Genetic Algorithms for Features Selection

Huiying Li

Abstract. Feature selections has been used to remove unneeded or unimportant features before training. Genetic algorithm is one of the techniques to find out the optimal combination of features that will increase prediction performance. According to the experiments results, using the optimal selected features combination from the genetic algorithm can increase the network's prediction accuracy. Moreover, another befit of the genetic algorithm is that it doesn't need any specific knowledge about the data set to do features selection. The cost to pay, however, is the expensive computation cost on running the genetic algorithm.

Keywords: genetic algorithm, features selection, neural network

1 Introduction

With the increasing use of deep neural network in machine learning and the expanding size of data available for network training, it is getting more and more difficult to train a neural network. There are some techniques developed aiming to simplify the network training without scarifying the network's prediction ability, such as network pruning and training data features selection (Aziz, Azar, Salama, Hassanien, & Hanafy, 2013).

In this paper I am going to focus on genetic algorithm. Inspired by the natural evolution procedures, where the genes of organism will evolve over generations to better adapt to the environment, the genetic algorithm is an optimizer to produce better pool of solutions of a given problem (Gómez & Quesada, n.d.).

In this paper I am going to experiment the effect of genetic algorithm for features selection, which will help removing unneeded or redundant features from the training data set (Li, 2006). I am going to compare the prediction results of features selected with different set of evolution hyperparameters, such as, the number of generation and the size of population, and with different fitness measurements, such as training loss as fitness and prediction accuracy as fitness. The genetics algorithm is written in Python.

The data set I used in this paper is named 'The Forest Cover Type dataset' from the UCI Machine Learning Repository. The data set has 581012 instances and 12 features to descript each instance.12 features are extended to 54 features in the data set due to 44 binary representations of 2 features. So, the actual feature size is 54. It is a

classification data set where data are classified into 7 classes or cover types and the 12 features are quantitative without any missing or null value. The features include: Elevation, Aspect, Slope, Horizontal Distance To Hydrology, Vertical Distance To Hydrology, Horizontal Distance To Roadways, Hillshade 9am, Hillshade Noon, Hillshade 3pm, Horizontal Distance To Fire Points, Willderness Area (4 binary columns) and Soil Type (40 binary columns).

The network I used for experiments is a three-layers fully-connected neural network, which is consist of an input layer, one hidden layer and an output layer. The three-layer network is trained for forest cover type classification by feedforward training and back-ward propagation weights updating. The loss function used in network training is cross entropy and the optimizer used is Adam. The network is built in Python Pytorch.

This paper can be divided into three sessions, where session 1 will detail the implementation steps of genetics algorithm, session 2 will demonstrate the results of the experiments followed by the discussion of the results and session 3 will conclude the experiments and give suggestion considering potential improvements.

2 Method Implementation

2.1 Data Set Preparation and Data Preprocess

In the experiments, the original data set is divided into two parts: 10% of the data, named GA_data, for features selection and 90% of the data, named NET_data, for normal network training and testing.

I further divided the GA_data into two parts. One data set contains 80% of the data from the GA_data and will be used for model training in feature selections. Another data set contains 20% of the data from the GA_data and will be used for model testing in feature selections.

The NET_data will be also divided by ratio of 8/2. 80% of the data from NET_data will be used for network training and 20% for testing.

All the features data from the original data set are normalized in the range of 0 and 1.

Forest Type	Number of Records
Spruce-Fir	21184
Lodgepole Pine	283301
Ponderosa Pine	35754
Cottonwood/Willow	2747
Aspen	9493
Douglas-fir	17367
Krumholz	20510
Other	0
Total records	581012

Table 1. Classification distribution of the forest cover type data set

Data Set Type	Number of Records	Percentage
GA Training Set	46481	10% * 80% = 8%
GA Testing Set	11620	10% * 20% = 2%
NET Training Set	418329	90% * 80% = 72%
NET Testing Set	104582	90% * 20% = 18%
Total	581012	100%

Table 2. Number of records in GA training set, GA testing set, NET training set and NET testing set

2.2 Network Structure and Hyperparameter Setting

The neural network used in this experiment is a basic three-layer fully connected network which includes one input layer, one hidden layer and one output layer. It is a feedforward network trained with backward propagation. All the units in the network are linear connected with weights from the previous layer. The below table lists all the hyperparameter I have used to train the network.

Network Hyperparameters	
Number of epochs	20
Batch size	2000
Learning rate	0.05
Hidden units	120

Table 3. Network hyperparameters setting

2.3 Genetic Algorithm Implementation

In general, the genetic algorithm is consisted of five steps: population initialization, fitness calculation, population selection, parent genes crossover, child genes mutation. A stopping criterion to stop the evolution should be given, which is the number of generation.

1. Initialization
   The population is randomly initialized. The size of genes is depending on the number of the features from the training data set. In this case, the size of the genes is 54 given 54 features form the original 'Forest Cover Types' data set. Population size should also be given for population initialization. During the experiments, the population size I have tried are 5, 10, 20.

   The genes are represented by an array of 0s and 1s. Indexes of 1s denoted the indexes of the selected features while indexes of 0s denoted the indexes of the removed features.

2. Fitness assignment
   Fitness is used to measure how well a DNA adapt to the environment. In the cause of feature selections, fitness means how well a network perform when trained with the selected features.

   During the experiments, I have used two fitness measurements: loss and accuracy. The lower the loss the better the fitness. The higher the accuracy the better the fitness.

   When using loss as fitness, the loss return will be the loss of the last feedforward training epoch in the feature selection process. When using accuracy as fitness, the accuracy will be the prediction accuracy using the GA_data testing set with the trained model in the feature selection process.

3. Selection
   DNAs from the population are randomly selected according to the finesses. The higher the DNA's fitness, the bigger chance it will be selected for reproduction.

4. Crossover
   The selected DNAs will be randomly paired up for reproduction. New child genes will be produced in this step.

5. Mutation

To increase the level of diversity of the offspring generated from the last generation, mutation will randomly change the features in the offspring (Gómez & Quesada, n.d.). For example, the 0s in the DNA array will randomly change to 1s according to the mutation rate and vice versa.

6. Stopping Criteria
   Sopping criteria of the evolution of defined by the number of generations. In the experiments, I have tried: 5, 10, 20

Population Size	Number of Generations	Fitness Measurement
5	5	Loss
5	5	Accuracy
10	10	Loss
10	10	Accuracy
10	20	Loss
10	20	Accuracy
20	20	Loss
20	20	Accuracy

Table 3. Experiments parameters combinations

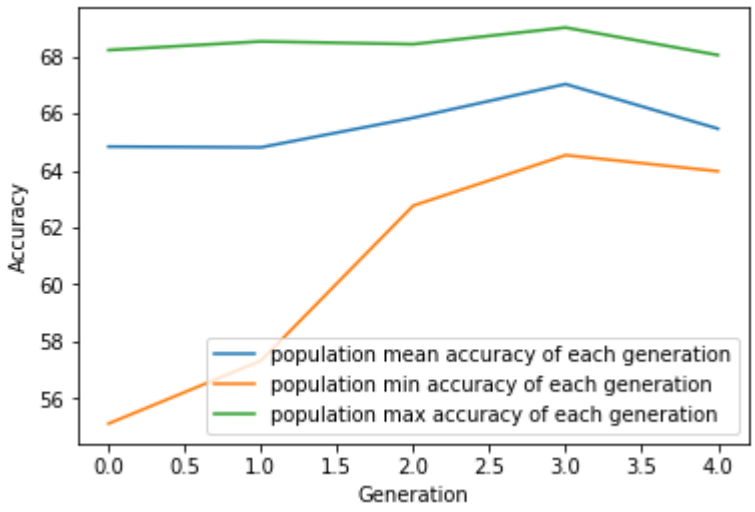
3 Experiment Results and Discussion

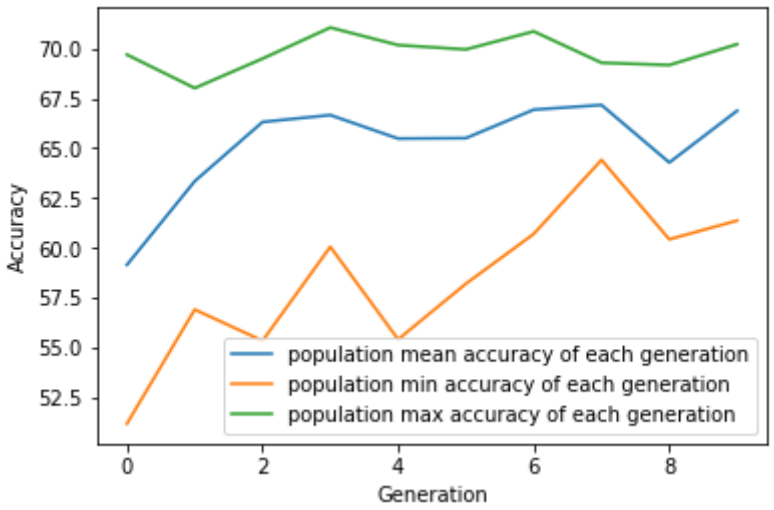
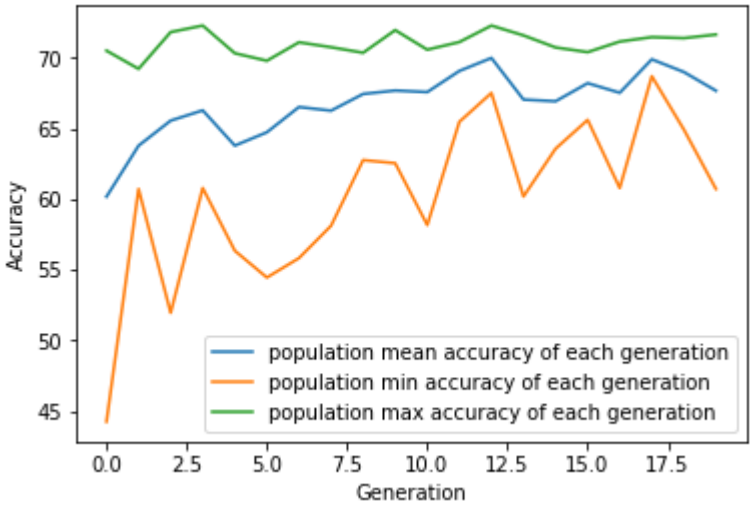
Base on the experiments results, I can tell that 1) comparing with the baseline model that trained without feature selection, feature selection with genetic algorithm can help increasing the network's prediction ability; 2) the higher the population size and the larger number of generation the genetic algorithm gets, the better the features selection it returns and the better the network perform in prediction; 3) using loss as fitness measurement is better than using accuracy. Below are the table and the plots of the results.

Population Size	Number of Generations	Fitness Measurement	Prediction Accuracy
0	0	Null	52.79 %
5	5	Loss	60.23 %
5	5	Accuracy	54.51 %
10	10	Loss	49.56 %
10	10	Accuracy	45.28 %
10	20	Loss	63.15 %
10	20	Accuracy	49.45 %
20	20	Loss	60.50 %
20	20	Accuracy	54.24 %

Table 4. Experiments prediction accuracy results

As the plots below shown, if accuracy is used as the fitness measurement, the generation mean accuracy and max accuracy will increase (although with fluctuations) as the number of generations increase. Comparing with the last two figures: 10*20 and 20*20, the latter plot has smoother mean accuracy line and max accuracy line, which means the larger the population size, the more stabilized the evolution will be. In other words, increasing the population size will help to decrease the existence of extremely bad features combinations (outlier) in a pool of good features combinations.

Pop_size * num_generation	Accuracy as fitness
5*5	

10*10	 70.0 67.5 65.0 62.5 60.0 57.5 55.0 52.5 Accuracy 0 2 4 6 8 Generation population mean accuracy of each generation population min accuracy of each generation population max accuracy of each generation
10*20	 70 65 60 55 50 45 Accuracy 0.0 2.5 5.0 7.5 10.0 12.5 15.0 17.5 Generation population mean accuracy of each generation population min accuracy of each generation population max accuracy of each generation

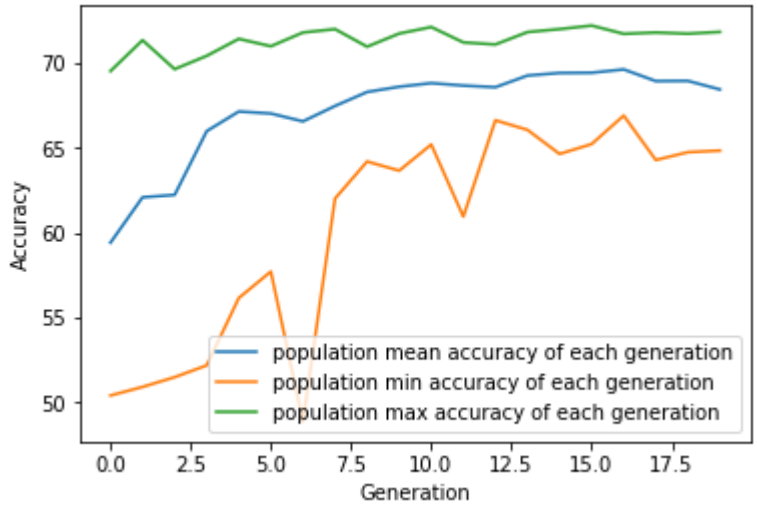
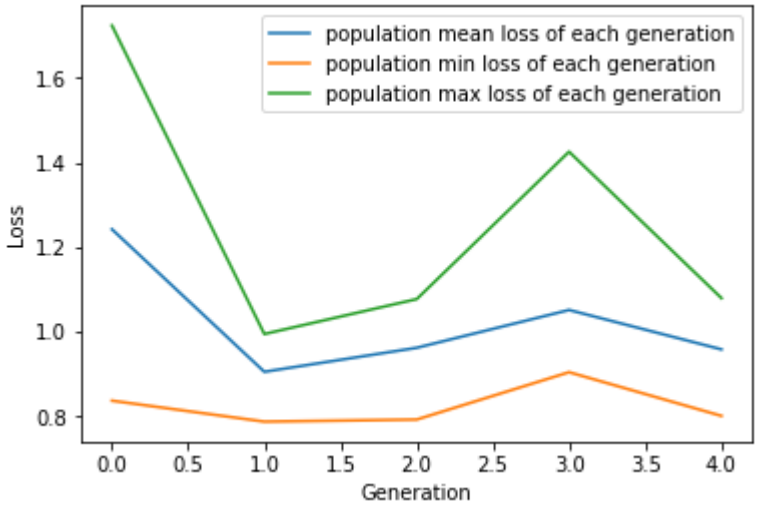
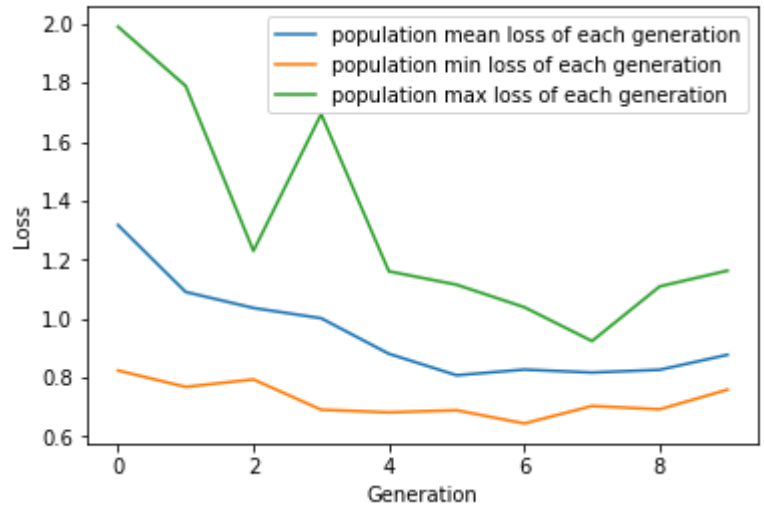
20*20	

Table 5. table of plots for the accuracy statistics change during the generation

As the plots below shown, if loss is used as the fitness measurement, the generation mean loss and min loss will drop before the number of generation reach 3 and will start to fluctuate after the 3rd generation. Same as using accuracy as fitness measurement, as the size of the population increase, the mean and min loss plots will be smother.

Pop_size * num_generation	Loss as fitness

5*5	 population mean loss of each generation population min loss of each generation population max loss of each generation Loss 1.6 1.4 1.2 1.0 0.8 0.0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0 Generation
10*10	 population mean loss of each generation population min loss of each generation population max loss of each generation Loss 2.0 1.8 1.6 1.4 1.2 1.0 0.8 0.6 0 2 4 6 8 Generation

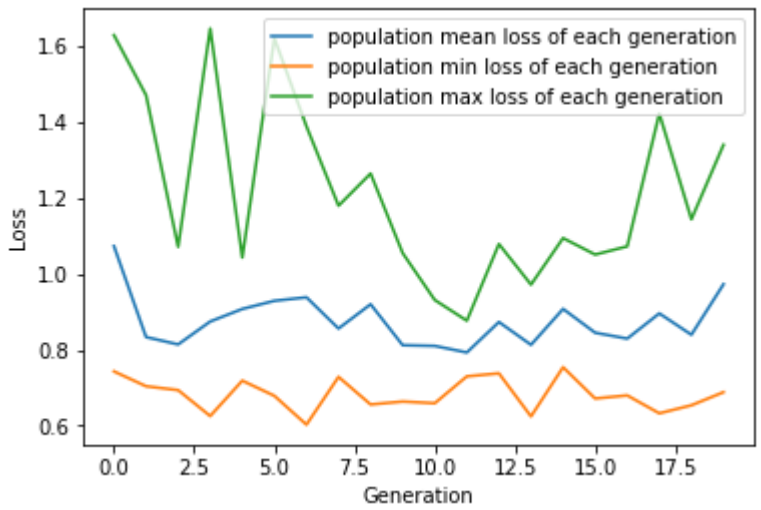
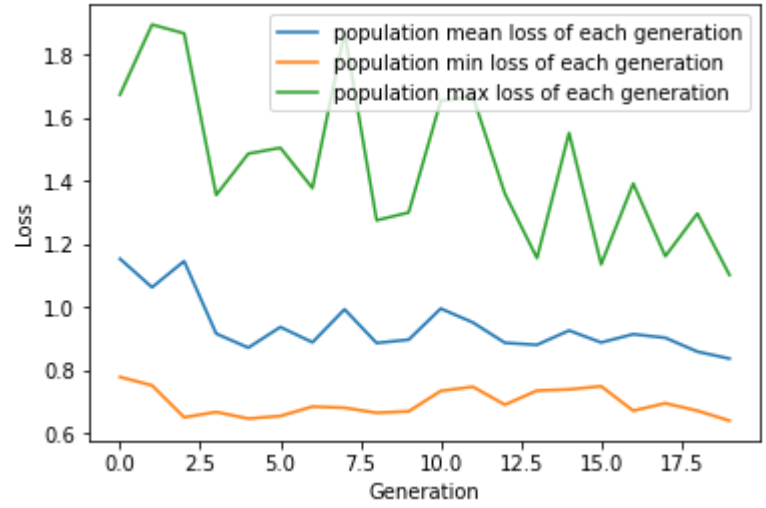
10*20	
20*20	

Table 6. table of plots for the loss statistics change during the generation

3 Conclusion and Improvement

To draw conclusion, the application of features selection by genetic algorithm on the 'Forest Types' helps to the model to better perform in prediction.

However, there are some limitations of the experiments I have done: 1) the experiments didn't consider the effect of crossover rate and mutation rate; 2) use of

small data set for feature selection may cause the problem of overfitting, which is the reason why losses are much lower and the accuracies are much higher during feature selection than the actual training and testing; 3) experiments with larger population size and number of generations are limited by the computation power. For example, I have try using population size of 50 * 50 generations and the features selection process takes more than 6 hours to finish; 4) limit use of loss and accuracy as fitness.

To improve the genetic algorithm on features selection, 1) tuning on mutation rate and crossover rate are worth trying; 2) more complex fitness measurement should be consider, such as Rank base fitness or the fitness will consider both loss and accuracy (Gómez & Quesada, n.d.); 3) putting larger portion of data in features selection can help to prevent overfitting.

References

Aziz, A. S., Azar, A. T., Salama, M. A., Hassanien, A. E., & Hanafy, S. E.-O. (2013). Genetic Algorithm with Different Feature Selection. *Federated Conference on Computer Science and Information Systems*, (pp. 769–774).

Gómez, F., & Quesada, A. (n.d.). *Genetic algorithms for feature selection in Data Analytics*. Retrieved from neuraldesigner: https://www.neuraldesigner.com/blog/genetic_algorithms_for_feature_selection

Li, T.-S. (2006). FEATURE SELECTION FOR CLASSIFICATION BY USING A. *Journal of the Chinese Institute of Industrial Engineers*, 55-64.