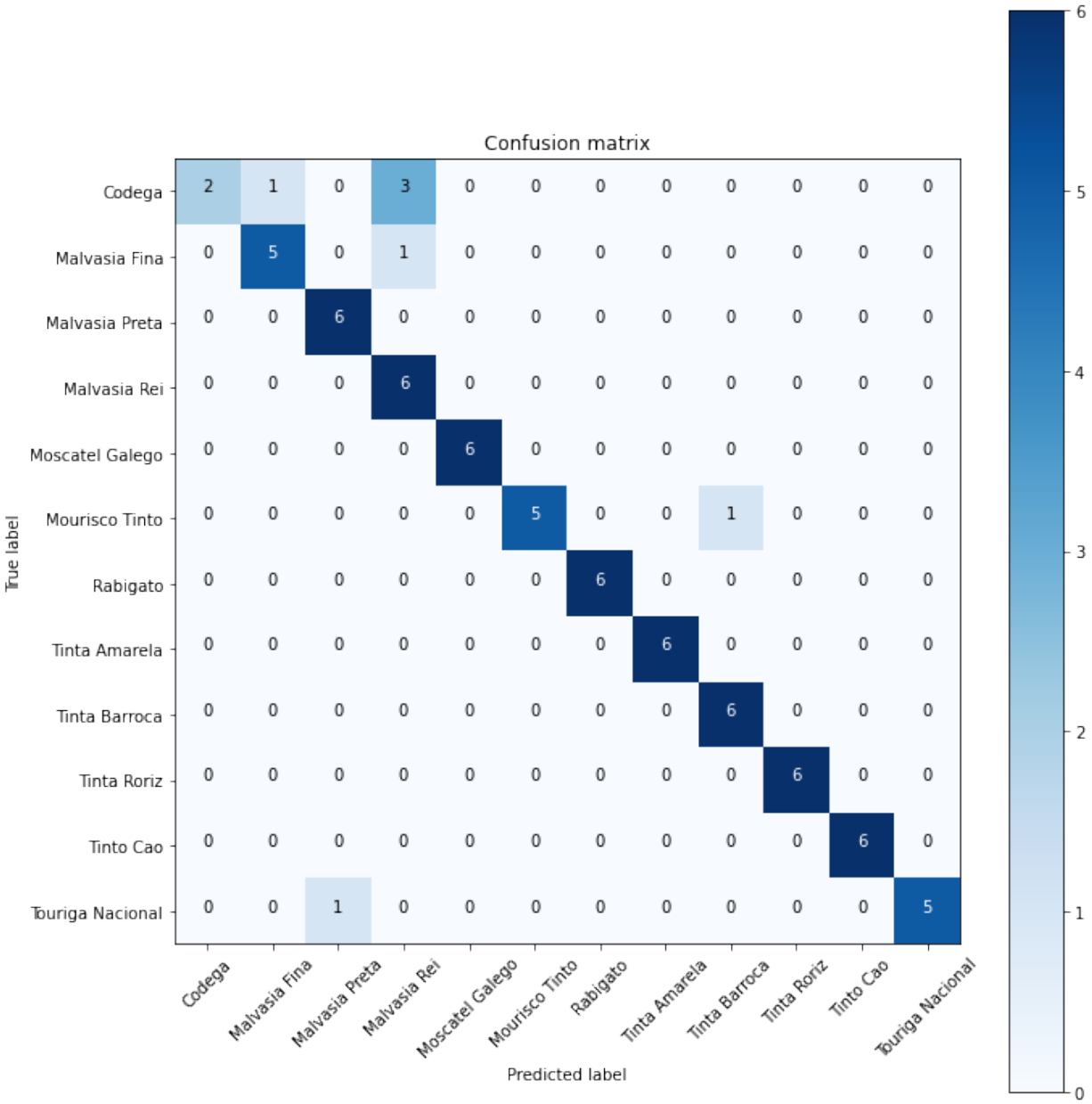

Figura 11. Matriz de Confusão do treinamento com decaimento exponencial.