Performance Analysis of Machine Learning-Based Systems for Detecting Deforestation

1st Michel de Araújo
Department of Computing
Federal Rural University of Pernambuco
Recife, Brazil
michel.arruda@ufrpe.br

2nd Ermeson Andrade
Department of Computing
Federal Rural University of Pernambuco
Recife, Brazil
ermeson.andrade@ufrpe.br

3rd Fumio Machida
Department of Computer Science
University of Tsukuba
Tsukuba, Japan
machida@cs.tsukuba.ac.jp

***Abstract*—Remote monitoring has become an important tool for recognizing land and ground objects through sensor data analysis. The use of Machine Learning (ML) algorithms for classification of remote monitoring images has increased in recent years. ML-based image classifiers have played an important role in detecting deforestation, illegal mining or fire. However, the precise classification of land use is a huge challenging task, especially in remote tropical regions, due to the complex biophysical environment and the limitations of the remote monitoring infrastructure. This work aims at studying the trade-offs between performance and accuracy of classification systems for the Brazilian Amazon rainforest, taking into account different computing platforms (server and edge), ML algorithms and images sizes. Although there is a direct relationship between image accuracy and quality, our experimental study shows that it is possible to use low-cost computational environments to perform image classification. The results indicate that Amazon rainforest can be monitored with affordable computing resources such as drones.**

***Index Terms*—Performance, Machine Learning, Deforestation**

I. Introduction

The use of Remote Sensing (RS) has become crucial for various application areas, such as urban recognition and environmental protection. The Amazon rainforest, for example, which is a symbol of biodiversity and also known as the lungs of the world, depends a lot on RS due to its vast territory [1]. Since the 1990s, the Brazilian government's INPE (National Institute of Space Research) together with international research institutions, such as United States Geological Survey (USGS) and National Aeronautics and Space Administration (NASA), have developed RS policies in the Amazon rainforest aiming to map the use of this vast territory [1]. The use includes housing, family farming, legal and illegal mining and deforestation. Several algorithms, specifically those in the Computer Vision (CV) and Pattern Recognition (PR) fields, have been successfully developed to classify satellites images such as Landsat 8 OLI (Operational and Land Imager) and MODIS (Moderate Resolution Imaging Spectroradiometer).

Generally, the RS is carried out as follows. Geospatial satellites take pictures of areas of interest over a period of time. Then, the images are sent to remote servers, so that specialized professionals analyze the images in search of changes, such as new deforestation, fires, illegal mining, among others [2]. This approach can be very time-consuming because it requires a lot of manual work and attention from the professionals. Over the years, several new approaches have been proposed for RS. For instance, Maretto [2] proposed an autonomous process for detecting Land Use and Land Cover (LULC) through Deep Learning (DL) algorithms, such as CNN (Convolutional Neural Network) and its variations. That work represents a great advance for the environmental protection, since images can be analyzed automatically by computers with almost human-like accuracy. Nevertheless, a major disadvantage of this approach is the size of the pictures taken by satellites, which are typically very large [3]. Consequently, it requires servers with high computational power to process DL algorithms which results in an expensive solution.

As Brazilian forests are vast, and the scale and intensity at which illegal deforestation occur grows every year [4], it is necessary to use tools capable of responding quickly to these situations. Drone-based surveillance systems have been used for that purpose [5]. With edge computing capabilities, drones are a viable alternative for executing operations to combat illegal activities in the Amazon basin. Additionally, drones have some advantages over traditional satellite monitoring system, such as (i) drones are much cheaper and more accessible, (ii) they are easy to use, (iii) drones can visit an area multiple times a day, and (iv) images taken by a drone do not suffer from the presence of clouds. However, the performance trade-offs of using DL algorithms for detecting deforestation in these edge-based platforms need to be investigated.

Several works have demonstrated the effectiveness of ML algorithms for LULC and other image classification problems [6]. However, to the best of our knowledge, none of the existing works has considered the differences in performance of these algorithms on edge and server platforms. Thus, this work aims to study the trade-offs between performance and accuracy of ML classification systems for detecting deforestation considering these platforms. More specifically, we experimentally investigate the impacts of image sizes (8x8, 16x16, 32x32, 64x64, and 128x128) and ML algorithms (k-NN (K-Nearest Neighbour), RF (Random Forests) and CNN) on the accuracy and performance of edge and server platforms. Our results reveal two important insights. First, the classification time on the edge platform can be almost twice as long as

on the server platform due to its lower processing power. Specifically for the CNN algorithm, the classification time was of 0.123 seconds for the edge and 0.07 seconds for the server, respectively. On the other hand, for the RF and k-NN algorithms, the difference is negligible. Second, even reducing the size of the images, it is still possible to obtain a good accuracy for detecting deforestation. For CNN, for example, an average accuracy of approximately 0.92 was obtained for the 32x32 image size and 0.95 for the 64x64 size. We hope this work can provide low-cost surveillance options that can be used in developing regions with little access to capital (e.g.: indigenous community).

The remainder of the paper is organized as follows. Section II presents the related work. Section III introduces fundamental concepts adopted in the paper. Section IV describes the proposed approach. Section V describes the experimental results. Finally, the Section VI presents the conclusions and briefly introduces the future work.

II. Related works

The use of ML for detecting deforestation is a relatively new topic that has attracted the attention of researchers. In [7], a detailed survey on the different ML algorithms for detecting deforestation is presented. To position our paper and indicate its contributions, we first summarize some related work that has been done in the classification of satellite images. Next, we discuss the related work with respect to drone applications. Lastly, we provide a comparison of our work in relation to the existing works in terms of context and evaluation.

Several techniques for classification of satellite images have been developed in recent years. L. Zhang, L. Zhang, and B. Du [8] showed the advantages of using RS and how DL algorithms can be combined with satellite images to extract value. Besides, the authors provided a technical explanation of CNN, autoencoder, Restricted Boltzmann Machines (RBM), and sparse coding algorithms, in addition to illustrating all the basic processes of using DL techniques for RS. *Zhu et al.* [9] performed an extensive review on the use of DL in RS images. They explained a greater variety of algorithms and how different spectral bands are used to perform different classification tasks in RS images such as target detection, semantic segmentation and pixel-wise classification.

Recently, drone-based systems have been used for image classification. In [10], H. Hildmann and E. Kovacs presented a review of the use of systems based on unmanned aerial vehicles (UAVs) for activities related to safety, such as immediate response to disasters and public and civil security. Peneque-Gálvez *et al.* [5] conducted a feasibility study on the use of drones by communities to monitor local deforestation. The work showed the benefits not only for the community itself, but also for partner organizations and the general public.

Considering the works available in the literature, none of the studies aimed at evaluating the performance of server and edge environments in the context of deforestation monitoring. In addition, this work aims to evaluate various machine learning algorithms developed for detecting deforestation in order to explore the *trade-offs* between performance and accuracy. As the computational resources required to run ML algorithms varies widely, we also investigate the performance impacts of these algorithms deployed in server and edge platforms.

III. Background

A. Deforestation

Brazil has the largest tropical forest in the world with an area of approximately five million square kilometers. This forest is home to the richest biodiversity of any ecosystem on the planet. Nevertheless, the rate of deforestation in Brazilian tropical forests is among the highest globally. According to INPE, 2,254 square kilometres of tropical forest were removed from July 2018 to July 2019, an increase of 278% over one year [11]. Deforestation is becoming a global problem with extensive environmental and economic consequences, since it reduces biodiversity, impacts on climate change by CO2 emission, and breaks up indigenous communities [12].

B. Remote sensing (RS)

RS is a way of acquiring, processing, and interpreting images and related data obtained from aircraft, satellites or drones [13]. It has been used for many decades. The advantages of RS include the ability to collect information over large spatial areas, to identify natural features or physical objects on the ground, to observe objects or areas on a systematic basis, among others [8]. In the context of deforestation, remote sensing is carried out using Satellite or drone-based systems. Satellite or drone-based systems can cover areas that are difficult to access, dangerous for humans or environmentally sensitive. It can also identify where mining, new agriculture fields, or other deforestation have replaced native forest and serve as evidence for authorities to take actions against violators. It is worth to highlight, however, that different from tradition satellite imagery, drones can obtain high-resolution images from cloudy areas and do not have to wait for satellites to pass over the monitoring area.

C. ML techniques for detecting deforestation

Machine Learning is a field of Artificial Intelligence. It consists of algorithms that allow the machine to learn from data, without being explicitly programmed. There are a good number of ML techniques for detecting deforestation, such as Support Vector Machine (SVM), CNN, Siamese Convolutional Network (S-CNN), among others. SVM is one of the most popular supervised machine learning algorithm used in image classification systems because it performs well when the data set has few labeled samples [14]. Random Forest methods [15] are also commonly used in image classification tasks. However, when it comes to accuracy, Deep Learning, which is a class of machine learning algorithms, is the choice. It can learn by employing several layers of neural networks, resulting in a much better overall accuracy. As DL algorithms are much more complex and typically require much more data to deliver better predictions, they require more computational resources to be trained and executed. Consequently, they might not be a good fit for edge-based environments like drones.

IV. Experimental Methods

A. Data Structure

In this paper, a dataset from a 2017 Kaggle competition called *Planet: Understanding the Amazon from Space* was used. The objective of this competition was to label around 40000 images from Brazilian rainforest and build algorithms to classify these images with respect to the land use. The dataset consists of 40479 jpg images with the size of 256x256 pixels. They are classified according to 17 different labels and each image can have more than one label - thus being a multi-label classification problem, as shown in Figures 1 and 2 and Table I. For more information concerning the meaning of each of the labels, the reader should refer to [16]. The dataset was divided into a ratio of 80:10:10, where 80% was used for training, 10% for test and 10% for validation. Library Scikit-Learn version 0.20.0 and TensorFlow version 2.2.0 were used for creating and executing the models.

TABLE I
Images names and its possible labels.

Image name	labels
train_0	haze, primary
train_1	agriculture, clear, primary, water
train_2	clear, primary
train_3	clear, primary
train_4	agriculture, clear, habitation, primary, road

B. Platforms

For the experiments, we adopted two platforms for the classification task: a server platform and an edge platform. As satellite images are, usually, sent to a remote server for processing, we adopted initially a server platform for the classification task. On the other hand, drone-based systems perform its classification task locally. Consequently, we adopted an edge computing platform for the drone using a Raspberry Pi model 4. The settings for the adopted platforms are detailed in Table II.

TABLE II
Adopted platforms.

Conf.	Server	Edge
CPU	Intel i5 4°Gen	Cortex-A72
	2.4 GHz Dual core	1.5 GHz Quad core
	8 MB Cache	1 MB Cache
RAM Memory	8Gb DDR3 1300 Mhz	4Gb DDR3 2400 MHz
Disk Storage	500Gb HD	16Gb SD card
GPU	Intel HD 4400	Broadcom VideoCore IV 400MHz

C. ML algorithms

Two ML and one DL algorithms were adopted to perform the classification task, which are k-NN, RF and CNN. The k-NN, which is the most basic one, was trained using $K = 19$ (lowest error rate) for the number of closest neighbors. k-NN is a peculiar algorithm for our problem in question because to classify a new image it loads the entire training set into memory and then performs the comparison one by one. Thus, we anticipate it might not be suitable for the edge platform, which has limited computational resources.

The second algorithm adopted was the RF. Despite the fact that the RF do not have high capacity for abstraction of characteristics, it has been widely used for classification of remote monitoring images. We conducted the training and classification by varying the number of trees (n_trees=100 and n_trees=500) in order to study the trade-offs between accuracy and performance.

The third algorithm adopted in our analysis is a well known algorithm in the field of Computer Vision called Convolutional Neural Networks. This algorithm has a high capacity for classify images. For CNN, we used the VGG architecture [17], that uses 3 blocks with 2 convolution layers followed by a 2x2 Maxpooling. We chose this architecture for its well known in the Computer Vision literature and it is easy to use and understand. Note that DL algorithms are generally highly complex and carry a lot of information due to their high level of feature abstraction. Consequently, the impact on the performance of this type of algorithm is important to be studied, especially for edge-based platforms.

As edge platforms are resource-constrained platform, all algorithms were trained considering different image sizes in order to analyze the trade-offs between image size and accuracy. Thus, the images were reduced from their original size to sizes of 128x128, 64x64, 32x32, 16x16 and 8x8. It is worth to highlight that the algorithms were trained on the server platform, but the classifications were performed on both server and edge platforms.

D. Execution of the experiments

Two testbeds were implemented for the experiments: i) a server testbed and ii) an edge testbed. The server testbed consists of a server executing the image classification models (k-NN, RF and CNN) for all the image sizes (8x8, 16x16, 32x32, 64x64, and 128x128). That is, for each model, all the images of the validation dataset were classified considering the image sizes of 8x8, 16x16, 32x32, 64x64, and 128x128. After that, the average classification time was calculated. We also obtained the confusion matrix for each of these classifications in order to evaluate the models, as explained in the next subsection. Similar to the server testbed, the edge testbed consists of an edge computer that executes the same image classification models for all the image sizes. Finally, we remark that we only computed the average classification times for the edge testbed, since the edge platform is no suitable for training models due to its lack of computational resources.

E. Evaluation of the models

Since we consider a multi-label classification problem for the data, we evaluated the models by Accuracy, precision, recall and F1 score [18]. Accuracy is the most intuitive measure and it is simply a ratio of correctly predicted observations to the total observations. precision, on the other hand, is the ratio of correctly predicted positive observations to the total predicted positive observations. recall is the ratio of correctly

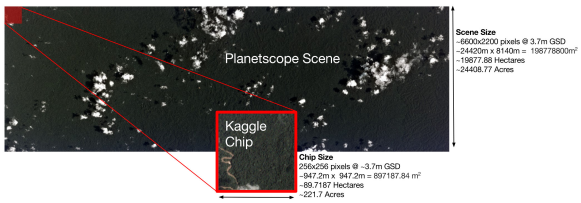

Fig. 1. Kaggle scene and chips.

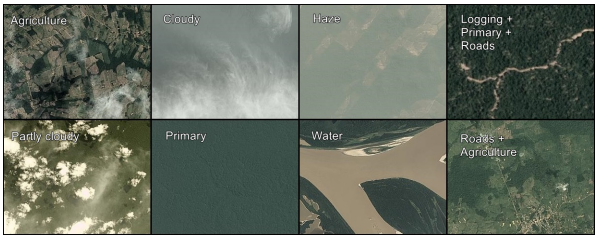

Fig. 2. Kaggle chips and possible labels for each image.

predicted positive observations to the all observations. Lastly, F1 score is the weighted average of precision and recall, so that this score takes both false positives and false negatives into account. As we deal with land use, images classified with potentially dangerous situations (such as illegal mining or selective deforestation), we chose to prioritize the recall metric because it provides the lowest possible number of false negatives. Note that, in the context of deforestation, authorities do not want deforestation activities to go unnoticed. Equations 1, 2, 3 and 4 show the definition of each metric, where TP, TN, FP, and FN represent true positives, true negatives, false positives, and false negatives, respectively.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{1}$$

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$F1\ Score = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{4}$$

V. Results

This section shows the results regarding the accuracy and performance in terms of classification times of the ML algorithms adopted to predict deforestation using server and edge platforms.

A. Server platform

As the server platform was used for training the models, we first present the results regarding this training. After that, we show the classification times. We begin by presenting the results for the CNN. After that, the results for the RF is presented. Lastly, we detail the results for the k-NN.

For the CNN-based classifier, we used the threshold to evaluate the accuracy of detecting deforestation. *Threshold* is the probability of the prediction being classified as positive or negative. For example, when the threshold is set to 0.4 and the CNN predicts the existence of a road in the image with a probability of 0.5, the system classifies it as positive. Otherwise, it is classified as negative. The purpose of the threshold is to adjust the model's sensitivity to the needs of the problem. As it is important that no illegal deforestation activity goes unnoticed, the higher the recall, the greater the probability of detecting illegal deforestation.

Figure 3 shows the impacts of the threshold on Accuracy. In this graph, the x-axis represents the values for the threshold, while the y-axis represents the accuracy obtained by the CNN-based classifier. As can be seen, the higher the threshold and image sizes, the better the accuracy. However, accuracy by itself may not be the best evaluation metric, specially for the deforestation case. Thus, it is necessary to analyze other metrics like precision and recall.

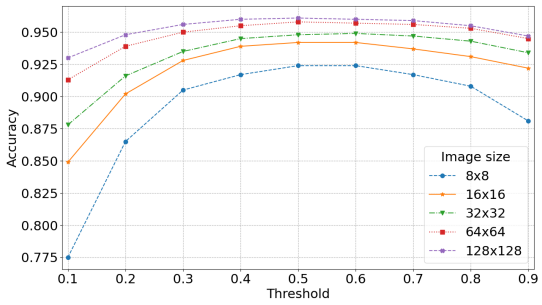

Fig. 3. Accuracy as function of the Threshold for for the CNN-based classifier.

Figure 4 shows the impacts of the threshold on precision and recall. The left graph shows the relationship between threshold

and precision. The right graph shows the relationship between threshold and recall. It is expected that by increasing image sizes, the quality of the CNN-based classifier also increases. However, we noticed that the increase in precision and recall is almost imperceptible when the image size goes from 64x64 to 128x128. The results also show that the values of these metrics are inversely proportional to each other and thus understanding their differences is important when building an efficient classification system. Therefore, the value chosen for the threshold is 0.3, as it provides the most balanced values of precision and recall. Table III summarizes the results for the CNN-based classifier considering the threshold equal to 0.3.

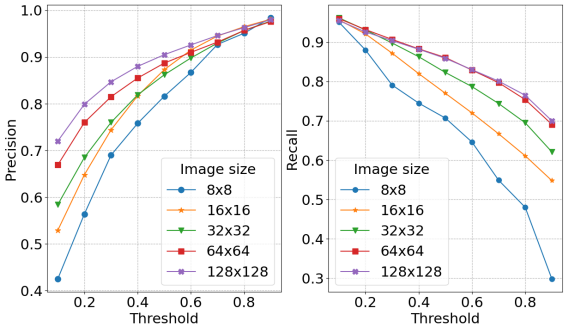

Fig. 4. precision and recall as function of the Threshold for the CNN-based classifier.

TABLE III
RESULTS FOR THE CNN MODEL

Image size	Precision	Recall	Accuracy	F1 Score
8x8	0.69	0.791	0.905	0.737
16x16	0.744	0.872	0.928	0.803
32x32	0.76	0.898	0.935	0.823
64x64	0.815	0.907	0.95	0.859
128x128	0.847	0.904	0.956	0.875

Figure 5 shows the average times that the CNN model takes to classify images of different sizes in the server platform. The x-axis represents the image sizes, while the y-axis represents the average times (in seconds) that the model takes to classify these images. As expected, the results show that the larger the image size is, the greater is its dimensionality and, therefore, the longer the model takes to process all the information. Note that the ratio between the longest and the shortest time is 1.57.

The next classifier trained and tested was the one based on the RF. In this classifier, the parameter n_trees represents the number of estimators that the model should use in the training. The greater the number of trees, the greater the complexity of the model, and therefore, the heavier it becomes. Figure 6 and Table IV show the results of the training for the RF. The left graph shows the relationship between precision and number of trees, while the right graph shows the relationship between recall and number of trees. The results reveal that the image size does not seem to have a linear influence on the results, since the size of the image with the best recall and precision was 32x32.

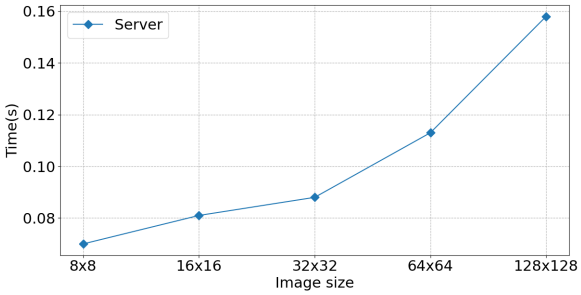

Fig. 5. Average time to classify images for the CNN in the server platform.

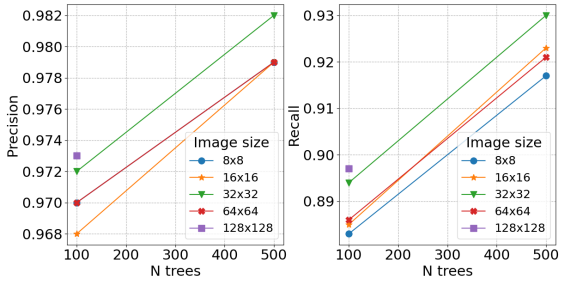

Fig. 6. Precision and recall as function of the number of trees for the Random Forest algorithm.

TABLE IV
RESULTS FOR THE RF MODEL

Image size	N_trees	Precision	Recall	Accuracy	F1 Score
8x8	100	0.970	0.883	0.976	0.924
8x8	500	0.979	0.917	0.983	0.947
16x16	100	0.968	0.885	0.976	0.925
16x16	500	0.979	0.923	0.984	0.955
32x32	100	0.972	0.894	0.978	0.931
32x32	500	0.982	0.930	0.985	0.955
64x64	100	0.970	0.886	0.976	0.926
64x64	500	0.979	0.921	0.983	0.949
128x128	100	0.973	0.897	0.979	0.933
128x128	500	-	-	-	-

Figure 7 shows the average time to classify the images of different sizes in the server platform concerning the RF-based classifier. In this graph, the x-axis represents the image sizes and the y-axis represents the classification time for the images. As expected, the model takes longer to classify images with $n\_trees = 500$ because it is more complex. The ratio between the longest and the shortest classification time is 6.02. However, the server platform was unable to train the RF model for $n\_trees = 500$ and image size of 128x128 due to the lack of memory. Additionally, the results show that the image sizes do not have much influence on the classification time.

The last classifier adopted for the server platform is the one based on the k-NN. For training this classifier, we vary the parameter $k\_neighbors$ from 1 to 40. $k\_neighbors$ is used to reflect the potential relationship of the error derivation at the time of forecasting. Figure 8 shows the the error rate results for the values of $k\_neighbors$, while Table V presents the $k\_neighbors$ that lead to the minimum error rate for each

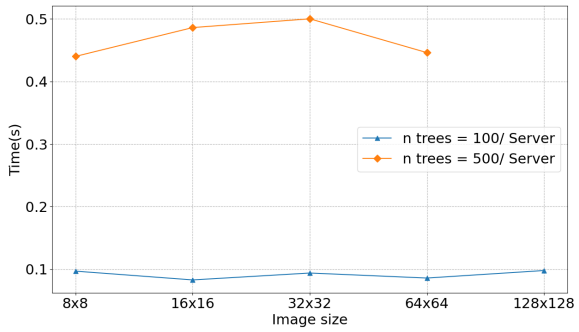

Fig. 7. Average Time to classify images for the RF in a server platform.

image size. For instance, the $k\_neighbors = 19$ is the best parameter for 64x64 image size.

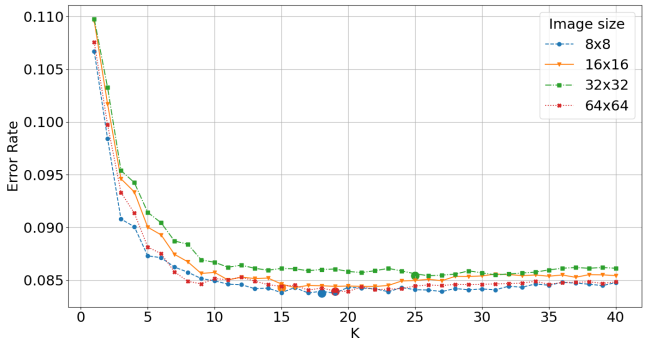

Fig. 8. Error rate as function of the $k\_neighbors$.

TABLE V
K NEIGHBORS NUMBERS FOR THE LEAST ERROR RATE.

Image size	$K\_neighbors$	Error rate
8x8	18	0.0837
16x16	15	0.0842
32x32	25	0.0854
64x64	19	0.0839

Figure 9 and Table VI present the results metrics for the k-NN-based classifier considering the best $K\_neighbors$ showed in Table V. It shows that the accuracy increases with the increase of the image sizes. However, recall does not have a monotonic relationship with the image sizes (see Figure 9). It is also worth to highlight that the ratio between the highest and the lowest value for the precision and recall is only 1.02. Consequently, the best option for k-NN is 64x64 size images. Additionally, the server platform was unable to train the model for images of 128x128 due to lack of RAM. This show that although CNN models are more complex due to their deep layers of feature extraction, they are much more optimized than the k-NN and RF models when it comes to the use of computing resources.

Figure 10 shows the result of the classification analysis for the k-NN-based classifier. The results clearly show that as the

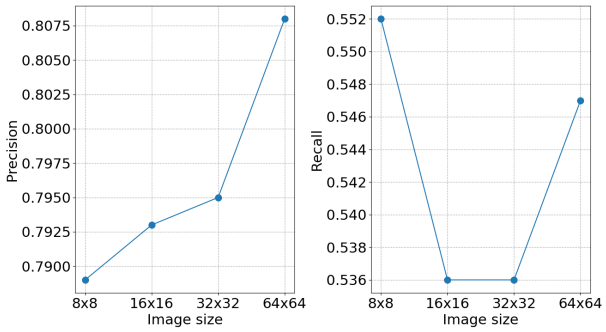

Fig. 9. Precision and recall as function of the image sizes for the k-NN-based classifier.

image sizes increase, the classification times also increase. The ratio between the highest and the lowest classification time for the k-NN is 35.5. This means it is necessary to reduce the image size in order to increase the server performance.

TABLE VI
RESULTS FOR K-NN MODEL

Image size	Precision	Recall	Accuracy	F1 Score
8x8	0.789	0.552	0.896	0.628
16x16	0.793	0.536	0.898	0.640
32x32	0.795	0.536	0.899	0.640
64x64	0.808	0.547	0.902	0.652

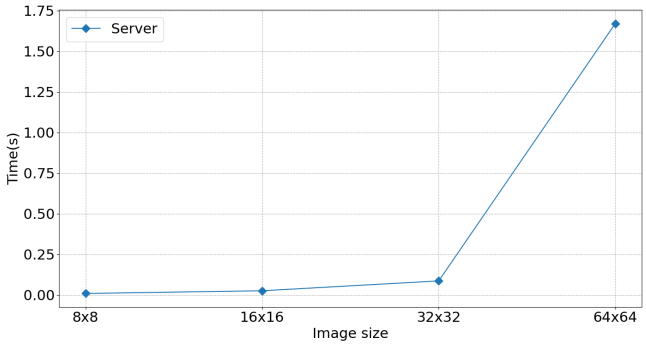

Fig. 10. Average classification times for the k-NN in a server platform.

B. Edge platform

In this subsection, we describe the results for the edge platform (Raspberry Pi Model 4). In this platform, as explained earlier, only the classifications for the images were performed, since the edge platform is not suitable for training the models due to the lack of computational resources. Figure 11 shows the classification times for the CNN model. Similar to the server platform, the classification time increases as the image size increases. Note that the ratio between the longest and the shortest time is only 1.089, which is smaller than in the server platform. However, the edge platform was unable to classify 128x128 images due to the lack of RAM.

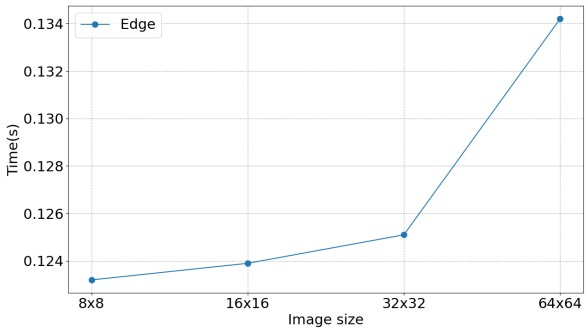

Fig. 11. Average classification times for the CNN in an edge platform.

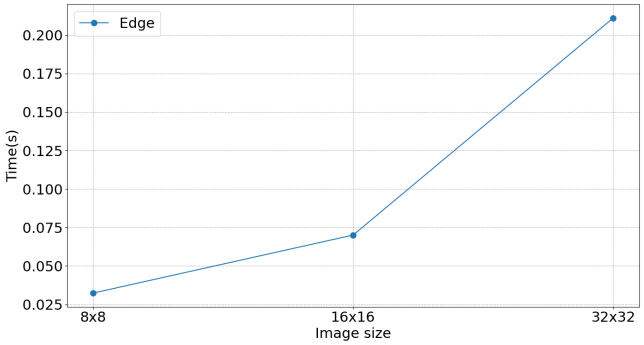

Fig. 13. Average classification times for the k-NN in an edge platform.

Figure 12 shows the results of the classification for the RF-based classifier Note that it only shows the results for $n_{trees} = 100$. The result for $n\_trees = 500$ is not shown because the edge platform was unable to run the RF model due to lack of memory. The result shows that the classification time depends on the size of the image. Nevertheless, the ratio between the longest and the shortest time is only 1.03.

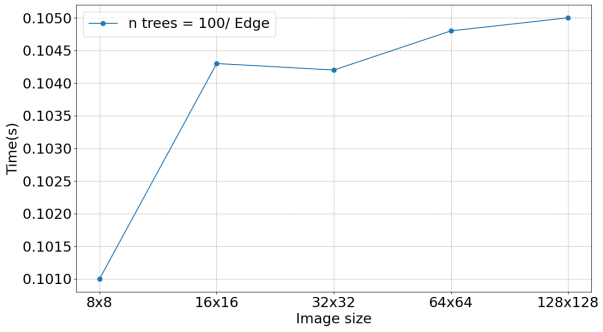

Fig. 12. Average classification times for the RF in an edge platform.

Finally, Figure 13 shows the results of the classification time for the k-NN model. As expected, the results show that the larger the image size, the longer the classification time. However, the edge did not have enough memory (RAM) to run the k-NN model for 64x64 image size. The ratio between the longest and the shortest time is 6.6, which is quite significant.

C. Performance Comparison

In this subsection, we present a comparative summary of the results reported in the previous subsections. Table VII summarizes all classification times for each algorithm used in this paper. The "Ratio" column means the ratio between the classification times on server and edge platforms. The smaller the ratio, the greater the difference between the performance of the platforms. As the ratio approaches one, the performance of the platforms gets close. The first fact to highlight is that there were cases where the edge was unable to classify the images and others that the server was unable to train the model. For these cases, the symbol $-$ was designated. Specifically, the server was unable to train the RF model for 128x128 size image with $n\_trees = 500$ nor the k-NN model for 128x128 size image. The edge, on the other hand, was unable to perform the classification for the CNN with 128x128 size image, the RF with $n\_trees = 500$ and the k-NN with 128x128 and 64x64 size images.

For the CNN-based classifier, it is possible to observe that the variation of the classification time for the edge platform, considering different image sizes, is very small. This is a positive result, as drone-based systems can use larger image sizes (e.g.: 64x64) without losing too much performance. Considering the server platform, however, the difference is more significant. The longest classification time is almost 2.3 times the shortest. Therefore, using 128x128 image size on the server platform requires a small reduction in performance compared to other image sizes. Finally, it is important to notice that, as the image size increases, the classification time of both platforms becomes closer, as can be seen in Table VII.

It is also worth to notice that the variation of classification times for RF model, considering the same number of trees, is small. The only exception is for the 128x128 image size, where the edge is 2.45 times slower than the server. In that way, it is possible to use larger size images without worrying too much about performance. That is, drone-based systems could benefit from this classifier, since they are resource-constrained. For the k-NN-based classifier, the result show a clear positive correlation between the image size and the classification time for both platforms. Additionally, for the k-NN model, the relationship between recall and image size is not well defined. One possible hypothesis is that the k-NN takes into account only a pixel-by-pixel comparison with other images in the dataset, being unable to extract more complex features than the pixel value itself.

VI. Conclusions

Drone-based real-time image analysis using ML algorithms is a promising solution for monitoring deforestation in the vast Brazilian Amazon rainforest. This work analyzed the performance-accuracy trade-offs of three common ML-based classifiers, two different computing platforms, and five different sizes of images. Our experimental study revealed that

TABLE VII
CLASSIFICATION TIMES (IN SECONDS)

Alg.	Img. size	N_trees	Server	Edge	Ratio (server/edge)
CNN	8x8	-	0.07	0.1232	0.568
	16x16	-	0.081	0.1239	0.653
	32x32	-	0.088	0.1251	0.703
	64x64	-	0.113	0.1342	0.842
	128x128	-	0.158	-	
RF	8x8	100	0.097	0.101	0.96
	8x8	500	0.44	-	-
	16x16	100	0.083	0.1043	0.795
	16x16	500	0.486	-	-
	32x32	100	0.094	0.1042	0.902
	32x32	500	0.5	-	-
	64x64	100	0.086	0.1048	0.82
	64x64	500	0.446	-	-
	128x128	100	0.098	0.405	0.241
	128x128	500	-	-	-
k-NN	8x8	-	0.009	0.032	0.281
	16x16	-	0.025	0.070	0.357
	32x32	-	0.086	0.211	0.407
	64x64	-	0.320	-	-
	128x128	-	-	-	-

ML-based image classifiers are a viable solution for monitoring deforestation even when using low quality images and resource-constrained edge devices that can be used in drone-based systems. Specific to this study, the k-NN was the worst-performing algorithm, with very low recall score and medium classification time. The best performing algorithm was the CNN-based, with average classification time of 0.134 seconds with 0.907 recall score. Therefore, limited hardware computers will benefit from the CNN classifier when executing ML classification tasks. To investigate the actual performance of drone-based image classification, in our future work, we plan to consider some environmental factors such as the status of wireless communication link, the battery usage, and the interference from other processes.

ACKNOWLEDGMENT

This research was partially funded by CNPq - Brazil, grant 406263/2018-3. The work was also supported in part by the grant of University of Tsukuba Basic Research Support Program Type S.

REFERENCES

[1] A. Voiland, "Tracking amazon deforestation from above," 2019. [Online]. Available: https://earthobservatory.nasa.gov/images/145988/tracking-amazon-deforestation-from-above

[2] B. São José dos Campos, "Automating land conver change detection: a deep learning based approach to map deforested areas," Ph.D. dissertation, Instituto Nacional de Pesquisas Espaciais, 2020.

[3] "Global forest change 2000–2018," 2019. [Online]. Available: https://earthenginepartners.appspot.com/science-2013-global-forest/download_v1.6.html

[4] INPE, "Monitoramento do desmatamento da floresta amazônica brasileira por satélite," 2020. [Online]. Available: http://www.obt.inpe.br/OBT/assuntos/programas/amazonia/prodes

[5] J. Paneque-Gálvez, M. K. McCall, B. M. Napoletano, S. A. Wich, and L. P. Koh, "Small drones for community-based forest monitoring: An assessment of their feasibility and potential in tropical areas," *Forests*, vol. 5, no. 6, pp. 1481–1507, 2014.

[6] L. Ma, Y. Liu, X. Zhang, Y. Ye, G. Yin, and B. A. Johnson, "Deep learning in remote sensing applications: A meta-analysis and review," *ISPRS journal of photogrammetry and remote sensing*, vol. 152, pp. 166–177, 2019.

[7] H. Mayfield, C. Smith, M. Gallagher, and M. Hockings, "Use of freely available datasets and machine learning methods in predicting deforestation," *Environmental modelling & software*, vol. 87, pp. 17–28, 2017.

[8] L. Zhang, L. Zhang, and B. Du, "Deep learning for remote sensing data: A technical tutorial on the state of the art," *IEEE Geoscience and Remote Sensing Magazine*, vol. 4, no. 2, pp. 22–40, 2016.

[9] X. X. Zhu, D. Tuia, L. Mou, G.-S. Xia, L. Zhang, F. Xu, and F. Fraundorfer, "Deep learning in remote sensing: A comprehensive review and list of resources," *IEEE Geoscience and Remote Sensing Magazine*, vol. 5, no. 4, pp. 8–36, 2017.

[10] H. Hildmann and E. Kovacs, "Using unmanned aerial vehicles (uavs) as mobile sensing platforms (msps) for disaster response, civil security and public safety," *Drones*, vol. 3, no. 3, p. 59, 2019.

[11] B. Specktor, "Amazon deforestation shot up by 278% last month, satellite data show," 2019. [Online]. Available: https://www.livescience.com/66120-amazon-rainforest-deforestation-bolsonaro.html

[12] M. Kanninen, D. Murdiyarso, F. Seymour, A. Angelsen, S. Wunder, and L. German, *Do trees grow on money? The implications of deforestation research for policies to promote REDD*. Cifor, 2007, vol. 4.

[13] U. S. G. Survey, "What is remote sensing and what is it used for?" [Online]. Available: https://www.usgs.gov/faqs/what-remote-sensing-and-what-it-used?qt-news_science_products=0#qt-news_science_products

[14] S. Dhingra and D. Kumar, "A review of remotely sensed satellite image classification." *International Journal of Electrical & Computer Engineering (2088-8708)*, vol. 9, no. 3, 2019.

[15] M. Pal, "Random forest classifier for remote sensing classification," *International journal of remote sensing*, vol. 26, no. 1, pp. 217–222, 2005.

[16] Planet, "Planet: Understanding the amazon from space," 2021. [Online]. Available: https://www.kaggle.com/c/planet-understanding-the-amazon-from-space/data

[17] M. ul Hassan, "Vgg16 – convolutional network for classification and detection." [Online]. Available: https://neurohive.io/en/popular-networks/vgg16/

[18] N. S. Chauhan, "Model evaluation metrics in machine learning," 2020. [Online]. Available: https://www.kdnuggets.com/2020/05/model-evaluation-metrics-machine-learning.html