# Improving Soybean Disease Prediction by Performing Late-Stage Re-Training Using Fireworks Algorithm

*Abstract*— **This paper proposes a novel method of training and optimizing a model to get as optimal a result as possible. Contemporary models emphasize bigger dataset sizes and better pre-processing. This study aims to shed light on alternative methods of using multiple optimizers with different properties to get better results. This incremental increase is achieved by switching the optimizer after training once, hence performing late-stage re-training.**

**The present study shows the use of 2 distinct optimizers for their unique properties to achieve Late-Stage Re-Training (LSRT). Here, a different optimizer is used to tune our parameters (i.e., "Re-Training") after a model has achieved a certain level of accuracy using another optimizer.**

**Existing methods of creating ANNs are utilized by using the APIs within Tensorflow and Keras. Furthermore, the Tensorflow Optimizer class is used to create our swarm algorithm within Tensorflow.**

**The source code can be found at: https://github.com/angadsinghsandhu/fireworks**

*Keywords*—**Optimizers, Fireworks Algorithm, Adam, Soybean Dataset Classification, Artificial Neural Network, Artificial Intelligence, Machine Learning**

## I. Introduction

The growing human population needs more and more resources each year. One of the key resources is food and grain. The growth demand thus means an increase in crop demand. Plant diseases and infections largely cause agricultural losses around the world. Plant ailments, weeds, insects, and chemicals are factors resulting in a steady decline in the worldwide production of crops. It is therefore extremely necessary to detect diseases and prevent their spread in order to curb losing crops. Agriculture has adopted cultural practices to protect crops from hazards—usually combined with crop rotation and solarization—to provide preservation from fatal pests to an extent. Practices also include the development of crop varieties that resist pests, as well as the use of biological instruments.

A healthy agroecosystem is dependent upon the effective detection of diseases and ailments. With steady development in the fields of molecular biology and biotechnology, detecting plant diseases has become more efficient. An automatic recognition system is required for the prevention and control of Soybean diseases. The obvious goal is to maximize yield and minimize economic losses, which can, in turn, be achieved by curbing pesticide residue on the land and improving crop quality. It is imperative to organize and categorize the diseases efficiently to predict Soybean diseases early. In order to predict diseases, pest attacks, and infestations in crops, numerous learning algorithms have been used. Using spectral imaging data, an algorithm has been developed to compare aerial parts of healthy and diseased plants. Aside from observing the crops' morphological traits, there is considerable evidence to show that Machine Learning methods are successful for disease detection as well.

| | |
|---|---|
| Diaporthe Stem Canker | Charcoal Rot |
| Rhizoctonia Root rot | Phytophthora rot |
| Brown Stem rot | Powdery Mildew |
| Downy Mildew | Brown Spot |
| Bacterial Blight | Bacterial Pustule |
| Purple Seed Stain | Anthracnose |
| Phyllosticta Leaf Spot | Alternarialeaf Spot |
| Frog Eye Leaf Spot | Diaporthe Pod and Stem Blight |
| Cyst Nematode | 2 4 d Injury |

Fig 1: Soybean Diseases

The diverse nature of crop biology, however, may lead to faulty predictions owing to variations in symptomatology. Therefore, disease identification based on appearances alone cannot be trusted to identify diseases correctly, particularly at the inception of growth. It is necessary to have appropriate methods of disease detection to correctly identify the responsible agents of decay because symptoms do not emerge until the crops' mid-season. The methodology for predicting the occurrence of charcoal rot disease used in the present study includes the plants' morphological characteristics (for instance, traits that are connected to growth and yield) along with their physiological characteristics. A hybrid set of features derived from both healthy and diseased Soybean plants are used to train and assess machine learning algorithms.

The present study applies machine learning, to discern healthy plants from those that are affected in some manner. It is also seen that for disease classification modelling, supervised machine learning algorithms may be used. These algorithms can be utilized for the classification of diseases. The aim of this study is to present a feature set to enhance the prediction of Soybean diseases. Crop yield is significantly impacted by infestations and diseases. To optimize the yield, early recognition is essential to keep crop diseases in check.

Soybean is categorized as a 'Kharif' crop and is a swiftly growing crop in India. The largest growth of Soybean in India is done in the states of Madhya Pradesh, Maharashtra and Rajasthan [2]. Worldwide, Soybean is considered a 'wonder' crop. Soybean is an essential food commodity owing to its elevated nutritional value (especially protein, which is present at levels greater than 38%) and high presence of oil (more than 20%). As the soya has enough protein to provide important amino acids, it is also known as a complete protein. There is no cholesterol in soybeans and it is also used as animal feed. Low yield is a major issue in the country's soya industry; crop diseases are one of the leading causes of such scarcity.

Diseases such as downy mildew, pod, stem blight, brown spot, Cercosporin leaf blight, purple seed stain, phytophthora root, stem rot, and frog eye leaf spot, amongst others can fatally infect Soybeans. The present study handles the categorization of diseases that affect soybean crops based on weather data, physical and other plant properties, and crop management strategies. The dataset is accessible in the UCI Machine Learning Repository. ML algorithms such as ANNs will be used for this.

## II. Literature Review

Separately, a lot of research about the classification of Soybean as well as various diseases associated with it and the fireworks algorithm has been done; some of these studies are mentioned below.

Elham et al. [10], as well as the subsequent work done by Nanda and Uday [1], show the empirical analysis and comparison between a variety of Machine Learning Techniques (K nearest neighbours, naive Bayes, decision tree, neural network algorithms, etc.). After choosing the preferred parameters through feature selection and then processing the data, it was put through a variety of machine learning techniques. This finally tells us that the best classical alternative among them is Gradient Boosting Tee (GBT) which is built using decision trees, giving an average accuracy of close to 96.13%. In comparison, Rajashree Krishna and Prema K V [2] used techniques such as Multi-layer Perceptron, Naive Bayes, Gaussian, and Bayesian Classifiers to further expand the work done on Soybean disease prediction.

Apart from classification based on values, there are a multitude of studies done on the classification of Soybeans and their corresponding diseases with the use of images. Shuang Liu et al [11] propose a method for classifying soybean frogeye leaf spot (FLS) by first building a dataset from collected leaf images and hyperspectral reflectance data of healthy and FLS diseased soybean. After cleaning and selecting attributes (spectral index (SI), principal component analysis (PCA), and competitive adaptive reweighted sampling (CARS)), the data was passed through SVM classifiers and Prediction Models, giving an average overall classification accuracy of 97.3%. Similarly, Robert W. Bruce et al [12] use an SVM radial basis function (RBF) classifier to group ariel images of crops to their analogous class.

Another field that is paramount in the working of the present study is the existing research on the Fireworks Algorithm.

In many studies, it was found that researchers have attempted to compare their results using other statistical tools. Each approach has its advantages and downsides; one study can use them to reduce the disadvantages of another. These findings come with a strong motivation for modelling predictable tools to predict soybean diseases. In addition to applying wavelet-based data pre-processing, this work uses the FWA to optimize the intermediate values of the ANN to enhance the accuracy of obtained results of the soybean disease prediction.

## III. Methodology

At the beginning of the present study, the idea of late-stage training came from the need to create more accurate models using an amalgamation of presently used technologies to create a procedure to train a model that is better than the individual sum of its parts. Most of these models are essential in fields where accuracy or efficiency may be paramount and extract actionability [4]. Such models are used in sectors such as Healthcare, Finance, Automobiles, etc., where the Bayes error rate is minuscule (close to or less than 5%). During the first round of training, making big waves in accuracy and loss metrics might be more convenient than closing that last 10% gap. This is where the finding of the present study comes in.

The architecture to best experiment on such a method is best suited to be a classification model, a technology whose implementations are quickly permeating through modern life. Hence, for the present study, the `Soyabean Disease` Dataset [5][6] was chosen due to its large number of classes (~19).

Another critical step of this study was to choose a pair of optimization algorithms that might work well together for two different parts of the training process. The first part of the training is optimized using the Adam Optimizer [7] for its quick convergence and the use of momentum; conversely, the second part of the training uses an implementation of the swarm intelligence algorithm, that is, the Fireworks Algorithm.

The following study discusses and presents the outcomes of this idea.

When deciding what algorithm to choose for the optimization step during the training of neural network models, there is also a need to take time into consideration. Thus, the present study considers numerous factors regarding training time and the best fit in terms of the data. However, it can also be intuitively understood that different sections of training can benefit most from different optimization techniques.

### A. Dataset

The soybean dataset is retrieved using the 'pmbl' library, where the data has more than 30 columns (such as: 'date', 'plant-stand', 'precip', 'temp', 'hail', 'crop-hist'). This data is imported, converted into a data frame, and then the necessary splits (X, Y, test, train) are made.

### B. Fireworks & Creating Custom Optimizer

Using the built-in Optimizer class of TensorFlow, the present study uses a custom-created implementation of the Fireworks Algorithm as a new class that can be called inside the main driver code. To create an optimizer function, an Object Function is needed to create and evaluate the sparks array inside the Fireworks class that is used to initialize new possible optimal values. The Object Function and the number of dimensions and maximum iterations are taken as parameters.

Fireworks is a swarm intelligence algorithm that uses 'explosions' to semi-randomly find out the next optimal step in our solution space. It is a great method to get out of any local minima that the study might have gotten trapped into due to the linear descending structure of Adam.

When fireworks itself is used as the only optimizer used during training, it does not give appropriate results, nearing only a measly 60% train accuracy on average and therefore being much behind industry-standard results that are generated from legacy algorithms. On further inspection, however, the present study finds that after the initial burst, the rate of increase of the 'train' accuracy decreases exponentially. This makes Fireworks a great secondary algorithm, while it performs poorly as a primary optimization algorithm.

In the present study, first, the Soybean classifier is trained using the Adam Optimizer, which has been used in several implementations for the solution of this problem statement. Nevertheless, as the algorithm gets trapped in a local minimum, the Firework algorithm is called upon to use built-in sparks to train further the model for better results for both the training and testing data.

Ying Tan [4] introduced a new iteration on the effective swarm intelligence algorithm, i.e., the

Fireworks Algorithm (FWA). FWA is a method through which intelligent classes work in collaboration with each other; similar to fireworks in the night sky, FWA is a technique that simulates the act of a firework explosion. The algorithm is normally used for optimization tasks. This is done by obtaining the optimum value inside the solution space of problems of specific types.

After being shot into the sky, an explosion of these fireworks occurs. Moving into a suitable position, they explode into radiant lights of various shapes, which is dependent upon the fireworks' design.

Each firework has specific characteristics—for instance, there can be a tapering explosion with a lot of sparks, a vast explosion with lesser sparks, a vast explosion with many sparks, and more. Every firework is defined by a unique explosion and the various positions at which they explode. In this instance, FWA as an algorithm is used to train ANNs and to determine the appropriate biases and weights. These values are taken from processing the error difference between the predicted and real results. This algorithm works in a repeating cycle until a fitting result is obtained. The tasks done in the present study is divided into four major parts, as elaborated in the Methodology section.

***Algorithm: Pseudocode of FWA***

```
Take in Dataset as Input
Filter and Perform Normalization
Create train and test split sets
Init ANN and FWA parameters (with max training cycles)
For the first iteration, generate random Fireworks
For i=1 to the max fireworks do
    Compute Err;
End for
Init from the optimal location
For j=1 to the max training cycles do
    For i=1 to the max fireworks do
        Compute amplitude;
        Compute the number of sparks;
        Derive regular sparks;
    End for
    Derive Gaussian Sparks;
    Select sparks for the next firework;
    Derive new fireworks;
End for
Return the optimal weights and biases values
```

*C. Training Model*

Now, the process of creating and training the model begins. As this happens, two instances of optimizers are created—one being Adam and the other being Fireworks. A sequential model is created with three dense layers with 56, 28, and 19 nodes, respectively.

Then, a new TensorFlow session is started, followed by compiling, training, and finally evaluating the results and data.

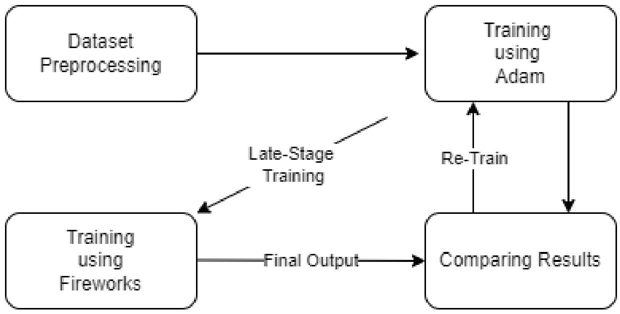



Fig 2: Methodology Flow

*D. Limitations & Alternatives*

One of the known drawbacks of the Adam optimizer is that in some scenarios, Adam does not converge to the optimal solution. Also, slow convergence and low accuracy are some major issues with the Fireworks Algorithm. But if these operations are used in tandem, they proceed to rectify each other's drawbacks, i.e., The initial training that is done using Adam provides a quick and accurate starting point. Further passes using Fireworks takes care of any local minimums and brings the solution to the global minimum.

This approach still has several issues, such as the Weight Decay problem in Adam and Explosion Tuning in Fireworks. These problems can be circumvented using alternative versions of these algorithms themselves, i.e., using the dynamic search firework algorithm (dynFWA) and AdamW or AMSGrad.

Furthermore, the pair of optimizers used in the present study might not be the best in terms of all-rounded use with other datasets as well. This makes the process of choosing the appropriate pair of optimizers more difficult.

## IV. Results and Discussion

| Operation | Train Accuracy | Test Accuracy |
|---|---|---|
| 1st Pass (Adam) | 94.47% | 90.58% |
| 2nd Pass (Adam) | 97.12% | 89.69% |
| 3rd Pass (Fireworks) | **98.89%** | **92.03%** |

Fig 3: The results of the 3-step training

In Fig. 3, the study shows that during training, the following processes take place:

### A. *First Pass (using Adam)*

During the first training pass, the model accuracy begins from around ~40% accuracy in the first epoch; this increases to nearly ~94.5% by the last epoch. The speed and accuracy of Adam are utilized to get a well-trained model as the starting point.


Fig 4: The results of the First pass of the training

### B. *Second Pass (using Adam)*

In this next phase of the training, in the current study, the `model.fit()` function is run again to re-train the existing model on the dataset using Adam once more. This operation gives a slightly better train accuracy but nearly the same test accuracy (if not lower). Subsequently, no matter how many times the model is re-trained, the test accuracy does not budge.

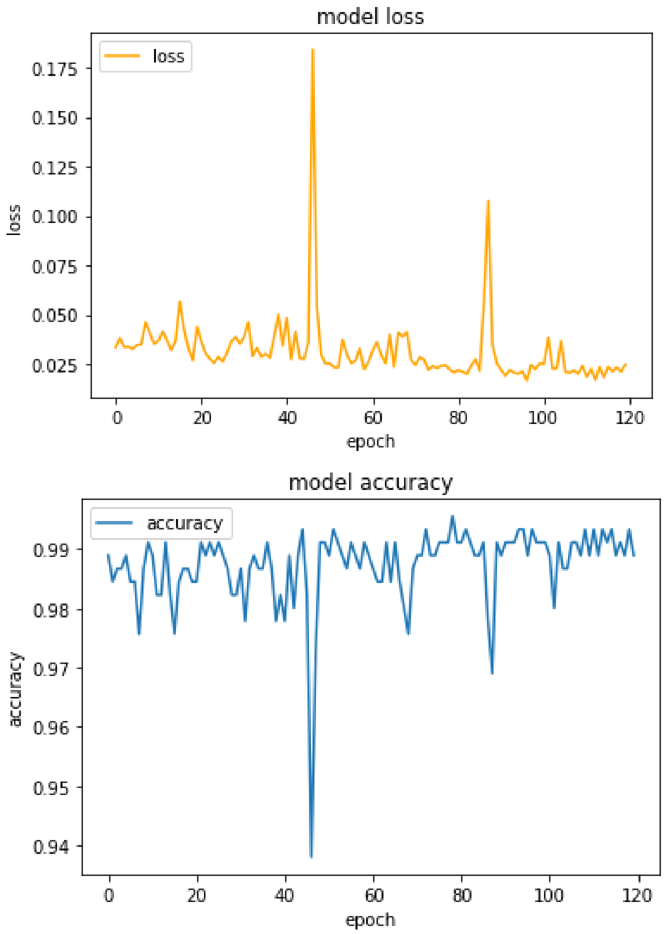
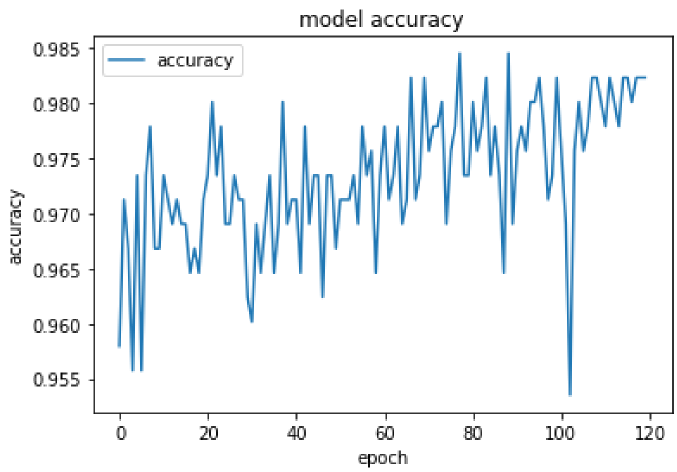



Fig 5: The results of the second pass of the training

### C. *Third Pass (using Fireworks)*



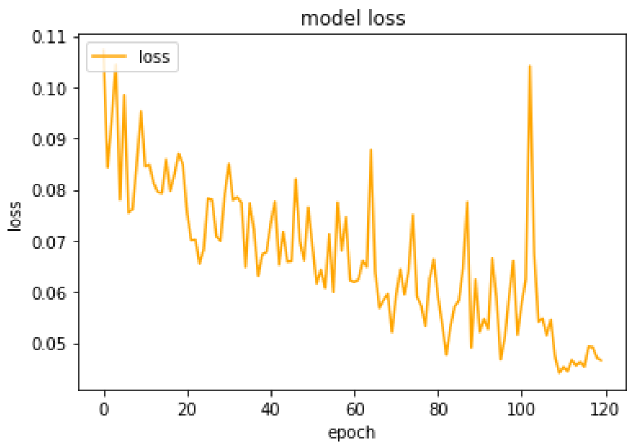


Fig 6: The results of the Third pass of the training

In the next phase of the training, Fireworks is now used as the optimizer, and the model is re-trained. After training, there is an equivalent increase in both the test and train accuracy of the dataset. While with Adam, the accuracy and loss were stagnant, Fireworks provides the last stage of training to get better results.

## V. Conclusion and Future Work

In conclusion, a new way of getting better results was discovered on both our test and train data with the help of the hybrid implementation of optimizers. Not only does the train accuracy increase after initially plateauing while using the original optimizer but the test accuracy increases by a huge margin after initially re-training with the Adam optimizer. Hence, using the Fireworks Algorithm and, therefore, LSRT (Late-Stage Re-Training), the present study was to create a model that was ~4% more accurate on train data and nearly ~2% on test data.

In the future, the focus will be on more hybrid optimization combinations and further research on consequent training. Furthermore, an optimizer can be created to use the multiple initializations of a swarm algorithm with the concept of momentum, alpha, and beta from traditional gradient descent algorithms.

The hypothesis provided by the current study needs to experiment on further datasets that need LSRT (Last-Stage Re-Training), such as COVID-19 prediction classification problems.

## VII. References

[1] [ Dr. Nanda Ashwin, Uday Kumar Adusumilli. ] A Machine Learning Approach to Prediction of Soybean Disease, ( ISSN: 2394-4099 )

[2] [ Rajashree Krishna, Prema K V. ] Soybean crop disease classification using machine learning techniques, ( INSPEC Accession Number: 2027-9158 )

[3] [ Saktaya Suksri, Warangkhana Kimpan. ] Neural Network Training Model for Weather Forecasting Using Fireworks Algorithm, ( INSPEC Accession Number: 16692911 )

[4] [ Qiang Lyu, Yixin Chen, Zhaorong Li, Zhicheng Cui, Ling Chen, Xing Zhang, Haihua Shen. ] Extracting Actionability from Machine Learning Models by Sub-optimal Deterministic Planning ( arXiv:1611.00873 [cs.AI] )

[5] [ Tan, M., & Eshelman, L. ] Using weighted networks to represent classification knowledge in noisy domains. ( Proceedings of the Fifth International Conference on Machine Learning (pp. 121-134). Ann Arbor, Michigan: Morgan Kaufmann. )

[6] [ Fisher, D.H. & Schlimmer, J.C. ] Concept Simplification and Predictive Accuracy. ( Proceedings of the Fifth International Conference on Machine Learning (pp. 22-28). Ann Arbor, Michigan: Morgan Kaufmann. )

[7] [ Diederik P. Kingma, Jimmy Ba ] Adam: A Method for Stochastic Optimization. ( arXiv:1412.6980 [cs.LG] )

[8] [ Haoran Lu, Weidi Xu, Ying Tan ] A Discrete Fireworks Algorithm for Solving Large-Scale Travel Salesman Problem. ( INSPEC Accession Number: 18147040 )

[9] [ Khuat Thanh Tung, Nguyen Thi Bich Loan ] Applying Artificial Neural Network Optimized by Fireworks Algorithm for stock price Estimation. ( ISSN: 2229-6956 )

[10] [ Elham Khalili, Samaneh Kouchaki ] Machine Learning Techniques for Soybean Charcoal Rot Disease Prediction. ( PMID: 33381132 )

[11] [ Shuang Liu, Haiye Yu ] Classification of soybean frogeye leaf spot disease using leaf hyperspectral reflectance ( PubmedID: 34478465 )

[12] [ Robert W. Bruce, Istvan Rajcan ] Classification of Soybean Pubescence from Multispectral Aerial Imagery. ( Article ID 9806201 )