**Abstract:**

Recognition of plants has become an active area of research as most of the plant species are at the risk of extinction. This paper uses an efficient machine learning approach for the classification purpose. This proposed approach consists of three phases such as preprocessing, feature extraction and classification. The preprocessing phase involves a typical image processing steps such as transforming to gray scale and boundary enhancement. The feature extraction phase derives the common DMF from five fundamental features. The main contribution of this approach is the Support Vector Machine (SVM) classification for efficient leaf recognition. Different leaf features which are extracted and orthogonalized into many principal variables are given as input vector to the SVM. Classifier tested with flavia dataset and a real dataset the proposed approach produces very good accuracy and takes very less execution time.

CHAPTER 1 INTRODUCTION

Plants play a vital role in the environment. There will be no existence of the earth’s ecology without plants. However, recently, several species of plants are at the danger of extinction. In order to protect plants and to catalogue various species of flora diversities, a plant database becomes very essential. There is huge volume of plant species worldwide. In order to handle such volumes of information, development of a rapid and competent classification technique has become an active area of research [1]. Moreover, along with the conservation feature, recognition of plants has also become essential to exploit their medicinal properties and using them as sources of alternative energy sources like bio-fuel. There are various ways to recognize a plant, like flower, root, leaf, fruit etc. Recently, computer vision and pattern recognition techniques have been applied towards automated process of plant recognition [2]

The classification of plant leaves is a vital mechanism in botany and in tea, cotton and other industries [3], [4]. Additionally, the morphological features of leaves are employed for plant classification or in the early diagnosis of certain plant diseases [5]. Plant recognition is an essential and challenging task. Leaf recognition plays an important role in plant classification and its key issue lies in whether the chosen features are constant and have good capability to discriminate various kinds of leaves. The recognition procedure is very time consuming. Computer aided plant recognition is still very challenging task in computer vision because of improper models and inefficient representation approaches. The main aim of plant recognition is to evaluate the leaf geometrical morphological and Fourier moment based features. This data is very vital in identifying the various classes of plants. Ji Xiang, Huang and Xiao Feng [6] carried out their investigation on recognizing the known plant species by salient features of the leaf such as physiological length, width, diameter, perimeter, area, smooth factor, aspect ratio and Fourier moments which could be employed to discriminate with each other. The extraction of leaf features from a plant is a key step in the plant recognition process [7, 8].

This feature extraction process creates a new challenge in the field of pattern recognition [9] [10]. The data acquisition from living plant automatically by the computer has not been implemented. This paper implements a leaf recognition algorithm using easy to extract features and high efficient recognition algorithm. The main phases involved in this research are feature extraction and the classification. All features are extracted from digital leaf image. manyfeatures are orthogonalized by Principal Components Analysis (PCA) [11] and are given to the classifier. The classifier used in this approach is Support Vector Machine [12, 13] for its fast speed and simple structure.

CHAPTER 2: LITURATURE SUVERY

**Plant Leaf Recognition Using Zernike Moments and Histogram of Oriented Gradients by Dimitris G. TsolakidisDimitrios I. KosmopoulosGeorge Papadourakis**

A method using Zernike Moments and Histogram of Oriented Gradients for classification of plant leaf images is proposed in this paper. After preprocessing, we compute the shape features of a leaf using Zernike Moments and texture features using Histogram of Oriented Gradients and then the Support Vector Machine classifier is used for plant leaf image classification and recognition. Experimental results show that using both Zernike Moments and Histogram of Oriented Gradients to classify and recognize plant leaf image yields accuracy that is comparable or better than the state of the art. The method has been validated on the Flavia and the Swedish Leaves datasets as well as on a combined dataset.

**Leaf and Flower Recognition Using Preferential Image Segmentation Algorithm by N. ValliammalS. N. Geethalakshmi**

Automatic plant classification systems are essential for a wide range of applications including environment protection, plant resource survey, as well as for education. With the aid of advanced information technology, image processing and machine learning techniques, automatic plant identification and classification will enhance such systems with more functionality, such as automatic labeling and flexible searching. Image segmentation and object recognition are two aspects of digital image processing which are being increasingly used in many applications including leaf recognition. In this paper, the Preferential Image Segmentation (PIS) method is used to segment an object of interest from the original image. A probabilistic curve evolution method with particle filters is used to measure the similarity between shapes during matching process. The experimental results prove that the preferential image segmentation can be successfully applied in leaf recognition and segmentation from a plant image.

**An Effective Tea Leaf Recognition Algorithm for Plant Classification Using Radial Basis Function Machine by Arunpriya Antony Selvadoss Thanamani**

A leaf is an organ of a vascular plant, as identified in botanical terms, and in particular in plant morphology. Naturally a leaf is a thin, flattened organ bear above ground and it is mainly used for photosynthesis. Recognition of plants has become an active area of research as most of the plant species are at the risk of extinction. Most of the leaves cannot be recognized easily since some are not flat (e.g. succulent leaves and conifers), some does not grow above ground (e.g. bulb scales), and some does not undergo photosynthetic function (e.g. cataphylls, spines, and cotyledons).In this paper, we mainly focused on tea leaves to identify the leaf type for improving tea leaf classification. Tea leaf images are loaded from digital cameras or scanners in the system. This proposed approach consists of three phases such as preprocessing, feature extraction and classification to process the loaded image. The tea leaf images can be identified accurately in the preprocessing phase by fuzzy denoising using Dual Tree Discrete Wavelet Transform (DT-DWT) in order to remove the noisy features and boundary enhancement to obtain the shape of leaf accurately. In the feature extraction phase, Digital Morphological Features (DMFs) are derived to improve the classification accuracy. Radial Basis Function (RBF) is used for efficient classification. The RBF is trained by 60 tea leaves to classify them into 6 types. Experimental results proved that the proposed method classifies the tea leaves with more accuracy in less time. Thus, the proposed method achieves more accuracy in retrieving the leaf type

**Computer-Aided Plant Species Identification (CAPSI) Based on Leaf Shape Matching Technique by** Ji-Xiang Du, De-Shuang Huang, Xiao-Feng Wang,

In this paper, an efficient computer-aided plant species identification (CAPSI) approach is proposed, which is based on plant leaf images using a shape matching technique. Firstly, a Douglas - Peucker approximation algorithm is adopted to the original leaf shapes and a new shape representation is used to form the sequence of invariant attributes. Then a modified dynamic programming (MDP) algorithm for shape matching is proposed for the plant leaf recognition. Finally, the superiority of our proposed method over traditional approaches to plant species identification is demonstrated by experiment. The experimental result showed that our proposed algorithm for leaf shape matching is very suitable for the recognition of not only intact but also partial, distorted and overlapped plant leaves due to its robustness .

**A Method of Plant Classification Based on Wavelet Transforms and Support Vector Machines Jiandu LiuShanwen ZhangShengli Deng**

As one of the most important morphological taxonomy features, plant leaf with many strong points has significant influence on research. In this paper, we propose a novel method of plant classification from leaf image set based on wavelet transforms and support vector machines (SVMS). Firstly, the leaf images are converted into the time-frequency domain image by wavelet transforms without any further preprocessing such as image enhancement and texture thinning, and then feature extraction vector is conducted. Then the effectiveness of the proposed method is evaluated by the classification accuracy of SVM classifier. The experimental results about the data set with 300 leaf images show that the method has higher recognition rate and faster processing speed.

Xiao Gu et al., [14] proposed a novel approach for leaf recognition by means of the result of segmentation of leaf’s skeleton based on the integration of Wavelet Transform (WT) and Gaussian interpolation. And then the classifiers,) are employed, based on Run-length Features (RF) obtained from the skeleton to identify the leaves. Ultimately, the efficiency of this approach is illustrated by several experiments. The results reveal that the skeleton can be effectively extracted from the entire leaf, and the recognition rates can be significantly improved. Xiao-Feng Wang et al., [15] proposed a technique of recognizing leaf images depending on shape features through a hypersphere classifier. Initially, the author employed image segmentation to the leaf images. Then, eight geometric features are extracted including rectangularity, circularity, eccentricity, etc., and seven moment invariants for classification. Ultimately, a moving center hypersphere classifier is presented to handle these shape features. Thus, there are more than 20 classes of plant leaves productively classified. The average correct recognition rate is up to 92.2 percent

**CHAPTER 3**

**SYSTEM REQUIREMENT SPECIFICATION**

**3.1 System Analysis**

Prediction of terrorism activities is an important area of concern for researchers. The large number of events makes it difficult to predict terrorist group responsible for some terrorist activity.

The current research is focused on finding out the correlation between terrorism and its causal factors. Existing efforts have not been good enough for prediction. Machine learning approaches can ad in predicting the likelihood of a terrorist attack, given the required data. The results of this work can help the security agencies and policy makers to eradicate terrorism by taking relevant and effective measures.

Hence there is an approach to analyzing terrorism region and country with the machine learning techniques and terrorism specific knowledge to fetch conclusions about terrorist behavior patterns.

* 1. **Functional Requirement**

The particular necessities are user interfaces. The outside clients are the customers.

Every one of the customers can utilize this product for ordering and looking.

* Hardware Interfaces: The outside equipment interface utilized for ordering and looking is PCs of the customers. The PC's might be portable PCs with remote LAN as the web association gave will be remote.
* Software Interfaces: The working Frameworks can be any rendition of windows.
* Performance Prerequisites: The PC's utilized must be atleast pentium 4 machine with the goal that they can give ideal execution of the item.

**3.3 Non-Functional Requirements**

Non utilitarian necessities are the capacities offered by the framework. It incorporates time imperative and requirement on the advancement procedure and models. The non useful prerequisites are as per the following:

* Speed: The framework ought to prepare the given contribution to yield inside fitting time.
* Ease of utilization: The product tought to be easy to understand. At that point the clients can utilize effortlessly, so it doesn't require much preparing time.
* Reliability: The rate of disappointments ought to be less then just the framework is more solid.
* Portability: It thought to be anything but difficult to actualize in any framework

# H/W System Configuration:

|  |  |
| --- | --- |
| Processor | Dual Core. |
| Speed | 1.1 G Hz. |
| RAM | 1GB. |
| Hard Disk | 500MB. |

# S/W System Configuration:

|  |  |
| --- | --- |
| Operating System | Windows 10. |
| Technology | Machine Learning. |
| **Front End** | GUI-tkinter. |
| **IDLE** | Python 3.7 or higher. |

**Hardware requirements**

The most widely recognized arrangement of prerequisites characterized by any working framework or programming application is the physical PC assets, otherwise called equipment, An equipment necessities list is frequently joined by an equipment similarity list, particularly if there should be an occurrence of working frameworks. A HCL records tried, perfect, and now and then incongruent equipment gadgets for a specific working framework or application. The accompanying sub-segments examine the different parts of equipment prerequisites.

All PC working frameworks are intended for a specific PC design. Most programming applications are restricted to specific working frameworks running on specific structures. In spite of the fact that engineering free working frameworks and applications exist, most should be recompiled to keep running on another design.

The energy of the focal preparing unit (CPU) is a central framework necessity for any product. Most programming running on x86 engineering characterize preparing power as the model and the clock speed of the CPU. Numerous different highlights of a CPU that impact its speed and power, similar to transport speed, store, and MIPS are frequently overlooked. This meaning of energy is regularly wrong, as AMD Intel Pentium CPUs at comparative clock speed frequently have distinctive throughput speeds.

• 10GB HDD(min)

• 128 MB RAM(min)

• Pentium P4 Processor 2.8Ghz(min)

**Software requirements**

Programming necessities manage characterizing programming asset necessities and requirements that should be introduced on a PC to give ideal working of an application.

These necessities or requirements are for the most part excluded in the product establishment bundle and should be introduced independently before the product is introduced.

* Python 3.7 or higher
* Pycharm
* opencv

**3.4 Tools and Technology details**

**Tool:**

IDLE is Python's Integrated Development and Learning Environment. It allows programmers to easily write Python code. Just like Python Shell, IDLE can be used to execute a single statement and create, modify, and execute Python scripts.

**Technology:**

**Machine Learning** is the field of study that gives computers the capability to learn without being explicitly programmed. ML is one of the most exciting technologies that one would have ever come across. As it is evident from the name, it gives the computer that makes it more similar to humans: The ability to learn. Machine learning is actively being used today, perhaps in many more places than one would expect.

**3.4 Python**

* Python is a general purpose high level programming Language (human understandable languages are High level programming languages)
* Python Developed by Guido Van Rossam
* 1989 National Research Institute(NRI) At Netherland
* Officially Python available to the public in 1991 :: FEB 20th 1991

Python was imagined in the late 1980s,[29] and its usage started in December 1989[30] by Guido van Rossum at Centrum Wiskunde and Informatica (CWI) in the Netherlands as a successor to the ABC dialect (itself roused by SETL)[31]capable of exemption dealing with and interfacing with the Amoeba working system.[6] Van Rossum remains Python's chief creator. His proceeding with focal part in Python's advancement is reflected in the title given him by the Python people group:

**Python Features**

* Simple and easy to learn

Python as only 33 keywords But JAVA as(53) keywords

* Free ware (There is no license we cannot pay anything)

& Open source (we can able to see source code if source is not good I can able to customize our requirements)

* High level programming language (human understandable language)

Python Is Platform Independent (It means I can write a program once and run any where(WORA)

* Portability

Moving python program from one platform to another platform without changing any thing

* Dynamically Typed Programming Language
* Both Object Oriented and Procedure Oriented Language
* Interpreted Language

It means we are not going to compile

* Extensible

We can use Other Programming Language in Python

**Limitations of python**

* Performance wise it is not up to the mark Beacause its an interpreted language

Interpreter able to see only one line

(JAVA is better performace compare to python in java JIT (just intime compiler) concept is there

* Mobile appications it is not up to the mark

Myth:-python is not suitable large scale enterprise applications

**Flavors of python**

* Cpython :- It can be standard, It can be used to c language python
* Jpython or jpython :- it is for JAVA application
* Iron python:-to work with microsoft .net platform
* Pypy :-Internally JIT (just intime compiler) compiler is there so performance wise too good
* Ruby python:- used for ruby application
* Anaconda python:- To handle Big-data happily go for Anaconda python
* Stackless (python for concurrancy):- parallely you execute (like mutithread) so go for stackless

**Applications of Python**

**1. GUI-Based Desktop Applications:** Python has simple syntax, modular architecture, rich text processing tools and the ability to work on multiple operating systems which make it a desirable choice for developing desktop-based applications. There are various GUI toolkits like wxPython, PyQt or PyGtk available which help developers create highly functional Graphical User Interface (GUI). The various applications developed using Python includes:

* **Image Processing and Graphic Design Applications:**

Python has been used to make 2D imaging software such as Inkscape, GIMP, Paint Shop Pro and Scribus. Further, 3D animation packages, like Blender, 3ds Max, Cinema 4D, Houdini, Lightwave and Maya, also use Python in variable proportions.

* **Scientific and Computational Applications:**

The higher speeds, productivity and availability of tools, such as Scientific Python and Numeric Python, have resulted in Python becoming an integral part of applications involved in computation and processing of scientific data. 3D modeling software, such as FreeCAD, and finite element method software, such as Abaqus, are coded in Python.

* **Games:**

Python has various modules, libraries and platforms that support development of games. For example, PySoy is a 3D game engine supporting Python 3, and PyGame provides functionality and a library for game development. There have been numerous games built using Python including Civilization-IV, Disney’s Toontown Online, Vega Strike etc.

**2. Web Frameworks and Web Applications:** Python has been used to create a variety of web-frameworks including CherryPy, Django, TurboGears, Bottle, Flask etc. These frameworks provide standard libraries and modules which simplify tasks related to content management, interaction with database and interfacing with different internet protocols such as HTTP, SMTP, XML-RPC, FTP and POP. Plone, a content management system; ERP5, an open source ERP which is used in aerospace, apparel and banking; Odoo – a consolidated suite of business applications; and Google App engine are a few of the popular web applications based on Python.

**3. Enterprise and Business Applications:** With features that include special libraries, extensibility, scalability and easily readable syntax, Python is a suitable coding language for customizing larger applications. Reddit, which was originally written in Common Lips, was rewritten in Python in 2005. Python also contributed in a large part to functionality in YouTube.

**4. Operating Systems:**Python is often an integral part of Linux distributions. For instance, Ubuntu’s Ubiquity Installer, and Fedora’s and Red Hat Enterprise Linux’s Anaconda Installer are written in Python. Gentoo Linux makes use of Python for Portage, its package management system.

**5. Language Development:** Python’s design and module architecture has influenced development of numerous languages. Boo language uses an object model, syntax and indentation, similar to Python. Further, syntax of languages like Apple’s Swift, CoffeeScript, Cobra, and OCaml all share similarity with Python.

**6. Prototyping:** Besides being quick and easy to learn, Python also has the open source advantage of being free with the support of a large community. This makes it the preferred choice for prototype development. Further, the agility, extensibility and scalability and ease of refactoring code associated with Python allow faster development from initial prototype.Since its origin in 1989, Python has grown to become part of a plethora of web-based, desktop-based, graphic design, scientific, and computational applications. With Python available for Windows, Mac OS X and Linux / UNIX, it offers ease of development for enterprises. Additionally, the latest release Python 3.4.3 builds on the existing strengths of the language, with drastic improvement in Unicode support, among other new features.

**Versions of python**

* Python 1.0 Introduced in jan 1994
* Python 2.0 Introduced in oct 2000
* Python 3.0 introduced in dec 2008

Latest version

python 3.9.1

Any new version should provide support for old version programs

* There is no- backward compatibility support
* Python 3 is not support to python 2 program

**Python Machine Learning**

Python is a popular platform used for research and development of production systems. It is a vast language with number of modules, packages and libraries that provides multiple ways of achieving a task.

Python and its libraries like NumPy, SciPy, Scikit-Learn, Matplotlib are used in data science and data analysis. They are also extensively used for creating scalable machine learning algorithms. Python implements popular machine learning techniques such as Classification, Regression, Recommendation, and Clustering.

Python offers ready-made framework for performing data mining tasks on large volumes of data effectively in lesser time. It includes several implementations achieved through algorithms such as linear regression, logistic regression, Naïve Bayes, k-means, K nearest neighbor, and Random Forest.

**Python in Machine Learning**

Python has libraries that enables developers to use optimized algorithms. It implements popular machine learning techniques such as recommendation, classification, and clustering. Therefore, it is necessary to have a brief introduction to machine learning before we move further.

**What is Machine Learning?**

Data science, machine learning and artificial intelligence are some of the top trending topics in the tech world today. Data mining and Bayesian analysis are trending and this is adding the demand for machine learning. This tutorial is your entry into the world of machine learning.

Machine learning is a discipline that deals with programming the systems so as to make them automatically learn and improve with experience. Here, learning implies recognizing and understanding the input data and taking informed decisions based on the supplied data. It is very difficult to consider all the decisions based on all possible inputs. To solve this problem, algorithms are developed that build knowledge from a specific data and past experience by applying the principles of statistical science, probability, logic, mathematical optimization, reinforcement learning, and control theory.

**Applications of Machine Learning Algorithms**

The developed machine learning algorithms are used in various applications such as:

Vision processing

Language processing

Forecasting things like stock market trends, weather

Pattern recognition

Games

Data mining

Expert systems

Robotics

**Libraries and Packages**

To understand machine learning, you need to have basic knowledge of Python programming. In addition, there are a number of libraries and packages generally used in performing various machine learning tasks as listed below:

**numpy** - is used for its N-dimensional array objects

**pandas** – is a data analysis library that includes dataframes

**matplotlib** – is 2D plotting library for creating graphs and plots

**scikit-learn** - the algorithms used for data analysis and data mining tasks

**seaborn** – a data visualization library based on matplotlib

**Installation**

You can install software for machine learning in any of the two methods as discussed here:

**Method 1**

Download and install Python separately from **python.org** on various operating systems as explained below:

To install Python after downloading, double click the **.exe** (for Windows) or .**pkg** (for Mac) file and follow the instructions on the screen.

For Linux OS, check if Python is already installed by using the following command at the prompt:

$ python --version. ...

If Python 2.7 or later is not installed, install Python with the distribution's package manager. Note that the command and package name varies.

On Debian derivatives such as Ubuntu, you can use **apt**:

$ sudo apt-get install python3

Now, open the command prompt and run the following command to verify that Python is installed correctly:

$ python3 --version

Python 3.6.2

**Python Machine Learning – Environment Setup**

Similarly, we can download and install necessary libraries like numpy, matplotlib etc. individually using installers like **pip**. For this purpose, you can use the commands shown here:

$pip install numpy

$pip install matplotlib

$pip install pandas

$pip install seaborn

**Method 2**

Alternatively, to install Python and other scientific computing and machine learning packages simultaneously, we should install **Anaconda** distribution. It is a Python implementation for Linux, Windows and OSX, and comprises various machine learning packages like numpy, scikit-learn, and matplotlib. It also includes **Jupyter Notebook**, an interactive Python environment. We can install Python 2.7 or any 3.x version as per our requirement.

To download the free Anaconda Python distribution from Continuum Analytics, you can do the following:

Visit the official site of Continuum Analytics and its download page. Note that the installation process may take 15-20 minutes as the installer contains Python, associated packages, a code editor, and some other files. Depending on your operating system, choose the installation process as explained here:

**For Windows:** Select the **Anaconda for Windows** section and look in the column with Python 2.7 or 3.x. You can find that there are two versions of the installer, one for 32-bit Windows, and one for 64-bit Windows. Choose the relevant one.

**For Mac OS:** Scroll to the **Anaconda for OS X** section. Look in the column with Python 2.7 or 3.x. Note that here there is only one version of the installer: the 64-bit version.

**For Linux OS:** We select the "Anaconda for Linux" section. Look in the column with Python 2.7 or 3.x.

Note that you have to ensure that Anaconda’s Python distribution installs into a single directory, and does not affect other Python installations, if any, on your system.

To work with graphs and plots, we will need these Python library packages: **matplotlib** and **seaborn**.

If you are using Anaconda Python, your system already has numpy, matplotlib, pandas, seaborn, etc. installed. We start the Anaconda Navigator to access either Jupyter Note book or Spyder IDE of python.

After opening either of them, type the following commands:

import numpy

import matplotlib Python Machine Learning

Now, we need to check if installation is successful. For this, go to the command line and type in the following command:

$ python

Python 3.6.3 |Anaconda custom (32-bit)| (default, Oct 13 2017, 14:21:34)

[GCC 7.2.0] on linux

Next, you can import the required libraries and print their versions as shown:

>>>import numpy

>>>print numpy.\_\_version\_\_

1.14.2

>>> import matplotlib

>>> print (matplotlib.\_\_version\_\_)

2.1.2

>> import pandas

>>> print (pandas.\_\_version\_\_)

0.22.0

>>> import seaborn

>>> print (seaborn.\_\_version\_\_)

**3.5 Machine Learning**

Machine Learning (ML)is an automated learning with little or no human intervention. It involves programming computers so that they learn from the available inputs. The main purpose of machine learning is to explore and construct algorithms that can learn from the previous data and make predictions on new input data.

The **input** to a learning algorithm is training data, representing experience, and the **output** is any expertise, which usually takes the form of another algorithm that can perform a task. The input data to a machine learning system can be numerical, textual, audio, visual, or multimedia. The corresponding output data of the system can be a floating-point number, for instance, the velocity of a rocket, an integer representing a category or a class, for example, a pigeon or a sunflower from image recognition.

In this chapter, we will learn about the training data our programs will access and how learning process is automated and how the success and performance of such machine learning algorithms is evaluated.

**Concepts of Learning**

Learning is the process of converting experience into expertise or knowledge.

Learning can be broadly classified into three categories, as mentioned below, based on the nature of the learning data and interaction between the learner and the environment.

Supervised Learning

Unsupervised Learning

Semi-supervised learning

Similarly, there are four categories of machine learning algorithms as shown below:

Supervised learning algorithm

Unsupervised learning algorithm

Semi-supervised learning algorithm

Reinforcement learning algorithm

However, the most commonly used ones are **supervised** and **unsupervised learning**.

# Data Preprocessing in Machine learning

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.

When creating a machine learning project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So for this, we use data preprocessing task.

## Why do we need Data Preprocessing?

A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

It involves below steps:

* **Getting the dataset**
* **Importing libraries**
* **Importing datasets**
* **Finding Missing Data**
* **Encoding Categorical Data**
* **Splitting dataset into training and test set**
* **Feature scaling**

**Supervised Learning**

Supervised learning is commonly used in real world applications, such as face and speech recognition, products or movie recommendations, and sales forecasting. Supervised learning can be further classified into two types: **Regression** and **Classification**.

Regression trains on and predicts a continuous-valued response, for example predicting real estate prices.

Classification attempts to find the appropriate class label, such as analyzing positive/negative sentiment, male and female persons, benign and malignant tumors, secure and unsecure loans etc.

In supervised learning, learning data comes with description, labels, targets or desired outputs and the objective is to find a general rule that maps inputs to outputs. This kind of learning data is called labeled data. The learned rule is then used to label new data with unknown outputs.

Supervised learning involves building a machine learning model that is based on labeled samples. For example, if we build a system to estimate the price of a plot of land or a house based on various features, such as size, location, and so on, we first need to create a database and label it. We need to teach the algorithm what features correspond to what prices. Based on this data, the algorithm will learn how to calculate the price of real estate using the values of the input features.

Supervised learning deals with learning a function from available training data. Here, a learning algorithm analyzes the training data and produces a derived function that can be used for mapping new examples. There are many supervised learning algorithms such as Logistic Regression, Neural networks, Support Vector Machines (SVMs), and Naive Bayes classifiers.

Common examples of supervised learning include classifying e-mails into spam and not-spam categories, labeling webpages based on their content, and voice recognition.

**Unsupervised Learning**

Unsupervised learning is used to detect anomalies, outliers, such as fraud or defective equipment, or to group customers with similar behaviors for a sales campaign. It is the opposite of supervised learning. There is no labeled data here.

When learning data contains only some indications without any description or labels, it is up to the coder or to the algorithm to find the structure of the underlying data, to discover hidden patterns, or to determine how to describe the data. This kind of learning data is called unlabeled data. Suppose that we have a number of data points, and we want to classify them into several groups. We may not exactly know what the criteria of classification would be. So, an unsupervised learning algorithm tries to classify the given dataset into a certain number of groups in an optimum way.

Unsupervised learning algorithms are extremely powerful tools for analyzing data and for identifying patterns and trends. They are most commonly used for clustering similar input into logical groups. Unsupervised learning algorithms include Kmeans, Random Forests, Hierarchical clustering and so on.

**Semi-supervised learning**

In this type of learning, the algorithm is trained upon a combination of labeled and unlabeled data. Typically, this combination will contain a very small amount of labeled data and a very large amount of unlabeled data. The basic procedure involved is that first, the programmer will cluster similar data using an unsupervised learning algorithm and then use the existing labeled data to label the rest of the unlabeled data. The typical use cases of such type of algorithm have a common property among them – The acquisition of unlabeled data is relatively cheap while labeling the said data is very expensive.

**Reinforcement Learning**

This learning data gives feedback so that the system adjusts to dynamic conditions in order to achieve a certain objective. The system evaluates its performance based on the feedback responses and reacts accordingly. The best known instances include self-driving cars and chess master algorithm AlphaGo.

**Purpose of Machine Learning**

Machine learning can be seen as a branch of AI or Artificial Intelligence, since, the ability to change experience into expertise or to detect patterns in complex data is a mark of human or animal intelligence.

As a field of science, machine learning shares common concepts with other disciplines such as statistics, information theory, game theory, and optimization.

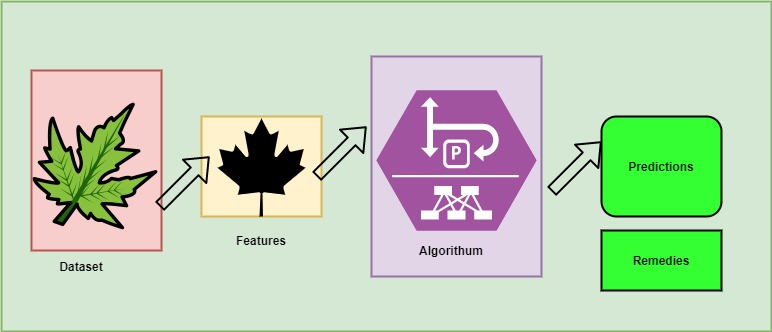
As a subfield of information technology, its objective is to program machines so that they will learn.

However, it is to be seen that, the purpose of machine learning is not building an automated duplication of intelligent behavior, but using the power of computers to complement and supplement human intelligence. For example, machine learning programs can scan and process huge databases detecting patterns that are beyond the scope of human perception

**SYSTEM DESIGN**

System design is the process of defining the architecture, components, modules, interfaces and data for a system to satisfy specified requirements. One could see it as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering. If the broader topic of product development "blends the perspective of marketing, design, and manufacturing into a single approach to product development," then design is the act of taking the marketing information and creating the design of the product to be manufactured. Systems design is therefore the process of defining and developing systems to satisfy specified requirements of the user.

**4.1 Overall system architecture**

****

**Activity Diagram**

We use Activity Diagrams to illustrate the flow of control in a system and refer to the steps involved in the execution of a use case. We model sequential and concurrent activities using activity diagrams. So, we basically depict workflows visually using an activity diagram. An activity diagram focuses on condition of flow and the sequence in which it happens. We describe or depict what causes a particular event using an activity diagram.

Dataset

Algorithm

Train data

Prediction

Preprocess

**State Transition Diagram**

State-transition diagrams describe all of the states that an object can have, the events under which an object changes state (transitions), the conditions that must be fulfilled before the transition will occur (guards), and the activities undertaken during the life of an object (actions). State-transition diagrams are very useful for describing the behavior of individual objects over the full set of use cases that affect those objects. State-transition diagrams are not useful for describing the collaboration between objects that cause the transitions.

**Data Flow Diagram(DFD)**

A data flow diagram is a graphical representation of the "flow" of data through an information system, modeling its process aspects. Often they are a preliminary step used to create an overview of the system which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design). The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of the input data to the system, various processing carried out on these data, and the output data is generated by the system.

Dataset Input

Data Preproceesing

Feature extraction

Apply Algorthm

Predict leaf with remedies

**IMPLEMENTATION**

There are 3 steps for implementation;

1) Dataset collection

2) Data Feature Extraction

3) Machine learning algorithm apply

4) Prediction

**Dataset collection:**

We collect the leaf flavia dataset it includes total 32 leaf categories

**Data Feature Extraction**

There are many features extracted from the individual leaf with opencv mechanisms.,

Features['area','perimeter','pysiological\_length','pysiological\_width','aspect\_ratio','rectangularity','circularity,'mean\_r','mean\_g','mean\_b','stddev\_r','stddev\_g','stddev\_b', 'contrast','correlation','inverse\_difference\_moments','entropy'}

Then this will go to the svm algorithm

**Machine learning algorithm apply**

Support Vector Machine(SVM) is a supervised machine learning algorithm used for both classification and regression. Though we say regression problems as well its best suited for classification. The objective of SVM algorithm is to find a hyperplane in an N-dimensional space that distinctly classifies the data points. The dimension of the hyperplane depends upon the number of features. If the number of input features is two, then the hyperplane is just a line. If the number of input features is three, then the hyperplane becomes a 2-D plane. It becomes difficult to imagine when the number of features exceeds three.

Let’s consider two independent variables x1, x2 and one dependent variable which is either a blue circle or a red circle.

****

From the figure above its very clear that there are multiple lines (our hyperplane here is a line because we are considering only two input features x1, x2) that segregates our data points or does a classification between red and blue circles. So how do we choose the best line or in general the best hyperplane that segregates our data points.

Selec One reasonable choice as the best hyperplane is the one that represents the largest separation or margin between the two classes.ting the best hyper-plane:

So we choose the hyperplane whose distance from it to the nearest data point on each side is maximized. If such a hyperplane exists it is known as the maximum-margin hyperplane/hard margin. So from the above figure, we choose L2.

Here we have one blue ball in the boundary of the red ball. So how does SVM classify the data? It’s simple! The blue ball in the boundary of red ones is an outlier of blue balls. The SVM algorithm has the characteristics to ignore the outlier and finds the best hyperplane that maximizes the margin. SVM is robust to outliers.



So in this type of data points what SVM does is, it finds maximum margin as done with previous data sets along with that it adds a penalty each time a point crosses the margin. So the margins in these type of cases are called soft margin. When there is a soft margin to the data set, the SVM tries to minimize (1/margin+∧(∑penalty)). Hinge loss is a commonly used penalty. If no violations no hinge loss.If violations hinge loss proportional to the distance of violation.

Prediction:

Then trained model with sample input to test which category also their remedies

**TESTING:**

Testing is a critical element which assures quality and effectiveness of the proposed system in (satisfying) meeting its objectives. Testing is done at various stages in the System designing and implementation process with an objective of developing an transparent, flexible and secured system. Testing is an integral part of software development. Testing process, in a way certifies, whether the product, that is developed, complies with the standards, that it was designed to. Testing process involves building of test cases, against which, the product has to be tested.

**Test objectives**

* Testing is a process of executing a program with the intent of finding an error.
* A good case is one that has a high probability of finding an undiscovered error.
* A successful test is one that uncovers a yet undiscovered error. If testing is conducted successfully (according to the objectives) it will uncover errors in the software. Testing can't show the absences of defects are present. It can only show that software defects are present.

**Testing principles**

Before applying methods to design effective test cases, a software engineer must understand the basic principle that guides software testing. All the tests should be traceable to customer requirements.

**Testing design**

Any engineering product can be tested in one of two ways:

**White box Testing**

This testing is also called as glass box testing. Inthis testing, by knowing the specified function that a product has been designed to perform test can be conducted that demonstrates each function is fully operation at the same time searching for errors in each function.

it is a test case design method that uses the control structure of the procedural design to derive test cases.

**Black box Testing**

Inthis testing by knowing the internal operation of a product, tests can be conducted to ensure that "all gears mesh", that is the internal operation performs according to specification and all internal components have been adequately exercised. It fundamentally focuses on the functional requirements of the software.

The steps involved in black box test case design are:

* Graph based testing methods
* Equivalence partitioning
* Boundary value analysis
* Comparison testing

**Testing strategies**

A software testing strategy provides a road map for the software developer. Testing is a set of activities that can be planned in advanced and conducted systematically. For this reason a template for software testing a set of steps into which we can place specific test case design methods should be defined for software engineering process.

**Any software testing strategy should have the following characteristics:**

* 1. Testing begins at the module level and works outward toward the integration of the entire computer based system.
  2. Different testing techniques are appropriate at different points in time.
  3. The developer of the software and an independent test group conducts testing.
  4. Testing and debugging are different activities but debugging must be accommodated in any testing strategy.

**Levels of Testing**

Testing can be done in different levels of SDLC. They are:

**Unit Testing**

The first level of testing is called unit testing. Unit testing verifies on the smallest unit of software designs-the module. The unit test is always white box oriented. In this, different modules are tested against the specifications produced during design for the modules. Unit testing is essentially for verification of the code produced during the coding phase, and hence the goal is to test the internal logic of the modules. It is typically done by the programmer of the module. Due to its close association with coding, the coding phase is frequently called “coding and unit testing.” The unit test can be conducted in parallel for multiple modules.

**Integration Testing**

The second level of testing is called integration testing. Integration testing is a systematic technique for constructing the program structure while conducting tests to uncover errors associated with interfacing. In this, many tested modules are combined into subsystems, which are then tested. The goal here is to see if all the modules can be integrated properly.

There are three types of integration testing:

* + - *Top-Down Integration*: Top down integration is an incremental approach to construction of program structures. Modules are integrated by moving downwards throw the control hierarchy beginning with the main control module.
    - *Bottom-Up Integration*: Bottom up integration as its name implies, begins Construction and testing with automatic modules.
    - *Regression Testing*: In this contest of an integration test strategy, regression testing is the re execution of some subset of test that have already been conducted to ensure that changes have not propagated unintended side effects.

**9.5.3 Functional test**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Table X: Functional Testing items

|  |  |
| --- | --- |
| Valid Input | Identified classes of valid input must be accepted. |
| Invalid Input | Identified classes of invalid input must be rejected. |
| Functions | Identified functions must be exercised. |
| Output | Identified classes of application outputs must be exercised. |

***Systems/Procedures:*** Interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**Validation testing**

At the culmination of integration testing, software is completely assembled as a package; interfacing errors have been covered and corrected, and final series of software tests-validating testing may begin. Validation can be defined in many ways, but a simple definition is that validation succeeds when software functions in a manner that can be reasonably expected by customers. Reasonable expectation is defined in the software requirement specification- a document that describes all user visible attributes of the software. The specification contains a section title “validation criteria”. Information contained in that section forms the basis for validation testing approach

**Alpha testing**

It is virtually impossible for a software developer to forsee how the customer will really use a program. Instructions for use may be misinterpreted; strange combination of data may be regularly used and output that seemed clear to the tester may be unintelligible to a user in field.

When custom software is built for one customer, a series of acceptance tests are conducted to enable the customer to validate all requirements by the end user rather than system developer and acceptable test can range from an informal “test drive” to a planned and systematically executed series of tests. In fact, acceptance testing can be conducted over a period of weeks or months, thereby uncovering cumulative errors that might degrade the system over time. If software is developed as a product to be used by many customers, it is impractical to perform formal acceptance test with each one. Most software product builders use a process called alpha and beta testing to uncover errors that only the end user seems able to find.

A customer conducts the alpha test at the developer’s site. The software is used in a natural setting with the developer “Looking over the shoulder” of the user and recording errors and usage problems. Alpha tests are conducted in controlled environment.

**Beta testing**

The beta test is conducted at one or more customer sites by the end user of the software. Unlike alpha testing, the developer is generally not present. Therefore, the beta test is a “live” application of the software in an environment that cannot be controlled by the developer. The customer records all problems that are encountered during beta testing and reports these to the developer at regular intervals. As a result of problems reported during beta test, the software developer makes modification and then prepares for release of the software product to the entire customer base.

**System Testing and Acceptance Testing**

System testing is actually a series of different tests whose primary purpose is to fully exercise the computer-based system. Include recovery testing during crashes, security testing for unauthorized user, etc.

Acceptance testing is sometimes performed with realistic data of the client to demonstrate that the software is working satisfactorily. This testing in FDAC focuses on the external behaviour of the system.

**Conclusion**

A new approach of plant classification based on leaves recognition is proposed in this paper. An efficient machine learning approach for plant leaf recognition is presented in this research. The approach consisted of three phases namely the preprocessing phase, feature extraction phase and the classification phase. The computer can automatically classify 32 kinds of plants via the leaf images loaded from digital cameras or scanners. 12 commonly used Digital Morphological Features (DMFs) obtained from 5 basic features are extracted in the feature extraction phase. SVM classifier is adopted for the classification approach as it has better accuracy, fast training speed and simple structure. 12 features are extracted and processed by PCA to form the input vector of SVM. The performance of the proposed approach is evaluated based on the accuracy and execution time. For Further research by incorporating efficient kernel functions the performance of the classifier can be improved.

References

] Jyotismita Chaki, and Ranjan Parekh, “Plant Leaf Recognition using

Shape based Features and Neural Network classifiers,” International

Journal of Advanced Computer Science and Applications (IJACSA),

2011, vol. 2, no. 10.

[2] J. Pan, and Y. He,”Recognition of plants by leaves digital image and

neural network,” International Conference on Computer Science and

Software Engineering, 2008, vol. 4, pp. 906 – 910.

[3] Cotton Incorporated USA, The classification of Cotton, 2005,

http://www.cottoninc.com/ClassificationofCotton.

[4] National Institute for Agricultural Botany, Chrysanthemum Leaf

Classification. Cambridge, 2005.

[5] N. Kumar, S. Pandey, A. Bhattacharya, and P.S. Ahuja, “Do leaf surface

characteristics affect agro bacterium infection in tea,” J. Biosci., vol. 29,

no. 3, 2004, pp. 309–317.

[6] Ji-Xiang Du,De-Shuang Huang, Xiao-Feng Wang, and Xiao Gu,

“Computer-aided plant species identification (capsi) based on leaf shape

matching technique,” Transactions of the Institute of Measurement and

Control, vol. 28, 2006, pp. 275-284.

[7] Y. Li, Q. Zhu, Y. Cao, and C. Wang, “A Leaf Vein Extraction Method

based on Snakes Technique,” Proceedings of IEEE International

Conference on Neural Networks and Brain, 2005.

[8] H. Fu, and Z. Chi, “Combined thresholding and Neural Network

Approach for Vein Pattern Extraction from Leaf Images,” IEEE

Proceedings-Vision, Image and Signal Processing, 2006, vol. 153, no. 6.

[9] J.-X. Du, X.-F. Wang, and G.-J. Zhang, “Leaf shape based plant species

recognition,” Applied Mathematics and Computation, 2007, vol. 185.

[10] F. Gouveia, V. Filipe, M. Reis, C. Couto, and J. Bulas-Cruz, “Biometry:

the characterization of chestnut-tree leaves using computer vision,”

Proceedings of IEEE International Symposium on Industrial Electronics,

1997.

[11] J. Shlens, “A tutorial on principal component analysis”, 2005. [Online].

Available: http://www.cs.cmu.edu/\_elaw/papers/pca.pdf.

[12] Vladimir N. Vapnik, “The Nature of Statistical Learning Theory” New

York: Springer-Verlag, 1995.

[13] N. Cristianini, and J. Shawe-Taylor, “An Introduction to Support Vector

Machines”, Cambridge University Press, 2000.

[14] Xiao Gu, Ji-Xiang Du, and Xiao-Feng Wang, “Leaf Recognition Based

on the Combination of Wavelet Transform and Gaussian Interpolation,”

Advances in Intelligent Computing, vol. 3644, 2005, pp. 253-262.

[15] Xiao-Feng Wang, Ji-Xiang Du, and Guo-Jun Zhang, “Recognition of

Leaf Images Based on Shape Features Using a Hypersphere Classifier,”

Advances in Intelligent Computing, vol. 3644, 2005, pp. 87-96.

[16] http://flavia.sf.net.

[17] Stephen Gang Wu, Forrest Sheng Bao, Eric You Xu, Yu-Xuan Wang,

Yi-Fan Chang, and Qiao-Liang Xiang, “A Leaf Recognition Algorithm

for Plant Classification Using Probabilistic Neural Network,” IEEE

International Symposium on Signal Processing and Information

Technology, 2007.