Searching for feature pre-processing methods in MLP binary classification with Genetic Algorithms

Jandrik Lana

February 22, 2022

Abstract

Feature pre-processing is an essential step for most machine learning algorithms. Pre-processing usually involves transforming the input data to a form that is more suitable for the learning algorithm. Various techniques can be used for pre-processing, such as feature scaling, feature extraction, and feature selection which help to improve the performance of the learning algorithm. This study develops novel pre-processing methods through a genetic programming approach. Genetic Algorithms were used to search for a combination of pre-processing operations that produced the best results of Multilayer Perceptrons on a set of binary classification datasets. The search findings show that these discovered methods, when combined with existing methods, statistically outperform existing methods by themselves on new datasets. Visualization of the effects its on synthetic data show that these discovered methods extend the range of the data and direct values away from the center of the data. This study provides practitioners with new methods that can be used as pre-processing techniques for machine learning algorithms.

1 Introduction

In machine learning, data is often pre-processed in order to improve the performance of a model or a learning algorithm. One of these pre-processing methods is feature scaling, a method used to scale the values of features in a dataset. Scaling is done because different features range differently, and the relative differences between each datapoint in a dataset may be affected by this, thus affecting a model's performance (Bollegala, 2017). Feature scaling also ensures that features with extreme high or low values do not have extreme or almost no effects on the outputs, respectively.

Features scaling also often increases the rate of convergence of gradient descent to a minimum in the loss function of neural networks (Wan, 2019). This is significant especially in larger tasks with more complex neural networks such as convolutional neural networks (CNNs) (Tan and Le, 2019, He et al., 2016), object detection (Zhang et al., 2018, Long et al., 2020) and Generative Adversarial Networks (Park et al., 2019, Yuan et al., 2018) which are trained on extremely large datasets (Deng et al., 2009, Lin et al., 2014).

For distance-based learning algorithms (e.g. Support Vector Machines, k-Nearest Neighbors, K Means Clustering), higher numerical values affect distances between data by dominating lower numerical values (Alshaher, 2021). Since such features in data as "*cell size*" or "*year*" may not always occupy the same range of values, higher or lower numerical values in data may not always mean larger or smaller effects. As such, feature scaling plays a significant role in the performance of the classifier.

Another pre-processing technique in machine learning is the transformation of features such that their probability distribution follows a Gaussian/Normal distribution. This is because features often have a skewed or unknown distribution. Although data following a Gaussian/Normal distribution does not always improve performance, it may give learning algorithms an easier fit to the data (Raymaekers and Rousseeuw, 2021).

Having established the importance of feature pre-processing in machine learning, an genetic programming approach is conducted to improve existing pre-processing methods or techniques by developing novel methods. This paper approaches this using genetic algorithms (GA) as a search method to discover pre-processing operations that can improve existing methods. Genetic algorithms, a class of evolutionary algorithms, have long been used as an optimization method for a variety of technical problems, including feature pre-processing and selection (Tan et al., 2014, Babatunde et al., 2014, Prathama et al., 2017b). In the context of this study, a genetic algorithm will be employed to search for a number of candidates of pre-processing operations that can be used to improve the performance of a machine learning algorithm. These operations that are discovered are evaluated and tested on multilayer perceptrons (MLPs) across a variety of binary classification datasets.

MLPs, also often referred to as fully-connected networks in many different neural network architectures, are widely used in the area of machine learning such as computer vision and natural language processing (NLP) as an essential part in their architectures. For example, fully-connected networks are employed after convolutional and pooling blocks in CNNs for image recognition (He et al., 2016), and in transformers for NLP as part of their encoder and decoder blocks (Vaswani et al., 2017). As such, this study will focus on MLPs and improving their performance through discovering novel pre-processing methods using GAs.

2 Background

2.1 Commonly used feature scaling and transformation methods

This section describes commonly used techniques to pre-process features in machine learning algorithms. These include methods of standardization, normalization, and techniques of transformation of data into a Gaussian distribution. The descriptions also describe Scikit-learn's (Pedregosa et al., 2011) implementations of these methods, if applicable, as Scikit-learn's implementations are used in this study's methods.

2.1.1 Z - Score Normalization

Z - score normalization, also referred to as Standard Scaling, scales a feature x using the mean and standard deviation of all of the features. The scaled feature x' is given as $x' = x - \mu/\sigma$ where μ and σ are the mean and standard deviation of the features, respectively.

2.1.2 Min - Max normalization

In Min - Max normalization, features are scaled to conform to a particular range of values, typically from 0 to 1. The scaled feature x' is given as $x' = x - min/max - min$ where the original unscaled feature is x, while min and max denote the minimum and maximum values that the features will range in, respectively.

2.1.3 Scaling to unit length

This method is purely normalization of a vector in the context of linear algebra. The transformed feature vector X' from the original feature vector X is given as $X' = X/||X||$.

2.1.4 Maximum Absolute scaling

As given by the name, Maximum absolute scaling scales a feature x by dividing it by the maximum absolute value of the feature column. Given a feature column F_x, the scaled feature x' is given as $x' = x/\text{Max}(|F_x|)$.

2.1.5 Robust Scaling

This scaling method scales data using their interquartile range (IQR), given as the difference between the 3^{rd} and 1^{st} quartiles. A scaled feature x' from an unscaled feature x is given as $x' = x - \tilde{x}/\text{IQR}$, where $\tilde{x}$ denotes the median of the features. As suggested by the name, Robust scaling is robust to outliers in datasets since it uses percentiles to scale the features.

2.1.6 Yeo-Johnson Transformation

Yeo and Johnson (2000) developed a transformation that transforms data to conform to a Gaussian distribution dependent on a parameter λ. Given a feature x, the transformed feature x' is given by:

$$x' = \begin{cases} \frac{((1+x)^{\lambda}-1)}{\lambda} & \text{if } \lambda \neq 0 \text{ and } x \geq 0 \\ \log(1+x) & \text{if } \lambda = 0 \text{ and } x \geq 0 \\ -\frac{((1-x)^{2-\lambda}-1)}{2-\lambda} & \text{if } \lambda \neq 2 \text{ and } x < 0 \\ -\log(1-x) & \text{if } \lambda = 2 \text{ and } x < 0 \end{cases}$$

The parameter lambda is chosen as the one the best fits the data, through maximum likelihood.

2.1.7 Quantile Normalization

Originally used in the anlayses of microarrays, Quantile Normalization attempts to make two distributions similar to each other (Bolstad et al., 2003). Quantile normalization can be used to transform data with respect to a reference distribution. In the case of machine learning, the reference distribution can be the Gaussian distribution. To transform data, Scikit-learn's implementation first computes quantiles of the data given n number of quantiles. Then, interpolants are created at points [n$^{\text{th}}$ quantile value, n$^{\text{th}}$ quantile] evaluated at each feature in the feature column. Finally, using the inverse Cumulative distribution function, the result is profiled to a normal distribution.

2.2 Genetic Algorithms

A Genetic Algorithm is a heuristic search process developed from the inspiration of natural selection. It was developed by Holland (1975) originally to study adaptation in nature. The process starts with an initial population of chromosomes consisting of genes. Genes can represent elements of a possible solution to a optimization problem encoded by alleles (e.g. 0 or 1, letters a-z, etc.). GA searches the best performing chromosome through applying operators which makes the next generation (population) (Mitchell, 1998). This is done iteratively over each new generation until convergence or until a specified number of iterations.

There are typically three main operators used in GAs, although there may be more. The selection operator selects fractions of the population based on the chromosome's fitness score to be used in generating the next generation. The chromosome's fitness score is computed through a function based on the optimization problem. Chromosomes with higher fitness tend to be selected more often. The mutation operator involves changing an allele to another random allele in a chromosome from the search space. Crossover combines characteristics of two (or more) selected parent chromosomes from the selection operator. This done by simply exchanging sections or fractions of the chromosome between parents. The resulting new chromosome then becomes part of the new generation.

3 Related Work

Prior studies have also employed evolutionary algorithms in the pre-processing of data, such as feature selection and feature importance scaling. The aim of these methods is to improve the performance of the learning algorithm by removing irrelevant or noisy features from the data, and by adjusting the feature importances so that the most important features are given more weight. These studies have shown that evolutionary algorithms are able to improve performance of the models that are subsequently trained on the pre-processed data. In addition, this studies show evaluation on a wide variety of tasks, exhibiting the GAs are exceptionally efficient for as a pre-processing method, and improving other pre-processing methods.

3.1 GAs in Feature selection

Feature selection is the process of selecting a feature subset from a dataset, usually with the objective of improving accuracy by selecting only the best and least amount of features. Various methods of feature

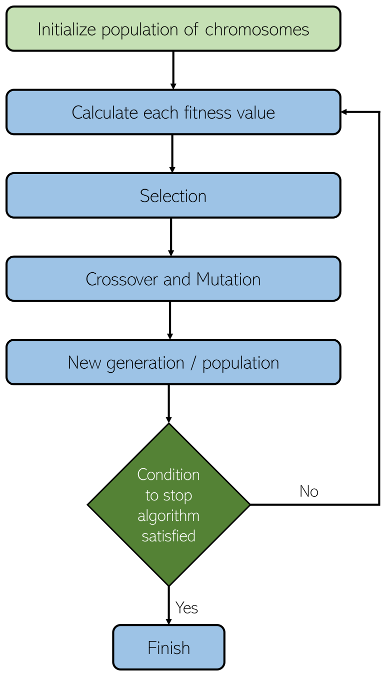

Initialize population of chromosomes
Calculate each fitness value
Selection
Crossover and Mutation
New generation / population
Condition to stop algorithm satisfied
No
Yes
Finish

Figure 1: The process of GAs in a flowchart

selection include Particle swarm optimization (Zhang et al., 2014), Variable neighborhood search (García-Torres et al., 2016), Scatter search (García-Torres et al., 2021) and GAs (Soufan et al., 2015).

Tan et al. (2014) proposed a multi-objective genetic algorithm called Modified micro Genetic Algorithm whose objectives were to improve neural network accuracy and choose a small number of input features. They compared their approach with other methods on several benchmarks. Evaluating their proposed method, results show that improved performance on classification.

Sayed et al. (2019) used a nested GA for identifying genes that were significant to a cancer disease. Their nested GA was ran on two types of datasets, Microarray gene experssion datasets and DNA Methylation datasets, in order to get the most favorable feature subset from combining both. Comparing the nested GA to other feature selection algorithms that also utilized both types of datasets, their approach was found to perform the best on classification.

A binary GA was adapted by Babatunde et al. (2014) using a novel function to calculate the fitness from a k-Nearest Neighbors based error. Their approach was applied on the Flavia dataset (Wu et al., 2007), a leaf dataset for classification. Extracting a numerous features from the images, Babatunde et al. achieved better results than other feature selection algorithms, specifically those from the WEKA software (Hall et al., 2008).

Desale and Ade (2015) used a GA for feature selection in a Naïve Bayes classifier and WEKA's J48 classifier in two datasets (Ikonomovska, 2009, Canadian Institute for Cybersecurity, 2009). Their results show improvements in accuracy and time complexity in both datasets, however no improvements were shown for the J48 classifier.

3.2 GAs in scaling importance of features

Feature importance scaling can be considered a type of feature selection, where the each feature's importance is given a scale based by an algorithm. These algorithms typically assign importance to be either 0 or 1, or ranging from 0 to 1, with 0 being of least importance and 1 being of high importance.

In their study, Prathama et al. (2017a) used GAs along with a classifier to scale features' importances in the Common Spatial Pattern (CSP) algorithm. CSP is a method of filtering information from Electroencephalographybased Brain-Computer Interface (Lu et al., 2009). While there are many processes of improving CSP, Prathama et al. utilized GAs for frequency band selection by scaling the importance of each band from 0 to 1, with 1 conveying most importance. In comparison with feature selection and vanilla CSP, their approach showed better results.

Yu et al. (2005) demonstrated that GAs improve performance in SVMs, particularly in recognition tasks compared to normal SVMs. Their method drops out features scaled to zero by the GA, as well as using the least amount of features.

4 Methods

4.1 Experimental Framework

The experiment will focus on binary classification performance of MLPs, and using GAs as a search method to discover pre-processing operations aimed at increasing the performance of MLPs, in terms of accuracy on several datasets. These will be evaluated on different datasets and statistically compared to the existing techniques described in section 2.1. Scikit-learn's (Pedregosa et al., 2011) implementations of these existing techniques are used if available.

4.2 Genetic Algorithm Framework

4.2.1 Defining and Representing pre-processing operations as Chromosomes

A pre-processing operation that a chromosome represents will be defined in this study as a simple composite function of n functions. As such, the terms *chromosome*, *pre-processing operation/operation*, and *chromosome* will refer to the same thing and will be used interchangeably in this paper. An pre-processing operation in the GA is defined as:

$$P(x) = (f_1 \circ f_2 \circ \cdots \circ f_n)(x) \tag{1}$$

with n being the maximum number of compositions. Technically, this composite function is a point in the GA search space, and each function f_i where $i = 1, 2, 3, \ldots n$ is considered an allele. A function part of the composite function f_n can take various domains, and is applied element-wise to an input vector. This ensures that the original dimensions or shape of the data is preserved in pre-processing.

A composite function will be have a maximum of six compositions. Flexibility of the number of functions in the composite chosen by the GA is incorporated through the addition of the identity function, as identity functions can be cancelled out. This in essence allows for a composite function to consist of $n \leq 6$ functions when simplified.

4.2.2 Objective, Fitness function, and Process of the Genetic Algorithm

Since the objective of the GA is to find operations that improve performance on binary classification, the output of the fitness function is simply the validation accuracy of an MLP model on a dataset.

To incorporate generalization of the GA results, the mean accuracy of an MLP trained on 5 binary classification datasets (Dua and Graff, 2017, Praveen, 2020, Islam et al., 2019, Eschlbeck, 2021, Issadeen, 2020) were used as the fitness score of the composite function. To further increase generalization, the MLP that will be trained is to have randomly set hyperparameters (various number of layers, different activation functions, random number of dropout regularization layers, and random number of units in each layer) every time a fitness score of a chromosome was calculated. This ensures that composite functions that only perform well on a certain structure of MLP will be discarded. Figure 2 shows the process of the genetic algorithm used in this study.

4.2.3 Genetic Algorithm Configurations

The length of a chromosome in a population is defined as the number of functions that comprises the pre-processing operation represented by the chromosome. This will range from 1 to 6, mentioned in section 4.2.1. The initial population size P was set to 16 chromosomes. The GA was ran for 200 iterations.

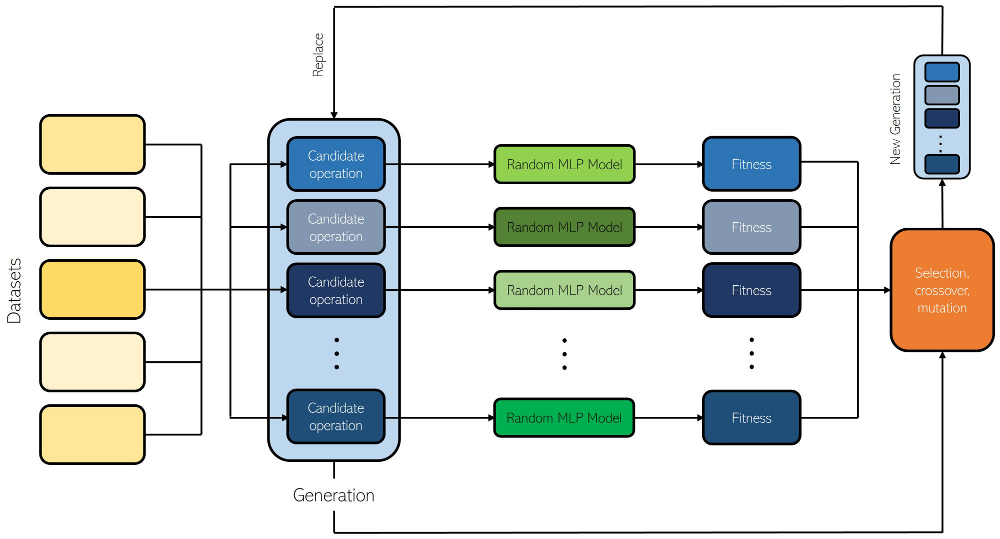


Figure 2: Process of the GA

A chromosome is represented by a bitstring $f_1 f_2 f_3 \dots f_n$ when undergoing selection, crossover, and mutation operations. Each allele in the chromosome bitstring represents a corresponding function.

The GA first runs selection operations in the population. This is done through tournament selection. Tournament selection picks k chromosomes from the population, then choosing the chromosome with the best fitness from these k chromosomes. This chosen chromosome is then processed further (Shukla et al., 2015). In this study, $k = 5$ and is done for P times resulting in P selected chromosomes. Note that this means that there are duplicate chromosomes selected.

Then, the selected chromosomes are paired. Given a chromosome C_i where $i = 1, 2, 3, \dots P$, a pair is plainly C_i, C_{i+1} where P is even for simplicity. Crossover is applied to these pairs by choosing a random slice point in the chromosome bitstring and exchanging alleles at the slice point. A crossover rate of 85% is used.

Finally, mutation is applied to the chromosomes. This study uses a modified implementation of vanilla mutation and the regenerate operation. The regenerate operation is applied from the work of Bingham and Miikkulainen (2020) which, in the context of this study, replaces each allele in a chromosome with a random allele. In essence, the regenerate function replaces a chromosome with a random point (random chromosome) in the GA search space. This is done to increase exploration. Vanilla mutation replaces each allele in the chromosome with a new random allele with a certain probability. A probability of 10% was used. Both vanilla mutation and regenerate operations are implemented as a single mutation operation. If mutation is applied to a chromosome, regeneration has a 0.1% chance of happening, and vanilla mutation having a 99.9% chance. These percentages were based on preliminary tests described in Appendix A.

4.3 Statistical Testing for Evaluation of GA results

Pre-processing operations discovered by the GA will be tested on 400 random configurations of an MLP for its mean validation accuracy on 5 different datasets (Wolberg et al., 1995, Bootwala, 2017, Dedhia, 2020, Hannousse and Yahiouche, 2021, Naranjo et al., 2016). A discovered pre-processing operation is combined with commonly used existing pre-processing techniques to form the final pre-processing method. Each of these methods are compared and tested against existing techniques listed in section 2.1 using Wilcoxon Signed Rank Tests on each discovered method. These methods are applied to evaluation datasets, and 400 randomized MLPs are trained on these pre-processed datasets whose mean accuracy is recorded. The null hypothesis is $\mathrm{H}_0 : \tilde{x}_1 \leq \tilde{x}_2$ and the alternative hypothesis is $\mathrm{H}_A : \tilde{x}_1 > \tilde{x}_2$ where $\tilde{x}_1$ is the median of the mean accuracy on the discovered methods, and $\tilde{x}_2$ are is the median of the mean accuracy on existing methods. An $\alpha = 0.01$ level of significance is used.

4.4 Datasets used

This section describes each dataset used in running the GA and the evaluation of results. Datasets that were collected vary in characteristics; the number of samples range from 250 to 11000 samples, the number features range from 2 to 87, which include low to extremely high values. The labels in each dataset were balanced, and any sample that had missing values were omitted. Categorical features were integer encoded. Each dataset was split in 70% training data and 30% validation data. Model performance on the validation data was used in the computation of the fitness score.

4.4.1 Datasets used in fitness calculation

- Cleveland UCI Heart Disease Dataset: This dataset is composed of 303 instances of patients with and without heart disease of 13 features each. Features are categorical and positive. The dataset was sourced from UCI machine learning repository (Dua and Graff, 2017). This dataset was scaled with standard scaling before being used to ensure that the GA finds operations that do not disrupt already pre-processed data.

- Credit Risk Classification Dataset: This dataset describes instances of credit risk among customers, labeling each as risky or not. Each sample in the dataset contains 11 features, containing negative and positive values. Labels were balanced to contain 376 samples (Praveen, 2020).

- Early Diabetes Classification Dataset: This dataset contains samples of patients and their characteristics, amounting to 520 instances each with 17 features (Islam et al., 2019). Labels describe the patient's diagnosis, either having diabetes or not. Values in the dataset are comprised of categorical and integer (mostly 0 and 1) values.

- Beginner's Classification Dataset: This dataset is comprised of 297 samples each containing only 2 features. The dataset labels each sample as "successful" or not in the context of sports. The dataset includes both positive and negative values (Eschlbeck, 2021).

- Gender Classification Dataset: Comprised of 5001 samples containing 7 facial features, this synthetic dataset labels each sample as either male or female (Issadeen, 2020). Values in the dataset are positive.

4.4.2 Datasets used for evaluation of search results

- Breast Cancer Wisconsin Data Set: Using images of fine needle aspiration (a type of diagnostic procedure) of breast mass, 32 characteristics (features) were computed for the dataset (Wolberg et al., 1995, Dua and Graff, 2017). The features are composed of categorical and positive values. This dataset consists of 569 instances labeled either as malignant or benign.

- Titanic Dataset: This dataset uses a cleaned version of the data (Bootwala, 2017) from a Kaggle competition (Kaggle, n.d.) composed of 792 samples (684 when balanced) of 14 features of passengers, labeling each as "survived" or not. The cleaned version contains values ranging from 0 to 1.

- Bike Buyers 1000 Dataset: This dataset describes 1000 customers' backgrounds (11 features) and label each as having purchased a bike or not (Dedhia, 2020). Values are all positive.

- Webpage Phishing Detection Dataset: Containing 87 extracted features from 11430 URLs, this dataset labels each sample as either "legitimate" or "phishing" (Hannousse and Yahiouche, 2021).

- Bank Marketing Data Set: Associated with marketing campaigns of a bank institute, this dataset contains attributes of clients and whether or not they subscribed a term deposit. Originally containing 20 features, the features recording the *month* and *day of the week* were removed before being used. Both positve and negative values are present in the dataset, and was balanced to have 902 instances. This dataset was collected from the UCI Machine Learning repository (Dua and Graff, 2017) and was created and used in the work of Moro et al. (2014).

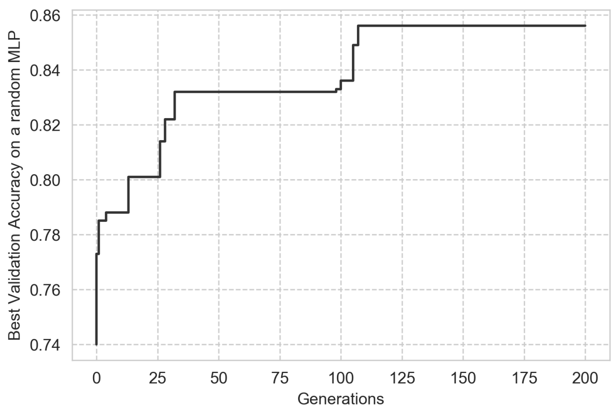


Figure 3: Best validation accuracy on the evaluation datasets obtained by a chromosome (operation) with respect to the number of generations (iterations) in the GA.

5 Results and Discussion

5.1 Operations discovered

Not all discovered operations that achieved high mean accuracies during the running of the GA also achieved high mean accuracies in the evaluation datasets. This means that some operations did not succeed in generalizing well to other data. Out of a number discovered operations that were tested, two were found to generalize well in terms of accuracy. These operations will be described. For easier description of the operations, variables μ and s that have a subscript will be defined as the mean and variance of its subscript values, respectively.

The first well-performing operation was found to utilize the same functions in Z-Score normalization in addition to other functions. This was found to be:

$$P(x) = (\text{DivideVar} \circ \text{SubtractMean} \circ \text{softsign} \circ \text{swish} \circ \text{SubtractMean} \circ \text{softsign})(x) \tag{2}$$

where $\text{DivideVar}(x) = \frac{x}{s_x}$ and $\text{SubtractMean}(x) = x - \mu_x$, softplus, swish, and softsign are recently developed activation functions with the same name (Dugas et al., 2001, Ramachandran et al., 2017, Glorot and Bengio, 2010). In another form, equation 2 is:

$$P(x) = \frac{p(x) - \mu_{p(x)}}{s_{p(x)-\mu_{p(x)}}} \quad \text{where} \quad p(x) = \text{softsign}(\text{swish}(\text{softsign}(x) - \mu_{\text{softsign}(x)})) \tag{3}$$

The second operation that performs equally well on the evaluation datasets was found to be:

$$P(x) = (\text{DivideVar} \circ \text{SubtractMean} \circ \text{softplus} \circ \tanh^{-1} \circ \text{Negative} \circ \text{softsign})(x) \tag{4}$$

where $\text{Negative}(x) = -x$. For a more compact form:

$$P(x) = \frac{p(x) - \mu_{p(x)}}{s_{p(x)-\mu_{p(x)}}} \quad \text{where} \quad p(x) = \text{softplus}(\tanh^{-1}(-\text{softsign}(x))) \tag{5}$$

To simplify the referencing of these operations, the first operation at equation 2 will be called P1, and the second operation at equation 4 P2.

A common trait of high performing operations that were searched was the frequent use of $\text{softsign}(x)$, $\text{SubtractMean}(x)$ and $\text{DivideVar}(x)$. Often $\text{softsign}(x)$ is applied as the first function in the composite function, while $\text{SubtractMean}(x)$ and $\text{DivideVar}(x)$ are applied last. Figure 4 shows the top 10 most frequently used functions among operations that achieved mean accuracies of more than 80%.

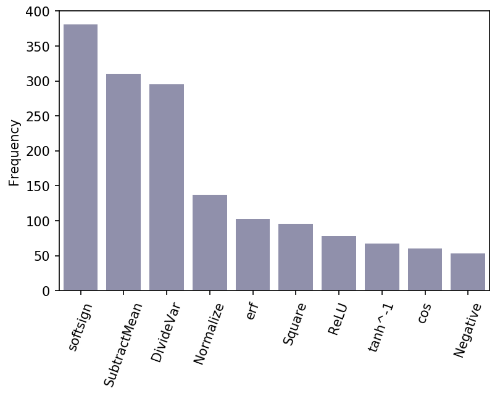


Figure 4: Top 10 most frequently used functions in the best performing operations

5.2 Statistical Analysis

It is important that the search findings generalize well to other datasets and in various MLP hyperparameters. The mean accuracies of 400 MLP samples with random hyperparameters (same random generation in section 4.2.2) on the evaluation datasets pre-processed using existing techniques (section 2.1) and P1 and P2 pre-processing were recorded. Mean accuracies of a combination of Z-score normalization and P1 or P2 pre-processing were also recorded (Z-score normalization was chosen to be combined with P1 or P2 based on preliminary tests showing the best performance). Shapiro-Wilks tests on the recorded mean accuracies show that the data is extremely non-normal, thus Wilcoxon Signed-Rank Tests are conducted.

Wilcoxon Signed-Rank Tests on P1, P2, Z-score normalization + P1 and Z-score normalization + P2 compared to each existing technique (from a 0.01 significance level) reports that medians of P1 and P2 pre-processing alone showed that these achieve better performance than existing techniques except for the Yeo-Johnson Transformation, Z-score normalization, and Quantile Transformer. However, when Z-score normalization is applied first to the data, then pre-processed using P1 or P2, this statistically shows higher performance than all of the listed existing techniques. P-values of the test are displayed in Table 1 and medians are presented in Table 2.

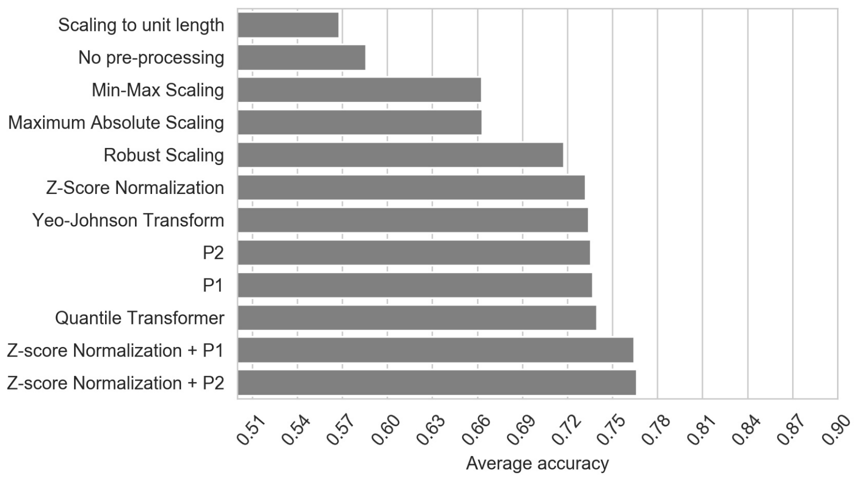
5.3 Visual Comparison

Further comparison is conducted between the discovered operations by plotting the distributions of the mean accuracies of each pre-processing method. Histograms and Kernel Density Estimations (KDE) show that the better-performing pre-processing methods are extremely left-skewed. The distributions of all the mean accuracies of pre-processing methods are illustrated in a histogram at Figure 6a and in a KDE plot at Figure 6b. For visual clarity, distributions of P1, P2, Z-score normalization + P1, and Z-score normalization + P2 are shaded in Figures 7 and 8.

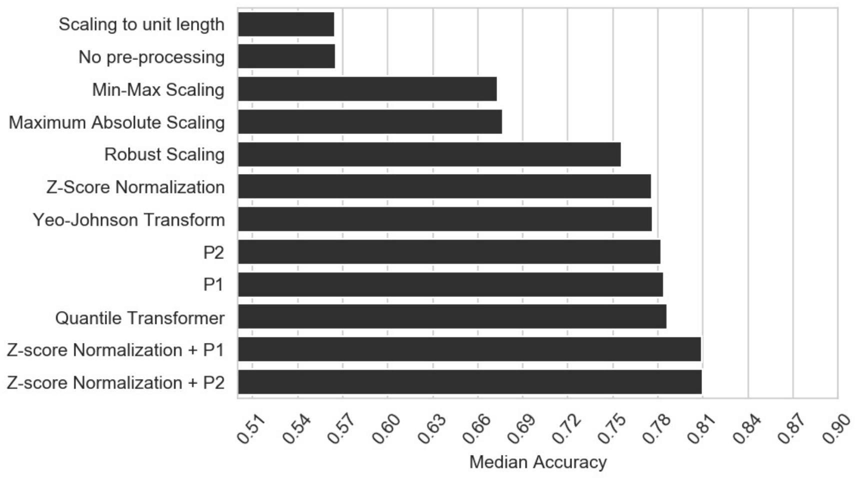
Visual comparison of data before and after pre-processing by P1, P2, Z-score normalization + P1, and Z-score normalization + P2 to other methods is presented. Pre-processing methods were applied to a linearly-separable, synthetic binary classification dataset. This dataset consists of both negative and positive values. Scatterplots of the data were plotted, showing the effects of each pre-processing method illustrated in Figure 9. Inspection of the plots show that P1 and Z-score normalization + P1 extend the range of the data. This also displays that these direct points away from the center of the plot resulting in less density in the center and more density around the edges to a degree. This holds true but is less apparent in P2 and Z-score normalization + P2.

5.4 Implications and Further Applications

The results of this study suggest that the pre-processing techniques of Z-score normalization + P1 or P2 may be used to improve the performance of MLPs. This potentially has important implications for the field of machine learning, as MLPs are used in various applications such as in medical aid (Mossa et al., 2019,



(a) Average of the mean accuracy of MLPs on evaluation datasets of different pre-processing techniques and the discovered pre-processing operations



(b) Median accuracies of MLPs on evaluation datasets of different pre-processing techniques and the discovered pre-processing operations

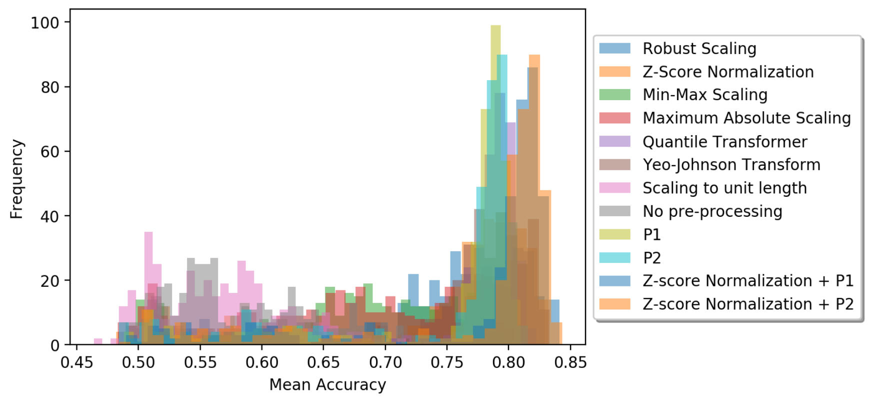
Figure 5: Average and median accuracies of each pre-processing method.

Table 1: Wilcoxon Signed-Rank Tests on P1, P2, Z-score normalization + P1 and Z-score normalization + P2. The level of significance is set at $\alpha = 0.01$.

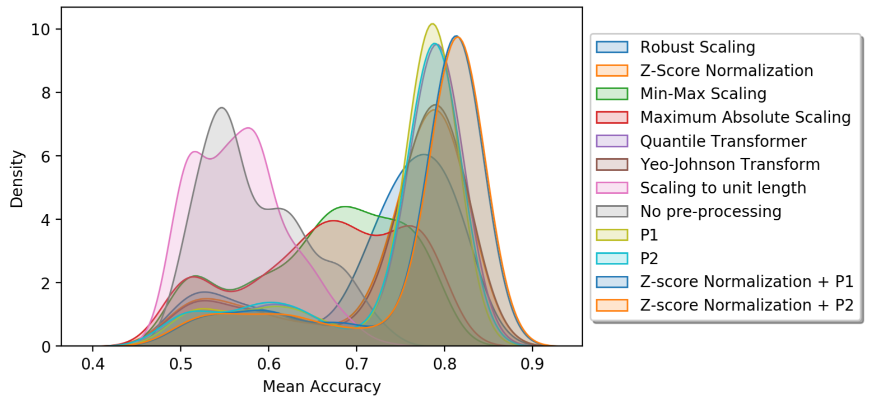
P1		
vs	p-value	Inference
Robust Scaling	3.60E-17	Reject H_0
Z-Score Normalization	0.011844	Do not Reject H_0
Min-Max Scaling	1.04E-60	Reject H_0
Maximum Absolute Scaling	1.98E-60	Reject H_0
Yeo-Johnson Transform	0.081415	Do not Reject H_0
Scaling to unit length	3.09E-67	Reject H_0
Quantile Transformer	0.999892	Do not Reject H_0
No pre-processing	2.03E-65	Reject H_0
P2		
vs	p-value	Inference
Robust Scaling	3.71E-17	Reject H_0
Z-Score Normalization	0.027059	Do not Reject H_0
Min-Max Scaling	4.17E-60	Reject H_0
Maximum Absolute Scaling	7.10E-59	Reject H_0
Yeo-Johnson Transform	0.247079	Do not Reject H_0
Scaling to unit length	8.42E-67	Reject H_0
Quantile Transformer	0.999897	Do not Reject H_0
No pre-processing	1.92E-64	Reject H_0
Z-score norm. + P1		
vs	p-value	Inference
Robust Scaling	1.38E-52	Reject H_0
Z-Score Normalization	3.92E-42	Reject H_0
Min-Max Scaling	1.79E-66	Reject H_0
Maximum Absolute Scaling	5.38E-67	Reject H_0
Yeo-Johnson Transform	1.64E-42	Reject H_0
Scaling to unit length	1.74E-67	Reject H_0
Quantile Transformer	2.87E-47	Reject H_0
No pre-processing	1.49E-66	Reject H_0
Z-score norm. + P2		
vs	p-value	Inference
Robust Scaling	9.19E-59	Reject H_0
Z-Score Normalization	7.90E-51	Reject H_0
Min-Max Scaling	4.64E-67	Reject H_0
Maximum Absolute Scaling	4.45E-67	Reject H_0
Yeo-Johnson Transform	6.41E-48	Reject H_0
Scaling to unit length	1.76E-67	Reject H_0
Quantile Transformer	4.83E-48	Reject H_0
No pre-processing	1.33E-66	Reject H_0

Table 2: Comparison of the Average Accuracy ± SD and Median of discovered methods and existing methods

Method	Average Accuracy	Median
Scaling to unit length	56.785 ± 5.328	56.471
No pre-processing	58.583 ± 5.999	56.541
Min-Max Scaling	66.282 ± 8.677	67.307
Maximum Absolute Scaling	66.317 ± 9.127	67.679
Robust Scaling	71.765 ± 9.956	75.584
Z-Score Normalization	73.209 ± 10.048	77.607
Yeo-Johnson Transform	73.426 ± 9.951	77.669
P2	73.536 ± 9.506	78.249
P1	73.671 ± 9.157	78.413
Quantile Transformer	73.978 ± 9.402	78.652
Z-score Normalization + P1	76.449 ± 9.454	80.927
Z-score Normalization + P2	76.596 ± 9.552	81



(a) Histogram of the mean accuracies of each pre-processing method.



(b) Kernel Density Estimate plot of the mean accuracies of each pre-processing method.

Figure 6: Distributions of the mean accuracies of each pre-processing method visualized.

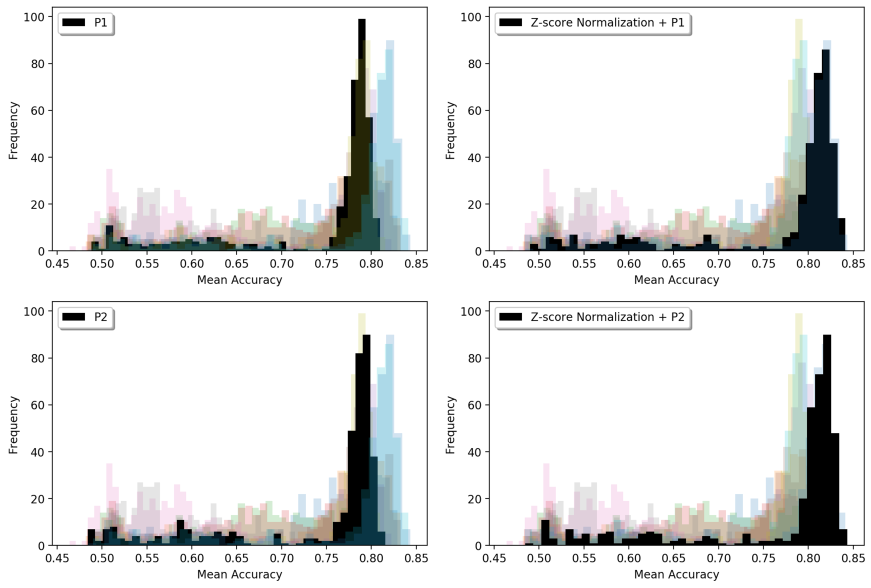


Figure 7: Histogram of the mean accuracies of each pre-processing method with emphasis on P1, P2, Z-score normalization + P1, and Z-score normalization + P2.

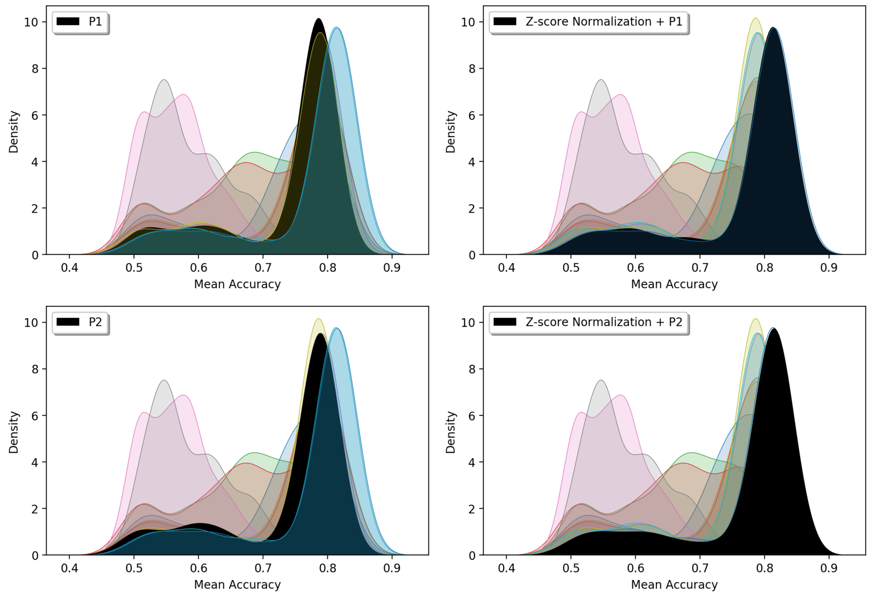


Figure 8: Kernel Density estimate plots of the mean accuracies of each pre-processing method with emphasis on P1, P2, Z-score normalization + P1, and Z-score normalization + P2.

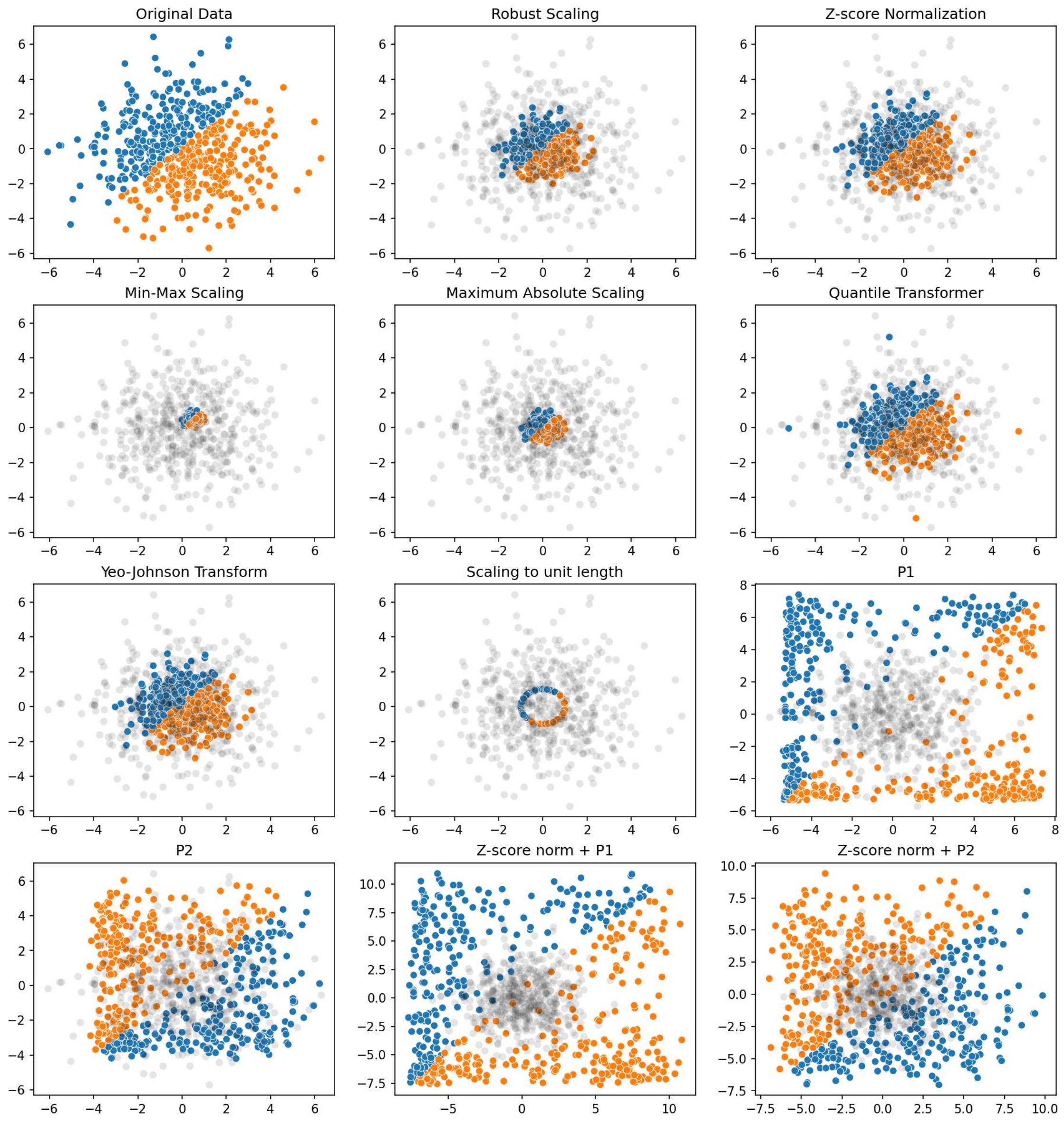


Figure 9: Scatterplots of data pre-processed by various methods, including P1, P2, Z-score normalization + P1, and Z-score normalization + P2. Orange and blue values indicate positive and negative labels, respectively. Transparent, gray points indicate the original data before pre-processing.

Akinnuwesi et al., 2021, Desai and Shah, 2021), software analysis (Qiao et al., 2020, Iqbal and Aftab, 2020), and even in computer vision by modern MLP models (Touvron et al., 2021, Tolstikhin et al., 2021).The results of this study may help to improve the performance of MLPs in these areas.

5.5 Future Work

There are several possible directions for future work in this area. One possibility is to widen the search space of this study's GA approach and methods. This involves using larger and more datasets, more varieties of MLPs (and hyperparameters), and increasing the number of iterations of the GA.

In addition to scaling this study's methods and approaches, future work is suggested to test the discovered operations on fully-connected blocks or linear layers in other neural network architecture which are similar (if not the same) in structure to MLPs, especially in convolutional neural networks. Also, more insight should be gained from testing other novel pre-processing methods in comparison to this study's findings.

6 Conclusions

In this paper, novel pre-processing operations for multilayer perceptrons are developed using genetic algorithms are presented. The operations are based on a composition of functions wherein data are processed initially before training. A genetic algorithm was used to evolve operations to maximize validation accuracy on a number of datasets. Some pre-processing methods discovered in this study yielded statistically higher performance on several new datasets than existing commonly used data pre-processing techniques. These findings suggest that the discovered pre-processing operations may be capable of improving MLP models. Future work is suggested to conduct analysis of the impact of the discovered operations on MLP blocks of convolutional neural networks and transformer architectures.

References

Boluwaji A. Akinnuwesi, Stephen G. Fashoto, Elliot Mbunge, Adedoyin Odumabo, Andile S. Metfula, Petros Mashwama, Faith-Michael Uzoka, Olumide Owolabi, Moses Okpeku, and Oluwaseun O. Amusa. Application of intelligence-based computational techniques for classification and early differential diagnosis of covid-19 disease. *Data Science and Management*, 4:10–18, 2021. ISSN 2666-7649. doi: https://doi.org/10.1016/j.dsm.2021.12.001. URL https://www.sciencedirect.com/science/article/pii/S2666764921000473.

Hanan Alshaher. Studying the effects of feature scaling in machine learning. Ph.D. dissertation, North Carolina Agricultural and Technical State University, 2021.

Oluleye H Babatunde, Leisa Armstrong, Jinsong Leng, and Dean Diepeveen. A genetic algorithm-based feature selection. *IJECCE*, 2014.

Garrett Bingham and Risto Miikkulainen. Discovering parametric activation functions. *CoRR*, abs/2006.03179, 2020. URL https://arxiv.org/abs/2006.03179.

Danushka Bollegala. Dynamic feature scaling for online learning of binary classifiers. *Knowledge-Based Systems*, 129:97–105, 2017.

Benjamin M Bolstad, Rafael A Irizarry, Magnus Åstrand, and Terence P. Speed. A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics*, 19(2):185–193, 2003.

Azeem Bootwala. Titanic dataset, 2017. Kaggle, V1. Retrieved from https://www.kaggle.com/azeembootwala/titanic.

Canadian Institute for Cybersecurity. Nsl-kdd dataset, 2009. URL https://www.unb.ca/cic/datasets/nsl.html.

Heeral Dedhia. Bike buyers 1000, 2020. Kaggle, V2. `https://www.kaggle.com/heeraldedhia/bike-buyers?select=bike_buyers_clean.csv`.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. IEEE, 2009.

Meha Desai and Manan Shah. An anatomization on breast cancer detection and diagnosis employing multi-layer perceptron neural network (mlp) and convolutional neural network (cnn). *Clinical eHealth*, 4:1–11, 2021.

Ketan Desale and Roshani Ade. Preprocessing of streaming data using genetic algorithm. *International Journal of Computer Applications*, 120(17), 2015.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL `http://archive.ics.uci.edu/ml`.

Charles Dugas, Yoshua Bengio, François Bélisle, Claude Nadeau, and René Garcia. Incorporating second-order functional knowledge for better option pricing. In T. Leen, T. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems*, volume 13. MIT Press, 2001. URL `https://proceedings.neurips.cc/paper/2000/file/44968aece94f667e4095002d140b5896-Paper.pdf`.

Sven Eschlbeck. Beginner's classification dataset, 2021. Kaggle, V1. Retrieved from `https://www.kaggle.com/sveneschlbeck/beginners-classification-dataset`.

Miguel García-Torres, Francisco Gómez-Vela, Belén Melián-Batista, and J Marcos Moreno-Vega. High-dimensional feature selection via feature grouping: A variable neighborhood search approach. *Information Sciences*, 326:102–118, 2016.

Miguel García-Torres, Francisco Gómez-Vela, Federico Divina, Diego P Pinto-Roa, José Luis Vázquez Noguera, and Julio C Mello Román. Scatter search for high-dimensional feature selection using feature grouping. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pages 149–150, 2021.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian Witten. The weka data mining software: An update. *SIGKDD Explor. Newsl.*, 11:10–18, 11 2008.

Abdelhakim Hannousse and Salima Yahiouche. Web page phishing detection, 2021. Mendeley Data, V3.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

John Holland. *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence.* U Michigan Press, 1975.

Elena Ikonomovska. Airlines dataset, 2009. URL `https://moa.cms.waikato.ac.nz/datasets`.

Ahmed Iqbal and Shabib Aftab. A classification framework for software defect prediction using multi-filter feature selection technique and mlp. *International Journal of Modern Education & Computer Science*, 12 (1), 2020.

M. M. Islam, Rahatara Ferdousi, Sadikur Rahman, and Humayra Yasmin Bushra. Likelihood prediction of diabetes at early stage using data mining techniques. *Computer Vision and Machine Intelligence in Medical Image Analysis*, page 113–125, 2019. doi: 10.1007/978-981-13-8798-2_12.

Jifry Issadeen. Gender classification dataset, 2020. Kaggle, V1. Retrieved from `https://www.kaggle.com/elakiricoder/gender-classification-dataset/`.

Kaggle. Titanic - machine learning from disastert. `https://www.kaggle.com/c/titanic/data`, n.d.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

Xiang Long, Kaipeng Deng, Guanzhong Wang, Yang Zhang, Qingqing Dang, Yuan Gao, Hui Shen, Jianguo Ren, Shumin Han, Errui Ding, et al. Pp-yolo: An effective and efficient implementation of object detector. *arXiv preprint arXiv:2007.12099*, 2020.

Haiping Lu, Konstantinos N Plataniotis, and Anastasios N Venetsanopoulos. Regularized common spatial patterns with generic learning for eeg signal classification. In *2009 Annual International Conference of the IEEE Engineering in medicine and biology society*, pages 6599–6602. IEEE, 2009.

Melanie Mitchell. *An Introduction to Genetic Algorithms.* MIT Press, 1998.

Sérgio Moro, Paulo Cortez, and Paulo Rita. A data-driven approach to predict the success of bank telemarketing. *Decision Support Systems*, 62:22–31, 2014. ISSN 0167-9236. doi: https://doi.org/10.1016/j.dss.2014.03.001. URL `https://www.sciencedirect.com/science/article/pii/S016792361400061X`.

Abdela Ahmed Mossa, Abdulkerim Mohammed Yibre, and Ulus Çevik. Multi-view cnn with mlp for diagnosing tuberculosis patients using ct scans and clinically relevant metadata. In *CLEF (Working Notes)*, 2019.

Lizbeth Naranjo, Carlos J. Pérez, Yolanda Campos-Roca, and Jacinto Martín. Addressing voice recording replications for parkinson's disease detection. *Expert Systems with Applications*, 46:286–292, 2016. doi: 10.1016/j.eswa.2015.10.034.

Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Gaugan: semantic image synthesis with spatially adaptive normalization. In *ACM SIGGRAPH 2019 Real-Time Live!* Association for Computing Machinery, 2019.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Yohanes de Britto Hertyasta Prathama, Mohd Ibrahim Shapiai, Siti Armiza Mohd Aris, Zuwairie Ibrahim, Jafreezal Jaafar, and Hilman Fauzi. Common spatial pattern with feature scaling (fsc-csp) for motor imagery classification. In Mohamed Sultan Mohamed Ali, Herman Wahid, Nurul Adilla Mohd Subha, Shafishuhaza Sahlan, Mohd Amri Md. Yunus, and Ahmad Ridhwan Wahap, editors, *Modeling, Design and Simulation of Systems*, pages 591–604, Singapore, 2017a. Springer Singapore. ISBN 978-981-10-6463-0.

Yohanes de Britto Hertyasta Prathama, Mohd Ibrahim Shapiai, Siti Armiza Mohd Aris, Zuwairie Ibrahim, Jafreezal Jaafar, and Hilman Fauzi. Common spatial pattern with feature scaling (fsc-csp) for motor imagery classification. In Mohamed Sultan Mohamed Ali, Herman Wahid, Nurul Adilla Mohd Subha, Shafishuhaza Sahlan, Mohd Amri Md. Yunus, and Ahmad Ridhwan Wahap, editors, *Modeling, Design and Simulation of Systems*, pages 591–604, Singapore, 2017b. Springer Singapore. ISBN 978-981-10-6463-0.

Praveen. Credit risk classification dataset, 2020. Kaggle, V2. Retrieved from `https://www.kaggle.com/praveengovi/credit-risk-classification-dataset/data`.

Yanchen Qiao, Bin Zhang, and Weizhe Zhang. Malware classification method based on word vector of bytes and multilayer perception. In *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, pages 1–6, 2020. doi: 10.1109/ICC40277.2020.9149143.

Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Searching for activation functions. *CoRR*, abs/1710.05941, 2017. URL `http://arxiv.org/abs/1710.05941`.

Jakob Raymaekers and Peter J Rousseeuw. Transforming variables to central normality. *Machine Learning*, pages 1–23, 2021.

Sabah Sayed, Mohammad Nassef, Amr Badr, and Ibrahim Farag. A nested genetic algorithm for feature selection in high-dimensional cancer microarray datasets. *Expert Systems with Applications*, 121:233–243, 2019. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2018.12.022. URL `https://www.sciencedirect.com/science/article/pii/S0957417418307905`.

Anupriya Shukla, Hari Mohan Pandey, and Deepti Mehrotra. Comparative review of selection techniques in genetic algorithm. In *2015 International Conference on Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE)*, pages 515–519, 2015. doi: 10.1109/ABLAZE.2015.7154916.

Othman Soufan, Dimitrios Kleftogiannis, Panos Kalnis, and Vladimir B Bajic. Dwfs: a wrapper feature selection tool based on a parallel genetic algorithm. *PloS one*, 10(2), 2015.

Choo Jun Tan, Chee Peng Lim, and Yu–N Cheah. A multi-objective evolutionary algorithm-based ensemble optimizer for feature selection and classification with neural network models. *Neurocomputing*, 125:217–228, 2014. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2012.12.057. URL `https://www.sciencedirect.com/science/article/pii/S0925231213005353`. Advances in Neural Network Research and Applications Advances in Bio-Inspired Computing: Techniques and Applications.

Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in Neural Information Processing Systems*, 34, 2021.

Hugo Touvron, Piotr Bojanowski, Mathilde Caron, Matthieu Cord, Alaaeldin El-Nouby, Edouard Grave, Gautier Izacard, Armand Joulin, Gabriel Synnaeve, Jakob Verbeek, et al. Resmlp: Feedforward networks for image classification with data-efficient training. *arXiv preprint arXiv:2105.03404*, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL `http://arxiv.org/abs/1706.03762`.

Xing Wan. Influence of feature scaling on convergence of gradient iterative algorithm. In *Journal of physics: Conference series*, 1213. IOP Publishing, 2019.

William H Wolberg, W Nick Street, and Olvi L Mangasarian. Image analysis and machine learning applied to breast cancer diagnosis and prognosis. *Analytical and Quantitative cytology and histology*, 17(2):77–87, 1995.

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. In *2007 IEEE international symposium on signal processing and information technology*, pages 11–16. IEEE, 2007.

In-Kwon Yeo and Richard A Johnson. A new family of power transformations to improve normality or symmetry. *Biometrika*, 87(4):954–959, 2000.

Ying Yu, Xiaolong Wang, and Bingquan Liu. Combining feature scaling estimation with svm classifier design using ga approach. *Journal of Electronics (China)*, 22(5):550–557, 2005.

Yuan Yuan, Siyuan Liu, Jiawei Zhang, Yongbing Zhang, Chao Dong, and Liang Lin. Unsupervised image super-resolution using cycle-in-cycle generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 701–710, 2018.

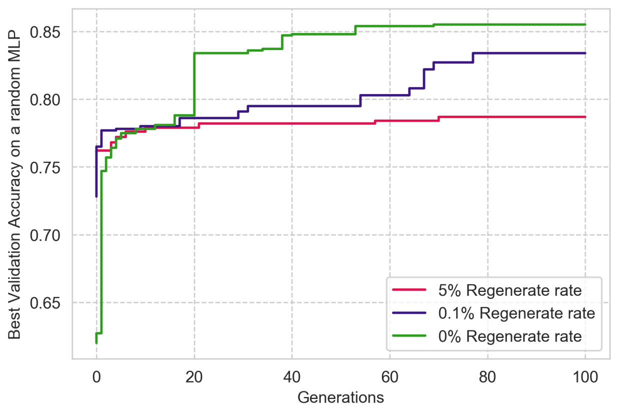


Figure 10: Comparison of the evolution of chromosome accuracy for each regenerate rate

Changzheng Zhang, Xiang Xu, and Dandan Tu. Face detection using improved faster rcnn. *arXiv preprint arXiv:1802.02142*, 2018.

Yudong Zhang, Shuihua Wang, Preetha Phillips, and Genlin Ji. Binary pso with mutation operator for feature selection using decision tree applied to spam detection. *Knowledge-Based Systems*, 64:22–31, 2014. ISSN 0950-7051. doi: https://doi.org/10.1016/j.knosys.2014.03.015. URL `https://www.sciencedirect.com/science/article/pii/S095070511400104X`.

Appendices

A Preliminary Tests

Preliminary Tests were conducted to tune the rate of the regenerate operator in section 4.2.3. Tests were ran on the GA running for 100 iterations experimenting 0%, 0.01% and 5% rates. Plotting the best achieved accuracy on a random MLP with respect to the number of generations shown in Figure 10, it was found that not using the regenerate operation at all in fact produces better results. However, since the regenerate operator increases exploration in the search space, the GA in this study used a 0.01% rate and was ran for an additional 100 iterations to account for the lesser performance in addition to increased exploration.

B Generation of an MLP with random hyperparameters

This appendix describes the randomization process of MLP hyperparameters used in the methods and evaluation of the study's results. An MLP with randomly set hyperparameters can have 2 - 4 densely connected layers, and the number of units in each layer ranges from 16 to 50, different in each layer. A dropout layer with 10% dropout rate has a 33% chance of being added after each densely connected layer. Regarding the activation function used in for the hidden layers, this can either be *Swish* (Ramachandran et al., 2017), *ReLU*, *sigmoid*, or *tanh*. All MLPs had the sigmoid activation at the output layer.

C Supplementary Figures

A heatmap is shown in Figure 11 showing the individual mean accuracies achieved by randomized MLPs on evaluation datasets pre-processed by the commonly used existing methods and including the discovered methods.

Scatterplots for non-linearly separable synthetic datasets are also plotted in Figure 12.

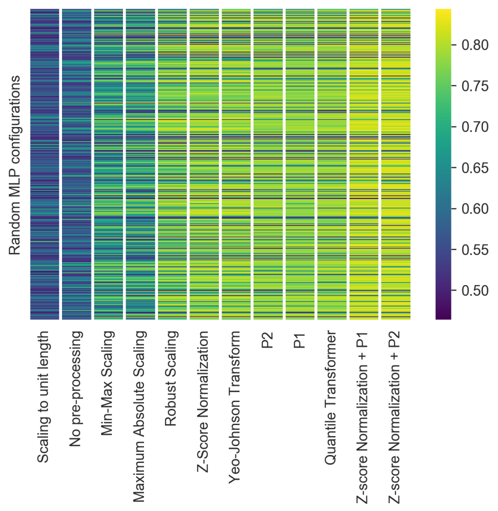


Figure 11: Heatmap of mean accuracies acheived by randomized MLPs on datasets pre-processed by the commonly used existing methods and including the discovered methods. Brighter vertical bands indicate higher accuracies.

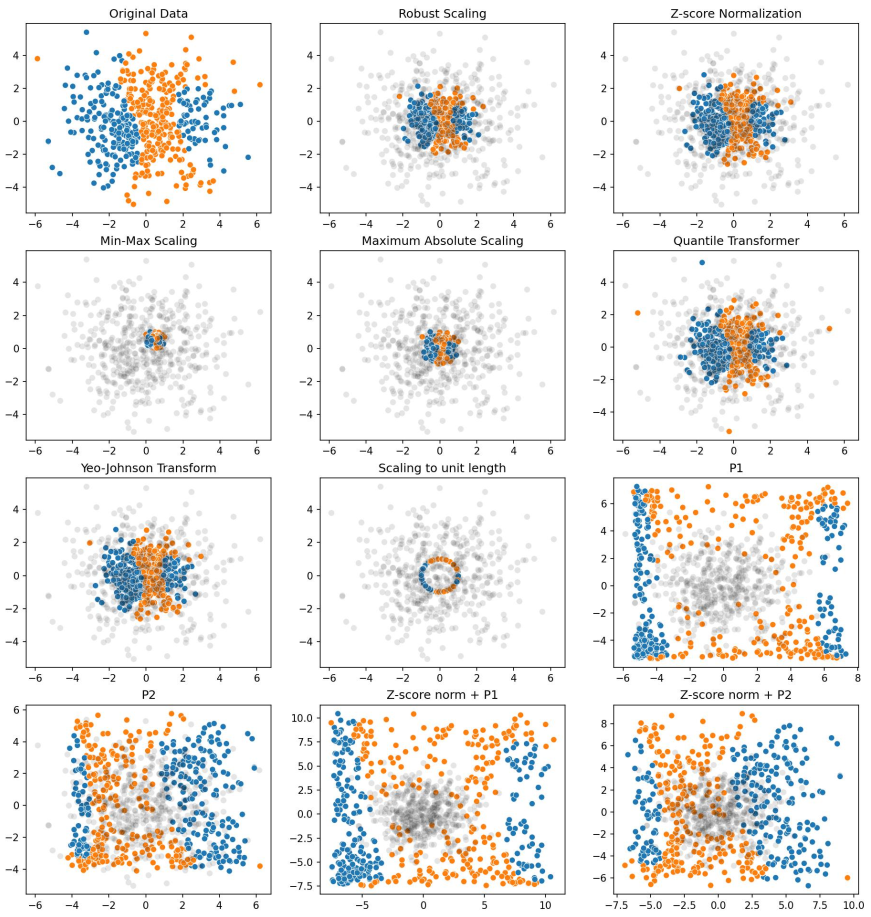


Figure 12: Scatterplots of a non-linearly separable synthetic binary class dataset pre-processed by existing methods and the discovered methods. Transparent, gray points indicate original data without pre-processsing.