Revista de Informática Teórica e Aplicada - RITA - ISSN 2175-2745
Vol. 32, Num. 1 (2025) 220-226

RESEARCH ARTICLE

Castor Beans Identification in Sugarcane Plantations Using Active Learning

Identificação de Mamonas em Plantações de Cana-de-açúcar Utilizando Aprendizado Ativo

Juan Carlos Conceição de Lima Sales[1]*, Mário de Araújo Carvalho[1], José Marcato Junior[2], Wesley Nunes Gonçalves[1]

Abstract: Castor beans have multiple applications but can become weeds in several crops. Manual identification is unfeasible in large plantations. Machine learning models can make identification efficient but require a lot of labeled data. Many images of a plantation would be similar and wouldn't improve the model's performance. Active learning (AL) allows labeling only relevant data, often surpassing the performance of models trained on entire datasets. In this work, we detect castor beans, with an AL method that uses self-supervised pretext tasks to separate data for labeling. Models based on pretext tasks presented a decrease in recall relative to the model trained in the whole dataset, which has 93%. We also trained a pseudo-task that separates data with a reasonable concentration of castor bean images. The pseudo-task classifier obtained a 92% recall, being trained in less than 1% of the dataset.
Keywords: Active Learning — Machine Learning — Weed Control — Castor Beans

Resumo: Mamonas tem múltiplas aplicações, mas podem se tornar ervas daninhas em diversas culturas. A identificação manual é inviável em grandes plantações. Modelos de aprendizado de máquina podem tornar a identificação mais eficiente, mas exigem uma grande quantidade de dados rotulados. Muitas imagens de uma plantação seriam semelhantes e não melhorariam o desempenho do modelo. O aprendizado ativo (AL) permite rotular apenas dados relevantes, muitas vezes superando o desempenho de modelos treinados em conjuntos de dados inteiros. Neste trabalho, detectamos mamonas com um método de AL que usa tarefas auxiliares auto-supervisionadas para separar dados para rotulagem. Os modelos baseados em tarefas auxiliares apresentaram uma diminuição na revocação em relação ao modelo treinado no conjunto de dados inteiro, que atingiu 93%. Também treinamos uma pseudo-tarefa que separa dados com uma concentração razoável de imagens de mamonas. O classificador baseado na pseudo-tarefa obteve uma revocação de 92%, sendo treinado em menos de 1% do conjunto de dados.
Palavras-Chave: Aprendizado Ativo — Aprendizado de Máquina — Controle de Daninhas — Mamonas

[1] *Faculty of Computer Science, Universidade Federal de Mato Grosso do Sul (UFMS), Brazil*
[2] *Faculty of Engineering, Architecture and Urbanism and Geography, Universidade Federal de Mato Grosso do Sul (UFMS), Brazil*
*__Corresponding author__: juan_carlos@ufms.br
DOI: http://dx.doi.org/10.22456/2175-2745.143495 • **Received:** 23/10/2024 • **Accepted:** 03/01/2025

1. Introduction

Ricinus communis, commonly known as castor bean, is a tropical plant with applications in various industrial sectors. The main byproduct of castor bean is the oil extracted from its seeds, with some of its applications being the production of biofuel and its use by the pharmaceutical, cosmetic, chemical, food, and textile industries [1][2]. Some byproducts of the castor bean oil extraction process can be used as organic fertilizer [3].

Despite its uses, the castor bean can become a weed in various crops. It can grow aggressively in multiple environments, competing with the crop for essential resources such as water, nutrients, light, and space. Their rapid growth and dense foliage can deprive the crop of sunlight, leading to reduced photosynthesis and subsequently lower crop yields. Castor bean fruits burst when exposed to the sun, launching seeds over long distances. One of the main crops affected by the presence of castor beans is sugarcane [4].

The economic impact of castor bean infestation is considerable and extends beyond crop productivity, also increasing

the cost of agricultural production due to the need for additional control practices, such as herbicide use. Furthermore, castor beans contain toxic compounds in their leaves and seeds that are harmful to humans and various animal species and can be lethal if consumed. [5][6].

Weed detection is essential for maintaining sustainable agricultural production. With weed detection, it is possible to determine the best course of action for controlling them, preventing their establishment and spread. Traditional methods of weed detection, such as visual identification by individuals in the field, are often time-consuming, laborious, and ineffective in large areas [7].

Pure computer vision algorithms may not perform well in detecting castor beans in certain scenarios, such as regions with high plant density, where crops and weeds are very close, or when the plants vary in size. These algorithms may also perform differently under various lighting conditions. The use of machine learning algorithms opens new avenues to address the challenges of this task [8][9]. These algorithms are trained on labeled datasets to recognize the unique characteristics of castor beans, such as leaf shape, growth patterns, and coloration.

However, labeling this data presents new challenges, which diminishes the effectiveness of this method compared to others, even manual detection. The first challenge is related to the area covered by the crop. In a single flight, it is possible to capture tens of thousands of images, making the labeling process time-consuming. The second challenge lies in the computational cost and time required to train a machine learning model on a large dataset. It is necessary to balance the number of images and the desired accuracy, which makes it crucial to select images that yield the biggest improvement in the performance of the model.

The first challenge can be partially solved with *model-assisted labeling*, which involves training machine learning models to label data automatically. Thus, only a subset of the dataset, used to train the labeling model, needs to be labeled. Pham et al. [10] applies model-assisted labeling to create an extensive dataset for jellyfish detection. The second challenge, however, requires a more sophisticated solution, as in a traditional machine learning paradigm, it is impossible to know if a subset of data will improve the model's performance without first labeling and feeding it to the model, thus creating a circular dependency. This problem can be circumvented through the use of *active learning* (AL).

Active learning is a branch of machine learning in which the algorithm iteratively selects the most informative samples from the dataset for labeling [11]. By employing AL, it is possible to achieve similar or even superior performance to a model trained on the entire dataset, using a substantially smaller subset of the data, thus reducing the time and computational resources required for training. Yi et al. [12] proposes a self-supervised approach, using pretext tasks to determine the "degree of difficulty" of an image and consequently its impact on the model's performance.

Despite the benefits of using AL, the literature on its application to weed identification is scarce. Thus, the present work helps to fill this gap and applies AL to castor bean detection in sugarcane plantations.

In this work, we use the method proposed by Yi et al. [12]. The method was chosen for the results obtained by the authors in the original work, which outperforms other AL methods available in the literature. We tested four pretext tasks for data sampling and a pseudo-task that performs an "ideal" sampling. The results of the classifiers trained with AL were compared to a baseline classifier trained without AL on the entire dataset.

The tested pretext tasks did not adapt to the dataset, and their respective classifiers had a lower recall than the baseline classifier, which had a recall of 93%. The decrease in recall means the classifiers based on pretext tasks failed to identify a larger number of castor beans in the dataset. In a real scenario, the weed infestation could start over from the non-detected plants. The pseudo-task, however, had a recall of 92%, signaling the importance of a good concentration of castor beans in the data sampled by the pretext tasks. The classifier based on the pseudo-task achieved a performance equivalent to the baseline classifier while being trained on less than 1% of the dataset's images.

2. Literature Review

2.1 Weed detection

Automatic weed detection consists of two main tasks: extracting features that describe a plant and determining whether the identified object is a weed or not. Effectively, the methods differ in how these tasks are performed.

The features used in plant identification can be visual, such as color and shape, as well as spatial, like the plant's height and density.

In the plant classification stage, methods vary in complexity and efficiency. Computer vision algorithms and traditional machine learning algorithms laid the foundation for automatic object detection. However, advances in hardware, especially GPUs, over the last decade have enabled the training of increasingly larger and more accurate deep learning models. These models currently represent the state of the art in automatic object detection, although traditional algorithms are still widely used.

The method employed in the work by Molina-Villa et al. [13] involves using RGB image filtering techniques for weed detection among the crop. These operations reduce the visual features of the images to a subset that comprises more significant characteristics. Initially, the algorithm filters the objects in the image by color and shape, removing any objects that are not part of the vegetation. The resulting image is then converted into a binary image, where white pixels correspond to vegetation. Finally, the objects are classified as part of the crop or weeds according to their pixel area.

Similarly, Perez et al. [14] use color and shape analysis of objects for weed detection from RGB images. The proposed

method uses image histograms and a refinement process based on binary images to segment the plants. The result of the segmentation is fed into a shape analysis process, which uses a Bayes rule and *K-Nearest Neighbors* (KNN) to identify the weeds.

It is also possible to employ multispectral sensors in the data capture process. These sensors capture images at different wavelengths of the electromagnetic spectrum, constituting images with multiple color channels, instead of the traditional RGB channels. This allows the estimation of a unique color signature for a plant species.

In their research, Lin et al. [15] used high-resolution multispectral images to analyze the characteristics of corn plants and different weed species. After data capture and visual feature segmentation, the authors use decision trees to identify the weeds.

The spatial characteristics of plants allow them to be distinguished by how they occupy space. Andújar et al. [16] use ultrasonic distance sensors for weed detection based on the height difference between the plants. According to the authors, the system performs well in detecting weeds during the early growth stage of the crop. After this stage, the weeds may be occluded by the crop. Shahbazi et al. [17] propose the application of LiDAR for detection. The process also uses the height difference between weeds and crops for detection.

Deep learning techniques, such as convolutional neural networks (CNNs), have proven to be a viable solution for weed detection. CNNs tend to be less sensitive to factors such as lighting and occlusion, making them the best alternative in many cases.

Highly specialized models can be built using deep learning, allowing the architecture to be adapted to the specific needs of the problem domain. Punithavathi et al. [18] propose a specialized method that uses a Faster RCNN for plant detection and an Extreme Learning Machine (ELM) to distinguish weed and crop.

Deep learning algorithms have high computational and time costs for training, but they are efficient during the inference stage, which allows their integration into embedded systems for real-time weed control. The work developed by Tummapudi et al. [19] integrates plant classification with a robotic arm that acts in removing the detected weeds.

2.2 Active Learning

Active learning enables informed sampling from unlabeled datasets, selecting the most significant data for the training process. This allows applications with large datasets and/or limited availability of labelers to achieve the highest performance with the fewest data points, thereby minimizing labeling costs. Zhao et al. [20] applied AL to the training of *Neural Machine Translation* (NMT) algorithms, which directly translate text from one language to another based on common words in sentences. The algorithm requires a large amount of textual data, and consequently, a significant amount of human effort. By applying AL to the model training, the authors surpassed the performance of the baseline model, with only 20% of the labeling effort required for their dataset.

Moreover, AL techniques have been successfully applied in various domains, such as training classification models for signal modulation and classification [21], text processing in the healthcare domain to determine patient medical conditions [22], and the creation of synthetic data to test autonomous driving algorithms [23].

3. Materials and Methods

3.1 Dataset

The dataset used in this work consists of aerial images of sugarcane plantations with castor bean plants. The data was collected from various plantations across Brazil using different drone models, ensuring a wide coverage and diversity of planting conditions. Orthoimages of the plantations were generated with a GSD (Ground Sampling Distance) of approximately 3 cm, providing a high level of detail and precision in the captured images. This high resolution is crucial for accurately identifying castor bean plants and other relevant characteristics of the plantations.

Initially, the orthoimages were captured, and the castor bean plants were manually labeled by a specialist, who drew a polygon with their approximate shape around them. For the classification task, we divided the orthoimages into patches of 32x32 pixels. These dimensions are the same as those used in the original study. Then, for each image in the dataset, we checked if the corresponding annotation image contained castor bean plant pixels, and if so, we associated that class with the image. Fig. 1 shows a pair of an RGB image and an annotation image.

Given that the images are crops from a larger orthoimage, only a small portion of a castor bean plant may be contained in the image, possibly causing false negatives. To prevent such an image from being labeled as castor bean, we assigned the label only to images in which the object occupies more than 10% of the pixels. Images with less than 10% and more than 0% of castor bean pixels were discarded.

Figure 1. Annotation image (left) and corresponding RGB image (right)

The dataset resulting from the division process mentioned above contains a total of 1,633,744 images, with 48,156 images labeled as castor bean plants and 1,585,588 labeled as

background. The data was divided into stratified splits for training and validation, with 90% of the data from each class allocated for training and 10% for validation.

3.2 Chosen Method

The method chosen for this work was proposed by Yi et al. [12]. The study aimed to maximize the performance of classification and semantic segmentation tasks by applying AL. These tasks are referred to as the main tasks and are trained in cycles with AL. In each cycle, the main task samples the most challenging images from the unlabeled dataset without replacement and sends them to the *oracle* for labeling. The oracle is an abstraction of an image labeler, which could be an automatic labeling model or a human labeler. The images are added to the labeled dataset and used to train the main task. At the end of the training, the metrics for the main task are calculated, and the cycle is repeated. The algorithm's workflow is described in Fig. 2.

To determine which images are challenging and should be selected, the authors use pretext tasks, which are self-supervised models trained on the unlabeled dataset before the AL cycles. In the testing phase of the pretext tasks, the entire unlabeled dataset is used again, when the task calculates the loss for each image, indicating its difficulty level.

The pretext tasks chosen by the authors include rotation estimation of an image, solving a jigsaw puzzle with image pieces, colorizing black-and-white images, and a SimSiam network. These pretext tasks are not necessarily related to the main task in scope. However, the authors estimate a high correlation between the losses of the pretext tasks and the main task across different datasets, demonstrating that the performance of the pretext tasks is related to the main task and that the difficulty of an image indicates its potential information gain.

The images are sorted in descending order based on their loss and divided into batches. Sorting the data helps select challenging samples, while separating it into batches distributes the data from different classes, improving the representativeness of the sampling.

Thus, in each AL cycle, the main task selects the top images from a batch for training, and the batch creation process ensures that the selected images are the most challenging according to the pretext task. The maximum number of training cycles corresponds to the number of batches, and the main task can be trained until all cycles are completed or until the training curves stabilize.

3.3 Experiments

We conducted two experiments throughout this work: pretext task classifiers evaluation and performance vs data amount evaluation. All code was written in Python, using the PyTorch library for model training. All image classification models used in the experiments have ResNet18 as the backbone.

The first experiment consists of training image classifiers with AL, using data selected from different pretext tasks. We tested four configurations: random sampling, rotation, colorization, and pseudo-task.

To establish a baseline performance, we trained an image classification network on the entire dataset. We trained the classifier for 200 epochs.

For all experiment configurations, the pretext task generates batches of 500 images for training the main task. We started training with 250 images sampled from the first batch and included 250 images from subsequent batches to the labeled dataset with each training cycle. One cycle corresponds to training the main task for 200 epochs. Due to time constraints and the stabilization of the curves, the main task training was stopped before the final batch in all AL experiments.

In the first configuration, the batches were created with data sampled randomly from the original dataset. The initial images and the increment at each cycle were randomly sampled from the batches. This configuration is intended to establish a baseline performance for the main task when trained with pretext task data.

For the configurations that use pretext tasks, the main task uniformly samples data from the first batch in the first cycle. In subsequent cycles, the first 250 images from the batch are sampled and used for training the main task. This is the same sampling method used in the original work.

The second configuration uses a rotation prediction network as the pretext task. This task involves applying a random rotation to an image. The possible rotations are 0 degrees (no rotation), 90, 180, and 270 degrees. The image and the rotation label are then fed into a classification network, which attempts to predict the image's rotation.

The third configuration uses a self-supervised image colorization task. The input image is converted to grayscale and passed to the colorization network along with the original image as the label. The network estimates the color of each pixel and compares the resulting image with the label using Mean Squared Error (MSE).

Since the data distribution is extremely unbalanced in favor of the background images, if we pick the images at random when creating the batches, the castor beans will likely be underrepresented. With this in mind, the fourth configuration uses a pseudo-task that simulates a pretext task that excels at identifying castor bean images, thus creating batches with a higher concentration of the class' images. We achieve this by manually creating batches with 100 castor bean images and 400 background images. In the original dataset, the castor bean images correspond to 3% of the dataset, which is the expected concentration of the class in the randomly sampled batches. The pseudo-task increases this concentration to 20%.

For the second experiment, we used the same training configurations, but instead of evaluating performance at the end of the last training cycle, we evaluated performance at each cycle. This gives us the relationship between performance and the amount of data used in training.

For comparison, we also trained classifiers using DalMax [24], an AL framework that implements multiple AL sam-

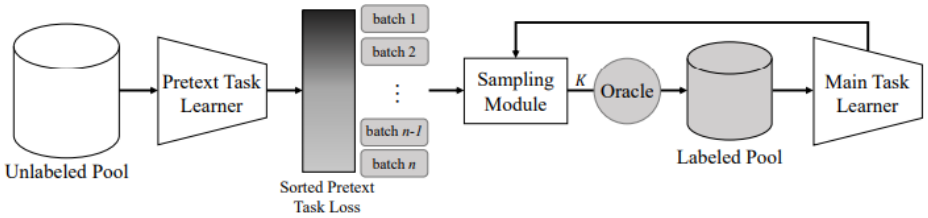

Figure 2. Method framework. Extracted from the original paper

pling strategies. The chosen strategies were Least Confidence Sampling [25] and Entropy Sampling [11]. Least Confidence Sampling selects the samples in which the model is least confident, while Entropy Sampling selects the samples with the highest prediction entropy. The training configuration is similar to the one for PT4AL: we trained the models in cycles of 200 epochs and increments of 250 images to the labeled data pool each cycle.

4. Results and Discussion

The test metrics for all classifiers are presented in Table 1.

Equations (1) and (2) describe the metrics discussed in this work. TP represents the number of correctly classified castor bean images or true positives; TN represents the number of correctly classified background images or true negatives; FP represents the number of background images classified as castor beans or false positives; and FN represents the number of castor bean images classified as background or false negatives.

Given the class imbalance, if the model always predicts the class of the images as background, the accuracy (1), which defines the rate of correct predictions relative to all predictions made, is approximately 97%. Therefore, a metric combining both predictions for castor beans and background does not provide relevant information about castor beans. For this reason, the table contains metrics only for the castor bean class.

The most relevant metric to evaluate the results of the experiments is the recall (2), as it indicates how many of the castor beans in the dataset the main task was able to identify. Since castor beans can disperse seeds over long distances, leaving even a single plant can lead to a new infestation, making it crucial to remove as many beans as possible from the plantation.

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

$$\text{Recall} = \frac{\text{TP}}{\text{TP}+\text{FN}} \tag{2}$$

Ideally, the pretext task used in creating batches for training with AL should perform better on images from one of the classes. The rotation task excels in images with lines and shapes that appear displaced when the image is rotated. The tested rotations can occur naturally in the dataset used in this work, as plants are not oriented objects. This results in a similar loss for all images across the entire dataset, so batches are effectively created with random images. This problem can be observed in Table 2, where both the random sampling and rotation tasks presented a similar class distribution. It is observable in Table 1 that the random sampling and rotation classifiers have close metrics, with 79% and 70% recall, respectively.

Table 1. Experiment 1: Recall at the end of training

Configuration	Recall
Baseline	0.93
Random Sampling	0.79
Rotation	0.70
Colorization	0.48
Pseudo-Task	0.92
Least Confidence	0.78
Entropy	0.84

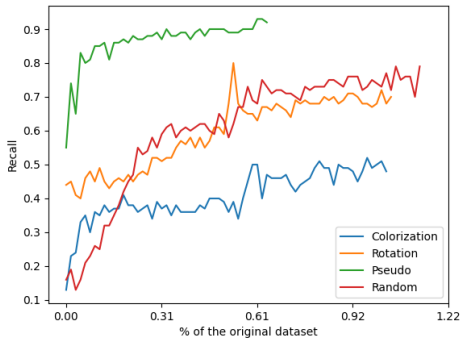

Figure 3. Experiment 2: Recall vs amount of data

Table 2. Class distribution per 250 images sample

	Castor beans	Background
Random Sampling	7.68 ± 2.64	242.32 ± 2.64
Rotation	8.49 ± 3.55	241.51 ± 3.55
Colorization	17.34 ± 7.01	232.66 ± 7.01
Pseudo-Task	53.05 ± 6.56	196.95 ± 6.56

Colorization had the worst performance, with 48% recall, despite being the non-artificial task with the highest concentration of castor bean images, at almost 7% of the batch. This indicates that the features of the selected images did not help the model learn to detect castor beans. Upon analyzing the

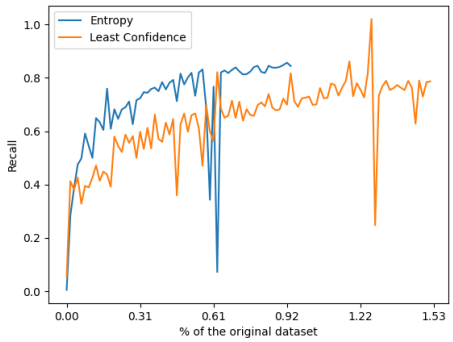

Figure 4. Experiment 2: Recall vs amount of data

sampled images, we observed that this task sampled more images with vegetation, not only more images with castor beans.

The classifiers based on Least Confidence Sampling and Entropy Sampling performed at, respectively, 78% and 84% recall. This puts the performance of the classifiers close to random sampling.

The classifier based on the pseudo-task had a performance equivalent to the baseline, at 92% recall.

Figures 3 and 4 show the relationship between recall and the amount of the original dataset used in training. The images are separated for visual clarity. The baseline classifier curve is not presented in the figures as the whole dataset was used to train it. For all pretext tasks, the curves stabilized before the 100th batch, corresponding to approximately 1.5% of the original dataset. This shows us that the information gain above this point is minimal compared to the training cost.

As expected, the colorization classifier showed the worst performance among the others, being trained up until cycle 68. As observed previously, the random sampling and rotation classifiers presented similar performances, which is evidenced by the curves of the two classifiers. The classifiers were trained up until cycles 74 and 69, respectively. The Least Confidence Sampling classifier was trained for 100 cycles, while the Entropy Sampling classifier was trained for 62 cycles, where the curve stabilized. The pseudo-task classifier outperformed the other pretext task classifiers in recall while requiring the least amount of data. We stopped the training in cycle 42, although it is possible to observe that the curve stabilized sooner. At cycle 42, the labeled dataset is made of 10500 images, corresponding to 0.6% of the original dataset.

5. Conclusion

Given the nature of the data set and the small amount of information contained in the images, the classifiers achieved a relatively high performance.

The extreme class imbalance makes the process of selecting pretext tasks difficult as these tasks need to force an increase in the concentration of the underrepresented class in the batches to ensure a significant number of examples.

The colorization classifier showed the worst performance among the baseline and the other classifiers. It successfully distinguished between the plants and the background but underperformed at distinguishing the castor beans from the crop. This indicates that color features extracted from RGB images alone may not be suitable for sampling the data.

The rotation task showed similar scores for images with and without castor beans. This caused the random sampling and rotation classifiers to perform similarly, with lower performances than the baseline classifier. The lower recall is not ideal in a real-world scenario, as these classifiers identified fewer castor beans.

The pseudo-task classifier achieved the highest recall among the pretext task classifiers. It performed similarly to the baseline, with a fraction of the required data. The most promising extension of this work is to evaluate new pretext tasks that generate batches with a higher concentration of castor bean images.

We observed that the image dimensions used in the study severely limit the amount of information available for training the models. Additionally, the use of a shallow ResNet18 backbone might negatively impact classification performance. By addressing these limitations, we expect to see improvements in both the classifiers and pretext tasks.

Moreover, this work focuses exclusively on castor bean detection, though other varieties of weeds may also be present in the crop. By increasing the number of classes, we expect a more general and robust model.

Acknowledgements

This research was funded by CNPq (p: 305814/2023-0; 403213/2023–1433783/2018- 4, 304052/2019-1, and 303559/2019-5), FUNDECT (p: 71/001.902/2022), and CAPES Print. The authors acknowledge the support of the UFMS (Federal University of Mato Grosso do Sul) and CAPES (Finance Code 001).

Author contributions

Juan Carlos Conceição de Lima Sales: Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology, Investigation, Conceptualization.
Mário de Araújo Carvalho: Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology, Investigation, Conceptualization.
José Marcato Junior: Writing – review & editing, Writing – original draft, Supervision.
Wesley Nunes Goncalves: Writing – review & editing, Writing – original draft, Supervision, Investigation, Conceptualization

References

[1] IN: CONGRESSO BRASILEIRO DE MAMONA, 3., 2008, SALVADOR. ENERGIA E .... *Utilização de*

coprodutos da mamona na alimentação animal. Disponível em: ⟨https://www.alice.cnptia.embrapa.br/alice/handle/doc/939880⟩.

[2] SAADAOUI, E. et al. Castor bean (ricinus communis l.) diversity, seed oil and uses. *Oilseed Crops: Yield and Adaptations under Environmental Stress*, Wiley Online Library, p. 19–33, 2017. Disponível em: ⟨https://doi.org/10.1002/9781119048800.ch2⟩.

[3] LIMA, R. et al. Casca e torta de mamona avaliados em vasos como fertilizantes orgânicos. Revista Caatinga, v. 21, n. 5, p. 102-106, supl., dez., 2008., 2008. Disponível em: ⟨https://www.alice.cnptia.embrapa.br/alice/handle/doc/277960⟩.

[4] ZERA, F. S. et al. Tolerância de mamona (ricinus communis) a herbicidas utilizados na cultura da cana-de-açúcar. *Nucleus*, Fundação Educacional Ituverava, v. 8, n. 1, p. 1–10, 2011. Disponível em: ⟨https://doi.org/10.3738/1982.2278.469⟩.

[5] FONSECA, N. B. da S.; SOTO-BLANCO, B. Toxicidade da ricina presente nas sementes de mamona. *Semina: Ciências Agrárias*, Universidade Estadual de Londrina, v. 35, n. 3, p. 1415–1424, 2014. Disponível em: ⟨https://doi.org/10.5433/1679-0359.2014v35n3p1415⟩.

[6] OLIVEIRA, R. C. de; FELIZARI, J. Toxicidade de extratos de ricinus communis e euphorbia milii para caramujo aquático e tilápia. *Revista Thêma et Scientia*, v. 1, n. 2, 2011. Disponível em: ⟨https://ojsrevistas.fag.edu.br/index.php/RTES/article/view/467⟩.

[7] COSTA, N. V. et al. Métodos de controle de plantas daninhas em sistemas orgânicos: breve revisão. *Revista Brasileira de Herbicidas*, v. 17, n. 1, p. 25–44, 2018. Disponível em: ⟨https://doi.org/10.7824/rbh.v17i1.522⟩.

[8] FERREIRA, A. d. S. Redes neurais convolucionais profundas na detecçao de plantas daninhas em lavoura de soja. 2017. Disponível em: ⟨https://repositorio.ufms.br/handle/123456789/3101⟩.

[9] ODA, Y. S.; NETO, J. C. d. C. Detecção de plantas daninhas e plantas de soja utilizando imagens multiespectrais e visão computacional. 2023. Disponível em: ⟨https://doi.org/10.11606/T.76.2023.tde-22022024-084207⟩.

[10] PHAM, T.-N. et al. Improved yolov5 based deep learning system for jellyfish detection. *IEEE Access*, IEEE, 2024. Disponível em: ⟨https://doi.org/10.1109/ACCESS.2024.3405452⟩.

[11] SETTLES, B. Active learning literature survey. University of Wisconsin-Madison Department of Computer Sciences, 2009. Disponível em: ⟨http://digital.library.wisc.edu/1793/60660⟩.

[12] YI, J. S. K. et al. Using self-supervised pretext tasks for active learning. *CoRR*, abs/2201.07459, 2022. Disponível em: ⟨https://arxiv.org/abs/2201.07459⟩.

[13] MOLINA-VILLA, M. A.; SOLAQUE-GUZMÁN, L. E. et al. Machine vision system for weed detection using image filtering in vegetables crops. *Revista Facultad de Ingeniería Universidad de Antioquia*, Universidad de Antioquia, n. 80, p. 124–130, 2016. Disponível em: ⟨https://doi.org/10.17533/udea.redin.n80a13⟩.

[14] PEREZ, A. et al. Colour and shape analysis techniques for weed detection in cereal fields. *Computers and electronics in agriculture*, Elsevier, v. 25, n. 3, p. 197–212, 2000. Disponível em: ⟨https://doi.org/10.1016/S0168-1699(99)00068-X⟩.

[15] LIN, F. et al. Detection of corn and weed species by the combination of spectral, shape and textural features. *Sustainability*, MDPI, v. 9, n. 8, p. 1335, 2017. Disponível em: ⟨https://doi.org/10.3390/su9081335⟩.

[16] ANDÚJAR, D.; WEIS, M.; GERHARDS, R. An ultrasonic system for weed detection in cereal crops. *Sensors*, Molecular Diversity Preservation International (MDPI), v. 12, n. 12, p. 17343–17357, 2012. Disponível em: ⟨https://doi.org/10.3390/s121217343⟩.

[17] SHAHBAZI, N. et al. Assessing the capability and potential of lidar for weed detection. *Sensors*, MDPI, v. 21, n. 7, p. 2328, 2021. Disponível em: ⟨https://doi.org/10.3390/s21072328⟩.

[18] PUNITHAVATHI, R. et al. Computer vision and deep learning-enabled weed detection model for precision agriculture. *Comput. Syst. Sci. Eng.*, v. 44, n. 3, p. 2759–2774, 2023. Disponível em: ⟨https://doi.org/10.32604/csse.2023.027647⟩.

[19] IEEE. *Deep Learning Based Weed Detection and Elimination in Agriculture*. 147–151 p. Disponível em: ⟨https://doi.org/10.1109/ICICT57646.2023.10134186⟩.

[20] *Active learning approaches to enhancing neural machine translation*. 1796–1806 p. Disponível em: ⟨https://doi.org/10.18653/v1/2020.findings-emnlp.162⟩.

[21] IEEE. *Modulation and signal class labelling with active learning and classification using machine learning*. 1–5 p. Disponível em: ⟨https://doi.org/10.1109/CONECCT55679.2022.9865826⟩.

[22] FERREIRA, M. D. et al. *Active learning for medical code assignment*. 2021. Disponível em: ⟨https://arxiv.org/abs/2104.05741⟩.

[23] IEEE. *Active machine learning to test autonomous driving*. 286–286 p. Disponível em: ⟨https://doi.org/10.1109/ICSTW52544.2021.00055⟩.

[24] CARVALHO, M. de A. *DalMax: Framework for Deep Active Learning Approaches*. 2024. ⟨https://github.com/MarioCarvalhoBr/dalmax-framework-deep-active-learning-python⟩. Available on GitHub.

[25] LEWIS, D. D.; GALE, W. A. A sequential algorithm for training text classifiers. *CoRR*, abs/cmp-lg/9407020, 1994. Disponível em: ⟨http://arxiv.org/abs/cmp-lg/9407020⟩.