iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 10 **Issue:** XII **Month of publication:** December 2022

DOI: https://doi.org/10.22214/ijraset.2022.48186

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 10 Issue XII Dec 2022- Available at www.ijraset.com

Ayurvedic Plant Species Detection System Using Image Processing

Anupriya Varshney[1], Kunal Dongare[2], Muskan Shrivastava[3], Janhavi Naik[4], Dr. Sanjesh Pawale[5]
Department of Computer Engineering, Vishwakarma University, Pune

Abstract: ***Plant species identification via plant leaves based on shape, color, and texture features using digital image processing techniques for classifying plant species using various machine learning algorithms. This research proposes a machine learning algorithm-based system to identify multiple plants of the same genus having different species in the form of an image using their leaf feature. The image processing technique is regarded as the primary method for classifying different plants based on various characteristics or specific portions or regions of the plant leaves, which will then be identified via image processing.***
Keywords: Image Processing, TensorFlow, Training, Testing, SSD Mobilenet, Object Detection

I. INTRODUCTION

Ayurveda is a traditional medical system that originated thousands of years back in India. Because it is entirely natural and free from side effects, it is still used extensively today. It has remained well-liked throughout time due to its capacity to treat chronic illnesses. In Ayurveda, the formulation of remedies generally uses parts like the leaf, flower, root, bark, and fruit. Expert doctors or taxonomists now identify plants manually, which can lead to human error in many situations. Previous research done in this area has successfully identified different plant genera. But there is still less accuracy in identifying different species of the same plant as they share quite similar features. Yet they have different medicinal properties which are useful in preparing ayurvedic medicines. To avoid the human errors caused in identifying plant species, this research provides an automated system for the identification of medicinal plants that takes a plant leaf as an input and outputs a classification of plants. This study describes a method for identifying ayurvedic medicinal herbs using leaf samples and image processing. More than 80% of ayurvedic plants come from forests and wastelands. There exists no predefined database of Ayurvedic plant leaves. Classifying medicinal plants can be done using both internal and exterior properties. Exterior characteristics like color, form, texture and edge histogram serve as their identification markers. As information technology is progressing rapidly, techniques like image processing, pattern recognition, and so on are used for the identification of plants on basis of leaf shape description and venation which is the key concept in the identification process. This paper suggests an automated system to identify the taxonomy of Ayurvedic plants by extracting features from their leaf images, using TensorFlow to obtain the leaf factor, and then comparing the obtained leaf factor with the trained leaf factor in the database to match the input leaf image and classify the ayurvedic plant species.

II. LITERATURE SURVEY

The number of plant species is extremely huge, with about 450,000 plant species all over the world [1]. Currently, machine learning, a subfield of artificial intelligence (AI), is a popular and widely used technique, that has been applied in various domains including biology, medicine, computer vision, speech recognition, and others [2]. Deep learning is a modern AI approach, which contributes a robust framework for supervised learning [6]. It is able to map an input vector rapidly and efficiently to an output vector even in a large dataset [7]. Image enhancement is a process that is used to emphasize the features of an image [8]. The texture is one of the important features of the plant identification system, which can be used to characterize the leaves based on the surface structure of the leaves. It is a non-consistent spatial distribution pattern of different image intensities, which concentrates mainly on every single pixel of an image. Lee et al. [15] proposed a CNN technique to identify 44 plant species acquired from the Royal Botanic Gardens of Kew, England. The extracted features were then classified with a Multilayer Perceptron (MLP) and a SVM. Two different datasets were used, namely, whole image (D1) and leaf patches (D2). Both datasets achieved an accuracy of more than 97%. Furthermore, researchers had combined both local and global features together in [18] and achieved more than 91% accuracy. Furthermore, Sladojevic et al. [19] employed CNN for plant diseases recognition. Various researches and studies have been conducted in this area but

©IJRASET: All Rights are Reserved

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 10 Issue XII Dec 2022- Available at www.ijraset.com

III. SYSTEM PROPOSED ARCHITECTURE

The proposed system is vision based, uses image processing techniques and inputs from a computer or laptop camera. Our system is specific towards detecting different species of the same plant. The input frame would be captured from the digital camera and the system is broken down into four stages, Image Acquisition, Pre-processing, Feature Extraction, and Leaf Factor calculation. Image acquisition in image processing and machine vision is the process of obtaining an image from a source, which is typically a hardware system like a camera, sensor, etc. Since the system cannot perform any meaningful processing without an image, it is the first and most crucial stage in the workflow sequence. The second step is Image Processing. It is a kind of signal processing where an image serves as the input, and either an image or properties or features related to that picture serves as the output. In this case, it is the structure of leaves. Now we know that the leaves of plants and trees are of different sizes, shapes and colors like mango tree leaves are simple leaves and neem tree leaves are compound leaves. We need to understand this even to distinguish between leaves of different species of the same plant. Feature Extraction and Description which identifies the features of the given leaf/leaves. Feature extraction is one of the most significant steps in image processing and thus features describing various aspects of the plant leaf must be considered before classification. And at the last step, we classify the leaves according to the structure and features identified and display the final name and the properties associated with it. To train the machine learning model, a set of leaf images of medicinal plants was collected from the botanical garden. To improve the efficiency of the plant identification system, machine learning techniques were used over human visual perception as it is more effective. A dataset of high-quality images was created containing images of plants of the Ocimum (Basil) genus. It consisted of 458 images of 3 Basil species to train and test the Machine Learning model initially. This Data Acquisition process was done using One Plus Android phone's high-quality rear camera.

TABLE I. Plant Species COnsidered for Classification

Ocimum Genus		
Scientific Name	***Common Name***	***Image***
Ocimum Gratissimum	Clove Basil	
Ocimum Tenuiflorum	Holy Basil	
Ocimum Africanum	Lemon Basil	

Table 1 consists of 3 different species of the Ocimum (Basil) genus that are used for classification in the proposed model.

©IJRASET: All Rights are Reserved

The images in the dataset were taken considering various environmental conditions like illuminated background, dark background, shadowed objects, overlapped objects, etc.

For machine learning, the datasets are divided into two subsets. The first subset is known as the training data - it is a portion of the actual dataset that is fed into the machine learning model to discover and learn patterns. Once the machine learning model is built (with the training data), it needs unseen data to test the model. This data is called testing data, it can be used to evaluate the performance and progress of the algorithms' training and adjust or optimize it for improved results. Fig. 1. represents the data flow diagram of the object detection process. It indicates the training and testing phase of the built system.

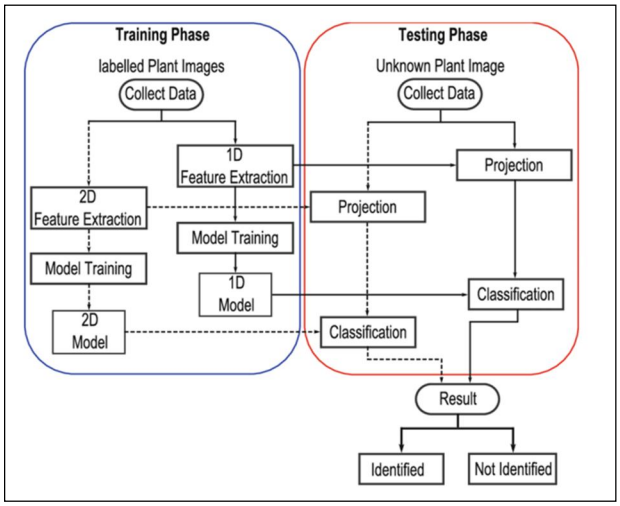


Fig. 1. Data Flow Diagram

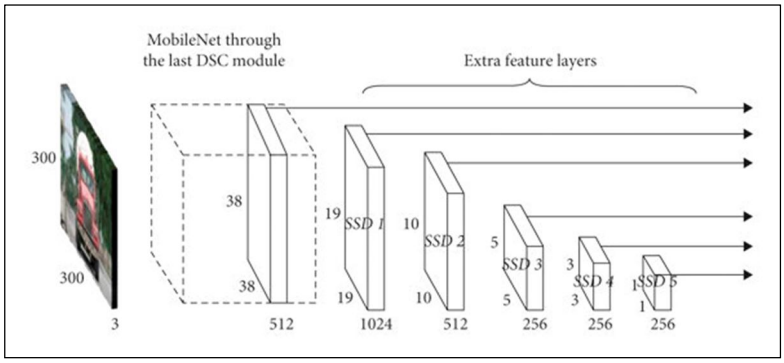


Fig. 2. SSD-MobileNet Model Architecture

IV. OBJECT DETECTION

A. SSD-MobileNet Architecture

A single convolutional network makes up the SSD architecture, which learns to predict bounding box positions and classify these places in a single run. Hence, end-to-end training could be obtained using SSD. The SSD network consists of a base architecture (MobileNet in this case) followed by several convolution layers as shown in Fig. 2.

In contrast to regional proposal network (RPN) based systems like the R-CNN series, which require two shots—one for generating region proposals and the other for identifying the item of each proposal—using SSD, we can detect many objects within an image with just one shot. Therefore, SSD is much faster than two-shot RPN-based methods.

B. Tensorflow Object Detection API

The platform for building a deep learning network that solves object detection problems is the TensorFlow object detection API. In their framework, known as Model Zoo, there are pre-trained models already. This comprises a set of models that have already been trained using the COCO, KITTI, and Open Images datasets. They are also useful for initializing the models when training on the novel dataset.

V. METHODOLOGY

A. Hardware and Software Requirements

1) *Hardware Requirements:* PC/Laptop with specifications such as 8GB RAM, i3 processor, and Mobile Camera with minimum 13 Megapixel specification.
2) *Software Requirements:* TensorFlow, Python, Anaconda3, Open CV, Operating System (Windows 8 or higher), CUDA

B. Steps

1) *Install TensorFlow or TensorFlow-CPU:* Initialize with "pip install upgrade tensorflow "(For CPU)or "pip install -upgrade tensorflow-gpu" command. These commands will download TensorFlow-GPU v1.5. Install cuDNN version 7 and CUDA 9.0, because TensorFlow-GPU v1.5 supports them. As future versions of TensorFlow are released, we will likely need to continue updating the CUDA and cuDNN versions to the latest supported version. Be sure to install Anaconda with Python 3.10 or higher.
2) *Set Up TensorFlow Directory and Anaconda Virtual Environment:* The specific directory structure mentioned in the TensorFlow Object Detection API's GitHub repository must be used. It also needs a number of extra Python packages, particular modifications to the PATH and PYTHONPATH variables, and a number of other things To set up everything for running or training an object detection model, a few extra setup commands are required.

a) *Download TensorFlow Object Detection API repository from GitHub:* Directly in C: drive, create a folder with the name "tensorflow1". The entire TensorFlow object detection framework, along with your training photos, training data, trained classifier, configuration files, and anything else required for the object detection classifier, will be located in this working directory.

b) *Download the ssd_mobilenet_v1_coco model from TensorFlow's model zoo:* In its model zoo, TensorFlow offers a number of object detection models (pre-trained classifiers with particular neural network architectures). While some models, like the Faster-RCNN model, provide slower detection but with greater accuracy, others, like the SSD-MobileNet model, offer an architecture that enables faster detection but with less accuracy. Since laptops with minimum specifications have lower setups that allow models to be trained using the CPU without the use of a graphics processing unit, we initially started with the SSD-MobileNet-V1 model for this research (GPU). A faster-RCNN-Inception-V2 model can be utilised instead if the laptop has a good NVIDIA graphics card and a higher setup, and the detection performs significantly better but at a notably slower speed. The SSD mobilenet v1 coco model was selected.

c) *Set up new Anaconda virtual environment*: In the command terminal, create a new virtual environment called “tensorflow1” by issuing the following command:

- C:> conda create -n tensorflow1 pip python=3.5
  Then, activate the environment by issuing:
- C:> activate tensorflow1
  Install tensorflow in this environment by issuing:
- (tensorflow1) C:> pip install --ignore-installed --upgrade tensorflow
  Install the other necessary packages by issuing the following commands:
- (tensorflow1) C:> conda install -c anaconda protobuf
- (tensorflow1) C:> pip install pillow
- (tensorflow1) C:> pip install lxml
- (tensorflow1) C:> pip install Cython
- (tensorflow1) C:> pip install jupyter
- (tensorflow1) C:> pip install matplotlib
- (tensorflow1) C:> pip install pandas
- (tensorflow1) C:> pip install opencv-python

d) *Compile Protobufs and run setup.py:* Compile the Protobuf files, which are used by TensorFlow to configure model and training parameter. Construct the TensorFlow Protobuf files that TensorFlow uses to set model and training parameters. Unfortunately, Windows cannot be used with the short protoc compilation command that is listed on the installation page for TensorFlow's Object Detection API. The command must call every .proto file in the directory "object detection" individually.

e) *Test TensorFlow setup to verify it works:* The TensorFlow Object Detection API is now all set up to train a new model for object detection.

3) *Gather and Label Images:* To train a successful detection classifier, TensorFlow needs several images. The training images should have random plants in the image along with the desired plants and should have a variety of backgrounds and lighting conditions to train a robust classifier. There must be some images where the desired species is partially obscured, overlapped with something else, or only partially visible in the picture. The next step is to label the objects in the images. Labeling of all the desired objects in the Ocimum Genus dataset was done in each individual image captured. LabelImg tool was used for labeling the images, whose source files were found on GitHub. Fig. 3. Is a sample of how the labeling was done for each image.

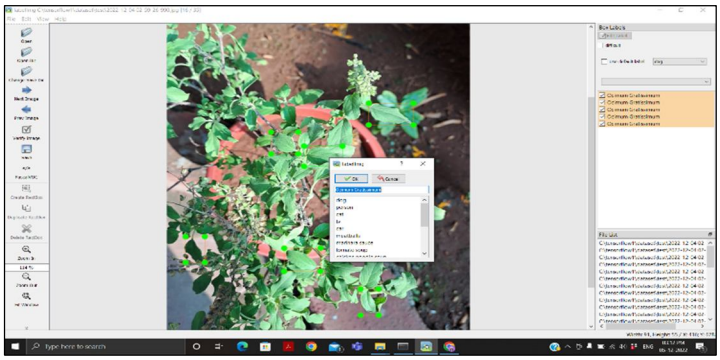


Fig. 3. Labelling Image using LabelImg tool

4) *Generate Training Data:* First, the image .xml data is used to create .csv files containing all the data for the train and test images.
5) *Create LabelMap and Configure Training:* The label map tells the trainer what each plant is by defining a mapping of class names to class ID numbers.
6) *Run the Training:* The pipeline for object detection training must then be set up. It specifies the model to be used as well as the training parameters. Every stage of training reports the loss. It will begin high and gradually decrease as you train. It initially peaked around 3.0 during training on the Faster-RCNN-Inception-V2 model and then quickly fell below 0.8. Starting with a loss of around 20, the MobileNet-SSD model was trained until the loss was consistently around 2. Progress of the training job can be monitored using TensorBoard. To test the object detector, move a picture of the object or objects into the \object detection folder, and mention the Image_Name variable in the a simple python script to detect object from the image. Alternatively, a video of the objects can also be used to detect the image from a live video, or from a USB webcam pointing at the objects by running a simple python script.

VI. IMPLEMENTATION RESULTS

After some initialization time, the object detector displays a window showing any objects it has detected in the image. Information and graphs illustrating the training's progression are available on the TensorBoard page. The loss graph, is one of the crucial graphs and it displays the overall loss of the classifier over time. Fig. 4. Is the loss graph for our classifier. Our implementation had a few minor setbacks and a few improvements. We learned from our mistakes and minimized the detected errors as much as we could over time.

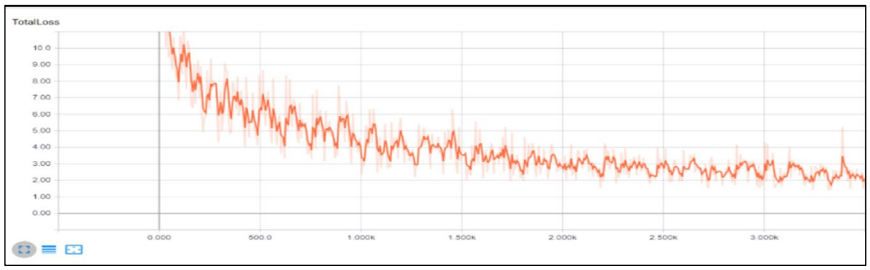


Fig. 4. Loss Graph of Object Detection Classifier

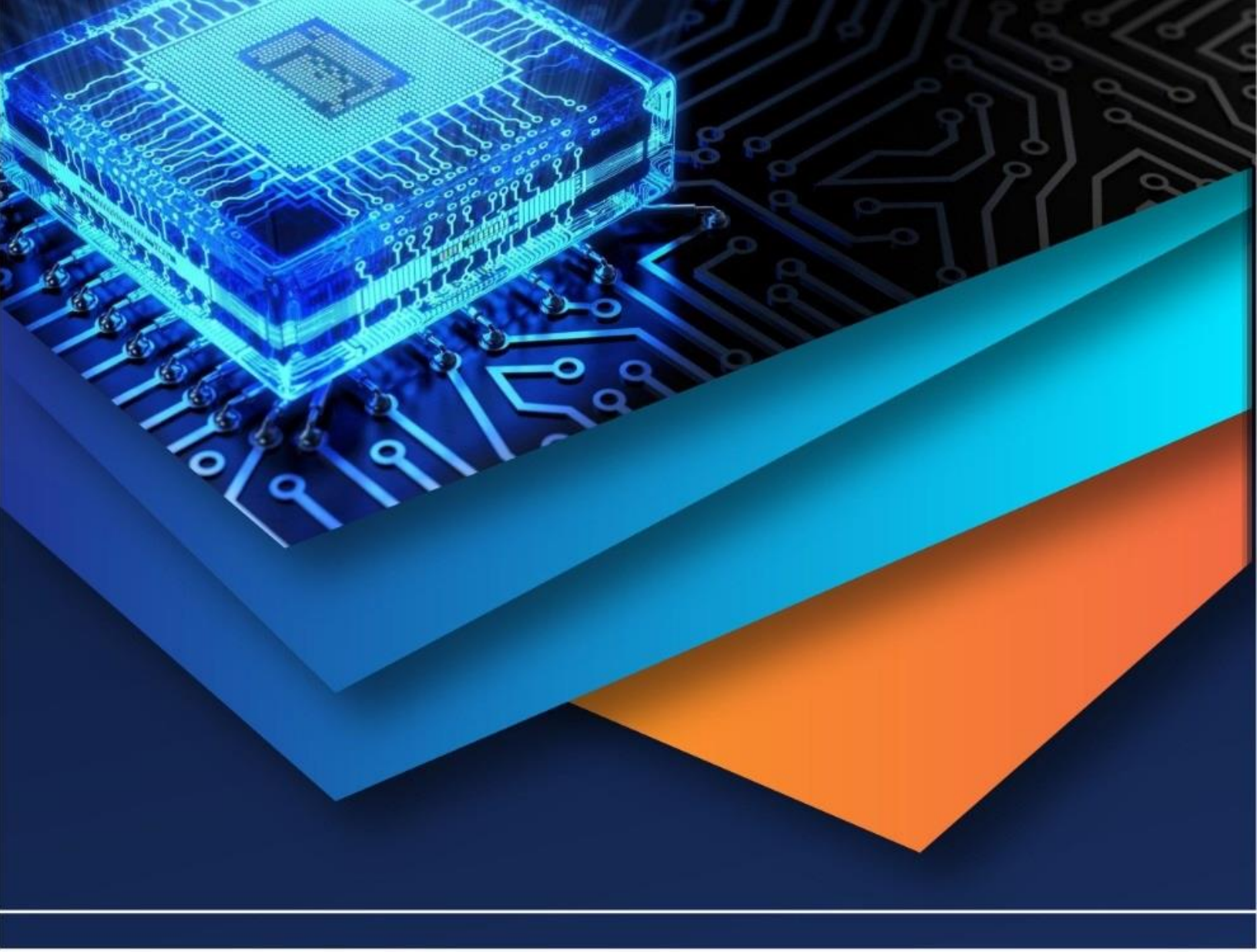
VII. CONCLUSION

The proposed methodology has been tested with different sample leaf images of the Ocimum genus. Qualified medical professionals or expert taxonomists are required to identify a medicinal plant. Through this work, the amount of physical labor and time needed to perform Ayurvedic species recognition can be decreased. This paper has proposed an automatic plant species identification approach which is employed using computer vision and machine learning techniques to classify plant leaf images. The study has been conducted in phases like image pre-processing, image segmentation, feature extraction, and finally classification of the image. The strength of this approach includes its simplicity, time management, accuracy, and ease of implementation. However, dataset creation is one of the major tasks to increase accuracy percentage and detect several plant species. The originality of the dataset is questionable hence it is important to verify it from experienced botanists. The proposed work can be extended to find the defective leaves to increase the accuracy and find if there is a loss in the medicinal properties due to nutrition deficiency in any plant.

REFERENCES

[1] Lottes, Philipp, et al. "UAV-based crop and weed classification for smart farming." 2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2018.
[2] Ghazi, Mostafa Mehdipour, Berrin Yanikoglu, and Erchan Aptoula. "Plant identification using deep neural networks via optimization of transfer learning parameters." Neurocomputing 235 (2018): 228-235.
[3] D. G. Adler and C. J. Gostout, “Wireless capsule endoscopy”, Hospital Physician, pp. 14-22, 2003.
[4] L. Baopu and M. Q. H. Meng, “Computer-aided detection of bleeding regions for capsule endoscopy images”, IEEE Trans. Biomedical Engineering, vol. 56, no. 4, pp. 1032-39, Apr. 2009.
[5] S. Liangpunsakul, L. Mays and D. K. Rex, “Performance of given suspected blood indicator”, American Journal of Gastroenterology, vol. 98, no.12, pp. 2676-8, Jan. 2004.
[6] Kolivand, Hoshang, et al. "A new leaf venation detection technique for plant species classification." Arabian Journal for Science and Engineering 44.4 (2019): 3315-3327.
[7] Wäldchen, Jana, and Patrick Mäder. "Plant species identification using computer vision techniques: A systematic literature review." Archives of Computational Methods in Engineering 25.2 (2018): 507-543.
[8] Masemola, Cecilia, Moses Azong Cho, and Abel Ramoelo. "Assessing the effect of seasonality on leaf and canopy spectra for the discrimination of an alien tree species, Acacia mearnsii, from Cooccurring native species using parametric and nonparametric classifiers." IEEE Transactions on Geoscience and Remote Sensing 57.8 (2019): 5853-5867.
[9] J. M. Buscaglia et. al., “Performance characteristics of the suspectedblood indicator feature in capsule endoscopy according to
[10] indication for study”, Clinical gastroenterology and hepatology: the official clinical practice journal of the American Gastroenterological Association, vol. 6, no. 3, pp. 298-301, Mar. 2008.
[11] Guolan Lv, Guozheng Yan and Zhiwu Wang, “Bleeding detection in wireless capsule endoscopic images based on color invariants and spatial pyramids using support vector machines”, 33rd annual international conference of IEEE EMBS Boston, USA, pp. 6643-46, 2011.
[12] Y.Fu, M. Mandal and G. Guo, “Bleeding region detection in wce images based on color features and neural network”, in Proc. IEEE 54th Int. Midwest Symp. Circuits Syst., pp. 1-4, Aug. 2011.
[13] Jie Li, Jinwen Ma, Tammam Tillo, Bailing Zhang and Eng Gee Lim, “A training based support vector machine technique for blood detection in wireless capsule endoscopy images”, IEEE EMBS Conference on Biomedical Engineering and Sciences (IECBES), pp. 826-830, Dec. 2012.
[14] Y. Li, Q. Zhu, Y. Cao, and C. Wang, “A leaf vein extraction method based on snakes technique,” in Proceedings of the International Conference on Neural Networks and Brain (ICNN&B '05), pp. 885–888, 2005.
[15] Fan, Jianping, et al. "Hierarchical learning of tree classifiers for largescale plant species identification." IEEE Transactions on Image Processing 24.11 (2018): 4172-4184.
[16] “Convolutional Neural Network.” Wikipedia, Wikimedia Foundation, 14 Mar. 2018, en.wikipedia.org/wiki/Convolutional_neural_network.
[17] Kaur, Surleen & Kaur, Prabhpreet. (2019). Plant Species Identification based on Plant Leaf Using Computer Vision and Machine Learning Techniques. Journal of Multimedia Information System. 6. 49-60. 10.33851/JMIS.2019.6.2.49.
[18] Purohit, Suchit, et al. "Automatic plant species recognition technique using machine learning approaches." 2015 International Conference on Computing and Network Communications (CoCoNet). IEEE, 2015
[19] Lee, Sue Han, et al. "How deep learning extracts and learns leaf features for plant classification." Pattern Recognition 71 (2018): 113.
[20] Saleem, G., et al. "Automated analysis of visual leaf shape features for plant classification." Computers and Electronics in Agriculture 157 (2019): 270-280.
[21] Xuanxin Liu, Fu Xu, Yu Sun, Haiyan Zhang, and Zhibo Chen, “Convolutional Recurrent Neural Networks for Observation-Centered Plant Identification,” Journal of Electrical and Computer Engineering, vol. 2018, pp. 1–7, 2018.
[22] Quoc Bao, Truong, et al. "Plant species identification from leaf patterns using histogram of oriented gradients feature space and convolution neural networks." *Journal of Information and Telecommunication* 4.2 (2020): 140-150.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)