Citrus Tree Classification from UAV Images: Analysis and Experimental Results

Felipe Kawashita Kobayashi*, Andrea Britto Mattos*, Maysa M. G. Macedo* and Bruno H. Gemignani‡
* IBM Research
Sao Paulo, Brazil
felipe.kawashita@ibm.com, {abritto,mmacedo}@br.ibm.com
‡3DGEO
bruno@3dgeo.com.br

***Abstract*—The use of unmanned aerial vehicles (UAVs) and computer vision for automating farm operations is growing rapidly: time-consuming tasks such as crop monitoring may be solved in a more efficient, precise, and less error-prone manner. In particular, for estimating productivity and managing pests, it is fundamental to characterize crop regions into four classes: (i) full-grown trees, (ii) tree seedlings, (iii) tree gaps, and (iv) background. In this paper, we address the classification of images from citrus plantations, acquired by UAVs, into the previously mentioned categories. While Deep learning-based methods allow to achieve high accuracy values for classification, explainability remains an issue. Therefore, our approach is to run an experimental analysis that allows to derive the effects of different parametrizations (involving descriptors, classifiers, and sampling methods) when applied to our citrus dataset.**

***Index Terms*—Computer vision, Unmanned aerial vehicles (UAVs), tree classification, citrus trees.**

I. Introdução

O gerenciamento do inventário de uma fazenda requer um monitoramento contínuo das plantações, a fim de calcular índices de produtividade e evitar eventuais prejuízos para o produtor. Esse acompanhamento pode ser um processo extremamente demorado e exaustivo quando feito de forma manual, pois a contagem e o registro do estado de cada árvore podem demorar diversos dias, a depender da área total da plantação.

A utilização de sensoriamento remoto mostra-se interessante para lidar com esse problema, pois com o uso de imagens aéreas da fazenda, juntamente com técnicas de visão computacional e algoritmos de aprendizado de máquina, é possível reduzir radicalmente o tempo em que os dados são adquiridos, processados e atualizados.

Nesse artigo, serão apresentados experimentos com imagens de uma fazenda com plantações de árvores de citrus, obtidas por um veículo aéreo não tripulado (VANT), com o objetivo de classificar regiões da imagem de acordo com a característica do plantio. Foram utilizados diferentes tipos de classificadores, métodos de balanceamento e descritores de características. Os melhores resultados alcançados em termos da eficácia na detecção das diferentes classes existentes serão mostrados e discutidos.

O artigo está estruturado da seguinte maneira: a Seção II relata alguns trabalhos relacionados e na Seção III é apresentada a metodologia que foi empregada para a realização dos experimentos. A quantidade de experimentos que foram realizados e os resultados atingidos são exibidos e posteriormente discutidos na Seção IV. Por fim, o trabalho é concluído na Seção V.

II. Trabalhos Relacionados

O Brasil responde por mais de três quartos das exportações globais de suco de laranja, sendo o maior produtor do mundo; em 2019, a produção de citrus pelo Brasil está prevista em 17,8 milhões de toneladas [1]. Diversos trabalhos anteriores abordaram a deteção de árvores de citrus em imagens provenientes de sensoriamento remoto, utilizando métodos como transformada de Hough circular [2], transformada simétrica radial [3]–[5], casamento de padrões [6], e mais recentemente, redes neurais convolucionais [7].

No entanto, todos esses trabalhos focam na detecção de árvores independente do seu grau de maturidade, isto é, não se preocupando em diferenciar entre árvores adultas e mudas, por exemplo. Essa diferenciação, porém, é fundamental para certas análises de crescimento e produtividade, uma vez que processos que operam durante as fases de sementes e mudas são cruciais para a compreensão de padrões, dinâmicas e sucessão em comunidades de plantas [8].

No contexto de classificação, López *et al.* [9] avaliaram métodos que diferenciam entre *citrus* e *não-citrus*, no entanto, como a detecção de mudas também era necessária, características específicas tiveram que ser extraídas e classificadores de mudas foram implementados. Os autores utilizaram características baseadas na imagem NDVI (Normalized Difference Vegetation Index) e avaliaram um conjunto de classificadores – Decision tree, Support Vector Machine (SVM) e Multi Layer Perceptron neural network (MLP). Nesse trabalho, os melhores valores de acurácia foram obtidos com o SVM (93%). Já Zortea *et al.* [10], utilizaram imagens de árvores com diferentes características para treinar uma CNN com o objetivo de detectar árvores de citrus; porém, apenas o resultado para árvores adultas foi reportado (94%).

CNNs também foram utilizadas em outros trabalhos recentes de classificação em agricultura, por exemplo, para classificar mudas de 12 espécies diferentes [11] ou para análise de saúde em Agricultura de Precisão [12]. Embora esses trabalhos reportem altos valores de acurácia na classificação,

pouco é discutido, por exemplo, sobre os descritores capazes de gerar as características mais relevantes para o processo ou sobre o impacto do balanceamento dos dados de treinamento. Para explorar tais aspectos, esse artigo aborda uma análise experimental de classificação de árvores de citrus, que será descrita nas próximas seções.

III. Metodologia

A. Geração do conjunto de dados

As imagens foram capturadas durante o monitoramento de uma fazenda localizada no município de Santa Cruz do Rio Pardo no estado de São Paulo. O sistema de aeronave remotamente pilotada (RPA) utilizado foi o Batmap® (Nuvem UAV®, Presidente Prudente-SP, Brazil) equipado com o sensor Sony A6000 (Sony®, Tokyo, Japan) de 24MP APS-C. O Software utilizado para o processamento das imagens foi o Pix4D® (Pix4D S.A., Prilly, Switzerland) que, a partir da configurações de vôo utilizadas e das técnicas de fotogrametria, resultou num ortomosaico georreferenciado da propriedade em questão de cerca de 9,5cm/pixel de GSD (ground sample distance). O Ortomosaico resultante é um arquivo raster no formato TIFF (Tagged Image File Format), o qual permite que informações sobre as coordenadas geográficas reais sejam salvas em cada pixel. A Figura 1 mostra um exemplo de uma das imagens do plantio, na qual é possível observar as quatro classes diferentes a seguir:

- Árvore (destacada pelo círculo vermelho): árvores adultas de citrus;
- Muda (destacada pelo círculo amarelo): árvores jovens de citrus;
- Falha (destacada pelo círculo azul): locais onde deveriam haver uma muda ou uma árvore de citrus;
- Outros: demais regiões onde encontram-se quaisquer elementos diferentes dos anteriores.

Figura 1. Exemplo de imagem da plantação.

A partir das imagens originalmente obtidas, acompanhadas de um arquivo do formato shapefile que contém as coordenadas de cada amostra de *árvore*, *muda* e *falha*, marcadas de forma manual, o conjunto de dados usado para a realização dos experimentos foi gerado, no qual é composto por imagens de tamanho 50×50 pixels (4.75×4.75 m), recortadas da imagem original, onde cada imagem contém uma amostra de alguma das quatro classes mencionadas.

As amostras de *árvore*, *muda* e *falha* foram recortadas combinando as coordenadas do shapefile com as da imagem completa, enquanto as imagens da classe *outros* foram selecionadas automaticamente a partir das regiões das imagens onde não haviam interseções com amostras das classes anteriores. A Figura 2 exibe algumas imagens de exemplo do conjunto de dados utilizado e a Tabela I mostra o número total de amostras para cada classe. Note que o conjunto de dados possui um alto grau de desbalanceamento.

Tabela I
Número de amostras em cada classe

Classe	Número de amostras
Árvore	268.624
Muda	2.400
Falha	19.779
Outros	84.373

O processo de construção do shapefile trata-se de um trabalho manual exaustivo devido ao grande número de amostras, e assim, notou-se que uma quantidade razoável de pontos foram marcados de maneira imprecisa, e consequentemente, gerou algumas imagens inconsistentes para as classes *árvore*, *muda* e *falha*, como pode-se ver na Figura 3.

Embora esse artigo aborde somente a tarefa de classificação, nosso objetivo final é gerar um sistema capaz de receber uma imagem de entrada, como a que é exibida na Figura 1, e utilizar uma janela deslizante para identificar automaticamente árvores, mudas e falhas. Dessa maneira, torna-se essencial a inclusão da classe *outros*, para diferenciar as classes anteriores de regiões de campo, floresta, ou qualquer outro elemento que possa estar presente na imagem obtida pelo VANT.

B. Extração das características

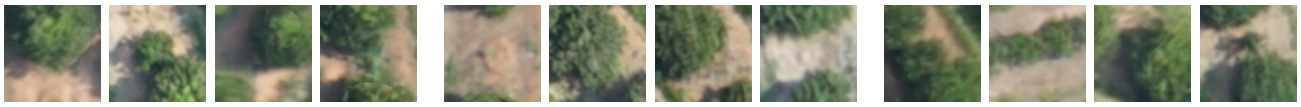
Ao analisar visualmente as imagens, foi possível perceber que informações sobre a textura e a cor eram claramente discriminantes para diferenciar as classes. Logo, para representar cada imagem, foram extraídos os seguintes descritores para serem utilizados no vetor de características:

- Local Binary Pattern (LBP): é um eficiente descritor de textura proposto em [13]. Os parâmetros utilizados para computar o LBP foram: *oito pontos*, *raio um* e *método uniforme*. A partir disso, foi criado um histograma normalizado que foi utilizado no vetor de características.
- Gray-Level Co-Occurrence Matrix (GLCM): é um dos métodos mais conhecidos para análise de textura, cuja ideia geral é criar uma matriz de co-ocorrência que armazena as mudanças de intensidade dos níveis de cinza que ocorrem na imagem [14]. A partir dessa matriz, as medidas estatísticas a seguir foram calculadas para representar as características de textura da imagem: *contraste*, *dissimilaridade*, *homogeneidade*, *segundo momento angular*, *energia* e *correlação*.
- Hue-Saturation-Value (HSV): é um espaço de cores que considera matiz, saturação e brilho. Optou-se por usar

(a) Árvore (b) Muda

(c) Falha (d) Outros

Figura 2. Exemplos das imagens 50×50 pixels do banco de dados utilizado.



(a) Imagens da classe *árvore* que podem ser interpretadas como muda ou falha (b) Imagens da classe *muda* que podem ser interpretadas como árvore ou falha (c) Imagens da classe *falha* que podem ser interpretadas como árvore ou muda

Figura 3. Exemplos de imagens com falhas de anotação presentes no banco de dados.

esse espaço de cores ao invés do RGB pois no banco de dados utilizado existem variações consideráveis na iluminação e na orientação das sombras [15]. Foi feito um histograma normalizado para cada componente e posteriormente os histogramas foram concatenados.

C. Configuração dos experimentos

Os classificadores utilizados para realizar os experimentos foram: K-Nearest Neighbors (KNN), Random Forest (RF) e Support Vector Machine (SVM). Além disso, alguns de seus parâmetros foram selecionados, baseados em [16]–[20], para serem modificados a cada experimento, nos quais estão representados na Tabela II. Os demais parâmetros se mantiveram no modo *default* de acordo com a biblioteca scikit-learn do Python.

Tabela II
PARÂMETROS UTILIZADOS NOS EXPERIMENTOS

Classificador	Parâmetros	Valores
KNN	Valor de K	[1, 3, 5, ..., 29]
	Tamanho da folha	[1, 5, 10, 15, 20]
RF	Número de estimadores	[50, 100, 150, 200]
	Max. Features	[sqrt, log2, auto]
SVM	Valor de gamma	[0.001, 0.01, 0.1, 1]
	Valor de C	[0.01, 0.1, 1, 10]
	Kernel	[linear]

Para conjuntos de dados desbalanceados, como o que foi utilizado nesse trabalho, a análise dos resultados de algoritmos de classificação deve ser analisada com cautela, pois como eles maximizam a taxa de acurácia do classificador, a classe majoritária tende a ser beneficiada em relação a minoritária. Para resolver esse problema, as seguintes técnicas de balanceamento foram empregadas:

- Undersampling: tem como objetivo balancear o conjunto de dados diminuindo o número de amostras das classes majoritárias, de forma que se iguale com as das minoritárias. Nos nossos experimentos, utilizamos a técnica de random undersampling, na qual as amostras das classess majoritárias são selecionadas de maneira aleatória.
- Oversampling: realiza o balanceamento do conjunto de dados ao criar ou replicar as amostras das classes minoritárias, de forma que se iguale com as das majoritárias. Nos nossos experimentos, o oversampling foi feito utilizando as seguintes técnicas: Synthetic Minority Over-sampling (SMOTE) [21], Adaptive Synthetic (ADASYN) [22] e o random oversampling, onde as duas primeiras criam amostras sintéticas e a última replica aleatoriamente amostras já existentes. Além disso, o método SMOTE possui diversas variações [23], e dentre essas, o Borderline-SMOTE e o SVM-SMOTE também foram utilizados. Utilizamos a biblioteca imbalanced-learn para executar todos esses métodos de oversampling.

Além das diversas configurações dos classificadores e dos diferentes métodos de balancamento utilizados para realizar os experimentos, o vetor de características também foi variado de acordo com as diferentes combinações possíveis dos três descritores, que são: (i) LBP; (ii) HSV; (iii) GLCM; (iv) LBP e HSV; (v) LBP e GLCM; (vi) HSV e GLCM; (vii) LBP, HSV e GLCM.

A métrica utilizada para avaliar o desempenho do classificador foi o F1 score (1), que é uma medida que combina *precision* (2) e *recall* (3).

$$F1score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{1}$$

$$Precision = \frac{VerdadeiroPositivo}{VerdadeiroPositivo + FalsoPositivo} \tag{2}$$

$$Recall = \frac{VerdadeiroPositivo}{VerdadeiroPositivo + FalsoNegativo} \tag{3}$$

IV. EXPERIMENTOS

Para realizar os experimentos, foi utilizado o *stratified* 10-*fold cross-validation*, com a proporção de 90% para treinamento e 10% para teste, em que as amostras de cada classe foram selecionadas aleatoriamente a partir de todo o conjunto de dados. Porém, como foi dito anteriormente, o banco possui um número razoável de dados inconsistentes, e como as amostras são escolhidas de maneira arbitrária, pode existir um caso em que muitas dessas amostras ruins são selecionadas, prejudicando assim o resultado final. Para lidar com esse problema, em cada experimento foi realizado o *cross-validation* cinco vezes, cada um com um conjunto de amostras gerado aleatoriamente.

O oversampling foi aplicado apenas após o particionamento do *cross-validation*, pois isso garante que as amostras geradas para as classes minoritárias sejam usadas apenas para o treinamento do modelo e que o teste seja feito exclusivamente com as amostras reais.

A Tabela III mostra a quantidade de experimentos executados.

Tabela III
NÚMERO DE EXPERIMENTOS REALIZADOS

Classificador	Nº de amostras em cada classe	Nº de experimentos
KNN	2400	525
	19779	3150
	30000	3150
RF	2400	84
	19779	504
	30000	504
SVM	2400	112
	19779	672
	30000	672

A. Resultados e discussão

A Tabela IV exibe os cinco melhores resultados obtidos por cada classificador, tomando como métrica a média do F1 score das quatro classes. Como pode-se observar, o maior desempenho obtido de todos os experimentos realizados foi um F1 score de 80,3%, utilizando o Random Forest.

A Tabela V mostra qual a configuração que adquiriu a melhor performance dentre as sete combinações diferentes de vetores de características. Como era esperado, o descritor de cor é muito relevante para a classificação dessas classes, e sua presença no vetor de características se torna fundamental para que se tenha um bom resultado.

Na Tabela VI estão os maiores desempenhos alcançados pelos classificadores em cada método de balanceamento diferente. O melhor F1 score, de 78,7%, desconsiderando o maior resultado já citado anteriormente, foi conquistado pelo Random Forest com o uso do Borderline2-SMOTE. Ao analisar essa mesma tabela, é possível perceber que a classe *muda* é muito comprometida quando o oversampling é realizado em conjunto com o Random Forest, pois quando isso ocorre, todos os piores F1 scores atingidos para essa classe pertencem à esse classificador. Por outro lado, ocorre totalmente o oposto nas outras classes, pois esse classificador foi o que atingiu o maior resultado em todos os métodos de balanceamento.

Utilizando esses experimentos da Tabela VI, foi calculada a média dos F1 scores obtidos nos seis métodos de oversampling, para cada uma das quatro classes, e ao fazer a diferença dessas médias com os valores do F1 score do random undersampling, obteve-se os seguintes dados para cada classificador:

- KNN: as classes *árvore*, *falha* e *outros* tiveram um aumento no F1 score de aproximadamente 2,13%, 5,52% e 3,45% respectivamente. Já a classe *muda* teve um decréscimo de 0,22%.
- Random Forest: as classes *árvore*, *falha* e *outros* tiveram um aumento no F1 score de aproximadamente 1,62%, 1,17% e 1,82% respectivamente. Já a classe *muda* teve um decréscimo de 20,83%.
- SVM: as classes *árvore*, *muda*, *falha* e *outros* tiveram um aumento no F1 score de aproximadamente 0,42%, 0,47%, 1,05% e 0,8% respectivamente.

A Tabela VII apresenta quais foram os melhores resultados obtidos em cada classe pelos três classificadores. O Random Forest conquistou o melhor F1 score para todas as classes, porém para as classes *árvore*, *falha* e *outros* os melhores resultados foram com o uso do oversampling, e para a classe *muda* o maior valor obtido foi com o uso do undersampling. Como esperado, nota-se também que as classes majoritárias (*árvore* e *outros*) possuem acurácia consideravelmente superior a das classes minoritárias (*mudas* e *falhas*)

A Tabela VIII mostra quais foram os cinco F1 scores mais baixos que cada classificador teve, porém só foi considerado os experimentos com o vetor de características com os três descritores, visto que o objetivo principal dessa tabela é comparar com a Tabela IV a influência que os parâmetros dos classificadores têm.

Todas essas comparações feitas acima mostram que existem alguns padrões nos melhores resultados obtidos nesses experimentos. O Random Forest se destacou muito em relação aos outros dois classificadores, pois além de atingir o maior F1 score médio utilizando o undersampling e o oversampling, também foi superior no resultado de cada uma das classes e em praticamente todos os descritores diferentes. O único fator em que ele foi inferior em relação aos outros dois foi na classe *muda* nos casos em que o oversampling foi realizado, no qual prejudicou consideravelmente o F1 score final. O motivo dessa queda de desempenho nesses casos ainda não se sabe ao certo, seria necessário um trabalho mais a fundo para identificar a causa.

Os experimentos indicam que de um modo geral, o oversampling melhora o F1 score, porém é necessário verificar se esse aumento realmente vale a pena, levando em conta que quando esse procedimento é feito, o tempo de execução do experimento fica consideravelmente maior. No caso do Random Forest, a escolha do método de oversampling foi mais relevante, pois o F1 score do Borderline2-SMOTE (78,7%) foi muito superior ao do Random Oversampling (70,2%). Já para o caso do KNN e SVM, a técnica de oversampling utilizada

Tabela IV
MELHORES RESULTADOS DE CADA CLASSIFICADOR

Classificador	Nº de amostras em cada classe	Parâmetros		Descritores	Método de balanceamento	F1 score					Tempo de execução (h:m:s)
						Árvore	Muda	Falha	Outros	Média	
KNN		Valor de K	Tamanho da folha								
	30000	11	10	LBP, HSV e GLCM	SMOTE	0.886	0.686	0.634	0.928	0.784	0:54:4.7
	30000	13	1	LBP, HSV e GLCM	SMOTE	0.885	0.692	0.632	0.926	0.784	1:26:25.31
	30000	9	10	LBP, HSV e GLCM	SMOTE	0.886	0.682	0.639	0.929	0.784	0:52:2.9
	30000	9	15	LBP, HSV e GLCM	SMOTE	0.886	0.682	0.638	0.93	0.784	0:49:34.14
	19779	15	20	LBP, HSV e GLCM	SVM-SMOTE	0.883	0.686	0.643	0.919	0.783	3:5:20.21
RF		Nº de estimadores	Max. features								
	2400	200	auto	LBP, HSV e GLCM	Random undersampling	0.884	0.733	0.664	0.931	0.803	0:27:0.44
	2400	100	sqrt	LBP, HSV e GLCM	Random undersampling	0.885	0.726	0.664	0.93	0.801	0:13:52.57
	2400	200	log2	LBP e HSV	Random undersampling	0.89	0.724	0.652	0.937	0.801	0:23:26.99
	2400	200	sqrt	LBP, HSV e GLCM	Random undersampling	0.886	0.727	0.659	0.932	0.801	0:26:54.04
	2400	100	auto	HSV	Random undersampling	0.888	0.726	0.652	0.935	0.8	0:11:50.71
SVM		Valor de gamma	Valor de C								
	19779	0.001	10	LBP, HSV E GLCM	Random oversampling	0.876	0.706	0.641	0.895	0.78	4:22:41.15
	19779	0.01	10	LBP, HSV E GLCM	Random oversampling	0.877	0.706	0.641	0.895	0.78	4:24:28.9
	19779	1	10	LBP, HSV E GLCM	Random oversampling	0.877	0.706	0.641	0.895	0.78	4:30:34.06
	30000	0.001	10	LBP, HSV E GLCM	Random oversampling	0.877	0.706	0.643	0.895	0.78	10:25:1.81
	30000	0.01	10	LBP, HSV E GLCM	Random oversampling	0.877	0.706	0.643	0.896	0.78	10:53:59.02

Tabela V
MELHOR RESULTADO DE CADA VETOR DE CARACTERÍSTICAS

Descritores	Classificador	Nº de amostras em cada classe	Parâmetros		Método de balanceamento	F1 score					Tempo de execução (h:m:s)
						Árvore	Muda	Falha	Outros	Média	
LBP	RF	2400	150 (nº de estimadores)	auto (max. features)	Random undersampling	0.567	0.512	0.341	0.511	0.483	0:18:23.65
HSV	RF	2400	100 (nº de estimadores)	auto (max. features)	Random undersampling	0.888	0.726	0.652	0.935	0.8	0:11:50.71
GLCM	SVM	30000	0.001 (valor de gamma)	10 (valor de C)	Random oversampling	0.604	0.494	0.459	0.771	0.582	7:31:44.68
LBP e HSV	RF	2400	200 (nº de estimadores)	log2 (max. features)	Random undersampling	0.89	0.724	0.652	0.937	0.801	0:23:26.99
LBP e GLCM	RF	2400	200 (nº de estimadores)	sqrt (max. features)	Random undersampling	0.674	0.623	0.512	0.799	0.652	0:25:36.29
HSV e GLCM	RF	2400	100 (nº de estimadores)	sqrt (max. features)	Random undersampling	0.885	0.718	0.647	0.928	0.795	0:13:2.74
LBP, HSV e GLCM	RF	2400	200 (nº de estimadores)	auto (max. features)	Random undersampling	0.884	0.733	0.664	0.931	0.803	0:27:0.44

Tabela VI
MELHORES RESULTADOS EM CADA MÉTODO DE BALANCEAMENTO

Método de balanceamento	Classificador	Nº de amostras em cada classe	Parâmetros		Descritores	F1 score					Tempo de execução (h:m:s)
						Árvore	Muda	Falha	Outros	Média	
Random Undersampling	KNN	2400	11 (valor de K)	10 (tamanho da folha)	LBP, HSV e GLCM	0.863	0.688	0.577	0.89	0.754	0:1:23.7
	RF	2400	200 (nº de estimadores)	auto (max. features)	LBP, HSV e GLCM	0.884	0.733	0.664	0.931	0.803	0:27:0.44
	SVM	2400	0.1 (valor de gamma)	10 (valor de C)	LBP, HSV e GLCM	0.873	0.694	0.629	0.887	0.771	0:3:45.42
ADASYN	KNN	19779	7 (valor de K)	7 (tamanho da folha)	LBP, HSV e GLCM	0.884	0.693	0.618	0.927	0.781	0:26:52.3
	RF	30000	200 (nº de estimadores)	auto (max. features)	LBP, HSV e GLCM	0.902	0.569	0.682	0.95	0.776	6:25:27.59
	SVM	19779	0.001 (valor de gamma)	10 (valor de C)	LBP, HSV e GLCM	0.878	0.71	0.631	0.895	0.778	4:24:2.58
Random Oversampling	KNN	30000	11 (valor de K)	20 (tamanho da folha)	LBP, HSV e GLCM	0.886	0.672	0.635	0.927	0.78	0:29:59.18
	RF	19779	50 (nº de estimadores)	auto (max. features)	LBP, HSV e GLCM	0.896	0.32	0.647	0.944	0.702	0:49:46.93
	SVM	19779	0.001 (valor de gamma)	10 (valor de C)	LBP, HSV e GLCM	0.876	0.706	0.641	0.895	0.78	4:22:41.15
Borderline1-SMOTE	KNN	19779	7 (valor de K)	10 (tamanho da folha)	LBP, HSV e GLCM	0.882	0.687	0.633	0.926	0.782	0:24:53.68
	RF	30000	100 (nº de estimadores)	sqrt (max. features)	LBP, HSV e GLCM	0.902	0.553	0.68	0.951	0.772	4:10:57.63
	SVM	19779	0.001 (valor de gamma)	10 (valor de C)	LBP, HSV e GLCM	0.876	0.702	0.64	0.894	0.778	4:12:21.69
Borderline2-SMOTE	KNN	19779	13 (valor de K)	20 (tamanho da folha)	LBP, HSV e GLCM	0.885	0.691	0.63	0.92	0.782	0:25:24.26
	RF	30000	150 (nº de estimadores)	auto (max. features)	LBP, HSV e GLCM	0.902	0.606	0.687	0.954	0.787	5:0:50.33
	SVM	30000	0.001 (valor de gamma)	10 (valor de C)	LBP, HSV e GLCM	0.878	0.699	0.628	0.893	0.774	11:37:6.47
SMOTE	KNN	30000	11 (valor de K)	10 (tamanho da folha)	LBP, HSV e GLCM	0.886	0.686	0.634	0.928	0.784	0:54:4.7
	RF	19779	150 (nº de estimadores)	sqrt (max. features)	LBP, HSV e GLCM	0.899	0.564	0.679	0.948	0.773	2:23:40.15
	SVM	30000	1 (valor de gamma)	10 (valor de C)	LBP, HSV e GLCM	0.877	0.705	0.643	0.896	0.78	10:10:46.34
SVM-SMOTE	KNN	19779	15 (valor de K)	20 (tamanho da folha)	LBP, HSV e GLCM	0.883	0.686	0.643	0.919	0.783	3:5:20.21
	RF	19779	200 (nº de estimadores)	sqrt (max. features)	LBP, HSV e GLCM	0.9	0.536	0.679	0.948	0.766	6:50:12.02
	SVM	30000	0.01 (valor de gamma)	10 (valor de C)	LBP, HSV e GLCM	0.878	0.67	0.654	0.897	0.775	19:14:39.56

Tabela VII
MELHORES RESULTADOS DE CADA CLASSE

Classe	Classificador	Nº de amostras em cada classe	Parâmetros		Descritores	Método de balanceamento	F1 score	Tempo de execução
Árvore	KNN	30000	11 (valor de k)	1 (tamanho da folha)	HSV e GLCM	SMOTE	0.887	0:54:25.89
	RF	30000	200 (nº de estimadores	sqrt (max. features)	LBP, HSV e GLCM	SVM-SMOTE	0.904	15:7:16.03
	SVM	30000	0.01	10	LBP, HSV e GLCM	ADASYN	0.879	11:7:39.01
Muda	KNN	19779	25	1	LBP, HSV e GLCM	SMOTE	0.711	0:42:13.47
	RF	2400	200	auto	LBP, HSV e GLCM	Random undersampling	0.733	0:27:0.44
	SVM	30000	0.1	10	LBP, HSV e GLCM	ADASYN	0.715	11:7:51.55
Falha	KNN	19779	3	1	LBP, HSV e GLCM	Random oversampling	0.655	0:24:45.48
	RF	30000	150	auto	LBP, HSV e GLCM	Borderline2-SMOTE	0.687	5:0:50.33
	SVM	19779	0.001	10	LBP, HSV e GLCM	SVM-SMOTE	0.656	6:32:15.26
Outros	KNN	30000	3	1	LBP, HSV e GLCM	SMOTE	0.932	0:56:35.62
	RF	30000	100	auto	LBP e HSV	Borderline2-SMOTE	0.954	2:54:51.53
	SVM	30000	0.1	10	LBP, HSV e GLCM	Borderline1-SMOTE	0.898	11:19:40.97

Tabela VIII
PIORES RESULTADOS DE CADA CLASSIFICADOR

Classificador	Nº de amostras em cada classe	Parâmetros		Descritores	Método de balanceamento	F1 score					Tempo de execução (h:m:s)
						Árvore	Muda	Falha	Outros	Média	
KNN		Valor de K	Tamanho da folha								
	30000	1	20	LBP, HSV e GLCM	Random oversampling	0.844	0.482	0.638	0.927	0.723	0:20:23.65
	30000	1	1	LBP, HSV e GLCM	Random oversampling	0.846	0.484	0.639	0.927	0.724	0:27:39.37
	30000	1	5	LBP, HSV e GLCM	Random oversampling	0.845	0.486	0.639	0.928	0.724	0:24:18.26
	30000	1	15	LBP, HSV e GLCM	Random oversampling	0.848	0.486	0.64	0.927	0.725	0:20:57.88
	19779	1	10	LBP, HSV e GLCM	Random oversampling	0.85	0.487	0.643	0.923	0.726	0:14:17.32
RF		Nº de estimadores	Max. features								
	30000	150	log2	LBP, HSV e GLCM	Random oversampling	0.895	0.241	0.636	0.942	0.679	3:10:37.92
	30000	200	log2	LBP, HSV e GLCM	Random oversampling	0.895	0.24	0.636	0.942	0.679	4:0:37.56
	30000	100	log2	LBP, HSV e GLCM	Random oversampling	0.896	0.25	0.637	0.942	0.681	2:0:27.93
	30000	150	sqrt	LBP, HSV e GLCM	Random oversampling	0.897	0.252	0.639	0.945	0.683	3:58:2.29
	30000	50	log2	LBP, HSV e GLCM	Random oversampling	0.893	0.261	0.637	0.941	0.683	1:2:28.25
SVM		Valor de gamma	Valor de C								
	2400	0.01	0.01	LBP, HSV E GLCM	Random undersampling	0.788	0.62	0.498	0.775	0.67	0:7:35.18
	2400	1	0.01	LBP, HSV E GLCM	Random undersampling	0.79	0.626	0.49	0.78	0.671	0:7:34.22
	2400	0.001	0.01	LBP, HSV E GLCM	Random undersampling	0.79	0.62	0.497	0.78	0.672	0:7:36.94
	2400	0.1	0.01	LBP, HSV E GLCM	Random undersampling	0.788	0.622	0.498	0.779	0.672	0:7:38.56
	30000	0.001	0.01	LBP, HSV E GLCM	ADASYN	0.839	0.679	0.49	0.84	0.712	15:9:3.29

não modificou tanto os resultados: a diferença entre o melhor e o pior resultado é de apenas 0,4% no KNN e 0,6% no SVM.

V. CONCLUSÃO

Nesse artigo, exploramos diferentes configurações experimentais para a classificação de imagens aéreas de citrus em quatro categorias: (i) árvores adultas, (ii) mudas, (iii) falhas, e (iv) outros. Os resultados dos experimentos realizados indicam que a escolha de parâmetros e métodos de balanceamento possuem grande impacto na classificação (especialmente para as classes minoritárias – mudas e falhas), além de reforçar descobertas intuitivas, como a importância da informação de cor para a classificação das imagens consideradas. De acordo com o nosso conhecimento, nenhum trabalho prévio explorou uma quantidade tão elevada de experimentos para endereçar a tarefa abordada.

Como trabalho futuro, pretendemos utilizar modelos baseados em Deep learning e avaliar se conseguimos melhores resultados em comparação com os modelos empregados, tanto para extração de features quanto para classificação, mantendo uma mesma abordagem experimental a fim de avaliar diferentes configurações para os dados e para a rede.

REFERÊNCIAS

[1] United States Department of Agriculture (USDA), "Citrus: World markets and trade," https://www.fas.usda.gov/data/citrus-world-markets-and-trade, Accessed: 2019-05.

[2] D. Koc-San, S. Selim, N. Aslan, and B. T. San, "Automatic citrus tree extraction from UAV images and digital surface models using circular hough transform," *Computers and Electronics in Agriculture*, vol. 150, pp. 289 – 301, 2018.

[3] A. Ozdarici-Ok, "Automatic detection and delineation of citrus trees from vhr satellite imagery," *International Journal of Remote Sensing*, vol. 36, no. 17, pp. 4275–4296, 2015.

[4] A. O. Ok and A. Ozdarici-Ok, "Detection of citrus trees from UAV DSMS," *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. IV-1/W1, pp. 27–34, 2017. [Online]. Available: https://www.isprs-ann-photogramm-remote-sens-spatial-inf-sci.net/IV-1-W1/27/2017/

[5] A. O. Ok and A. O. Ok, "Automated detection of citrus trees from a digital surface model," in *2017 25th Signal Processing and Communications Applications Conference (SIU)*, May 2017, pp. 1–4.

[6] P. Maillard and M. F. Gomes, "Detection and counting of orchard trees from vhr images using a geometrical-optical model and marked template matching," *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. III-7, pp. 75–82, 2016. [Online]. Available: https://www.isprs-ann-photogramm-remote-sens-spatial-inf-sci.net/III-7/75/2016/

[7] O. Csillik, J. Cherbini, R. Johnson, A. Lyons, and M. Kelly, "Identification of citrus trees from unmanned aerial vehicle imagery using convolutional neural networks," *Drones*, vol. 2, no. 4, 2018. [Online]. Available: http://www.mdpi.com/2504-446X/2/4/39

[8] K. Shivanna and R. Tandon, "Seedling recruitment," in *Reproductive Ecology of Flowering Plants: A Manual*. Springer, 2014, pp. 145–162.

[9] J. A. López, E. I. Verdiguier, L. G. Chova, J. M. Marí, J. R. Barreiro, G. C. Valls, and J. C. Maravilla, "Land cover classification of vhr airborne images for citrus grove identification," *ISPRS journal of photogrammetry and remote sensing*, vol. 66, no. 1, pp. 115–123, 2011.

[10] M. Zortea, M. Macedo, A. Britto Mattos, B. C Ruga, and B. H Gemignani, "Automatic citrus tree detection from uav images based on convolutional neural networks," in *2018 31th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, 11 2018.

[11] D. K. Nkemelu, D. Omeiza, and N. Lubalo, "Deep convolutional neural network for plant seedlings classification," *CoRR*, vol. abs/1811.08404, 2018. [Online]. Available: http://arxiv.org/abs/1811.08404

[12] H. S. Abdullahi, R. Sheriff, and F. Mahieddine, "Convolution neural network in precision agriculture for plant image recognition and classification," in *2017 Seventh International Conference on Innovative Computing Technology (Intech), Ieee, Londrés*, 2017, pp. 1–3.

[13] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," *Pattern recognition*, vol. 29, no. 1, pp. 51–59, 1996.

[14] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Textural features for image classification," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-3, no. 6, pp. 610–621, Nov 1973.

[15] S. Sural, G. Qian, and S. Pramanik, "Segmentation and histogram generation using the hsv color space for image retrieval," in *Proceedings. International Conference on Image Processing*, vol. 2. IEEE, 2002.

[16] Mohtadi Ben Fraj, "In depth: Parameter tuning for KNN," https://medium.com/p/in-depth-parameter-tuning-for-knn-4c0de485baf6, Accessed: 2019-05.

[17] "In depth: Parameter tuning for random forest," https://link.medium.com/QxBb1qVBRW, Accessed: 2019-05.

[18] "In depth: Parameter tuning for SVC," https://link.medium.com/acuwGvW83W, Accessed: 2019-05.

[19] H. Dongyuan and C. Xiaoyun, "Tuning svm hyperparameters in the primal," in *2010 Second International Conference on Computational Intelligence and Natural Computing*, vol. 1. IEEE, 2010, pp. 201–204.

[20] S. Bernard, L. Heutte, and S. Adam, "Influence of hyperparameters on random forest accuracy," in *International Workshop on Multiple Classifier Systems*. Springer, 2009, pp. 171–180.

[21] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[22] H. He, Y. Bai, E. A. Garcia, and S. Li, "Adasyn: Adaptive synthetic sampling approach for imbalanced learning," in *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*. IEEE, 2008, pp. 1322–1328.

[23] A. Fernández, S. Garcia, F. Herrera, and N. V. Chawla, "Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary," *Journal of artificial intelligence research*, vol. 61, pp. 863–905, 2018.