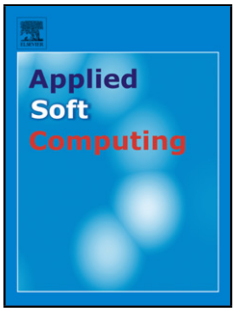


Applied
Soft
Computing

Title: A new evolutionary algorithm for mining top-*k* discriminative patterns in high dimensional data

Author: TarcÃ-sio Lucas TÃºlio C.P.B. Silva Renato Vimieiro Teresa B. Ludermir

PII: S1568-4946(17)30313-7
DOI: http://dx.doi.org/doi:10.1016/j.asoc.2017.05.048
Reference: ASOC 4252

To appear in: *Applied Soft Computing*

Received date: 30-12-2016
Revised date: 17-5-2017
Accepted date: 24-5-2017

Please cite this article as: TarcÃ-sio Lucas, TÃºlio C.P.B. Silva, Renato Vimieiro, Teresa B. Ludermir, A new evolutionary algorithm for mining top-*k* discriminative patterns in high dimensional data, *<![CDATA[Applied Soft Computing Journal]]>* (2017), http://dx.doi.org/10.1016/j.asoc.2017.05.048

A new evolutionary algorithm for mining top-*k* discriminative patterns in high dimensional data

Tarcísio Lucas[a,*], Túlio C. P. B. Silva[a], Renato Vimieiro[a], Teresa B. Ludermir[a]

[a] *Centro de Informática, Universidade Federal de Pernambuco,*
Av. Jornalista Anibal Fernandes, s/n - Cidade Universitária (Campus Recife)
50.740-560 - Recife - PE - BRAZIL

Abstract

This paper presents an evolutionary algorithm for Discriminative Pattern (DP) mining that focuses on high dimensional data sets. DPs aims to identify the sets of characteristics that better differentiate a target group from the others (e.g. successful vs. unsuccessful medical treatments). It becomes more natural to extract information from high dimensionality data sets with the increase in the volume of data stored in the world (30GB/s only in the internet). There are several evolutionary approaches for DP mining, but none focusing on high-dimensional data. We propose an evolutionary approach attributing features that reduce the cost of memory and processing in the context of high-dimensional data. The new algorithm thus seeks the best (top-*k*) patterns and hides from the user many common parameters in other evolutionary heuristics such as population size, mutation and crossover rates, and the number of evaluations. We carried out experiments with real-world high-dimensional and traditional low dimensional data. The results showed that the proposed algorithm was superior to other approaches of the literature in high-dimensional data sets and competitive in the traditional data sets.

Keywords: subgroup discovery, evolutionary algorithms, discriminative patterns, high dimensional data.

1. Introduction

This paper presents an evolutionary algorithm for Discriminative Pattern (DP) mining that focuses on high dimensional data sets. Discriminative pattern mining is a data mining task that has the objective of identifying sets of items that distinguish a target group from the others, for example: successful from unsuccessful treatments, unhealthy from healthy cells, spam from other emails, or even positive from negative sentiments in sentiment analysis. The necessity to investigate new methods, especially heuristics methods mining these patterns, comes from the fact that data generated/collected from many domains have different characteristics from those of last decade. The vast amount and high dimensionality of data sets in this so-called *Era of Big Data* render the application of existing methods infeasible.

Studies estimate that in the internet alone around 30GB of data, including texts, images and videos, are produced each second. Another important source of data is the biomedical sciences, particularly the *Omics* (genomics, proteomics, transcriptomics, ...), since the price for sequencing samples has dramatically dropped in the last years. These areas have two things in common: (1) they are major contributors to today's massive amount of available data (big data); (2) data from these domains is usually very high dimensional with tens of thousands to millions of attributes. In this sense they present new challenges to data mining and machine learning researchers. Among these challenges is the need for new tools for exploratory data analysis.

Discriminative patterns are an important tool for exploratory data analysis, since recurring patterns in the data are summarized in a simple way [1]. This is particularly suitable to explaining/describing differences among groups

[*] Corresponding author: Tel.: +55 81 2126-8430 (x4074), fax: + 55 81 2126-8438
Email addresses: tdpl@cin.ufpe.br (Tarcísio Lucas), tcpbs@cin.ufpe.br (Túlio C. P. B. Silva), rv2@cin.ufpe.br (Renato Vimieiro), tbl@cin.ufpe.br (Teresa B. Ludermir)

of samples in the data. Discriminative pattern mining has simultaneously evolved with different terminologies, *Subgroups Discovery* [2, 3]; *Emerging Patterns* [4, 5]; and *Contrast Sets* [6], until they were unified by Novak et al. [7]. There are many applications reported in the literature in different domains such as: medicine [8, 9], bioinformatics [10, 11], marketing [12, 13], e-learning [14] and traffic accidents [15, 16].

Little attention has been given to mining discriminative patterns in high dimensional domains in spite of the great number of applications in the literature. High dimensionality of data sets is an intricate problem for current methods for discriminative pattern mining. It represents a computationally difficult problem for most of existing methods because of their combinatorial nature. Most of the exact methods, e.g. [17, 18], enumerate subsets of attributes, avoiding and discarding paths in the search space that exclusively yield uninteresting patterns. In fact Vimieiro [19] already discussed in 2012 the issues related to exact methods for mining discriminative patterns. He argues that the feasibility of such methods is not only limited by computational aspects (time and memory usage), but also by the number of returned patterns. In many occasions the problem is just shifted from analyzing raw data to analyzing a huge number of patterns. This motivates the investigation of heuristics for mining discriminative patterns.

There are plenty of heuristics for mining discriminative patterns, including many based on *evolutionary computing* [13, 20–26]. The vast majority of these methods target traditional, low-dimensional data sets. As their exact counterparts, they also use interestingness measures to guide the search. These constraints are mostly related to the frequency (support) and discriminative power of patterns. Thus, they explicitly deal with the computational issues associated to exact methods, but might not solve the second issue related to the number of patterns. The algorithms for mining discriminative patterns usually return the best patterns in one of two ways: (1) based on constraints, which return patterns that satisfy some constraint, as minimum support; and (2) based on top-k, which return the k best patterns. Both options have their relevance depending on the analysts' goals, but the top-k approach provides more flexibility [2]. Notwithstanding, an evolutionary top-k DPs mining approach has not been proposed yet.

This context motivates us to pose the following research question: *is it possible to devise a new evolutionary heuristic that tackles both the combinatorial issues and huge amount of patterns associated with high dimensional data?* To address this question, we present a new evolutionary heuristic SSDP (Simple Search Discriminative Patterns). We aim at providing end-users a viable and easy to use tool for analyzing high dimensional data. Our approach allows the user to choose the most appropriate interestingness measure and requires only the number of patterns that she intends to analyze. The algorithm then seeks the best (top-k) patterns, hiding from the user many common parameters in other evolutionary heuristics such as population size, mutation and crossover rates, and the number of evaluations.

SSDP was first presented as a preliminary work at the 5th Brazilian Conference on Intelligent Systems (BRACIS 2016) [27]. However, we made additional progress as following. We improved our experiments to assess the performance of our approach with both real-world high-dimensional and traditional low dimensional data. We compared the results from our algorithm with other traditional evolutionary methods, which had not been previously done. The aim of these new experiments was to evaluate both the effectiveness of SSDP on mining high-dimensional data, which it has been designed for, and its suitability to different contexts (low dimensional data, which it has not been designed for). Since we omit many common parameters as discussed above, we also conducted experiments to investigate different settings of these parameters and their impact on our method. Such an analysis had not been done in the previous conference paper, despite being extremely important to confirm whether the choices made indeed return relevant patterns compared to other settings. Finally, we also revised the entire manuscript and made significant changes to improve its readability.

The remainder of this manuscript is organized as follows. We formalize the problem of mining discriminative patterns in Section 2. We formally define the concept of a discriminative pattern and the interestingness measures to assess its relevance. In Section 3, we review the literature, providing a critical analysis of the state of the art. We identify the issues related to the current methods for mining discriminative patterns. We present our algorithm in Section 4. Then we discuss the experiments conducted to assess the performance of our algorithm and compare the results with other algorithms in Section 5. We conclude the manuscript with some final remarks in Section 6.

2. Discriminative Patterns

Let D be a labeled data set with a set A of categorical/discrete attributes. According to the class label, the set of samples from D can be partitioned into $D^+ = \{e_1^+, e_2^+, ..., e_{|D^+|}^+\}$ and $D^- = \{e_1^-, e_2^-, ..., e_{|D^-|}^-\}$, respectively the positive

Table 1: A toy example of a data set. In this simulated data, the target is to identify the differences between successful and unsuccessful medical treatments for a given disease.

example	genre	age	medicine	label
e_1	M	senior	B	success
e_2	F	senior	B	success
e_3	M	senior	A	success
e_4	M	adult	A	success
e_5	F	child	A	success
e_6	F	child	A	failure
e_7	M	child	B	failure
e_8	F	child	B	failure
e_9	M	adult	A	failure
e_{10}	F	adult	A	failure

Table 2: Universe of items I and respective covered examples for the data presented in Table 1. In this table, items are in rows and examples in columns. There is a cross if an example has the corresponding item.

I	(attribute, value)	D^+					D^-				
		e_1	e_2	e_3	e_4	e_5	e_6	e_7	e_8	e_9	e_{10}
i_1	(genre, M)	×		×	×			×		×	
i_2	(genre, F)		×			×	×		×		×
i_3	(age, child)		×			×	×		×		
i_4	(age, adult)				×					×	×
i_5	(age, senior)	×	×	×							
i_6	(medicine, A)			×	×	×	×			×	×
i_7	(medicine, B)	×	×					×	×		

(target) examples and the remaining (negative examples). Let $dom(A_i)$ be the domain of values for attribute $A_i \in A$. We call features or items the set of all pairs $(attribute, value)$, that is $I = \bigcup A_i \times dom(A_i) = \{i_1, i_2, ..., i_{|I|}\}$. We say that an example d has an item $x = (A_i, v) \in I$ if d has value v for the attribute A_i.

We call a *discriminative pattern* a set $dp \subseteq I$. The *size* of a discriminative pattern dp is the number of items in dp, that is $size(dp) = |dp|$. Every dp might be associated (cover) a set of positive and negative examples, which we formally define as $c^+(dp) = \{d \in D^+ \mid d \text{ has all items in } dp\}$, and $c^-(dp) = \{d \in D^- \mid d \text{ has all items in } dp\}$. The size of these two sets define the *positive* and *negative support* of a discriminative pattern, i.e. its frequency among positive and negative examples, and their sum defines the overall support of the patterns.

Table 1 contains a toy example of data set, for which the aim is to identify the differences between successful and unsuccessful medical treatments for a given disease. In this example, Table 1 represents the data set D and $label = success$ is the target of investigation . Thus, $D^+ = \{e_1, e_2, e_3, e_4, e_5\}$ are the positive examples (where $label = success$) and $D^- = \{e_6, e_7, e_8, e_9, e_{10}\}$ are the negative examples (where $label \neq success$). Meanwhile Table 2 represents the universe of items $I = \{i_1, i_2, i_3, i_4, i_5, i_6, i_7\}$ and the respective positive and negative covered examples. In this context, $dp = \{i_5\}$ is an interesting discriminative pattern, once $c^+(dp) = |\{e_1 e_2, e_3\}| = 3$ and $c^-(dp) = |\emptyset| = 0$. On the other hand, $dp = \{i_1, i_7\}$ is not an interesting pattern as it is equally frequent among positive and negative samples ($c^+(dp) = |\{e_1\}| = 1$ and $c^-(dp) = |\{e_7\}| = 1$).

The definition of the relevance/interestingness of a discriminative pattern is given by a measure [28]. Lavrac et al. [28] present a thorough review on several types of evaluation/interestingness measures for discriminative patterns. They discuss how the measures relate to each other, often describing the same, while, in spite of it, there is still no consensus about the best one. This choice often depends on the problem or specialist's convictions. In this way, it is important for discriminative pattern mining algorithms to accept different options of evaluation metrics to meet user needs.

One of the most used evaluation metric is the weighted relative accuracy (WRAcc), given by Equation 1:

$$WRAcc(dp) = \frac{TP + FP}{|D|} \left(\frac{TP}{TP + FP} - \frac{|D^+|}{|D|} \right), \tag{1}$$

where $TP = |c^+(dp)|$ (the positive support) and $FP = |c^-(dp)|$ (the negative support).

As described by Lavrac et al. [28], the WRAcc is a trade-off between generality and accuracy. The first part of the equation (outside parethesis) accounts for the generality of the pattern. Patterns covering more samples, i.e. more general, are, at first, preferred to more specific patterns. The second part (inside parenthesis) corresponds to the relative accuracy of the pattern. Patterns showing a gain relative to the fixed rule assigning/describing all samples to/from the positive class are preferred. *WRAcc* values range from −0.25 to +0.25, or from −1 to +1 in its normalized form ($WRAcc_{normalized} = 4 \times WRAcc$) [29]. In this case +1 represents a totally pure pattern, describing all the positive examples and none of negative examples, while −1 represents a pattern describing all the negative examples and none of the positive, and 0 indicates that there is no gain relative to the fixed rule describing all examples of the base as positive.

Another well-known metric is the Q_g [30], given by Equation 2, where g is a generalization parameter, which defaults to 1. The value of g represents the tolerance to negative examples in relation to positives covered by a DP. The higher the g value, the more generic DPs will be. On the other hand, the closer to zero the value of g, the more specific and intolerant to FP the best DPs will be [30].

$$Q_g = \frac{TP}{FP + g}, \tag{2}$$

Q_g can be used as an alternative to the evaluation metric $GrowthRate = \frac{TP}{|D^+|} / \frac{FP}{|D^-|}$ [4], often used in works mining DPs from bioinformatics data [10, 18, 31] without the disadvantage of division by zero, though. Q_g can assume values between 0 and $+\infty$. A complete survey on many evaluation metrics can be found in [1, 3].

There are also some global metrics, whose purpose is to evaluate a DP set. One of them is the overall support $SUPP^+$ [32], witch measures the percentage of target examples D^+ covered by a DP set (Equation 3). This metric is important to evaluate if a DP set significantly covers D^+ or if it is restricted to a small subset of them. $SUPP^+$ can assume values between 0 and 1, where 1 means that a DP set completely cover D^+.

$$SUPP^+ = \frac{|c^+(dp_1) \cup ... \cup c^+(dp_k)|}{|D^+|}, \tag{3}$$

One of the challenges in discriminative patterns is the redundancy in a DP set. Two of the most common type are coverage and description redundancy [33]. Coverage redundancy occurs when a DP set has many positive examples in common (e.g. $dp_1 = \{i_2\}$ and $dp_2 = \{i_3\}$ in Table 2, where $c^+(dp_1) = c^+(dp_2) = \{e_2, e_5\}$). A DP set with high coverage redundancy usually has low $SUPP^+$. On the other hand, description redundancy occurs when a DP set has one or more items in common. The DPs $dp_1 = \{i_1\}$, $dp_2 = \{i_1, i_3\}$ and $dp_3 = \{i_1, i_3, i_6\}$ in Table 2, for example, describes only male patients ($i_1 \rightarrow (genre, M)$). Both can result in poor information for end user.

3. Related work

The area of discriminative pattern mining evolved in parallel from three different areas: *Subgroup Discovery* [2, 3], *Emerging Patterns* [4] and *Contrast Sets* [6]. *Subgroup Discovery* is the extraction of subgroups of interest related to the value of label [3]. *Emerging Patterns* are groups where the difference of frequency with respect to two classes diverges to a rate of gain [4]. At last, *Contrast Set* are conjunctions of attributes and values that significantly differ in their distributions [6]. In 2009, Novak et al. [7] discussed how these areas related to each and classified them as being the same problem. However, one of the first works to use the term *discriminative pattern* were the articles by Gao and Wang [34] and Pandey et al. [35]. Liu et al. [1] were the first to survey the area from the perspective of bioinformatics.

Discriminative pattern mining algorithms usually return the best patterns in one of two ways. The most popular way is constraint-based searching, where the algorithm traverse the search space keeping patterns that satisfy a given constraint while avoiding paths with unpromising patterns. This technique was borrowed from *association rule mining* algorithms [36] and, hence, often uses as constraints measures such as minimum support and confidence.

Nevertheless, setting the thresholds for those constraints might not be a simple task. If it is too large, the algorithm may not return any results [37]. On the other hand, if it is small, it does not effectively filter uninteresting patterns. This is particularly critical when dealing with high dimensional data sets as a huge number of patterns may satisfy a constraint [18, 19]. In other words the longer patterns found in high dimensional data may yield an exponential number of sub-patterns that also satisfy the constraint; this turns out to be always true if the interestingness measure is anti-monotonic. An alternative to the constraint-based approach is to find patterns based on (implicit) rankings. The aim of this second approach is to find the top-k patterns with the highest values for a given interestingness measure. In this scenario the user provides the number k of patterns to be found and the algorithm searches for the best ones accordingly.

There are several exact and heuristic data mining algorithms [1, 3, 32, 38]. Among the heuristic algorithms, the ones based on *beam search* [30, 33, 39] and *evolutionary computing* [13, 20–26, 38] are most important ones.

The algorithms based on *beam search* are initialized from a predefined number of DPs determined by a *beamSize* parameter. New patterns are generated from the *beamSize* ones from the previous iteration. Therefore, approaches based on beam search restrict memory usage by exploring only part of the search space. One of the first and most prominent algorithm for mining discriminative patterns based on beam search is SD [30]. The algorithm starts the search by taking the highest quality (according to Q_g and minimal support) items as singleton discriminative patterns. After that, the algorithm replaces the least relevant patterns by the most relevant ones with larger sizes. The SD stops the search when there is no change in list of relevant patterns over one iteration.

One of the greatest disadvantage of beam search algorithms is the lack of diversity. The algorithms usually target only individually good items, which, by the point of view of domain experts, might already be a well-known pattern [40, 41]. In this scenario, evolutionary algorithms represent a perfect fit, having many methods been proposed in the literature. We now review some of the most important evolutionary algorithms for discriminative pattern mining, and refer the reader to the work of Carmona et al. [38], which provides a thorough survey of the area.

SDIGA is a mono-objective approach that uses a global search followed by a local search for each iteration. The global search is performed by the genetic algorithm and the local search, via *Hill Climbing*. Two other algorithms are *MESDIF* and *NMEEF*. These algorithms are multi-objective, the first being based on the *SPEA2* [42] algorithm and the second one on the *NSGA-II* [43]. MESDIF uses elitism and the concept of *Pareto Front* in its search strategy, and NMEEF uses an operator to reset the population. NMEEF has been one of the most competitive approaches when compared with other algorithms [24, 25, 32]. *FuGePSD* [25] uses genetic programming [44] and represents individuals with trees. In addition, *FuGePSD* performs both local and global search while attempting to cover all positive examples of the D^+ database. Finally, *MEFASD-BD* is an approach focused on *big data* in relation to the number of examples. *MEFASD-BD* uses the *MapReduce* paradigm to partition the data set, and concepts of *NMEEF* to mine the DPs. These algorithms use fuzzy logic to deal with numeric attributes. There are, however, other evolutionary approaches for mining DPs that do not use the *Genetic Fuzzy System* [45]. *EDER* [23] is a mono-objective approach based on *HIDER* [46] (HIerarchical DEcision Rules). *EDER* focuses on issues with minority classes in unbalanced databases. *CGBA* [24], on the other hand, is an approach that uses evolutionary programming as a search strategy and *context-free grammar* to represent DPs in a readable and flexible way. In addition, *CGBA* dynamically defines crossover and mutation rates while searching without user interference. Table 3 summarizes the characteristics of the algorithms reviewed here.

Despite the large number of evolutionary approaches, none of them was developed with focus on high dimensionality and most of the performance tests considered data sets with less than 40 attributes. In addition, some features of such models can be problematic in the context of high dimensionality. The representation of individuals using one gene per item (or attributes), for example, can bring high cost of memory in the context of high dimensionality. At the same time, limiting the size of individuals in the initial population to percentages of $|I|$ tends to generate large random DPs that do not cover any example of D, which may restrain the convergence of the algorithm. Finally, such models usually have some non-trivial configuration parameters and none of them is top-k.

More recently, researchers are also reconsidering sampling algorithms as an alternative to exact/enumerative methods [47–50]. Sampling algorithms most often use Monte Carlo Markov Chain methods to find patterns via the distribution of their support or a quality measure based on it. These algorithms are particularly useful for interactive exploratory analysis as samples may be drawn sequentially from the given distribution [51]. Nevertheless, most of the works in this area are still focusing on low dimensional data. Boley et al. [48], for instance, restricted their experiments to UCI data with less than 300 dimensions (and 4000 samples). Since our focus is on batch analysis of

Table 3: Summary of the characteristics of the main evolutionary algorithms for mining discriminative patterns.

Algorithm	Objective	Size of individuals	Initial population	top-k	No. parameters
SDIGA	Mono-objective	$\|I\|$	?	NO	7
MESDIF	Multi-objective	$\|I\|$	?	NO	7
NMEEF	Multi-objective	$\|I\|$	75% of individuals with up to 25% of of items $i \in I$	NO	7
EDER	Mono-objective	$\|A\|$	Based on examples	NO	4
CGBA	Mono-objective	$size(dp)$	Random until all individuals are valid	NO	4
FuGePSD	Mono-objective	$size(dp)$	1% to 50% of items $i \in I$, until all individuals are valid	NO	14
MEFASD-BD	Multi-objective	$\|I'\|$ where $I' \subset I$	?	NO	8

very high dimensional data sets, such as those from unstructured textual or biomedical data, we do not consider these approaches in our experiments.

4. SSDP: Simple Search Discriminative Patterns

SSDP is a mono-objective evolutionary approach for discriminative pattern mining. Its main characteristics are: (1) being adapted to high dimensional data and (2) having few easily adjustable parameters.

In SSDP, individuals represent only the items used in the DP. The rationale for using such a representation lies on the fact that the best patterns usually contain less than 1% of the items. Therefore, each individual of the population is represented by one or more integers. Each integer (or index) corresponds to the position of an item i in I (assuming any total ordering of items). A two-dimensional discriminative pattern $dp = \{2043, 213\}$, for example, represents the set formed by items in positions 2043 and 213 of I. However, when representing individuals as sets of integers, it is necessary to ensure that there is no duplicity (eg. $dp = \{2043, 2043, 213\}$). We implemented individuals using hash tables to avoid duplicity and maintain the performance of the algorithm.

SSDP initializes the searches with patterns of size one and evolves to higher dimensions through its evolutionary operators. The initial population is composed of all one dimensional possible DPs (an individual for each $i \in I$). Such an initialization allows the population size to be determined automatically according to the problem ($populationSize = |I|$). Besides, it ensures that all items $i \in I$ are considered in the search. Initializing the search from one-dimensional solutions is a novelty among evolutionary approaches for mining DPs. However, it is widely used in algorithms based on Beam Search [17, 30, 52]. In addition, in high dimensional bases, initializing a search by randomly generated individuals may restrain the convergence of the algorithm, as we discuss in Section 5.1.

After the initial population is generated, SSDP uses the following genetic operators to generate new candidates. The selection is made by binary tournament. In mutation there are three possibilities with the same probability: (1) a random item is added to the individual (e.g. $i = \{a, b, c\} \rightarrow i' = \{a, b, c, d\}$); (2) a random item is replaced by another (e.g. $i = \{a, b, c\} \rightarrow i' = \{a, b, d\}$); and (3) a random item is removed from the individual (e.g. i=$\{a, b, c\} \rightarrow i' = \{a, b\}$). Therefore, in the mutation, an individual with size d randomly evolves to dimension d, $d - 1$ or $d + 1$. It is common that just one item changes in evolutionary approaches for DP mining. That happens because the change in a single item represents a significant transformation in the individual.

With respect to crossover method, there are two possibilities: *crossOverAND* and *crossOverUniform*. The first one generates an individual from the union of the two individuals' items (e.g. $i_1 = \{a\}$ and $i_2 = \{b\} \rightarrow i' = \{a, b\}$). This type of crossing is used only in the initial population, in which all individuals have size=1. While in *crossOverUniform* crossing, two individuals generate two new by uniform crossover with 50% mixing ratio (e.g. $i_1 = \{a, b\}$ and $i_2 = \{c, d\} \rightarrow i'_1 = \{a, d\}$ and $i'_2 = \{b, c\}$).

Crossover and mutation rates initialize at 0.6 and 0.4, respectively, and are adapted according to the search. If, at the end of a generation, there is improvement in the top-k DPs, the algorithm increases the crossover rate at 0.2 and reduces mutation rate at the same value. When there is no improvement in the top-k DPs, the mutation rate increases by 0.2 and the crossover rate decreases by the same value. Thus, the algorithm tends to intensify the search in depth when it is in a promising region, otherwise, it tends to intensify the search in breadth. The mutation and crossover rates always sum to one. This methodology is an adaptation of the one proposed by Luna et al. [24] for CGBA (described in Section 3).

The SSDP uses as stopping criterion the stabilization of the group of the k best DPs after the population has been reset twice. A population is reset when there is no change in top-k DPs for three consecutive generations and the mutation rate is equal to one. In this process the algorithm randomly generates individuals of fixed size between two and the average size of the top-k DPs. Moreover, 10% of individuals are generated using exclusively items present in top-k DPs.

SSDP does not allow the user to tune some common parameters, such as mutation and crossover rate, population size and minimal support. The algorithm has only two input parameters: the number of DPs (k) and evaluation metric (fitness). SSDP theoretically allows the use of any interestingness measure as fitness. Currently, SSDP implementation includes three evaluation metrics: Q_g, $WRAcc$ and $SUB = TP - FP$.

Algorithm 1 contains the pseudocode of SSDP. In the algorithm, the population P_k keeps the best k individuals that are relevant. An individual dp_i is considered irrelevant in relation to population P_k if $\exists dp \in P_k | c^+(dp_i) \subset c^+(dp) \wedge c^-(dp) \subset c^-(dp_i)$. The other populations (P, P_{new} and $P*$) allow the presence of duplicated individuals. The control over the individuals is made only in the population P_k in an attempt to minimize the computational costs of the algorithm and at the same time return only non-redundant DPs (P_k) to the end-user. The algorithm was implemented in Java and is available from our supporting website (`https://github.com/tarcisiodpl/ssdp`).

Algorithm 1 SSDP pseudocode

```
Require: $k$, $metricEvaluation$
  $P \leftarrow \{\{i_1\}, \{i_2\}, ..., \{i_{|I|}\}\}$
  $P_k \leftarrow kBestRelevants(P)$
  $reinializationCount \leftarrow 0$
  $mutationRate \leftarrow 0.4$
  $crossoverRate \leftarrow 0.6$
  while $reinializationCount < 2$ do
    while $P_k$ not improve three consecutive generations keeping $mutationRate == 1.0$ do
      if generation == 1 then
        $P_{new} \leftarrow crossoverAND(P)$
      else {generation > 1}
        $P_{new} \leftarrow evolutionaryOperator(P, mutationRate, crossoverRate)$
      end if
      $P* \leftarrow best(P, P_{new})$
      $P_k \leftarrow kBestRelevants(P_k, P_*)$
      $update(mutationRate, crossoverRate)$
      $P \leftarrow P*$
    end while
    $reinializationCount + +$
    $P \leftarrow$ restart
  end while
  return $P_k$
```

5. Experiments

The experiments were performed in two groups of data sets, one with high dimensionality and another one traditional. The high dimensionality group (Table 4) consists of 21 microarray bases, available in the package *datamicroarray* [53] from R software. The bases have between 456 and 54,613 numerical attributes, and between 31 and 248 examples. For each data set, the majority class was considered the target (positive) and the remaining were labeled as negative. The attributes were discretized prior to applying the algorithm. Since it is not our goal to discuss the implications of the discretization method on the discriminative pattern mining algorithms, we used the simplest

Table 4: Summary of the 21 high dimensional data sets used in our experiments to assess the performance of SSDP. The columns $|D|$, $|D^+|$ and $|D^-|$ contains the total number of examples, and the number of positive and negative examples after mapping the most frequent label in the data to positive and the remaining to negative. The column *Attributes* contains the number of numeric attributes in the original data, while the remaining columns contain the number of items in the discretized data set using either equal frequency or width with the corresponding number of bins.

					Size of *I* with Equal Frequency			**Size of *I* with Equal Width**		
Name	$\|D\|$	$\|D^+\|$	$\|D^-\|$	**Attributes**	**2 bins**	**4 bins**	**8 bins**	**2 bins**	**4 bins**	**8 bins**
alon	62	40	22	2000	4000	8000	16000	4000	7909	14734
borovecki	31	17	14	22283	44566	89132	178257	44566	87611	159671
burczynski	127	59	68	22283	44566	89132	178264	44566	88588	170312
chiaretti	128	74	54	12625	25250	50500	101000	25250	50429	98837
chin	118	75	43	22215	44430	88860	177719	44430	88254	169006
chowdary	104	62	42	22283	44566	89118	177748	44566	82456	133038
christensen	217	113	104	1413	2826	5652	11304	2827	5555	10425
golub	72	47	25	7129	14258	28516	57032	14258	28129	52989
gordon	181	150	31	12533	25066	50132	100264	25355	49807	91869
gravier	168	111	57	2905	5810	11620	23240	5811	11530	22165
khan	63	23	40	2308	4616	9232	18464	4616	9219	17930
nakayama	105	21	84	22283	44566	89132	178264	44566	87038	160167
pomeroy	60	39	21	7128	14256	28512	57024	14256	28202	53494
shipp	77	58	19	7129	14258	28516	57032	14258	27475	49714
singh	102	52	50	12600	25200	50132	97804	25200	49558	91633
sorlie	85	32	53	456	912	1824	3648	946	1860	3555
subramanian	50	33	17	10100	20200	40400	80800	20200	40191	77625
sun	180	81	99	54613	109227	218453	436905	109227	215499	410113
tian	173	137	36	12625	25250	50500	101000	25250	49780	94021
west	49	25	24	7129	14258	28516	57032	14258	27231	48924
yeoh	248	79	169	12625	25250	50500	101000	25250	50427	99689

methods based on equal frequency and width with 2 , 4 and 8, bins. Such a preprocessing step resulted in a total of 126 data sets composed exclusively of binary (interval based) attributes (items).

The traditional group (of low dimensionality) is formed by 20 data sets extracted from the UCI repository [54]. The bases have between 10 and $12,960$ examples, between 6 and 69 attributes, and are made exclusively by discrete attributes. Table 5 describes the bases with more details, where $|D|$, $|D^+|$ and $|D^-|$ are, respectively, the amount of examples, positive examples and negative examples of databases. The columns *attributes* and $|I|$ are, respectively, the number of attributes and items $i \in I$.

Figure 1 graphically summarizes all (the 20 UCI and the 126 high dimensional) data sets used in our experiments. As we can notice, the data sets are well distributed in the item-example space. The high dimensionality bases have a wide variation in the number of items, but all of them have a small number of examples. On the other hand, the UCI bases show higher variation in the number of examples, but a small number of items. We can also notice that the majority of bases have proportionally the same amount of positive and negative examples, with a slight tendency to have more positive than negative examples. However, we also see some bases where there is an imbalance between the number of positive and negative examples; this occurs for both the UCI and high dimensional bases.

We conducted four types of experiments in this work to assess the performance of SSDP from different perspectives. In our first batch of experiments (Section 5.1), we evaluated different parameter settings for SSDP. We tested the algorithm with different crossover and mutation rates (sometimes fixed during the entire execution, opposite to the original version, which auto-adjust these parameters) and two different stopping criteria. These experiments will help us elucidate whether the choices made for the original version (Section 4) are indeed good ones. Defined the best parameters for SSDP, we compared the effectiveness of the algorithm against SD and random search (Section 5.2), and also against the state of the art evolutionary algorithms (Section 5.3). Finally, in Section 5.4 we evaluated the SSDP in relation to D^+ coverage and in the redundancy among the returned DPs.

Statistical analysis of the results was performed by using the hypothesis tests *Wilcoxon* and *Friedman*. The *Wilcoxon* is a non-parametric test that has been indicated and used for performance analysis between two algorithms. *Friedman* test [55] is commonly indicated to assess whether there is statistical difference between more than two

Table 5: Summary of the 20 UCI data sets used in our experiments to assess the performance of SSDP. The columns $|D|$, $|D^+|$ and $|D^-|$ contains the total number of examples, and the number of positve and negative examples after mapping the most frequent label in the data to positive and the remaining to negative. The column *Attributes* contains the number of attributes in the data, while $|I|$ is the number of items (attribute,value) pairs.

Name	$\lvert D\rvert$	$\lvert D^+\rvert$	$\lvert D^-\rvert$	**Atributtes**	$\lvert I\rvert$
audiology	226	57	169	69	154
kr-vs-kp	3196	1669	1527	36	73
lung-cancer	32	13	19	56	157
molecular-biology_promoters	106	53	53	58	334
soybean	683	92	591	35	99
trains	10	5	5	32	77
splice	3190	1655	1535	61	3465
breast-cancer	289	201	85	9	41
bridges_version2	105	44	61	12	191
car	1728	1210	518	6	21
monks-problems-1_train	124	62	62	6	17
postoperative-patient-data	90	64	26	8	23
primary-tumor	339	84	255	17	37
shuttle-landing-control	15	9	6	6	16
solar-flare_2	1,066	331	735	12	42
spect_test	187	172	15	22	44
tic-tac-toe	958	626	332	9	27
vote	435	267	168	16	32
mushroom	8124	4208	3916	22	116
nursery	12960	4320	8640	8	27

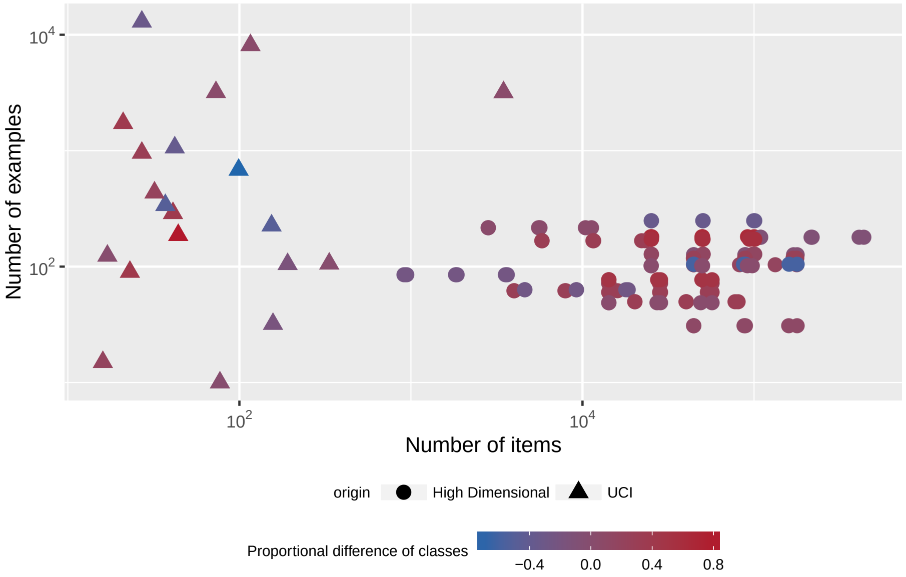


Figure 1: Visual summary of data sets used in experiments to assess the performance of SSDP. Data sets in the high dimensional group (Table 4) are represented by bullets, while UCI (Table 5) representatives are marked by triangles. The color of the points represent the proportional difference between the number of positive and negative examples in the data. Red represents data set with proportionally more positive examples than negative, while blue represents the opposite.

Table 6: Summary of the eight SSDP versions tested.

Version	Crossover rate	Mutation rate	Stop criterion
SSDP_Auto_3x3 (**or just SSDP**)	Auto	Auto	3x3
SSDP_90x10_3x3	90%	10%	3x3
SSDP_50x50_3x3	50%	50%	3x3
SSDP_100x100_3x3	100%	100%	3x3
SSDP_90x10	90%	10%	3x
SSDP_50x50	50%	50%	3x
SSDP_100x100	100%	100%	3x
SSDP_Auto	Auto	Auto	3x

algorithms [56].

When the *Friedman* test rejected the null hypothesis, the next step is to perform another hypothesis test to validate which one or which algorithms are standing out from the others. One of the options is the test with controller. Controller is the baseline algorithm that will be compared to all the others. This method is commonly used when a new algorithm is proposed and researchers needs to compare its performance with other existing methods in the literature [56]. The *Friedman* test statistic with controller is given by:

$$z = \frac{(R_i - R_j)}{\sqrt{\frac{k(k+1)}{6N}}},$$

where R_i-R_j is the mean ranking difference between two algorithms, N is the number of databases and k is the number of algorithms.

The value of z is calculated between the control algorithm and the other algorithms generating $k - 1$ values. The respective *p-values* are calculated from the values of z. The null hypothesis is rejected when $p < \alpha$. However, the α must be adjusted for multiple comparisons. We adjusted the significance values with the *Holm* [57] methodology, where the value of α is adapted by the equation $\frac{\alpha}{k-1}$, k being the number of algorithms. Hypothesis tests were done using the implementation available in [58].

As discussed above, we divided the experiments in four sections. Section 5.1 aims to test variations of the SSDP and experimentally validate some characteristics of the model in the approach of high dimensional bases. Furthermore, Section 5.1 shows the convergence and behavior of the SSDP in each generation for the largest database used in our experiments. Section 5.2 confronts SSDP with an approach based on beam search, a random search approach and another one based on a trivial search in high dimensionality bases. Next, Section 5.3 confronts the SSDP with three evolutionary approaches in traditional and high dimensional databases. Finally, in Section 5.4 we confront SSDP with three evolutionary approaches in relation to D^+ coverage and redundancy.

5.1. Assessing the impact of different parameter settings on SSDP's performance

In this section we first tested different configurations of the SSDP algorithm. We tried different mutation and crossover rates: 100-100, 10-90, 50-50 and *Auto* (Section 4). We also attempted to use different stopping criteria: (1) stop when there is no change in the top-k patterns for 3 generations, referred to as *(3x)*; and (2) reset the population when the algorithm stops because of the first criterion, and halt the execution when it reaches the first criterion for the third time, referred to as *(3x3)*. Thus, eight different versions/configurations of SSDP were tested, which are summarized in Table 6. The SSDP(3x3)_Auto (Table 6) version corresponds to the final version of the SSDP, described in Section 4. Each experiment was repeated ten times, with the metric evaluation Q_g ($g = 1$) and $k = \{5, 10, 20, 50\}$.

Figure 2 presents the average Q_g and time for the eight SSDP versions for $k = \{5, 10, 20, 50\}$. We notice that the versions with stopping criterion *(3x3)* stood out from the others with respect to the average Q_g. The improvement in quality is even more noticeable for higher values of k. We see that for $k = 50$ the use of the second stopping criterion yields an improvement of near 10%. We also notice that the original version of SSDP discussed in Section 4 and SSDP_100x100_3x3 present similar performances for all values of k. Nevertheless, we also observe in the figure that

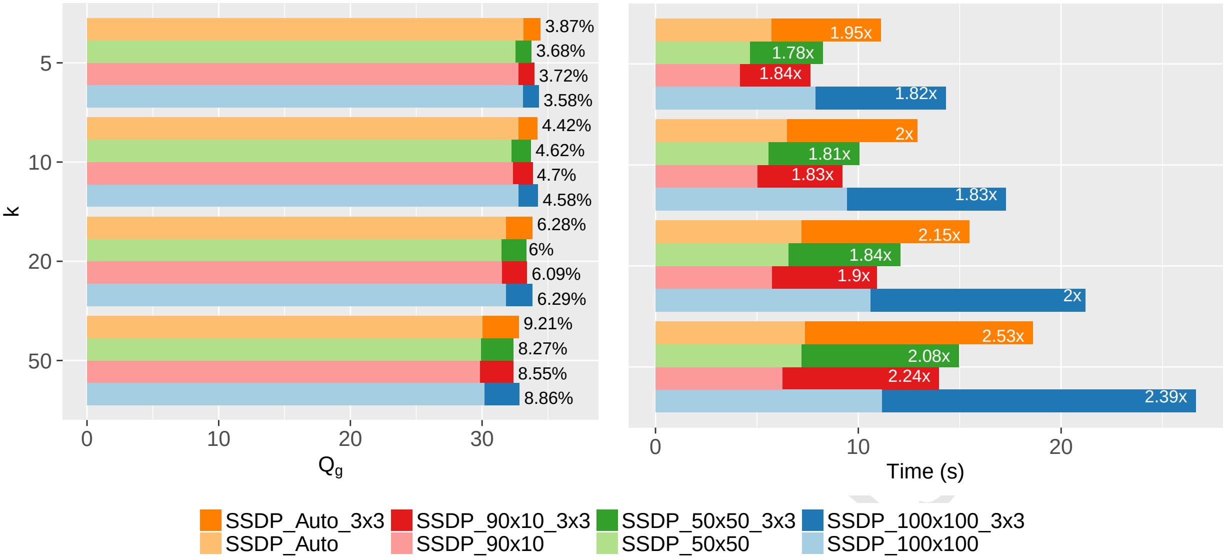


Figure 2: This figure graphically displays the average Q_g (left) and time in seconds (right) for the eight different configurations of SSDP. The algorithms were tested with the 126 microarray databases for $k = \{5, 10, 20, 50\}$. In both charts, the algorithms were grouped according to the choice of crossover and mutation rates. Dark-shaded colors represent the configurations for which population was reset twice, before the algorithm was halt. The numbers in both charts represent the increase in quality and time because of the choice to reset the population.

Table 7: *Friedman* test comparing eight versions of SSDP for $k = \{5, 10, 20, 50\}$ ($\alpha = 0.05$).

Version	**Ranking**			
	k=5	**k=10**	**k=20**	**k=50**
SSDP_100x100	4.85	5.07	5.25	5.24
SSDP_100x100_3x3	3.04	2.95	**2.80**	**2.56**
SSDP_Auto	4.82	4.98	5.19	5.70
SSDP_Auto_3x3	**2.95**	**2.91**	2.81	2.91
SSDP_90x10	5.81	5.75	5.89	6.04
SSDP_90x10_3x3	3.88	3.61	3.68	3.45
SSDP_50x50	6.33	6.38	6.32	6.40
SSDP_50x50_3x3	4.33	4.33	4.07	3.69
p-value	1.44E-10	1.48E-10	1.40E-10	1.81E-10

the latter has the highest average time for all values of k. In terms of time, we observe that the choice of the second stopping criterion roughly doubles computing time of all algorithms.

In order to evaluate whether the visual difference observed in Figure 2 was statistically significant, we applied the *Friedman* test with a null hypothesis that there is no variation in the mean Q_g between the various configurations. Table 7 displays the average rankings of the different settings for the different values of k and their respective *p-values*. We observe that in all cases there was a statistical difference in the performance of the configurations. We also note that the SSDP_100x100_3x3 and SSDP_Auto_3x3 presented very similar performances, and there were no significant discrepancies between them with different values of k. This leads us to carry out a second test, in which we will evaluate if there is statistical difference between the best ranked and the other configurations.

This second hypothesis test confronted the configuration of best average rank in the Friedman test (Table 7 in bold) with the other configurations for each value of k ($\alpha = 0.05$). Table 8 summarizes the *p-value* and the significance level required by the *Holm* method to reject the null hypothesis. We notice that SSDP_Auto_3x3 and SSDP_100x100_3x3 versions were statistically at least as good as the others for all value of k. However, from the perspective of average quality of the patterns, there is no difference between the best two configurations. Therefore, SSDP_Auto_3x3 configuration was chosen the best because it had the lowest computing time in all experiments.

Table 8: Friedman hypothesis test with control algorithm for $k = \{5, 10, 20, 50\}$, $\alpha = 0.05$ (adapted by the *Holm* method).

k=5			
Rank	**Version**	**p-value**	α **(Holm)**
7	SSDP_50x50	0.0000	0.0071
6	SSDP_90x10	0.0000	0.0083
5	SSDP_100x100	0.0000	0.0100
4	SSDP_Auto	0.0000	0.0125
3	SSDP_50x50_3x3	0.0000	0.0166
2	SSDP_90x10_3x3	0.0025	0.0250
1	**SSDP_100x100_3x3**	**0.7772**	**0.0500**
Control	**SSDP_Auto_3x3**	-	-

k=10			
Rank	**Version**	**p-value**	α **(Holm)**
7	SSDP_50x50	0.0000	0.0071
6	SSDP_90x10	0.0000	0.0083
5	SSDP_100x100	0.0000	0.0100
4	SSDP_Auto	0.0000	0.0125
3	SSDP_50x50_3x3	0.0000	0.0166
2	SSDP_90x10_3x3	0.0236	0.0250
1	**SSDP_100x100_3x3**	**0.8976**	**0.0500**
Control	**SSDP_Auto_3x3**	**-**	**-**

k=20			
Rank	**Version**	**p-value**	α **(Holm)**
7	SSDP_50x50	0.0000	0.0071
6	SSDP_90x10	0.0000	0.0083
5	SSDP_100x100	0.0000	0.0100
4	SSDP_Auto	0.0000	0.0125
3	SSDP_50x50_3x3	0.0000	0.0166
2	SSDP_90x10_3x3	0.0043	0.0250
1	**SSDP_Auto_3x3**	**0.9692**	**0.0500**
Control	**SSDP_100x100_3x3**	**-**	**-**

k=50			
Rank	**Version**	**p-value**	α **(Holm)**
7	SSDP_50x50	0.0000	0.0071
6	SSDP_90x10	0.0000	0.0083
5	SSDP_Auto	0.0000	0.0100
4	SSDP_100x100	0.0000	0.0125
3	SSDP_50x50_3x3	0.0002	0.0166
2	SSDP_90x10_3x3	0.0038	0.0250
1	**SSDP_Auto_3x3**	**0.2578**	**0.0500**
Control	**SSDP_100x100_3x3**	**-**	**-**

Table 9: Average Q_g obtained by SSDP using different initialization alternatives, where zero means that the algorithm did not return any valid solution, and "–" means there was a memory exhaustion (12GB limit).

Base	**|I|**	**original**	**0.1%**	**1%**	**5%**	**10%**
alon	4,000	24.1	24.1	**0**	**0**	**0**
gravier	5,860	46.6	45.9	**0**	**0**	**0**
tian	25,250	56.2	54.1	**0**	**0**	–
yeoh	25,300	76.2	75.6	**0**	**0**	–
sun	109,226	57.2	**0**	–	–	–

Regarding the initial population, we conducted experiments to compare the method used in SSDP and populations randomly generated with individuals of predetermined sizes equal to 0.1%, 1%, 5% and 10% of $|I|$. Table 9 presents the average Q_g obtained by using the different population sizes for five discretized data sets with two intervals. In this table *original* represents the method used by the SSDP, zero means that the algorithm did not return any valid solution, and "–" means there was a memory exhaustion (12GB limit). As we can see, the larger the size of the randomly generated initial population, the more difficult it is for SSDP to converge to valid solutions. This happens because large individuals randomly generated tend to not represent a valid solution as they do not cover any example. Besides, in an index representation such as SSDP, the average size of individuals has a strong impact on memory consumption. In this context, the strategy of initializing searches with one dimension individuals, besides helping in the convergence of the model, reduces memory consumption compared to the other tested options.

Finally, we ran experiments to evaluate SSDP's convergence. Figure 3a shows the values of average fitness of populations P and P_k for each generation of the model, applied to the *sun* database, for $k = 50$. The accentuated evolution of the fitness shows the SSDP's capacity to quickly converge. The points of strong fall in the average fitness of P are the moments in which the population is reset. We see that, for this example, the first reset of P was successful, since the algorithm continued to improve the population P_k for several times in sequence. We also observed that, at some moments, the average fitness of the population P is above the population P_k. This indicates that P has many duplicated high quality individuals. This duplicated is tackled in SSDP principal by mutation and reset operator, when the population P is recreated.

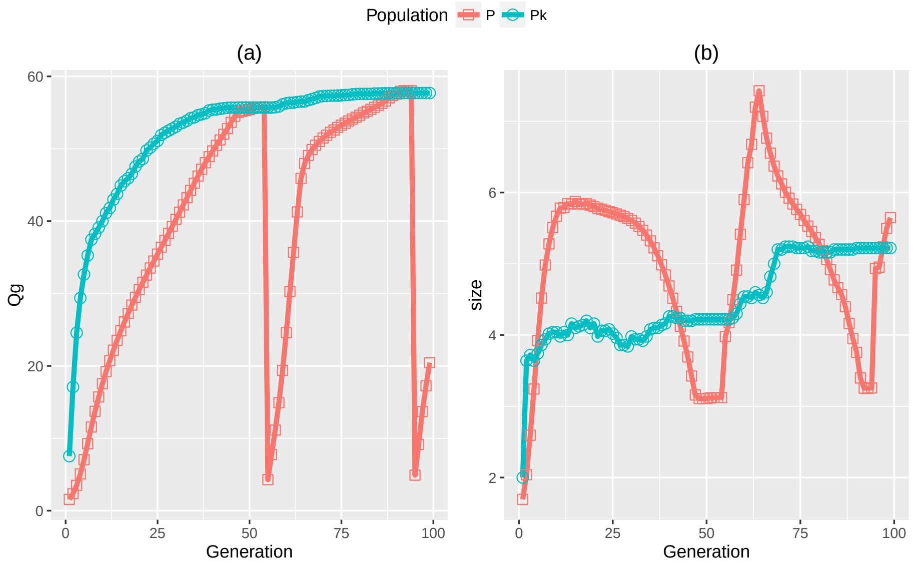


Figure 3: This figure depicts the evolution of fitness (left) and average size (right) of the populations P and P_k for each generation of SSDP for $k = 50$ with the data set *sun*.

Figure 3b shows the evolution of DPs average size in populations P and P_k. In the first generation P and P_k are composed strictly by singletons (DPs of size one). After that, poor quality patterns are replaced by better quality ones. We can observe from the average size of P that SSDP tends to initially direct the searches towards larger dimensions but it may also change direction to smaller dimensions if required.

5.2. Comparing SSDP to beam search

This section aims to confront SSDP with a heuristic approach based on beam search and validate it as a heuristic for discriminative pattern mining from high dimensional data. SSDP was compared with the approaches described below, all implemented in *Java*. Each experiment was repeated 10 times, with the objective function Q_g ($g = 1$) and $k = \{5, 10, 20, 50\}$.

- Random3M: three million DPs up to four dimensions randomly generated. The objective of this experiment is to compare SSDP to a random search.
- Trivial: DPs with highest fitness among all combinations of up to four dimensions, but using only the best k items. The purpose of this comparison is to validate SSDP's ability to find non-trivial DPs.
- SD: The aim is to confront SSDP with a competitive beam search heuristic. SD used the following parameters: $beamWidth = k$ and $minimumSupport = \frac{\sqrt{|D^+|}}{|D|}$ (this parameter was set according to the recommendations of Gamberger and Lavrac [30]).

Figure 4 shows the mean Q_g and time of the SSDP, SD, Trivial and Random3M approaches for $k = \{5, 10, 20, 50\}$. We notice that, while the beam search heuristic SD is very similar to random search, our evolutionary heuristic SSDP found higher quality patterns for all values of k. In fact, we notice that SSDP achieved roughly 50% higher quality than the random search (Trivial and Random3M) and 30% higher than SD. On the other hand, SD achieved only 15% improvement compared to random search. In terms of time, we observe a linear growth in computing time for SD and a sub-linear growth for SSDP. Interestingly, despite requiring 15% more computing time to find the top 50 patterns than SSDP, SD did not find higher quality patterns.

We applied the *Wilcoxon test* to verify whether the difference of performances between SSDP and SD was statistically significant. The null hypotheses that SSDP performs equally well to SD for the different k values were all

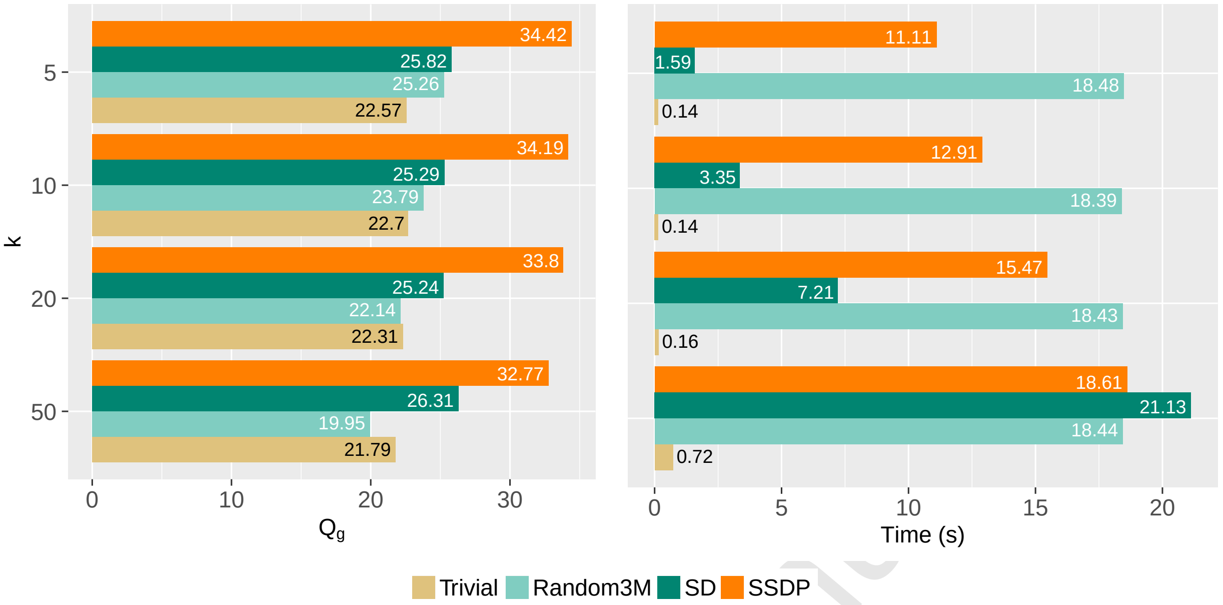


Figure 4: This figure graphically displays the average Q_g (left) and time in seconds (right) for SSDP, SD, Trivial and Random3M. The algorithms were tested with the 126 microarray databases for $k = \{5, 10, 20, 50\}$. The numbers in both charts represent the Q_g and time of the algorithm.

rejected for a level of significance $\alpha = 0.01$. The *p-values* obtained for $k = \{5, 10, 20, 50\}$ were respectively 6.17E-16, 4.59E-16, 5.67E-16 and 9.50E-14. Thus, SSDP was statistically superior to SD in the context of high dimensionality for $k = \{5, 10, 20, 50\}$. Furthermore, the SSDP's superiority over Random3M and Trivial approaches validate the proposed model in relation to random search and the ability to find non-trivial patterns.

The exact algorithm based on *Beam Search SDMap* [59] has been tested as well, using the available implementation on software *KEEL* [60] with default parameters. However, the algorithm had problems with memory exhaustion (12G limit) and processing time (2 hours time limit) in the tested high-dimensional databases (Table 4). With respect to traditional databases (Table 5), *SDMap* converged to valid results on only four of the 20 bases tested.

5.3. *Comparing SSDP to other evolutionary approaches*

This section compares SSDP to other evolutionary approaches using both traditional and high dimensional data sets. SSDP was confronted with SDIGA [13], MESDIF [20] and NMEEF [21], algorithms available in the KEEL machine learning suite [60]. Default parameters in KEEL were used for the algorithms, and $k = 5$ and *WRAcc* as fitness function for the SSDP. We changed the choice of the fitness function for SSDP because all other algorithms use *WRAcc* as their fitness function.

Table 10 shows the mean $WRAcc_{normalized}$ (Section 2), time, number of DPs (k) and size for the algorithms tested with the 20 UCI data sets (Table 5). We notice that NMEEF and SSDP were the best performing algorithms regarding the average $WRAcc_{normalized}$. Regarding computing time, SSDP proved to be faster than the others (it takes only a tenth of the time required by NMEEF). The table also shows the result for the *Friedman* hypothesis test considering the $WRAcc_{normalized}$. The test showed that there was a statistical difference between the algorithms (*p-value*=$1.39E-05$), with NMEEF as the best ranked algorithm, followed closely by SSDP. Then, Table 11 shows the result of the multiple *Friedman* using the NMEEF as control and *Holm* method for correcting the significance levels (α). The test showed that NMEEF was statistically better than MESDIF and SDIGA, but there is no evidence regarding SSDP. This confirms SSDP as a competitive approach also for traditional data without the need to adjust any parameters.

On the other hand, the experiments in high dimensional bases were limited to 10 of the bases described in Table 4 due to the high computational cost of the simulations. Table 12 presents the average $WRAcc_{normalized}$, time, number of DPs and size obtained by the evolutionary algorithms. In initial experiments SDIGA had difficult to converging in less than three hours with several databases and was excluded from the comparison. We notice however that the

Table 10: $WRAcc_{normalized}$, time, number of DPs (k), average size and rank obtained by MESDIF, NMEEF, SDIGA and SSDP algorithms with 20 UCI data sets.

Algorithm	$WRAcc_{normalized}$	time(s)	k	size	Avg. Rank
MESDIF	0.080	2.85	3	15.06	3.4
NMEEF	**0.412**	2.55	8.1	3.02	**1.775**
SDIGA	0.188	5.70	2.6	1.28	3.025
SSDP	0.376	**0.17**	5	2.25	1.8
p-value for the Friedman test					1.39E-05

Table 11: *Friedman* hypothesis test with control algorithm for MESDIF, SDIGA, NMEEF and SSDP algorithms ($\alpha = 0.05$)

Rank	Algorithm	p	Holm ($\alpha = 0.05$)
3	MESDIF	0.00006	0.016
2	SDIGA	0.0021	0.025
1	**SSDP**	**0.9511**	**0.05**
Control	**NMEEF**	–	–

other two algorithms, NMEEF and MESDIF, did not converge to valid solutions, despite using more computational resources than SSDP.

We performed a last experiment to compare NMEEF to SSDP. In this experiment we set a population of $1,000$ individuals and $1,000,000$ evaluations (NMEEF-1k-1M). Table 13 shows the $WRAcc_{normalized}$, number of DPs (k), time and number of tests did by the SSDP and NMEEF-1k-1M for the tested databases, where "–" means that the algorithm did not finish in less than 48 hours. We observe that NMEEF (the best performing evolutionary algorithm for traditional data) still did not return any valid discriminative pattern in six of the ten tested data sets. On the other bases, the average $WRAcc_{normalized}$ was lower than those obtained by SSDP. In addition, NMEEF's computing time was considerably higher than SSDP's for all bases.

We conclude from these experiments that the evolutionary models NMEEF, MESDIF and SDIGA are not suitable to high dimensionality, even if the parameters are tuned (in the case of NMEEF). Moreover, these algorithms proved to be costly in terms of processing time and returned poor results. SSDP, on the other hand, obtained valid DPs for all high dimensional data sets using considerably lower processing time.

5.4. Redundancy and coverage in SSDP

This section aims to evaluate SSDP in relation to D^+ coverage and in redundancy between top-k DP set. The experiments were made in UCI data sets (Table 5), for $k = 5$ and $WRAcc$ as metric evaluation. SSDP was confronted with the evolutionary algorithms NMEEF, MESDIF and SDIGA using default parameters.

The coverage in relation to D^+ was evaluated by overall support ($SUPP^+$, Equation 3). The algorithms SSDP, NMEEF, SDIGA and MESDIF obtained respectively 86.2%, 89.1%, 86.6% and 37.6% as mean $SUPP^+$ (Table 14). In this way, SSDP was competitive in relation to NMEEF and SDIGA and superior to MESDIF.

Already the coverage redundancy was evaluated by difference between the mean of overall support $SUPP^+$ and mean local support ($supp^+_{mean}$), where $supp^+_{mean} = \frac{1}{k}\sum supp^+(dp)$, $supp^+(dp) = \frac{|c^+(dp)|}{|D^+|}$ and $SUPP^+ = \frac{|c^+(dp_1)\cup...\cup c^+(dp_k)|}{|D^+|}$. In this way, $supp^+_{mean} \approx SUPP^+$ indicates that a DP set was restricted to describing approximately the same examples of D^+. Thus, Table 14 shows that SSDP generated less average coverage redundancy than NMEEF, MESDIF and SDIGA.

Table 12: $WRAcc_{normalized}$, time, number of DPs (k) and average size obtained by the MESDIF, NMEEF and SSDP algorithms in ten microarray databases.

Algorithm	$WRAcc_{normalized}$	time(s)	k	size
MESDIF	0.0039	1,241.4	3	16,458.03
NMEEF	0.0115	2,069.2	10.1	56.87
SSDP	**0.6304**	7.7	5	2.42

Table 13: WRAcc, *k*, time, number of tests and patterns obtained by SSDP and NMEEF-1k-1M algorithms in ten microarray databases.

Base	Algorithm							
	NMEEF-1k-1M				SSDP			
	$WRAcc_{normalized}$	k	Time(s)	Tests (10^6)	$WRAcc_{normalized}$	k	Time(s)	Tests (10^6)
alon	0.26	3	1,984	1	0.572	5	0.422	0.116
burczynski	0	0	63,697	1	0.684	5	8.254	1.247
chiaretti	0	0	36,882	1	0.584	5	4.789	0.808
chin	0.388	1	31,301	1	0.624	5	5.928	0.799
christensen	0.592	1	3,408	1	0.896	5	0.297	0.056
gravier	0.232	1	5,745	1	0.440	5	0.905	0.185
nakayama	0	0	58,419	1	0.600	5	7.893	1.515
tian	0	0	43,218	1	0.296	5	4.072	0.505
yeoh	0	0	104,533	1	0.848	5	8.798	0.782
sun	–	–	–	–	0.186	5	36.315	3.386

Table 14: Local mean support ($supp^+_{mean}$) and mean overall support ($SUPP^+$, Equation 3) for algorithms SSDP, NMEEF, MESDIF and SDIGA for 20 UCI databases (Table 5), where $supp^+_{mean} = \frac{1}{k}\sum supp^+(dp)$, $supp^+(dp) = \frac{|c^+(dp)|}{|D^+|}$ and $SUPP^+ = \frac{|c^+(dp_1) \cup ... \cup c^+(dp_k)|}{|D^+|}$.

Algorithm	**mean** $SUPP^+$	**mean** $supp{+}_{mean}$	mean $SUPP^+$ - mean $supp{+}_{mean}$
SSDP	0.862	0.582	0.28
NMEEF	0.891	0.766	0.125
MESDIF	0.376	0.222	0.154
SDIGA	0.866	0.676	0.19

Table 15 shows the mean local support ($supp^+_{mean}$) and the overall support ($SUPP^+$) of top-5 DPs returned by SSDP in each UCI database. In this way, Table 15 shows that SSDP covered more than 60% of D^+ in 19 of 20 databases and more than 90% in seven of them. The difference between $SUPP^+$ e $supp^+_{mean}$ also shows that DP set returned by SSDP usually were not restricted to cover the same examples of D^+.

Finally, description redundancy was analyzed by counting the number of databases where all returned DPs have a common item. This kind of redundancy occurred in SSDP, NMEEF, SDIGA and MESDIF in respectively 6, 8, 12 and 17 of 20 databases. Thus, all algorithms presented significant description redundancy, but with less frequency in SSDP.

So, in these experiments we conclude that the SSDP was more efficient than the NMEEF, MESDIF and SDIGA using default parameters in relation to redundancy between returned DP set. But the proposed model presented some difficulties. The covered in relation to D^+ was unstable, ranging between 100% and 58.4%. Although the description redundancy was more critical, presenting in 6 out of 20 databases. Thus, we believe that the proposed model still deals a little inefficiently with the redundancy problem.

Some contents like SSDP implementation, some high dimensionality databases used in the tests, tables with the results of each experiment of this paper, including other evaluation metrics such as support, confidence level, TP (true positive), FP (false positive) and *p-value* are available on this website (`https://github.com/tarcisiodpl/ssdp`).

6. Conclusion

This paper presents SSDP, the first evolutionary approach for mining top-*k* discriminative patterns in high dimensional data sets. Extraction of discriminating information from high dimensional data is a common challenge in areas such as bioinformatics and text mining. Evolutionary approaches have been shown to be an efficient option for mining discriminative patterns in traditional data sets. However, none of them were developed with a focus on high dimensionality.

SSDP has been designed from the beginning to the context of high dimensionality bases. The representation of individuals, for example, only considers the items used by DPs as a way to reduce the computational cost of memory. Our approach for generating the initial population seeks to increase the convergence of the algorithm and ensure that

Table 15: Mean local support ($supp^{+}_{mean}$) and the overall support ($SUPP^{+}$) obtained by the SSDP in 20 UCI databases (Table 5), for $k = 5$ and $WRAcc$ as metric evaluation, where $supp^{+}_{mean} = \frac{1}{k}\sum supp^{+}(dp)$, $supp^{+}(dp) = \frac{|c^{+}(dp)|}{|D^{+}|}$ and $SUPP^{+} = \frac{|c^{+}(dp_1)\cup \ldots \cup c^{+}(dp_k)|}{|D^{+}|}$.

Database	$supp^{+}_{mean}$	$SUPP^{+}$	$SUPP^{+} - supp^{+}_{mean}$
audiology	0.947	0.94	0.007
breast-cancer	0.86	0.767	0.093
bridges-version2	0.977	0.7	0.277
car	0.99	0.383	0.607
kr-vs-kp	0.715	0.715	0
lung-cancer	0.846	0.615	0.231
molecular-biology-promoters	0.924	0.339	0.585
monks-problems-1-train	0.661	0.316	0.345
mushroom	0.977	0.779	0.198
nursery	1	0.44	0.56
postoperative-patient-data	0.609	0.281	0.328
primary-tumor	0.619	0.59	0.029
shuttle-landing-control	1	0.666	0.334
solar-flare-2	1	0.721	0.279
soybean	0.978	0.939	0.039
spect-test	0.738	0.454	0.284
splice	0.807	0.249	0.558
tic-tac-toe	0.584	0.3	0.284
trains	1	0.56	0.44
vote	0.947	0.847	0.1

all items are considered in the search. At last, SSDP controls redundant individuals in the top-k DPs as a way to reduce the computational cost and increase the relevance of patterns.

The proposed model also seeks to hide some parameters from the user to become a simpler approach to apply and consequently help in the popularization of discriminative knowledge extraction. Population size, mutation and crossover rates are automatically defined by the algorithm. The stopping criteria is not defined by the number of tests or generations and has been developed to be kept clear for the final user.

The SSDP had some of its characteristics experimentally validated, such as mutation, stopping criteria and initial population. Its performance was assessed using high dimensional and traditional data sets. The algorithm was also compared with other approaches: random, trivial, based on beam search and based on evolutionary computing. In the context of high dimensionality, SSDP obtained statistically better results than all the others algorithms, in relation to the quality of DPs. In traditional databases, SSDP was shown to be a competitive approach without the necessity to make any adjustments in parameters. Finally, in relation to redundancy in DP set, SSDP was better than other evolutionary approaches, but it presented some problems.

Therefore, we concluded that SSDP is an efficient, flexible and simple alternative for the extraction of discriminant knowledge in high dimensional data sets. However, SSDP is the only evolutionary approach that deals exclusively with discrete data. Besides that, the model has few resources to deal with redundancy in top-k DPs, restricting itself to eliminating DPs equal or dominated by others. This opens new pathways in the direction of evolving SSDP to deal with numerical data and to find more efficient alternatives to eliminate redundancy.

Acknowledgment

This work has been partially funded by FACEPE and CNPq.

References

[1] Liu X, Wu J, Gu F, Wang J, He Z. Discriminative pattern mining and its applications in bioinformatics. Briefings in bioinformatics 2014;16(5):884–900. doi:10.1093/bib/bbu042.

[2] Atzmueller M. Subgroup discovery. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 2015;5(1):35–49. doi:10.1002/widm.1144.

[3] Herrera F, Carmona CJ, González P, Del Jesus MJ. An overview on subgroup discovery: foundations and applications. Knowledge and information systems 2011;29(3):495–525. doi:10.1007/s10115-010-0356-2.

[4] Dong G, Li J. Efficient mining of emerging patterns: Discovering trends and differences. In: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining. New York, NY, USA: ACM; 1999, p. 43–52. doi:10.1145/312129.312191.

[5] Blinova VG, Dobrynin DA, Finn VK, Kuznetsov SO, Pankratova ES. Toxicology analysis by means of the jsm-method. Bioinformatics 2003;19(10):1201. URL: +http://dx.doi.org/10.1093/bioinformatics/btg096. doi:10.1093/bioinformatics/btg096.

[6] Bay SD, Pazzani MJ. Detecting group differences: Mining contrast sets. Data Mining and Knowledge Discovery 2001;5(3):213–46. doi:10.1023/A:1011429418057.

[7] Novak PK, Lavrač N, Webb GI. Supervised descriptive rule discovery: A unifying survey of contrast set, emerging pattern and subgroup mining. The Journal of Machine Learning Research 2009;10:377–403. doi:10.1145/1577069.1577083.

[8] Carmona CJ, Chrysostomou C, Seker H, del Jesus M. Fuzzy rules for describing subgroups from influenza a virus using a multi-objective evolutionary algorithm. Applied Soft Computing 2013;13(8):3439–48. doi:10.1016/j.asoc.2013.04.011.

[9] Carmona CJ, González P, Del Jesus M, Navío-Acosta M, Jiménez-Trevino L. Evolutionary fuzzy rule extraction for subgroup discovery in a psychiatric emergency department. Soft Computing 2011;15(12):2435–48. doi:10.1007/s00500-010-0670-3.

[10] Li J, Wong L. Identifying good diagnostic gene groups from gene expression profiles using the concept of emerging patterns. Bioinformatics 2002;18(5):725–34. doi:10.1093/bioinformatics/18.5.725.

[11] Quackenbush J. Computational analysis of microarray data. Nature reviews genetics 2001;2(6):418–27. doi:10.1038/35076576.

[12] Carmona CJ, Ramírez-Gallego S, Torres F, Bernal E, del Jesús MJ, García S. Web usage mining to improve the design of an e-commerce website: OrOliveSur.com. Expert Systems with Applications 2012;39(12):11243–9. doi:10.1016/j.eswa.2012.03.046.

[13] Jesus M, Gonzalez P, Herrera F, Mesonero M. Evolutionary fuzzy rule induction process for subgroup discovery: A case study in marketing. IEEE Transactions on Fuzzy Systems 2007;15(4):578–92. doi:10.1109/TFUZZ.2006.890662.

[14] Romero C, González P, Ventura S, Del Jesús MJ, Herrera F. Evolutionary algorithms for subgroup discovery in e-learning: A practical application using moodle data. Expert Systems with Applications 2009;36(2):1632–44. doi:10.1016/j.eswa.2007.11.026.

[15] Kavšek B, Lavrac N. Analysis of example weighting in subgroup discovery by comparison of three algorithms on a real-life data set. Advances in Inductive Rule Learning 2004;:64.

[16] Kavšek B, Lavrac N, Bullas JC. Rule induction for subgroup discovery: a case study in mining uk traffic accident data. In: Proceedings of the international multi-conference on information society. 2002, p. 127–30.

[17] Kavšek B, Lavrač N, Jovanoski V. Apriori-sd: Adapting association rule learning to subgroup discovery. Applied Artificial Intelligence 2006;20(7):543–83. doi:10.1080/08839510600779688.

[18] Vimieiro R, Moscato P. A new method for mining disjunctive emerging patterns in high-dimensional datasets using hypergraphs. Information Systems 2014;40:1–10. doi:10.1016/j.is.2013.09.001.

[19] Vimieiro R. Mining disjunctive patterns in biomedical data sets. Ph.D. thesis; The University of Newcastle, NSW, Australia; 2012.

[20] Jesus M, Gonzalez P, Herrera F. Multiobjective genetic algorithm for extracting subgroup discovery fuzzy rules. In: IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making. 2007, p. 50–7. doi:10.1109/MCDM.2007.369416.

[21] Carmona C, Gonzalez P, Jesus M, Herrera F. Nmeef-sd: Non-dominated multiobjective evolutionary algorithm for extracting fuzzy rules in subgroup discovery. IEEE Transactions on Fuzzy Systems 2010;18(5):958–70. doi:10.1109/TFUZZ.2010.2060200.

[22] Pachón V, Mata J, Domínguez JL, Maña MJ. Multi-objective evolutionary approach for subgroup discovery. In: Hybrid Artificial Intelligent Systems: 6th International Conference (HAIS 2011). Berlin, Heidelberg: Springer Berlin Heidelberg; 2011, p. 271–8. doi:10.1007/978-3-642-21222-2_33.

[23] Rodríguez D, Ruiz R, Riquelme JC, Aguilar-Ruiz JS. Searching for rules to detect defective modules: a subgroup discovery approach. Information Sciences 2012;191:14–30. doi:10.1016/j.ins.2011.01.039.

[24] Luna JM, Romero JR, Romero C, Ventura S. On the use of genetic programming for mining comprehensible rules in subgroup discovery. IEEE Transactions on Cybernetics 2014;44(12):2329–41. doi:10.1109/TCYB.2014.2306819.

[25] Carmona CJ, Ruiz-Rodado V, del Jesús MJ, Weber A, Grootveld M, González P, et al. A fuzzy genetic programming-based algorithm for subgroup discovery and the application to one problem of pathogenesis of acute sore throat conditions in humans. Information Sciences 2015;298:180–97. doi:10.1016/j.ins.2014.11.030.

[26] Pulgar-Rubio F, Rivera-Rivas A, Pérez-Godoy M, González P, Carmona C, del Jesus M. MEFASD-BD: Multi-objective evolutionary fuzzy algorithm for subgroup discovery in big data environments-A MapReduce solution. Knowledge-Based Systems 2016;doi:10.1016/j.knosys.2016.08.021.

[27] Pontes T, Vimieiro R, Ludermir TB. SSDP: A simple evolutionary approach for top-k discriminative patterns in high dimensional databases. In: 2016 5th Brazilian Conference on Intelligent Systems (BRACIS). 2016, p. 361–6. doi:10.1109/BRACIS.2016.072.

[28] Lavrac N, Flach PA, Zupan B. Rule evaluation measures: A unifying view. In: Proceedings of the 9th International Workshop on Inductive Logic Programming. London, UK: Springer Berlin Heidelberg; 1999, p. 174–85. doi:10.1007/3-540-48751-4_17.

[29] Flach PA. The geometry of ROC space: Understanding machine learning metrics through ROC isometrics. In: Proceedings of the Twentieth International Conference on Machine Learning (ICML 2003). Washington, DC; 2003, p. 194–201.

[30] Gamberger D, Lavrac N. Expert-guided subgroup discovery: Methodology and application. Journal of Artificial Intelligence Research 2002;17(1):501–27.

[31] Yu LT, Chung Fl, Chan SC, Yuen S. Using emerging pattern based projected clustering and gene expression data for cancer detection. In: Proceedings of the second conference on Asia-Pacific bioinformatics; vol. 29. Darlinghurst, Australia: Australian Computer Society, Inc.; 2004, p. 75–84.

[32] Helal S. Subgroup discovery algorithms: A survey and empirical evaluation. Journal of Computer Science and Technology 2016;31(3):561–76. doi:10.1007/s11390-016-1647-1.

[33] Van Leeuwen M, Knobbe A. Diverse subgroup set discovery. Data Mining and Knowledge Discovery 2012;25(2):208–42. doi:10.1007/s10618-012-0273-y.

[34] Gao C, Wang J. Direct mining of discriminative patterns for classifying uncertain data. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '10; New York, NY, USA: ACM. ISBN 978-1-4503-0055-1; 2010, p. 861–70. doi:10.1145/1835804.1835913.

[35] Pandey G, Wang W, Gupta M, Fang G, Kumar V, Steinbach M. Mining low-support discriminative patterns from dense and high-dimensional data. IEEE Transactions on Knowledge & Data Engineering 2010;24:279–94. doi:10.1109/TKDE.2010.241.

[36] Agrawal R, Srikant R. Fast algorithms for mining association rules in large databases. In: Proceedings of the 20th International Conference on Very Large Data Bases (VLDB 94). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. ISBN 1-55860-153-8; 1994, p. 487–99.

[37] Han J, Wang J, Lu Y, Tzvetkov P. Mining top-k frequent closed patterns without minimum support. In: Proceedings of the 2002 IEEE International Conference on Data Mining. Washington, DC: IEEE Computer Society. ISBN 0-7695-1754-4; 2002, p. 211–8.

[38] Carmona CJ, González P, del Jesus MJ, Herrera F. Overview on evolutionary subgroup discovery: analysis of the suitability and potential of the search performed by evolutionary algorithms. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 2014;4(2):87–103. doi:10.1002/widm.1118.

[39] Lavrač N, Kavšek B, Flach P, Todorovski L. Subgroup discovery with cn2-sd. Journal of Machine Learning Research 2004;5:153–88.

[40] Fang G, Wang W, Oatley B, Ness BV, Steinbach M, Kumar V. Characterizing discriminative patterns. CoRR 2011;abs/1102.4104.

[41] Garriga G, Kralj P, Lavrač N. Closed sets for labeled data. The Journal of Machine Learning Research 2008;9:559–80.

[42] Zitzler E, Laumanns M, Thiele L. SPEA2: Improving the Strength Pareto Evolutionary Algorithm for Multiobjective Optimization. In: Giannakoglou K, et al., editors. Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001). International Center for Numerical Methods in Engineering (CIMNE); 2002, p. 95–100.

[43] Deb K, Pratap A, Agarwal S, Meyarivan T. A fast and elitist multiobjective genetic algorithm: Nsga-ii. IEEE Transactions on Evolutionary Computation 2002;6(2):182–97. doi:10.1109/4235.996017.

[44] Koza JR. Genetic Programming: On the Programming of Computers by Means of Natural Selection. Cambridge, MA, USA: MIT Press; 1992. ISBN 0-262-11170-5.

[45] Herrera F. Genetic fuzzy systems: taxonomy, current research trends and prospects. Evolutionary Intelligence 2008;1(1):27–46. doi:10.1007/s12065-007-0001-5.

[46] Aguilar-Ruiz JS, Ramos I, Riquelme JC, Toro M. An evolutionary approach to estimating software development projects. Information and Software Technology 2001;43(14):875–82. doi:10.1016/S0950-5849(01)00193-8.

[47] Bendimerad AA, Plantevit M, Robardet C. Unsupervised exceptional attributed sub-graph mining in urban data. In: Proceedings of the IEEE 16th International Conference on Data Mining (ICDM). 2016, p. 21–30. doi:10.1109/ICDM.2016.0013.

[48] Boley M, Lucchese C, Paurat D, Gärtner T. Direct local pattern sampling by efficient two-step random procedures. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '11; New York, NY, USA: ACM. ISBN 978-1-4503-0813-7; 2011, p. 582–90. URL: http://doi.acm.org/10.1145/2020408.2020500. doi:10.1145/2020408.2020500.

[49] Moens S, Boley M. Instant Exceptional Model Mining Using Weighted Controlled Pattern Sampling. Cham: Springer International Publishing. ISBN 978-3-319-12571-8; 2014, p. 203–14. URL: http://dx.doi.org/10.1007/978-3-319-12571-8_18. doi:10.1007/978-3-319-12571-8_18.

[50] Kaytoue M, Plantevit M, Zimmermann A, Bendimerad A, Robardet C. Exceptional contextual subgraph mining. Machine Learning 2017;:1–41URL: http://dx.doi.org/10.1007/s10994-016-5598-0. doi:10.1007/s10994-016-5598-0.

[51] Scholz M. Sampling-based sequential subgroup mining. In: Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining. KDD '05; New York, NY, USA: ACM. ISBN 1-59593-135-X; 2005, p. 265–74. URL: http://doi.acm.org/10.1145/1081870.1081902. doi:10.1145/1081870.1081902.

[52] Mueller M, Rosales R, Steck H, Krishnan S, Rao B, Kramer S. Subgroup discovery for test selection: a novel approach and its application to breast cancer diagnosis. In: Advances in Intelligent Data Analysis VIII: 8th International Symposium on Intelligent Data Analysis (IDA 2009). Berlin, Heidelberg: Springer Berlin Heidelberg; 2009, p. 119–30. doi:10.1007/978-3-642-03915-7_11.

[53] Ramey J. The datamicroarray r package. 2016. URL: https://github.com/ramhiser/datamicroarray.

[54] Lichman M. UCI machine learning repository. 2013. URL: http://archive.ics.uci.edu/ml.

[55] Friedman M. A comparison of alternative tests of significance for the problem of m rankings. The Annals of Mathematical Statistics 1940;11(1):86–92. doi:10.1214/aoms/1177731944.

[56] Demšar J. Statistical comparisons of classifiers over multiple data sets. The Journal of Machine Learning Research 2006;7:1–30.

[57] Holm S. A simple sequentially rejective multiple test procedure. Scandinavian journal of statistics 1979;6(2):65–70.

[58] Statistical inference in computational intelligence and data mining. 2016. URL: http://sci2s.ugr.es/sicidm.

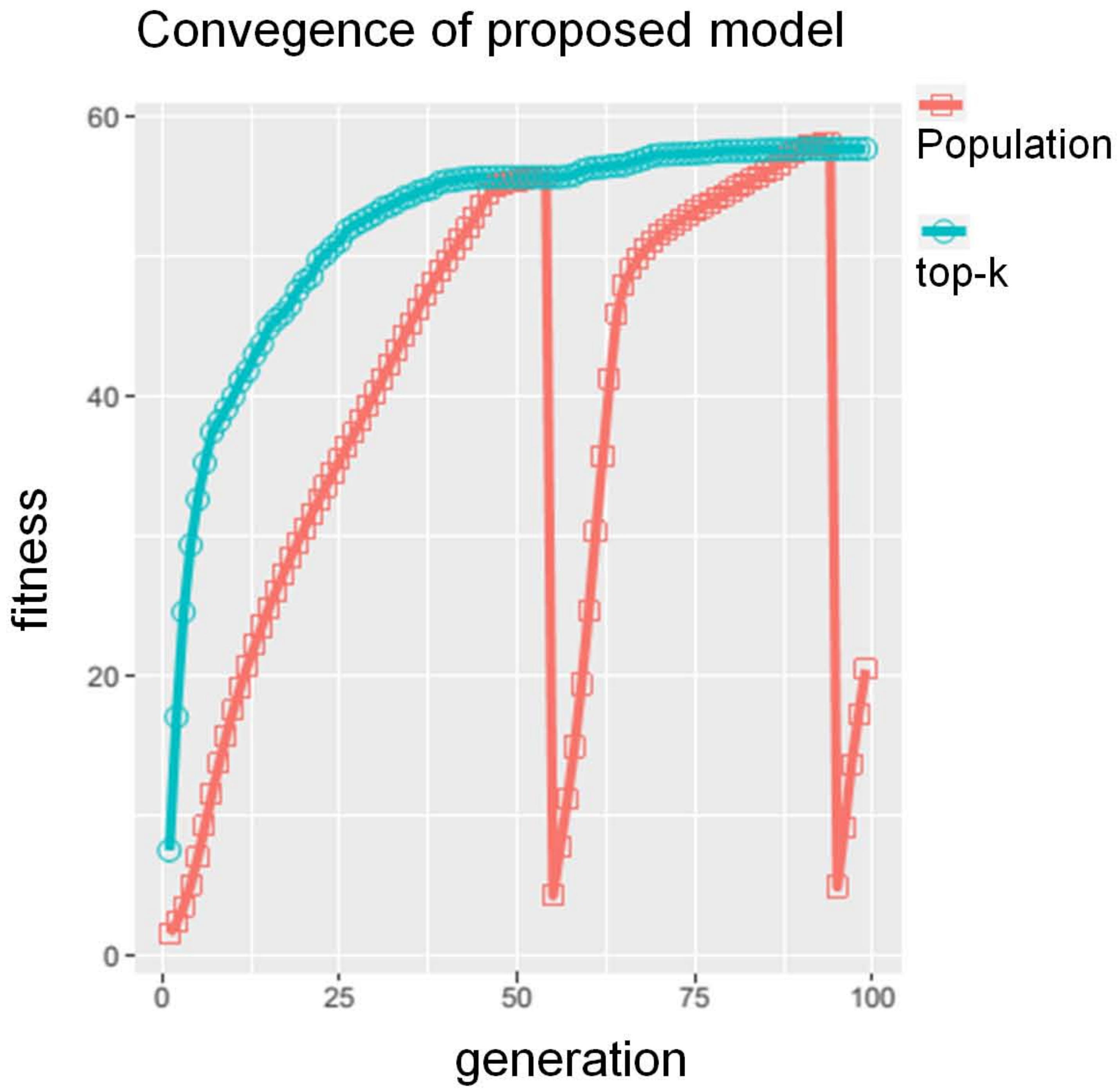
[59] Atzmueller M, Puppe F. Sd-map – a fast algorithm for exhaustive subgroup discovery. In: Knowledge Discovery in Databases: PKDD 2006: 10th European Conference on Principles and Practice of Knowledge Discovery in Databases. Berlin, Heidelberg: Springer Berlin Heidelberg; 2006, p. 6–17. doi:10.1007/11871637_6.

[60] Alcalá-Fdez J, Sánchez L, García S, Jesus M. Keel: a software tool to assess evolutionary algorithms for data mining problems. Soft Computing 2009;13(3):307–18. doi:10.1007/s00500-008-0323-y.

Highlights

- **New section of experiments to evaluate the SSDP with respect to redundancy and coverage.**
- **We removed part of section 5.1 (including the table mentioned). We understood that the deleted content was a bit long and did not result in very relevant information for the paper.**

Evolutionary approach	Focuses high dimensional	top-k	Number parameters
Proposed model	YES	YES	2
SDIGA	NO	NO	7
MESDIF	NO	NO	7
NMEEF	NO	NO	7
EDER	NO	NO	4
CGBA	NO	NO	4
FuGePSD	NO	NO	14
MEFASD-BD	NO	NO	8



Convegence of proposed model