Improving SWAY Performance: Better Explanations and Optimized Strategies

Ashley King
School of Engineering
North Carolina State University
Raleigh, North Carolina
Email: anking4@ncsu.edu

Pradyumna Prakash Khawas
School of Engineering
North Carolina State University
Raleigh, North Carolina
Email: ppkhawas@ncsu.edu

Jainam Ketan Shah
School of Engineering
North Carolina State University
Raleigh, North Carolina
Email: jkshah@ncsu.edu

***Abstract*—Semi-supervised multiobjective learning is a research area that combines multiobjective optimization and semi-supervised learning. Multi-objective optimization refers to the process of optimizing multiple, often competing objectives simultaneously. In our research article, we introduced a modification to `SWAY` and `EXPLAIN` algorithms. The main goal of the sway algorithm is to discover a set of low-dimensional coordinates for each data point that minimizes the differences between the pairwise distances observed in the high-dimensional space and the corresponding low-dimensional space. This is achieved by dividing the clusters into two parts, namely the "best" and the "rest". The EXPLAIN algorithm then attempts to identify simple rules that can be followed to distinguish the best data from the rest of the data. We propose an optimal set of hyperparameters that improves `SWAY`'s performance. We also propose an approach to modify `SWAY` by reducing the stochasticity of the algorithm by selecting initial candidates in `SPLIT` using α-shapes. The paper also explores the use of decision trees as a post-doc explainability algorithm. The paper compares the results of `SWAY` run with optimized hyperparameters, the modified `SWAY` algorithm, and the proposed explanation algorithm with the original `SWAY` and `EXPLAIN`.**

I. Introduction

In recent years, Multi-Objective Optimization (MOO) has become relevant in various fields, such as bioinformatics, wireless networks, natural language processing, image processing, astronomy and astrophysics, and many more [1]. MOO problems are much more complex than single objective problems as there may be contradicting objectives, and MOO techniques yield a set of (Pareto) optimal solutions [1].

Chen et. al. [2] proposes `SWAY`, which is a Multi-Objective Evolutionary Algorithm (MOEA) technique that recursively clusters the input space in order to isolate the superior cluster. In spite `SWAY` being proposed as a Search-Based Software Engineering (SBSE) [3] baseline method, its performance is comparable to state-of-the-art evolutionary algorithms while requiring significantly fewer evaluations. An improvement to `SWAY` and an increased number of evaluations could bring the performance of the algorithm to a level where it could become a good MOO technique.

Tuning hyperparameters is crucial to improving the performance of machine learning algorithms [13]. These parameters that control the training process in algorithms have to be tuned for the problem the algorithm is trying to solve. Tuning `SWAY`'s hyperparameters can yield better results.

Evaluating each of the combinations of hyperparameters can take a significant amount of time, posing a challenge [4]. Grid search is one of the widely used techniques for hyperparameter optimization. It involves running the algorithm for a grid of combinations of hyperparameters in the hyperparameter space. This technique may explore a large portion of the space that may not be significant in improving the performance of the algorithm.

For the purposes of this paper, we wish to improve `SWAY` by starting the `SPLIT` for a continuous space with a candidate that exists on the boundary of candidates in a projected 2D/3D space rather than selecting the candidate at random. The boundary candidates are found using α-shapes [5]. α-shape of a finite point set is a shape that envelops a set of points, and the shape's smoothness is determined by the parameter α. In our case, α is set to 0, which makes the formed α-shape its convex hull, which is the smallest convex set that includes all the points.

Also, we aim to find an approach to discover an optimal set of hyperparameters with minimal evaluations. Instead of exploring all possible combinations of hyperparameters, we seek to find a method to explore only a subset of representative hyperparameters. `SWAY` takes a large input space of possible explanatory variables and only evaluates examples to a log degree of the original dataset size. Using this same technique, we will use `SWAY` to explore the set of hyperparameters to pass to `SWAY`.

Finally, we will determine if `EXPLAIN` produces rules that select a "better" subset of the data than the "rest" of the data. We will then explore another rule set generator based on decision trees, and better understand if this decision tree classifier can also select a "better" subset of the data than the "rest" of the data.

A. Structure of this paper

Throughout this paper, the experiments we conduct attempt to answer three main research questions. We seek to answer these questions on 11 total datasets, but will generalize the results on datasets of small, medium, and large size. Ideally, our algorithms should be general enough to perform well on datasets of varying dimensionality.

1) RQ1: Can the default `SWAY` hyperparameters be optimized further?: The first research question deals with finding the optimal combination of hyperparameters for `SWAY`. The goal is to determine a technique to tune the hyperparameters and assess the improvements (if any) in the performance of the algorithm. We later show that optimizing `SWAY`'s hyperparameters leads to more optimal results.

2) RQ2: Does selecting the starting candidate from the boundary of candidates for `SPLIT` in `SWAY` improve the performance of the algorithm?: The second research question explores the effect of selecting the starting candidate for `SPLIT` from the boundary candidates which would otherwise be selected at random. This approach would involve mapping the candidates to a 2D/3D space using dimensionality reduction techniques. We show that this technique does not fare well against larger datasets, and in the case of the smaller datasets, it aggressively dominates some objectives and does slightly worse compared to `SWAY` in the rest of the objectives.

3) RQ3: Does using a tree-like explanation algorithm yield positive results?: The final research question will evaluate using a tree-like explanation algorithm, like a decision tree. To test this, we will examine the results of the decision tree technique with EXPLAIN. To see if the results are positive, we will check if the average rank of the "best" data is higher than 50%. We later show that using a tree-like explanation algorithm does yield positive results, and has slight improvements over the baseline EXPLAIN.

B. Overall contributions

Using SWAY to determine an optimal set of hyperparameters has vast implications for both machine learning and more specifically multi-objective semi-supervised learning. This novel technique will provide a semi-supervised method of optimizing the hyperparameters. This technique would not only be more cost-efficient than grid search and random search but may possibly provide a better hyperparameter combination. For algorithms such as neural nets that have a large hyperparameter subspace and a high cost of evaluations, using SWAY will drastically decrease the time required to tune such algorithms.

This paper improves the performance on SWAY by incorporating α-shapes [5], which is a technique from computational geometry. α-shapes creates a series of nested shapes around a collection of points or objects. The degree of complexity and amount of detail in each shape is determined by a unique α value. It capture various degrees of approximation of the input data by changing the alpha value. A smaller α value will produce an α-shape that is more detailed and resembles the input data more closely. Although the shape becomes smoother and simpler as the α value is raised, this could result in the loss of finer features in the original data [5]. Adding dimensionality reduction to the pipeline also minimizes over-fitting problems and normalizes the effect of outliers as well as mitigates domination of one of the objectives.

Using decision trees with the gini criterion as an explanation system will help further users' understanding of machine learning models and why they made their decisions. A tree-like structure, instead of bins, helps users "walk" through the tree and visually understand what features are important to the machine learning algorithm.

C. Caveats

When modifying SWAY using α-shapes, it is crucial to carefully choose the value of the α parameter, as the algorithm is sensitive to this choice. Obtaining a meaningful α-shape relies on selecting the correct α value, which can be challenging, especially in high-dimensional spaces, and may require experimentation or prior knowledge. Additionally, the α-shapes algorithm can produce ambiguous and non-unique results, particularly when dealing with point sets that contain noise, outliers, or non-uniform distribution. This is because the algorithm can capture irrelevant features or miss important ones, depending on the point density, distribution, and the shape and size of the alpha parameter.

When using the DecisionTreeClassifier for EXPLAIN2, for the purpose of this study, only the default hyperparameters were used, and not optimized further. This is because performing a hyperparameter optimization can be costly, and our research questions involve only optimizing hyperparameters for SWAY.

At each step of the recursion in SWAY, we take a small sample from the input data at random. There is a high chance that data points of significant importance may be ignored. Also, the stochastic nature of the algorithms generates an internal bias that has not been mitigated.

For the explanation algorithm, we did not explore post-hoc explanation algorithms, such as LIME. This is because those explanation algorithms require generating candidates and calling the evaluation function multiple times to understand better what features are important. For the purpose of this study, where we try to have a low budget, it does not fit into our requirements.

II. Related Work

Common MOEA algorithms include improving candidates using genetic algorithms, by multiple generations of mutating, crossover, and selection. Related work to hyperparameter tuning has actually shown for some applications, where the input space is large and requires exhaustive search, using the default parameters is often more advantageous than exploring via genetic algorithms [9].

Although those algorithms yield positive results, they also generate a larger population, and do not narrow down the current population. In addition, common algorithms call the evaluation function multiple times, where our goal is to minimize calling the evaluation function.

The current implementation of SWAY includes several hyperparameters. Although they might have worked on several datasets in the past, it is possible that these hyperparameters are not optimal for our datasets. So, more combinations of hyperparameters must be explored to better understand if the baseline hyperparameters are sufficient.

SWAY has been shown to outperform common MOEA algorithms, such as NSGA-II and SATIBEA [2]. Therefore, the baseline SWAY will be used to compare the performance of the new, optimized SWAY2.

α-shapes has applications [7] mainly in pattern recognition, digital shape sampling and processing, and structural molecular biology but has also been used in cluster analysis [6].

For explanation algorithms, most work includes post-hoc explanation algorithms, which include explaining why an algorithm made certain decisions, after it has been trained [10]. This is not effective for our purposes, as it leads to a high number of evaluations. Our baseline explanation algorithm is an ante-hoc explanation algorithm that generates rules from different "bins" of data that differ the "better" set of data from the "rest" of the data. This can lead to less flexibility, because there is not a notion of different if statements based on previous branches.

Algorithms include a notion of "levels" include tree-like classifiers where there are binary choices per node, leading to a "branches" of choices. Previous work has shown that decision tree classifiers are effective as anomaly detection schemes [11]. We can apply the same logic, using decision trees to detect "better" records. For our purposes, we will compare a new type of explanation systems, decision trees, to the baseline explanation system.

Dimensionality reduction methods produce a low-dimensional linear mapping of the original high-dimensional data. They compress data and reduce the compute time of running algorithms on high-dimensional data. Moreover, these methods help reduce the effects of noisy data on exploratory and predictive models. They also tackle the issue of overfitting when algorithms are run on small high-dimensional datasets (with fewer input rows and many features).

III. Methods

A. Algorithms

1) Sway: SWAY is an algorithm that recursively clusters candidates in order to isolate the superior cluster.

During each evaluation, the SWAY algorithm tries to find a set of low-dimension coordinates for each data point that minimizes the

TABLE I
SWAY2 (SWAY FOR HYPERPARAMS) SEARCH SPACE

Name	default	range	step size	result
Far	0.95	[0.7,0.95]	0.05	0.75
Halves	512	[100,50]	100	500
Min	0.5	[0.0, 0.60]	0.20	0.2
Max	512	[1, 126]	25	26
P	2	[1.0, 1.90]	0.1	1
Rest	10	[1, 4]	1	3
Reuse	True	[False, True]	-	True

discrepancy between the pairwise distances in the high-dimensional space and the low-dimensional space.

It does this by comparing rows using a betters function, which is based on domination via the Zitzler common domination predicate, which says that one example is preferred to another if we lost the least jumping to it.

Sways advantage is that is has a small number of evaluations. For example, to split the dataset 4 times, if the "reuse" parameter is set to true, there are only 5 evaluations taken. If the "reuse" parameter is set to false, there are only 8 evaluations. This is highly advantageous for large datasets, as it is roughly log(n) evaluations.

2) Sway for Hyperparameters: One disadvantage to `SWAY`, as described in the above subsection, is that it has multiple hyperparameters to tune. Using the default hyperparameters is sufficient for the task, but has a chance of not being optimal. Table I shows a summary of such hyperparameters, and the optial hyperparameter search space for `AUTO2`.

Currently, the approach to tuning hyperparameters is by approaching it using a "gridsearch". A gridsearch computes all combinations of hyperparameters, and runs the algorithm on each combination of hyperparameter, and returns the "best" result. The positive of this approach is that since it searches over all combinations of hyperparameters, you will get the exact hyperparameter combination that yields the best result. The negatives include that this requires a large amount of space and memory to store all combinations, and a large time complexity. In addition, for our purposes of requiring a low number of evaluations, this would very quickly use up our budget, without being able to explore all parameters.

An alternative is to instead use `SWAY` to find the most optimal hyperparameter set. This is done by creating a Data object with all combinations of hyperparameters to `SWAY`. Then, when `SWAY2` (`SWAY` for hyperparameters) is called, the "evaluation" is creating a new Data object of the original file, and performing `SWAY` on that Data object. When comparing two runs, the "better" set of hyperparameters is the one that performs best according to the Ziltzler Predicate of the two runs. This will be conducted on the `AUTO2` dataset, since it is a smaller dataset with average dimensionality.

The ranges of hyperparameters were chosen based on the default value, and the maximum value. When experimenting with step size and maximum value, if the algorithm was run and the result included a value close to one of the "axis" of the hyperparameters, the range for that hyperparameter was expanded.

Instead of running `SWAY` on the original dataset o(n * m) times, where n is the size of the product of hyperparameters and m is the number of rows in the dataset, it is instead o(log(n) * log(m)).

3) Sway with Principal Component Analysis and α-shapes: This approach tries to improve the performance of `SWAY` by reducing the stochasticity of the `SPLIT` method. In the `SPLIT` method of `SWAY`, the selection of the "left" most point is done at random. We propose that the selection of this initial candidate closer to the boundary would yield a better axis to split the candidates into the "best" and "rest" clusters. The boundary of a finite set of points in a two or three-dimensional space can be calculated by using α-shapes. α-shapes only work with data points in two or three-dimensional spaces.

The input data can be mapped to a two or three-dimensional space using dimensionality reduction techniques. In our algorithm, we use Principal Component Analysis (PCA) to reduce the dimensions. Also, before reducing the data, we normalize the data using Z-score normalization, which normalizes each point such that the mean is 0 and the standard deviation is 1.

Once the data is mapped to the reduced dimensions, all the operations ahead in the algorithm are done in this reduced space which would reduce the runtime of the operations. Cosine similarity is used to project all the points onto the axis formed by the "left" most and the "right" most points.

The disadvantage of this approach is that we are choosing the axis with the greatest range rather than the one with the highest dispersed density. While this approach leads to significant improvements, it may fail in certain situations.

4) Explain: The baseline explanation algorithm procedurally builds rules using "bins" of data. It attempts to find simple rules to follow to distinguish the best data from the rest of the data.

First, the `SWAY` algorithm is explored, separating the data into "best" and "rest". Then, the algorithm finds ranges that distinguish the "best" data from the "rest" of the data. Using those ranges, explain then attempts to find the best set of rules that select for the best cluster.

For each range, to place a value on how well it selects the best rows, it uses probability times support, and sorts each range by the highest value. Rules are then generated using the first item, first two items, etc.

The advantage of `EXPLAIN` is that it does not require any extra evaluations, and puts the classification into easy-to-understand rules. The disadvantages of explain include that it includes hyperparameters that need to be tuned, and returns only one rule, not multiple. Since the rules are conjunctive across attributes and disjunctive for one attribute with multiple ranges, it does not allow for as much flexibility as, for example, tree-based learners allow.

5) Decision Tree: Decision tree learning is a supervised learning approach where leaves represent class labels and branches represent conjunctions of features that lead to those class labels.

At each level, the decision tree chooses an attribute that divides the data into more "pure" subsets, where the split of data has a higher class proportion in one or both subsets of data. This process is repeated recursively, and the final "tree" involves yes or no questions at each node to determine the next node to proceed to.

An advantage of decision tree learning is that it is simple to implement and understand. The decision trees can be viewed visually such that programmers and stakeholders can visually understand what decisions were taken to classify the final result. The decision trees can also handle both numerical and categorical data and perform well with large datasets, which will be advantageous for some of our datasets.

For our purposes, decision trees will be used to classify data into the "best" and "rest" partitions.

B. Data

The data explored for this paper include 11 different datasets, of varying length and dimensionality. We will further explain the auto2 dataset, as it is used for hyperparameter tuning, but present a summary table for all other datasets.

TABLE II
AUTO2

Feature	Type	Mid	Div
Clndrs	Num	4	1.6
Volume	Num	146	100.8
Lbs-	Num	2800	887.2
Acc+	Num	15.5	2.7
Model	Num	76	3.9
origin	Sym	1	1.3
Mpg+	Num	20	7.8

1) auto93: auto93 is a dataset with 93 rows and 23 columns.The multi-goal optimization includes maximizing City and Highway miles per gallon while minimizing weight and class.

2) auto2: auto2 is a dataset with 398 instances and 8 attributes of cars, and is a slightly modified version of a StatLib dataset. The multi-goal involve attempting to maximize the acceleration and miles per gallon, while minimizing the weight of the car. This dataset is used for SWAY2 hyperparameter tuning, see Table II for more information about the distribution of the dataset. This dataset is useful for the hyperparameter tuning because it is a smaller dataset with average dimensionality, so the budget of running SWAY multiple times on this dataset will be smaller than other datasets, and it will generalize well for other datasets. When we ran SWAY for SWAY of the optimal hyperparameter space using AUTO2, it took 156 evaluations to determine the optimal hyperparameters, then all other runs took an average of 8 evaluations.

3) china: china is a dataset with 499 rows and 17 attributes, in the software project estimation domain. The multi-goal objective includes only minimizing the N_Effort

4) coc1000: coc1000 is a dataset with 1000 rows and 22 attributes, in the software project estimation domain. The multi-objective includes minimizing applications experience, platform experience, and risk, while maximizing KLOC.

5) coc10000: coc10000 is a dataset with 10,000 rows and 22 attributes, in the software project estimation domain. The goals and attributes are the same as coc1000.

6) healthCloseIsses12mths0001-hard: healthCloseIsses12mths0001-hard is a dataset with 10,000 rows and 8 attributes, in the issue close domain. The multi-goal optimization include minimizing the magnitude of relative error, and maximizing the accuracy and percentage of relative error deviation

7) healthCloseIsses12mths0011-easy: healthCloseIsses12mths0011-easy has the same domain as the above dataset, except with an "easier" set of data.

8) nasa93dem: nasa93 dem has 94 rows and 26 attributes, in the software effort and detects estimation domain. The multi-goal optimization includes maximizing Kloc while minimizing Effort, Defects, and Months.

9) pom: pom is a dataset with 10,000 rows and 13 attributes, in the agle project management domain. The multi-goal optimization include minizing cost and idle time while maximizing completion rate

10) SSM: SSM is a dataset with 239,360 rows and 15 attributes, in the computational physics domain. The multi-goal optimization involves parameters for a mesh solver, Trimesh, and involves minimizing the number of iterations and time to solution.

11) SSN: SSN is a dataset with 53,662 rows and 19 attributes, in the computational physics domain. The multi-goal optimization includes optimizing a video encoder, X264-DB, and minimizing the PSNR and Energy.

TABLE III
DATASETS

Dataset	Number of rows	Number of x	Number of y
auto2	398	4	3
auto93	93	19	4
china	499	16	1
coc1000	1000	17	5
coc10000	10000	22	3
healthHard	10000	5	3
healthEasy	10000	5	3
nasa93dem	93	22	4
pom	10000	10	3
SSM	239360	13	2
SSN	53662	17	2

TABLE IV
EXPERIMENT PARAMETERS

Name	Description	value	sway2 value
bins	initial number of bins	16	-
cliff	cliff's delta threshold	.147	-
D	different is over sd*d	.35	-
Far	distance to distant	.95	0.75
Halves	search space for clustering	512	500
Min	size of smallest cluster	.5	0.2
Max	numbers	512	26
P	dist coefficient	2	1
Rest	how many of rest to sample	10	3
reuse	child splits reuse a parent pole	true	true
Bootstrap	number of samples to bootstrap	512	-
Conf	confidence interval	0.05	-
Niter	number of iterations to run	20	-

Our datasets, as seen in table III, have a wide range of dimensionality. The smaller datasets, auto2, auto93, china, and nasa93dem will have the challenge of not having a large sample space, but will have the advantage of having a fast running time. The remaining larger datasets will have the challenge of a large dimension space to explore, and longer time to run.

Ideally, the SWAY2 and xpln2 algorithms will be flexible enough to perform well for small, medium, and large datasets.

C. Performance measures

Before all algorithms are run, using all attributes of data, the data was ranked using the Zitzler predicate, normalized from 1 to 100, inclusive. This is useful for the summarization methods, where we can quantify what results included the "better" rows of the data.

In order to evaluate the effectiveness of the algorithms, an experiment was run, where over 20 iterations, for a given limited budget B, each algorithm attempted to guess the top B items of the data. The default parameters can be seein in Table IV. Then, we evaluated the top B guesses, in order to understand the distribution of the data to see which algorithm performed the best. The distributions of those final B evaluations were collected, and summarized per algorithm.

The baseline sampling budget, B_0 is 10, but there is one exception, for the SWAY2 algorithm, which is SWAY using hyperparameter tuning, the budget will be higher, in some cases, than the default 10.

D. Summarization methods

Collecting and summarizing the distribution allows us to visually compare how the algorithms perform, but we will also use statistical measures and tests to further quantify if the distributions of results are different per algorithm.

Overall, better methods will have a lower sum of the final B results, ranked by using the Zitzler predicate. Over each iteration, the selected rows stats were summarized, as well as average number of evaluations, and average rank of results.

TABLE V
AUTO2 RESULTS

	Lbs-	Acc+	Mpg+	Avg evals	Avg rank	Avg Time Taken
all	2800	15.5	20	0	50.4	0
sway	2169.2	16	31	6	23.6	0
sway2	2079.8	17.8	34.5	8	18.2	0
sway3	2234.6	16.7	29.5	6	28	0.1
xpln	2216.5	16	30	6	28.7	0
xpln2	2161.1	16	31.5	6	23.7	0
top	1987	18.8	40	398	2.4	0

TABLE VI
COC10000 RESULTS

	Loc+	Risk-	Effort-	Avg evals	Avg rank	Avg Time Taken
all	931	5	21427	0	50.5	0
sway	979.6	3.2	16207.7	8	45.1	1
sway2	987.6	2.7	17632.5	12	38.2	1
sway3	992.6	4.7	18366.7	8	49.2	23.6
xpln	1049.2	5.8	19832	8	47.9	0.2
xpln2	1004.6	4.8	17713.3	8	49.2	0
top	1959	0	17733	10000	1.4	0

TABLE VII
SSN RESULTS

	PSNR-	Energy-	Avg evals	Avg rank	Avg Time Taken
all	45.7	1237.7	0	50.5	0
sway	44.6	954.3	9	41.3	4.1
sway2	39.8	736.3	14	33.6	4.1
sway3	44.3	1108.5	9	46.2	102.7
xpln	44.6	821	9	41.3	0.2
xpln2	45.1	910.4	9	44.4	0
top	26.7	465.9	53662	1.2	0

TABLE VIII
RESULTS OF ALL DATASETS

name	sway2 improved	sway3 improved	xpln2 improved
auto2	3/3	1/3	2/3
auto93	4/4	3/4	1/4
china	1/1	1/1	1/1
coc1000	3/5	3/5	2/5
coc10000	2/3	1/3	2/3
healthHard	2/3	2/3	0/3
healthEasy	1/3	0/3	0/3
nasa93dem	3/4	0/4	1/4
pom	2/3	0/3	3/3
SSM	2/2	0/2	1/2
SSN	2/2	0/2	0/2

To show that the distributions of results are statistically significant, a table was also generated for each pair of algorithms, and each y value, performing the conjunction of an effect size test and a significance test. The tests that were performed were the bootstrap and CliffsDelta test. If results were not statistically significant, they were omitted from the chart.

In order to determine if a given algorithm is "better" than another, we will examine the average rank, the mean of each y column, and ensure that the comparisons mark the two algorithms as not equal.

IV. RESULTS

The table VIII shows the aggregated results across all datasets. The column "sway2 improved" indicates the percentage of columns that have a better value in sway2 versus sway1. The column "xpln2 improved" indicates the percentage of columns that have a better value in xpln2 versus xpln1. Green indicates all columns improved, yellow indicates some columns improved, and red indicates no values improved.

The questions we seek to answer throughout this research paper is are our improvements to SWAY2, SWAY3, and xpln2 positive, and if they are generalized for datasets of different sizes. The algorithms were run on all 11 datasets, but in order to make the results concise, we only examine 3 dataset results, of different dimensionality. The results can be seen summarized in table VIII.

For a small dimensionality, auto2, the results in Table V show that for a small increase in evaluations, 8 versus 6, there are drastic increases in the performance of SWAY2. For the first evaluation where we run SWAY on SWAY to determine hyperparameters, the number of evaluations was 156. The average rank is 5 points lower and SWAY2 outperformed the baseline algorithm on all columns.

For a medium dimensionality, coc10000, the results in Table VI show that for a small increase in evaluations, 12 versus 8, there are drastic increases in the performance of SWAY2. The average rank is 7 points lower, and there is only one column, Effort- that SWAY2 did not outperform the baseline algorithm on.

For a large dimensionality, SSN, the results in Table VII show that for a small increase in evaluations, 14 versus 9, there are drastic increases in the performance of SWAY2. The average rank is 8 points lower, and for all columns, SWAY2 outperforms the baseline algorithm.

1) RQ1: Can the default SWAY hyperparameters be optimized further?: The goal for the first research question was to determine a technique to tune the hyperparameters and assess the improvements (if any) in the performance of the algorithm. Based on table VIII, we are able to see that SWAY2 outperformed the original parameters on all columns for 5 out of the 11 datasets. For the rest of the datasets, SWAY2 outperformed in atleast one column. An important thing to note is that SWAY2 always outperformed SWAY for at least one column.

We are able to conclude that for a small increase in the number of evaluations, SWAY2 shows that using SWAY for the hyperparameters of SWAY improves the performance.

2) RQ2: Does selecting the starting candidate from the boundary of candidates for SPLIT in SWAY improve the performance of the algorithm?: The second research question explores the effect of selecting the starting candidate for SPLIT from the boundary candidates which would otherwise be selected at random. This approach would involve mapping the candidates to a 2D/3D space using dimensionality reduction techniques and using α-shapes to select initial candidates from the boundary candidates. For most of the small datasets, SWAY3 performed significantly well for a few objectives and worse than SWAY in the rest. For the larger datasets, SWAY3 does not perform well.

One important point is that the compute time is high for SWAY which can be attributed to the dimensionality reduction step and calculation of the α-shape of the points.

3) RQ3: Does using a tree-like explanation algorithm yield positive results?: The final research question will evaluate using a tree-like explanation algorithm, like a decision tree. To test this, we will examine the results of the decision tree technique with EXPLAIN. To see if the results are positive, we will check if the average rank of the "best" data is higher than 50%.

The goal for the first research question was to determine if using a tree-like explanation algorithm yielded positive results. Based on our small, medium, and large representative datasets, the average rank for XPLN2 was less than 50%. This shows that XPLN2 does better than make random rules, which shows that the algorithm yields positive resuts. To quantify how positive the results were, and if it improved XPLN1, we refer to Table VIII. Based on table VIII, we are able to see that XPLN2 outperformed the original algorithm in all columns for 3 out of the 11 datasets. For 6 of the datasets, XPLN2 outperformed

in at least one column. It was only for two datasets, healthHard and healthEasy that XPLN2 did not improve on the original algorithm. We are able to conclude that XPLN2 yields positive results, although it does not improve on XPLN1 for all datasets.

V. Discussion

1) Threats to Validity: Dimensionality reduction (especially linear techniques) fails to preserve certain features of interest as the data points lose some details while being mapped to a 2D/3D space. Also, the use of dimensionality reduction increases the explanation tax of the algorithm as the data points are no longer directly representative of their original attributes. This approach has shown promising results for datasets with lower dimensions. However, this method may lead to significant information loss when dealing with higher dimensional data.

Integrating α-shapes to select boundary points may help get better clusters in some cases, but does not consider the actual density distribution of the data points in the space. This technique assumes that the best split for the clusters is perpendicular to the axis with the highest range. This assumption may not hold true for real-world data.

At each step of the recursion in SWAY, we randomly take a small sample from the input data. There is a high chance that data points of significant importance may be ignored. Also, the stochastic nature of the algorithms generates an internal bias that has not been mitigated.

2) Future work: Future work for SWAY2 includes exploring a broader range of hyperparameters. This can involve taking smaller step sizes, expanding the range, and exploring new distance functions or domination predicates. By making these changes in the future, the algorithms performance can be further enhanced. Future work for EXPLAIN2 includes experimenting with different tree-based classifiers including RandomForestClassifier and ExtraTreesClassifier.

Using AlphaShapes after applying PCA to the dataset has resulted in significant improvements over the baseline SWAY. It should be noted that the AlphaShapes algorithm is highly sensitive to the α parameter, which controls the level of detail and complexity of the shapes identified. If the value of alpha is too low, the algorithm may miss important structures and produce an incomplete or inaccurate representation of the data, while if it is too high, the algorithm may capture irrelevant or spurious features and produce an overly complex or noisy representation of the data. In our experiments, we used the same α parameter for all the datasets, which is why the baseline SWAY outperformed the modification by PCA and AlphaShapes in a few datasets. Therefore, determining the optimal value of alpha is crucial for achieving better results and a more meaningful representation of the data [12].

VI. Conclusion

Throughout this paper, our goal was to explore three different research questions:

1) RQ1: Can the default SWAY hyperparameters be optimized further?
2) Q2: Does selecting the starting candidate from the boundary of candidates for SPLIT in SWAY improve the performance of the algorithm?
3) Q3: Does using a tree-like explanation algorithm yield positive results?

By conducting an experiment across 11 different datasets, with 5 different algorithms, we were able to show that using SWAY2, or SWAY for SWAY yields better results than the default sway. SWAY3 does not perform well for large datasets. Using a decision tree as an explanation algorithm also yields weakly stronger results than the baseline explanation algorithm.

These results are widely applicable to multi-goal optimization, hyperparameter optimization, and explanation algorithm applications. Using SWAY with the more optimal parameters presents a more optimal SWAY algorithm, and can be used to re-evaluate previous experiments to determine if the default hyperparameters should be changed.

References

[1] N. Saini and S. Saha, "Multi-objective optimization techniques: a survey of the state-of-the-art and applications: Multi-objective optimization techniques." The European Physical Journal Special Topics 230, no. 10 (2021): 2319-2335.

[2] J. Chen et al, ""Sampling" as a Baseline Optimizer for Search-Based Software Engineering," IEEE Transactions on Software Engineering, vol. 45, (6), pp. 597-614, 2019.

[3] M. Harman, Y. Jia, J. Krinke, W. B. Langdon, J. Petke, and Y. Zhang. "Search based software engineering for software product line engineering: a survey and directions for future work." In Proceedings of the 18th International Software Product Line Conference-Volume 1, pp. 5-18. 2014.

[4] Marinov, D. and Karapetyan, D., "Hyperparameter optimisation with early termination of poor performers." In 2019 11th Computer Science and Electronic Engineering (CEEC), pp. 160-163. IEEE, 2019.

[5] N. Akkiraju, H. Edelsbrunner, M. Facello, P. Fu, E. P. Mücke, and C. Varela, "Alpha shapes: definition and software." In Proceedings of the 1st international computational geometry software workshop, vol. 63, no. 66. 1995.

[6] H. Edelsbrunner and E. P. Mücke, "Three-dimensional alpha shapes." ACM Transactions On Graphics (TOG) 13, no. 1 (1994): 43-72.

[7] H. Edelsbrunner, "Alpha shapes-a survey." In Tessellations in the Sciences: Virtues, Techniques and Applications of Geometric Tilings. 2011.

[8] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.

[9] A. Arcuri and G. Fraser, "Parameter tuning or default values? An empirical investigation in search-based software engineering," Empirical Software Engineering : An International Journal, vol. 18, (3), pp. 594-623, 2013.

[10] M. T. Ribeiro et al, ""why should I trust you?": Explaining the predictions of any classifier," in 2016, . DOI: 10.1145/2939672.2939778.

[11] T. Amraee, "Loss-of-field detection in synchronous generators using decision tree technique," IET Generation, Transmission & Distribution, vol. 7, (9), pp. 943-954, 2013.

[12] Yunfan Li, Guang Gao, Bo Cao, Liang Zhong and Yao Liu, "Building boundaries extaction from point clouds using dual-threshold Alpha Shapes," 2015 23rd International Conference on Geoinformatics, Wuhan, 2015, pp. 1-4, doi: 10.1109/GEOINFORMATICS.2015.7378619.

[13] L. Yang and A. Shami, "On hyperparameter optimization of machine learning algorithms: Theory and practice," Neurocomputing (Amsterdam), vol. 415, pp. 295-316, 2020.