# [Team 35] Leaf Wilting Detection in Soy Bean

Nikhil Sundaraswamy
*nsundar*
*200314268*

Shreyas Chikkballapur Muralidhara
*schikkb*
*200314024*

Chintan Gandhi
*cagandhi*
*200315238*

## I. Motivation

Weather stress and climate change are major challenges for agriculture. Drought tolerant soybean varieties are needed but breeding for drought tolerance in soybean, is difficult. The most widely used indicator of drought tolerance/sensitivity in soybean is leaf wilting during periods of water stress. We propose a CNN model which will allow farmers to gain information about these plants using images collected.

## II. Methodology

The architecture of our model is shown in Fig 1. The model first takes an RGB image an input. This image is converted into a grayscale image first. Next, the image is flattened for extracting one dimensional feature vector. We then apply Principal Component Analysis (PCA) on the data, over which we apply Support Vector Machine(SVM) for categorical classification.

We use grayscale image as an input the model, since features repeat across the channels, it is computationally expensive and redundant. A single channel input becomes computationally feasible and easy to handle.

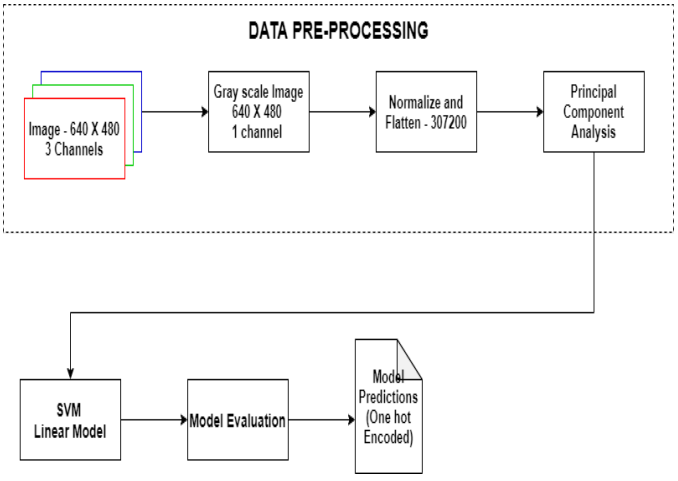



Fig. 1. Model Architecture

We use a flatten function to flatten the two dimensional input to a one dimensional feature vector. Classic machine learning approaches like SVM, require a one dimensional input. Our input image will have a dimension 480 X 640. The feature vector output of flatten function will now be of length 307200, which is now input for our classifier.

All features extracted from the image may not be useful. In other words, our classifier watches out for those feature vectors that show a particular amount of variability. Out of 307200 feature vectors, not many are significant. Thereby we perform PCA to reduce its dimension while retaining as much variance as possible in the dataset.

Since we are performing a multi-class classification, we want small mis-classification rate. A reliable algorithm in such a case would be Support Vector Machine. SVM is easier to train and take less memory as compared to a neural net in general. Support Vector Machine technique was firstly proposed for classification and Regression tasks by Vapnik [2]. SVM originated from the idea of structural minimization We build 5 binary classifiers for this project. The main idea behind such an approach is that the algorithm constructs a hyper-plane as a decision surface in such a way the margin of separation between different categories is maximised.

The toolbox and libraries along with the specific functions used are listed below:

TABLE I
Libraries and Functions

| **Libraries** | **Function names** |
|---|---|
| numpy | save, savetxt, bincount |
| pandas | read_csv |
| opencv | imread |
| scikit-learn | decomposition.PCA, svm.SVC, metrics.confusion _matrix, metrics.classification_report, model_selection.train_test_split |
| keras.utils | to_categorical |
| matplotlib | pyplot.plot, pyplot.savefig |
| pickle | load, dump |

## III. Model Training and hyper-parameter selection

### A. Model Training

The model was developed using the architecture explained in Section II. The images were read into memory, normalized and flattened to form a long 1-D vector. The vectors from all the images were appended to the data matrix. The data matrix vectors were shuffled and divided into training and validation vectors with a stratified train-test split of 80-20. The training

data matrix contained 820 vectors of length 307200 while the validation data matrix contained 205 vectors of length 307200.

We perform PCA on the training data vectors and reduce the dimensions from 307200 to 800. This reduction in vector length greatly helps in reducing the training time for the machine learning and model and saves up on memory while retaining most of the variance in the data. We train a Support Vector Machine model on the PCA transformed training vector matrix. We transform the validation data matrix using the fit PCA model, predict the classes using the trained SVM model, compute evaluation metrics and generate the confusion matrix.

### *B. Hyper-parameter tuning*

The images are loaded into memory and the dimension of the vector is reduced using PCA. The first hyper-parameter is the number of components to be retained in PCA.

| No. of retained components | % of explained variance |
|---|---|
| 500 | 89.74 |
| 600 | 94.24 |
| 700 | 97.68 |
| 800 | 99.85 |

We know that the maximum number of principal components is the minimum of number of training samples and number of features and hence, the maximum number of principal components are 820. We see that selecting more than 600 principal components captures approximately 95% variance in the dataset. Since, our original number of features are 307200, we select the top 800 principal components as they capture 99% variance.

.

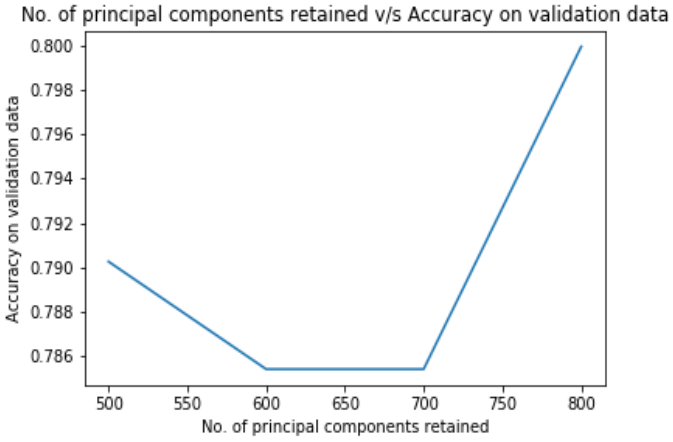
After reducing the number of features to 800, we train a SVM model for the training data vectors. We perform a grid search on the SVM parameters of kernel and regularization parameter.

| C\kernel | Linear | RBF | Poly |
|---|---|---|---|
| **0.01** | 84.39% | 47.80% | 47.80% |
| **0.1** | 84.39% | 47.80% | 47.80% |
| **1** | **84.39%** | 59.51% | 53.17% |
| **10** | 84.39% | 68.78% | 80% |
| **100** | 84.39% | 68.78% | 84.87% |

We use linear, RBF and polynomial kernels to transform the data and give different regularization strengths in the SVM. From the table it is clear that the best accuracy is achieved using the linear kernel. For simplicity, we will keep the regularization parameter C=1. The final model is a linear kernel SVM model with C=1.

## IV. Evaluation

Our final model is a SVM model with linear kernel and regularization parameter C=1. We split the data into training and validation in a stratified fashion and we achieve a validation accuracy of 79.02%.




The confusion matrix for the prediction looks like:

| Wilting Levels | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| **0** | 73 | 15 | 5 | 4 | 1 |
| **1** | 3 | 28 | 5 | 1 | 0 |
| **2** | 0 | 1 | 23 | 2 | 0 |
| **3** | 1 | 0 | 3 | 21 | 1 |
| **4** | 0 | 0 | 1 | 0 | 17 |

We know that our dataset has class imbalance with class 4 being the most under-represented. Hence, it is not enough to simply measure accuracy as our model can just predict the majority class and still get a high accuracy. Hence, we generate the precision, recall, F-1 score values for individual classes along with the macro and weighted average which are specified in the table below.

| Wilting Class | Precision | recall | f1-score | support |
|---|---|---|---|---|
| **0** | 0.95 | 0.74 | 0.83 | 98 |
| **1** | 0.64 | 0.76 | 0.69 | 37 |
| **2** | 0.62 | 0.88 | 0.73 | 26 |
| **3** | 0.75 | 0.81 | 0.78 | 26 |
| **4** | 0.89 | 0.94 | 0.92 | 18 |
| **accuracy** | | | 0.79 | 205 |
| **macro avg** | 0.77 | 0.83 | 0.79 | 205 |
| **weighted avg** | 0.82 | 0.79 | 0.80 | 205 |

## References


[1] Pujari, Devashish, Rajesh Yakkundimath, and Abdulmunaf S. Byadgi. "SVM and ANN based classification of plant diseases using feature reduction technique." IJIMAI 3, no. 7 (2016): 6-14.

[2] Vapnik, Vladimir. The nature of statistical learning theory. Springer science business media, 2013.

[3] Acır, Nurettin. "A support vector machine classifier algorithm based on a perturbation method and its application to ECG beat recognition systems." Expert Systems with Applications 31, no. 1 (2006): 150-158.