High Dimensional Fuzzy Outlier Detection

Vasudev Sharma[1] *and B.K. Tripathy*[2]
[1]*Computer Science, Vellore Institute of Technology, Vellore, India*
vasudevsharma74@yahoo.com

Information Technology, Vellore Institute of Technology, Vellore, India
tripathybk@vit.ac.in

Abstract. We propose a novel outlier detection approach, namely high dimensional fuzzy outlier detection (HDFOD), to address the pertinence of outlier results, i.e., to find outliers in high dimensions that lack pertinence and understandability. Our key idea is to use fuzzy constraint technology to prune irrelevant objects for outlier detection, during which the nearness measure theory in fuzzy mathematics is used for detecting similarities between objects and constraint information. HDFOD finds outlier by searching sparse subspace, where genetic algorithms can be extended and incorporated into HDFOD such that an optimum solution of an outlier is discovered. While constructing a sparse subspace, we present the sparse threshold concept to describe the sparse levels of data objects in a subspace, where data objects are regarded as outliers. Then, we demonstrate the effectiveness and scalability of our method on synthetic and UCI datasets. The experiment evaluations reveal that our fuzzy constraint-based outlier detection is superior to two existing high dimensional algorithms.

Keywords: Outlier Detection, Nearness Measure, Fuzzy Constraint, Sparse Subspace, Genetic Algorithm.

1 Introduction

Outlier detection, also called anomaly detection, has become an important data mining task for the detection of inconsistent or suspicious objects from large databases. Outlier mining has attracted increasing attention in many application fields, such as fraud detection for credit cards, intrusion detection in cyber-security, medical diagnosis, data cleaning, and financial analysis. Many prominent outlier mining approaches have been proposed for outlier detection from a global point of view, where each data object is extracted as outlier from the whole attribute dimension space of a dataset. However, these outlier results may be hard to understand and fail to attract experts or users. Some outlier detection results from partial dimensions are more interesting and helpful for the research of experts than those from the full dimensions.

For example, in physical indicators, outlier monitoring based on ”blood pressure” and ”blood sugar” may be important for outlier search ”heart disease”. Some attributes, such as ”body temperature”, have no relation with outlier detection, but they may be related to other outlier detections. Outlier detection often occurs in the subsets.Therefore, we can improve the efficiency of mining on the basis of constrained subset outlier mining.

We propose an outlier detection method that integrates the knowledge of domain experts (i.e., background knowledge) into the process of outlier mining on the basis of fuzzy constrains. We demonstrate that background knowledge is useful in pruning some irrelevant objects and in substantially improving outlier detection efficiency.

1.1 Motivation

Fuzzy constraint outlier mining addressed in this study is motivated by the following three observations:

- For high-dimensional data, finding meaningful outliers becomes substantially complex and non-obvious. In addition, the traditional outlier detection methods have very low efficiency.
- The mining results, which contain some meaningless outliers, are difficult to understand.
- Outlier detection based constraint in high-dimensional datasets is critical because obtaining constraint conditions is difficult. Moreover, fuzzy constraint representation techniques for outlier detection are lacking.

1.2 Contributions

Motivated by the above mentioned three observations, we propose HDFOD - a local outlier detection method based on fuzzy constraint. HDFOD seamlessly integrates three modules: namely, fuzzy-constraint representation, fuzzy-constraint- subspace searching, and constraint outlier detection. In fuzzy-constraint representation module, the nearness measure theory in fuzzy mathematics is used for the description of background knowledge, where priori information provided by a user can be quantified and expressed effectively through a reasonable nearness measure threshold. The fuzzy- constraint-subspace searching module extends the genetic algorithm (GA) for the search of subspaces that satisfy the constraint requirement. In the constraint outlier detection module, we introduce the concept of fuzzy constraint in the outlier mining method, and then use fuzzy constraint to improve the pertinence and understandability of outlier detection results. The module increases the efficiency and accuracy of HDFOD by pruning dissatisfied condition objects. Importantly, HDFOD achieves good interpretability, because constraint information in HDFOD facilitates insightful explanations on detected outliers.

Using synthetic and real world high dimensional datasets, we conduct extensive experiments to investigate the effectiveness of HDFOD. Our experimental results show that HDFOD significantly improves the overall performance of local outlier detection. Moreover, HDFOD significantly improves the efficiency of the existing outlier detection schemes by up to 34% and average of23%.

1.3 Roadmap

The remainder of this paper is organized as follows: Section II discusses the related work of outlier mining techniques. Section III reviews the preliminaries of this study. Section IV describes the background knowledge representation method on the basis of nearness measure. Section V presents outlier detection algorithm and implementation details of HDFOD. Section VI discusses the performance evaluation of HDFOD. Finally, Section VII concludes the paper with future research directions

2 Related Work

This section, reviews existing techniques related to our study.

Outlier detection is the process of finding abnormal data objects that deviate significantly from the normal data and do not satisfy the general pattern or behavior of the data. As an important technology of data mining, outlier mining has been widely used in some fields, such as detection of credit card fraud, network security analysis, and intrusion detection. In the past, many outlier detection methods have been proposed. These existing approaches can be divided into four categories, namely, statistical-based, clustering- based, density-based, and subspace-based approaches.

In statistics-based techniques[1] [2] [3] [4], knowledge of the underlying distribution is assumed, and some objects that deviate from the distribution are searched for outlier detection. Eskin[1] proposed a method in which machine learning techniques are used for the elucidation of anomalies and performed statistical tests to detect the anomalies. Chen et al.[2] presented robust estimation and outlier detection approaches based on their proposed generalized local statistical framework. Additionally, Hido et al.[3] used a statistical method to address the problem of inlier-based outlier detection, where their key idea is to use the ratio of training and test data densities as an outlier score. The disadvantage of these methods is that we do not always know the underlying distribution of the given datasets.

Clustering-based methods[5] [6] [7] [8] [9] [10] mainly rely on clustering techniques for the characterization of underlying data behavior. Some clusters contain far less points than other clusters, and are more likely to be outliers. The main point of Jiang et al. [5] is that the concept of object outlier is extended to the cluster, and they propose an outlier detection method on the basis of clustering (i.e., CBOD). Shi and Zhang[6] presented an iterative detection method for the detection clusters and outliers in another perspective for noisy datasets, where the adjustment of the relationship between clusters and outliers is executed repeatedly until a particular condition is met. We introduce the concept of cluster histograms, which provides an efficient way to estimate and summarize the most important data distribution profiles over different stream segments. In [7], the concept of cluster histograms was provided and applied to outlier detection for modelling and mining data streams. However, clustering-based approaches must build a clustering model, which limits the outlier detection performance.

In density-based methods, models are adjusted such that outliers occur far from the closest neighbors. Breunig et al. assigned an anomaly score to each object, namely, the local outlier factor (LOF [11]). In approach, similarities are computed according to the distance of an object from the sur- rounding, and the density estimate for each object with its k nearest neighbor are obtained. Several extensions of the LOF model have been proposed (e.g., uncertain local outlier factor (ULOF [12])), the flexible kernel density estimates (KDEOS [13]), and natural outlier factor (NOF [14]). Liu et al.[12] address outlier detection with imperfect data labels and incorporate LSH and support vector data description (SVDD). Their method introduces the likelihood values of each input datum into the SVDD training phase, where the local uncertainty is captured. Then, global classifiers are constructed by the incorporation of negative examples into likelihood values. After analyzing the interplay be- tween density estimation and outlier detection, Schubert et al.[13] proposed an outlier detection method and performed kernel density estimation, which can be modified for the detection of unusual local concentrations or trends in a dataset. Salehi et al.[15] presented the memory-efficient- incremental-local outlier algorithm, which has nearly the same accuracy as LOF but within a fixed memory bound. These approaches achieve good mining accuracy without relying on assumptions of the generative distribution of data. Unfortunately, these solutions have a high processing time and are complex in the testingphase.

Subspace-based methods[16] [17] [18] [19] [20] [21] are introduced for outlier detection in subspaces. Traditional outlier detection techniques are based on the full dimension space, which can be more complex and subtle than subspace-based methods. To solve this problem, Aggarwal et al.[16] proposed an outlier method based subspace, which can mine outliers in any possible subspace. Kriegel et al.[17] propose a local outlier model to distinguish exceptional outliers by considering combinations of different subsets of attributes. Muller ¨ et al.[18] propose an outlier ranking, which computes local density deviation by searching relevant subspaces for objects deviating in subspace projections. The work proposed by Zhang et al. [19] is a concept lattice based outlier mining algorithm for low dimensions. It improves the efficiency and accuracy of outlier mining.

3 PRELIMINARIES

This section introduces the fuzzy set and fuzzy similarity scale, followed by an introduction of the GA.

3.1 Fuzzy Set and Fuzzy Similarity Scale

Fuzzy techniques that are a generalized interval analysis method can address the issues about the analysis of uncertain and/or vague information. Fuzzy set theory was initially introduced by Zadeh[22] in 1965 as a generalization of classic logic. Nowadays, fuzzy set theory is widely used in many fields, for example, in artificial intelligence[23], pattern recognition[24], decision theory[25], and computer science[26]. In mathematics, fuzzy sets are sets whose elements have degrees of membership. A fuzzy number describes the relationship between an uncertain quantity and a membership function A(x), which ranges between 0 and 1. A fuzzy set is an extension of the classical set theory in which an x can be a member of a set with a certain membership function A(x). Fuzzy sets are regarded as fuzzy numbers if they are normal, convex and bounded. We give two definitions [27] [28] [29] as follows:

Definition 1: If X is a collection of objects where x is an element included X , then a fuzzy set F (X) on X which is a set of ordered pairs can be represented as the following mathematical symbol: F (X) = {(x, A(x)|x ∈ X)}.

A(x) is called the membership function, which ranges between 0 and 1. The membership function denotes the possible membership degrees of the element x ∈ X . A(x) = 1 means full membership, A(x) = 0 means non-membership and intermediate values between 0 and 1 mean partial membership.

Given A ∈ F (X), we need to know which class A should be identified with. To solve this problem, we need to measure how close two fuzzy sets are.

Definition 2: If $N: F(X) \times F(X) \rightarrow [0,1]$ satisfies that
1. $\mathrm{N}(\emptyset, \mathrm{X}) = 0$ and $\mathrm{N}(A, A) = 1$ whenever $A \in F(X)$,
2. $\mathrm{N}(\mathrm{A}, \mathrm{B}) = \mathrm{N}(\mathrm{B}, \mathrm{A})$ whenever $A, B \in F(X)$,
3. $N(A, C) \leq \min\,(N(A, B), N(B, C))$ whenever $\mathrm{A} \subseteq B \subseteq C$

then N is called a nearness measure.

3.2 GA

GA [30] can be described as a heuristic search and optimization technique inspired by the way nature evolves species using natural selection of the fittest individuals. This method transposes the notions of natural evolution to the world of computers and imitates natural evolution. The GA operation is based on the Darwinian principle of "survival of the fittest" and the possible solutions to the problem being solved are represented by a population of chromosomes. A chromosome is a string of binary digits, and each digit that makes up a chromosome is called a gene. The fit chromosomes (i.e., individuals) are likely to survive and have a significant chance of passing their good genetic features to the next generation. Moreover, a mutation scheme is also applied to ensure a sufficient amount in the population. GA uses three operators, which are described below, on its population.

- Selection: The selection strategy addresses which of the chromosomes in the current generation will be used to reproduce offspring, which will have even high fitness. Fitness can be defined as a capability of an individual to survive and reproduce in an environment. In this method, the selection probability of each individual is proportional to its fitness value. The greater the individual fitness, the higher the probability of being chosen, and vice versa. After the individual has been selected, the transaction can be paired randomly for later crossover operation.
- Crossover or recombination: After selection, the crossover operation is applied to the selected positions. This operation will generate some new individuals by swapping genes or sequence of bits in the string between two individuals. This process is repeated with different parent individuals until the next generation has enough individuals. After crossover, the mutation operator is applied to a randomly selected subset of the population.
- Mutation: Mutation alters the genes of an individual to introduce diversification into the population. In general, the basic steps of the mutation operator are as follows: first, to determine whether the individuals in the population are in accordance with the pre-set mutation probability, and second, to select randomly) the location of mutation and to change the value of gene.

4 Fuzzy Constraint Based on Nearness

In most outlier detection algorithms, mining results contain some valueless outliers for users. We only get a range or some fuzzy descriptions about the priori information; therefore, the range should be converted to a constraint information, which can help to detect outliers. In fuzzy set theory, the nearness measure can frequently recognize some fuzzy pattern or objects. For example, one often describes a woman by using words such as young, tall, thin, and long hair. From these fuzzy information, the right woman can be recognized by using the nearness measure. Therefore, the nearness measure is an improved scheme, which describes fuzzy knowledge in constraint-outlier detection.

4.1 Equations

The nearness measures can be defined by many methods, for example, Hamming distance-related, Euclidean distance-related, Minkowski distance-related, and lattice- based methods. The lattice-based method (i.e., lattice nearness measure) is defined by means of the inner and outer product of two fuzzy sets, and is an important index in the evaluation of the nearness of two fuzzy sets. We use lattice nearness measure to describe fuzzy constraints.
For matching the fuzzy constraints, we redefine the lattice nearness measure in accordance with the idea in [31] [32].
Definition 3: Given a dataset DS, $A = \{A_1, A_2, \ldots, A_d\}$ is an attribute set, and $O = \{O_1, O_2, \ldots, O_n\}$ is an object

set in DS, where $O = \{O_1, O_2, ..., O_n\}$. O_{ij} (i = 1, 2, . . . , n; j = 1, 2, . . . , d) is the value of object O_j on attribute A_i. Suppose $U = \{U_1, ..., U_i, ... U_d\}$ is a priori knowledge provided by users, where U_i is the value on attribute A_i. Let F (X) be a fuzzy set, X be a subset of attributes and O_i, $U \in$ F (X), is called the inner product of O_C and U.

$$O_C \oplus U = \vee x \in X(Oi(x) \vee U(x)) \quad (1)$$

is called the inner product of O_i and U

$$O_i \otimes U = \wedge_{x \in X} (O_i(x) \vee U(x)) \quad (2)$$

is called the outer product of O_C and U.

With the notion of inner and outer product, we define lattice nearness measure N_L as follows, $\forall\ O_i$, $U \in$ F (X),

$$N_L(O_i, U) = (O_i \oplus U) \wedge (1 - O_i \otimes U) \quad (3)$$

The $N_L(O_C, U)$ has the following properties:

P1. $N_L(\emptyset, X) = 0$;
P2. $N_L(A, A) = \bar{a} \wedge (1 - \underline{a})$ where $\bar{a} = \vee_{x \in X} A(x)$ and $\bar{a} = \wedge_{x \in X} A(x)$
P3. $N_L(A, B) = N_L(B, A)$
P4. $A \subseteq B \subseteq C \implies N_L(A, B) \leq min(N_L(B, A), N_L(B, C))$

These properties indicate that N_L is not a nearness measure in strict sense because $N_L(A, A) = 1$ does not necessarily hold. However, N_L is an important index in evaluating the nearness of two fuzzy sets and extensively used in the literature. Hence, from property 3 $N_L(A, A) = 1$ if a = 1 and a = 0, which are true for fuzzy sets in practice. The above mentioned properties imply that the nearness measure value of two fuzzy sets is not greater than 1, and that the two fuzzy sets have similarly increasing nearness measure value.
Definition 4: Given an object O_i, priori knowledge U, and threshold value σ, if $N_L(O_i, U) \geq \sigma$, then object O_i is called a positive object, which matches a constraint condition (i.e., an object that raises the interest of the user). If $N_L(O_i, U) < \sigma$, then object O_i is called a negative object, which does not match a constraint condition (i.e., an object that loses the interest of the user). In this study, threshold value σ is called nearness-threshold, which is provided by users.
We can compute the nearness measure between each object in dataset DS and priori knowledge U, and then divide DS into two subsets in accordance with definition 4, where the first subset satisfies constraint conditions and the other subset does not satisfy these conditions. In other words, some objects are pruned because they do not satisfy the constraint conditions, and the efficiency of outlier detection is improved significantly on the reduced dataset.

4.2 An Example

We use an exampDle to prune disinterested objects in the dataset (Table I) by using fuzzy constraint information. We also show how to compute the lattice nearness measure between each object and constraint knowledge.

TABLE I. A SAMPLE OF TEADATASET

Tid	twist	Color	Neatness	liquid color	Aroma	flavor
O_1	20.00	8.00	4.50	7.00	22.50	23.00
O_2	12.50	9.00	3.05	8.50	16.25	19.50
O_3	20.25	7.80	4.45	6.00	22.75	21.25
O_4	20.50	8.10	3.95	9.00	21.75	22.00
O_5	22.50	8.60	4.35	8.50	20.00	19.75
O_6	16.75	6.50	3.15	6.80	17.50	17.75
O_7	23.25	8.00	3.50	6.10	18.00	16.25
O_8	16.25	6.30	4.00	8.20	19.75	20.25
O_9	22.50	7.80	3.65	7.50	17.25	17.00
O_{10}	21.25	8.30	4.00	9.00	21.25	21.00

Table I is a dataset composed of tea quality standard information, including 7 attributes and 10 data objects. eliminate the difference among the evaluation indicators of the tea dataset. , we use the Min-Max normalization method[33] to normalize the original data, and then a normalized matrix of the tea dataset is generated (Table II). Constraint condition is assumed as U={0.5, 0.4, 0.3, 0.8, 0.5, 0.4} and σ = 0.4, that is, priori information consists of Twist=0.5, Color=0.4, Neatness=0.3, Liquid Color=0.8, Aroma=0.5, Flavor=0.4, and threshold is set to 0.4. We can use lattice nearness measure to divide the normalized matrix (Table II) into two subsets. We also describe the calculation process of the first object, which consists of four steps.

TABLE II. THE NORMALIZED MATRIX OF TEA DATASET

Tid	twist	Color	Neatness	liquid color	Aroma	flavor
O_1	0.70	0.63	1.00	0.33	0.96	1.00
O_2	0.00	1.00	0.00	0.83	0.00	0.48
O_3	0.72	0.56	0.97	0.00	1.00	0.74
O_4	0.74	0.67	0.62	1.00	0.85	0.85
O_5	0.93	0.85	0.90	0.83	0.58	0.52
O_6	0.04	0.07	0.07	0.27	0.19	0.22
O_7	1.00	0.63	0.31	0.03	0.27	0.00

O_8	0.35	0.00	0.66	0.73	0.54	0.59
O_9	0.93	0.56	0.41	0.50	0.15	0.11
O_{10}	0.81	0.74	0.66	1.00	0.85	0.70

Step 1. The inner product of $O_>$ and U is calculated by using equation 1.

$$\begin{aligned} O_> \oplus U &= \vee_{J\in K}(O_1(x) \vee U(x)) \\ &= \vee \{0.70 \wedge 0.50, 0.63 \wedge 0.40, 1.00 \wedge 0.30, 0.33 \wedge 0.80, 0.96 \\ &\quad \wedge 0.50, 1.00 \wedge 0.40, \} \\ &= \vee \{0.5, 0.4, 0.3, 0.33, 0.5, 0.4\} \\ &= 0.5 \end{aligned}$$

Step 2. The outer product of $O_>$ and U is calculated by using equation 2.

$$\begin{aligned} O_> \otimes U &= \wedge_{J\in K}(O_C(x) \vee U(x)) \\ &= \wedge \{0.70 \vee 0.50, 0.63 \vee 0.40, 1.00 \vee 0.30, 0.33 \vee 0.80, 0.96 \\ &\quad \vee 0.50, 1.00 \vee 0.40, \} \\ &= \vee \{0.7, 0.63, 1.00, 0.8, 0.96, 1.00\} = 0.63 \end{aligned}$$

Step 3. The lattice nearness measure of $O_>$ and U is calculated by using equation 3 and the computing results of step 1 and 2.

$$\begin{aligned} N_N(O_>, U) &= (O_> \oplus U) \wedge (1 - O_> \otimes U) \\ &= 0.5 \wedge (1-0.63) \\ &= 0.37 \end{aligned}$$

Step 4. We compare the lattice nearness measure to the threshold value. When the lattice nearness measure of object less than the threshold (i.e., $N_N < \sigma$), the object is pruned; otherwise, the object is kept. Because of the lattice nearness measure of $O_>$ is less than 0.4, $O_>$ is pruned.
Similarly, the nearness measure of the other objects in Table II can be computed by using above calculation process In computing results, three objects are pruned because their nearness measures are less than the threshold, so the reduced dataset consists of seven objects.

5 Subspace and Outlier Detection

After applying the fuzzy constraint approach to prune disinterested objects in a dataset, we detect outliers (Section 5.1) by searching subspace (Sections 5.2) in the reduced dataset.

5.1 Outlier detection based on subspace

Outlier mining of high-dimensional data has been a major challenge owning to the curse of dimensionality. Most existing approaches become substantially inefficient when the required outlier detection is measured among data objects in a full-dimensional space. Moreover, the mining results may become more significant in some applications when outliers are detected in partial dimensions. Subspace-based methods can effectively detect local outliers from partial dimensions.

Definition 5: Given that dataset DS includes d attributes and n objects, let $A = \{A_1, A_2, \ldots, A_d\}$ be its attribute set and $O = \{O_1, O_2, \ldots, O_n\}$ be its object set. A subspace is described as $S = (O', A)$, where A' is the attribute set and $A' \subset$ A, O' denotes the object set and $O' \subset O$. If the subspace consists of t attribute, we call the subspace a t-dimensional subspace.

Dataset DS can be divided into many subspaces, where the whole objects in each subspace should have similar features in their own t-dimensional attributes. To find such similar attribute values, we first perform a grid discretization of the data. Each attribute of the data is divided into φ ranges. These ranges are created on an equidepth basis; thus, each range contains a fraction f = 1/φ of the objects. These ranges form the units of locality that we will use to define low-dimensional subspaces. An outlier detection of subspaces is one in which the density of the data is exceptionally lower than average. Let us consider a t-dimensional subspace that is created by picking grid ranges from t different dimensions. The expected fraction of the objects in that region if the attributes were statistically independent would be equal to f^t . Given a dataset DS, n is its number of objects and d is its number of attributes. Let S is a t-dimensional subspace of DS, if the data are uniformly distributed, and then the presence or absence of any object in S are Bernoulli random variables with probability f^t . The expected fraction and standard deviation of the objects in S are given by $n * f^t$ and $(n * f^t * (1 - f^t))^{1/2}$ in accordance with the description in [16]. We measure the sparsity degree *F(S)* of the subspace S as follows:

$$F(S) = \frac{|S| - n * f^t}{\sqrt{n * f^t * (1 - f^t)}} \qquad (4)$$

Where |S| is the number of objects in subspace S. Given a sparsity degree threshold Δ (note that Δ is a negative number), if $F(S) \leq \Delta$, then S is called a sparse subspace (i.e., the density of the data in subspace S is exceptionally lower than average). Therefore, objects including subspace S are outliers in our paper.

5.2 Searching subspace with GA

The number of subspaces increases exponentially when the number of dimensions increases, and subspaces in high-dimensional data cannot be completely exhausted. Therefore, searching sparse subspaces becomes an urgent problem. This section, introduces a novel searching sub- space method, named GA, which is a method of searching for the optimal solution through the simulation of natural evolution process. Subspace matching conditions of sparsity degree threshold can be efficiently searched (i.e., Δ).

Given a reduced dataset DS , let $A = \{A_1, A_2, . . . , A_d\}$be its attribute set, and $O = \{O_1, O_2, . . . , O_n\}$ be its objects set. We randomly select m object from O, which generate a new object set $O' = \{O'_1, O'_2, . . . , O'_m\}$, where m < n. Significantly, $O' \subset O$. For each object in O', we randomly select (d−t) attributes which will be replaced by the value * denoting that the attribute is a ”don’t care attribute” or ”invalid attribute”. Conversely, other attributes are” valid attributes”. Therefore, each object in O' consists of t ”valid attributes” and (d−t) ”invalid attributes” after replacement, and the original object set O' will be transformed into a new dataset that is denoted O'' . In this paper, O'' is regarded as a population of the GA in HDFOD, and an object in O'' is considered to be an individual. In our HDFOD, we make use of equation 4 to denote the fitness function of GA. The pseudocode of HDFOD is detailed in Algorithm 1, which performs fuzzy-constraint operation (Line 1 or Algorithm 2) and subspace-search operation by using GA (Line 3-14).

HDFOD consists of the following five steps:

Step 1. Some objects can be pruned in accordance with fuzzy constraint, and then the reduced dataset DS is generated (Line 1 in Algorithm 1). The procedure of fuzzy constraint takes the following five phases to create DS’ . First, the inner product is calculated in accordance with equation 1 presented in Section 4.1 (also Line 4 in Algorithm 2). Second, the outer product is calculated in accordance with equation 2 presented in Section 4.1 (also Line 5 in Algorithm 2). Third, the lattice nearness measure is computed in accordance with equation 3 in Section 4.1 (also Line 6 in Algorithm 2). Fourth, a positive object that matches the constraint condition, can be obtained by comparing the lattice nearness measure to the threshold value (Lines 7-9 in Algorithm 2). Finally, the previous four phases are repeated until all objects in dataset DS are scanned; as a result, the reduced dataset DS’ isgenerated.

Step 2. A new population is generated by the selection operator of GA, whose purpose is to transfer the optimized individuals to the next generation. Many methods are used for the selection operator of GA, for example, roulette wheel selection, stochastic universal sampling, local selection, and truncation selection. Before selection operator, the population of GA needs to be initialized, where M individuals are randomly

selected from DS’ (the detailed description is presented in the second paragraph in Section 5.2). Moreover, the results are stored to array Indi[M] (Line 2 in Algorithm 1).

Algorithm 1 *HDFOD*: Outlier search based on fuzzy constraint

```
Input:
  The original dataset DS, the number M of individuals in population, the
number of iteration G, the sparsity degree threshold Δ;
Output:
  outliers OutlierArray;
 1: DS′ ← Constraint(DS); //The reduced dataset DS′ is generated
after the fuzzy constraint of DS.
 2: Indi[M]←Initial individuals population; //M individuals are
stored to Indi[M].
 3: for (i=0; i<G; i++) do
 4:     Indi[M] ←Select(Indi[M]);
 5:     Indi[M] ←Crossover(Indi[M]);
 6:     Indi[M] ←Mutation(Indi[M]);
 7:     for (j=0; j<M; j++) do
 8:       computing the sparsity degree F(Indi[j]) according to eq. 4
 9:         if F(Indi[j]) ≤ Δ then
10:            OutlierArray ←Output(Indi[j])
11:            A new gene is randomly selected and added to Indi[j];
12:         end if
13:     end for
14: end for
15: return (OutlierArray)
```

Algorithm 2 Fuzzy constraint

```
Input:
  The original dataset DS, the number of objects n, the number of attributes
d, the priori knowledge U, the nearness measure threshold σ;
Output:
  the reduced dataset DS′;
 1: DS′ = null
 2: for (i=0; i<n; i++) do
 3:     O ← {O1,O2,...,O };//select the i^th object and store it in O
 4:     inner←O ⊕ U ;
 5:     outer←O ⊗ U;
 6:     N_L ← inner∧(1−outer);
 7:     if  N_L ≥ σ then
 8:     end if
 9: end for
10: return (DS′)
```

At present, several common selection operators exist, such as fitness ratio, random traversal sampling, and local selection methods, in which roulette selection method is the most simple and most common scheme. In this method, the selection probability of each individual is proportional to its fitness value.

In our *HDFOD*, the tournament selection strategy is used in selection operator (Line 4 in Algorithm 1). The reasons are as follows. The tournament selection strategy is also one of the selection methods, whose main procedure is to select r individuals from population at a time, and then select the best one from r individuals and add it to the offspring population. This selection operation is repeated until the size of the offspring population is equal to that of the parent population. The tournament selection strategy can be used to validly solve the problem of maximization or minimization. Conversely, some strategies need to shift fitness values when the minimization problem is solved, for example, roulette wheel selection strategy. In this paper, GAs are used to solve the minimization problem. Therefore, we use the tournament selection strategy to select the optimized individuals.

Step 3. This step performs crossover operator of GA described in Line 5 in Algorithm 1. The crossover operator is an exchange of some gene positions that are randomly selected between two individuals, and then produce two new individuals as an input of the mutation operator. Traditional methods include single-point, two-point, multi- point, and uniform crossover. We used optimized crossover mechanism in GA for finding the optimum outlier. Aggarwal *et al.*[34] introduces a detail of the crossover operator.

Step 4. In this step, a mutation operator (Line 6 in Algorithm 1), which modifies the values in some positions of the gene, is executed to maintain various populations. In our *HDFOD*, if a picked position in an individual (i.e., an object) is an ”invalid attribute” (second paragraph in Section 5.2), then we change its value to a number between 1 and φ. At the same time, we need to select a randomly ”valid attribute” and change its value to ”*”. If a picked position is a ”valid attribute”, then we change its value to a random number between 1 and φ.

Step 5. We detect outliers by searching sparse subspace, where a subspace is regarded as a sparse subspace in accordance with the definition of sparse subspace (the last paragraph in Section 5.1). The sparsity degree of each individual (i.e., object) in the population is calculated by using equation 4 (the second paragraph in Section 5.2), and then we compare the sparsity degree to threshold Δ (Lines 9-12 in Algorithm 1). If the sparsity degree is less than or equal to the threshold (i.e., $F(Indi[j]) \leq \Delta$), then subspace, which contains the individual or the object, is a sparsity subspace.

Therefore, the objects in the subspace are regarded as outliers. Subsequently, a new object is randomly selected as individual and added to the population.

6 Experimental Evaluation

This section, experimentally evaluates *HDFOD* and compares with *Gen* and *NOF*, where *Gen* is presented in [16] and *NOF* is presented in [14]. For all reported results, the test platform is a Dual 2.4GHz Intel Core2 T9600 laptop with 4GB RAM. *Gen*, *NOF* and *HDFOD* algorithms are all coded in Java (jdk 1.6). We test the outlier detection on both synthetic and UCI datasets.

Synthetic Datasets. For scalability experiments, we use the data generator model described in [35] (available from http://dx.doi.org/10.1137/1.9781611972740.23) to create two groups of synthetic datasets. The first group has five datasets (i.e., Syn1-1, Syn1-2, Syn1-3, Syn1-4, and Syn1-5), which include 50,000 objects and various dimension numbers. The dimension numbers of the five datasets are 25, 50, 100, 150, and 200, respectively. Table III summarizes the characteristics of these datasets in the first group. The second group is composed of five datasets (i.e., Syn2-1, Syn2-2, Syn2- 3, Syn2-4, and Syn2-5), which include various object numbers and 200 dimensions. The object numbers of the five datasets are 500,000, 1000,000, 1500,000, 2000,000, and 2500,000, respectively. Table IV summarizes the characteristics of these datasets in the second group. In addition, we add a little outlier in each dataset.

TABLE III. THE FIRST GROUP SYNTHETIC DATASETS

Datasets	objects number	attribute number	outlier number

Syn1-1	50,000	25	50
Syn1-2	50,000	50	50
Syn1-3	50,000	100	50
Syn1-4	50,000	150	50
Syn1-5	50,000	200	50

TABLE IV. THE SECOND GROUP SYNTHETIC DATASETS

Datasets	objects number	attribute number	outlier number
Syn2-1	50,000	200	50
Syn2-2	100,000	200	100
Syn2-3	150,000	200	150
Syn2-4	200,000	200	200
Syn2-5	250,000	200	250

- **UCI Datasets.** To evaluate *HDFOD* in a real life situation, we select five real world benchmark datasets from the UCI machine learning repository[36] (available from http://archive.ics.uci.edu/ml): Census Income (i.e., UCI1), Letter Recognition (i.e., UCI2), HTRU2 (i.e., UCI3), Nursery (i.e., UCI4), and Thyroid Disease (i.e., UCI5). Since outlier mining is conceptually similar to detecting objects that belong to a rare class, we focus on datasets where the class definitions feature a clear minority class. We assume this class to contain the outliers in these datasets. In addition, the datasets are cleaned to deal with the categorical and missing attributes. Table V summarizes the characteristics of these four datasets.

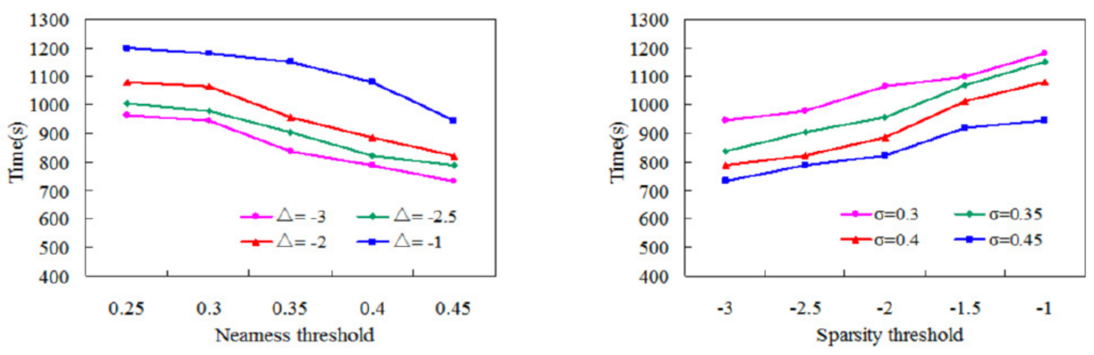
TABLE V. UCI DATASETS

Datasets	objects number	attribute number
Census Income(UCI1)	48842	14
Letter Recognition(UCI2)	20000	16
HTRU2(UCI3)	17898	9
Nursery(UCI4)	12960	8
Thyroid Disease(UCI5)	7200	21

6.1 Performance Measure

In this group of experiments, we evaluate the performance of *HDFOD* when the sizes of nearness and sparsity thresholds grow dramatically.

Fig. 1(a) illustrates that the running times of *HDFOD* reduce when the nearness threshold is increased. High nearness threshold improves *HDFOD*'s performance, and the reason is as follows: when the nearness threshold increases, a decreasing number of objects will satisfy the constrained condition. This phenomenon leads to the small reduction dataset, that is, the searching space for outlier detection becomes smaller than the original space. Therefore, the execution times of *HDFOD* is also decreasing, and its efficiency is improved Fig. 1(b) reveals that the executing times of HDFOD increase with an increasing sparsity threshold. When the sparsity threshold varies from -2 to -1, HDFOD's running time is slowly increasing. We conclude that a small sparsity threshold shortens running time by quickly detecting the outlier. From equation 4, if the sparsity threshold is a small value, then the number of sparsity subspace is decreased and the time of searching sparsity subspace is also decreased.



(a). Efficiency with various nearness threshold *(b). Efficiency with various sparsity threshold*

Fig. 1. Impacts of parameters σ and Δ on the efficiency of HDFOD. The synthetic dataset Syn2-2 is used to test used for massive data.

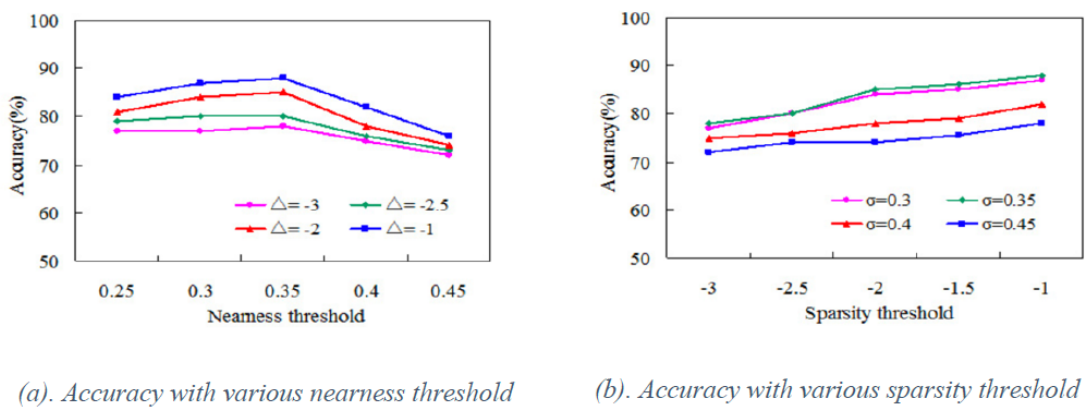


Fig. 2. Impacts of parameters σ and Δ on the accuracy of HDFOD. The synthetic dataset Syn2-2 is used to test used for massive data.

Fig. 2 depicts HDFOD's accuracy in outlier detection. More specifically, we draw two intriguing observations from Fig. 2(a). First, when the nearness threshold is increasing from0.25 to 0.35, the mining accuracy is improved. If the near- ness threshold σ is configured to a small value(e.g., 0.25,0.3 and 0.35), then a small number of objects are pruned. These reduced objects are meaningless for outlier detection and have adverse impacts on mining accuracy. The second observation is that the mining accuracy is worsened when σ is set to a big value (e.g., 0.4 and 0.45). The reason is that a high σ can prune a large number of objects that include some important information for maintaining high mining accuracy. Fig. 2(b) reveals that a large sparsity threshold Δ improves HDFOD's mining accuracy. When sparsity thresholdΔ is a large value, then more sparsity subspaces can be found. Therefore, that an increasing number of outliers can be searched from the sparsity subspace enhances the HDFOD's accuracy. On the contrary, if Δ is configured to a small value, the number of sparsity subspaces is decreased. Hence, some sparsity subspaces including outliers may not be detected, and the accuracy of HDFOD may be worsened.

6.2 Scalability

We evaluate the scalability performance of *HDFOD* by increasing the numbers of objects and attributes. Two groups of synthetic datasets are tested to drive the scalability analysis of *HDFOD*.

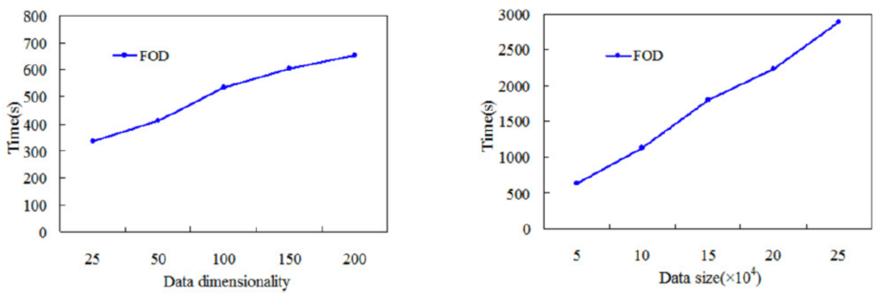
Fig. 3 exhibits the efficiency of HDFOD with an increasing number of data attributes and objects. The experimental results plotted in Fig. 4(a) indicate that the execution time of HDFOD increases when the number of attributes is sharply enlarged. The outlier detection process is slowed down because the number of subspaces is quickly increased by the excessive attribute number. Interestingly, the increasing speed of HDFOD's time is slower than that of the attribute number. This

phenomenon implies that our HDFOD can be applied to high-dimensional data. Fig. 3(b) shows that when the size of the dataset increases from 5×104 to 2.5×105, the time of HDFOD is quickly increasing. The range of searching subspace sharply increases with the increasing size of datasets, which leads to the worsening of the efficiency of outlier detection. The running time increases approximately in a line with the increase of dataset size. Thus, our *HDFOD* can be used for massive data.

Fig. 4 reveals that the accuracy of *HDFOD* remains almost unchanged when data dimensionality and size are varied. The experimental results illustrated in Fig. 4(a) show that *HDFOD*'s accuracy is declined from *90%* to *87%* when the number of attributes increases from *25* to *200*. This range of decline is small, the reason is that we use GA in *HDFOD to* search sparsity subspace, where the fitness function (e.g., sparsity degree, Formula 4) has nothing to do with the number of attributes. Fig. 4(b) presents a similar experimental result when the number of objects increases from 5×10^4 to 2.5×10^5. Such a high accuracy is attributed to GA that can find the optimal solution from a large number of data. Hence, from the perspective of accuracy, our *HDFOD* is suitable for not only high-dimensional data, but also massive data.

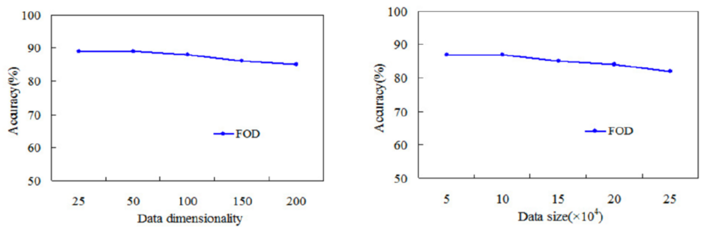
Fig. 5(a) reveals that *HDFOD* is able to achieve highly efficient results and its performance is generally consistent. Fig. 5(a) shows that *HDFOD* takes less time than the other two algorithms. The reason is that *HDFOD* uses fuzzy constraint to prune some unrelated objects for outlier detection. Hence, in *HDFOD* method, outliers are detected from a reduction dataset; otherwise, the other two algorithms detect outlier from an original dataset. Thus, our *HDFOD* has a little searching space, which leads to a high efficiency, compared with the other two algorithms.

Fig. 5(b) significantly depicts the accuracies of three algorithms. From the experimental results, *HDFOD* has a higher accuracy compared with *Gen* and *NOF*. In our *HDFOD*, a fuzzy constraint technology is used to prune negative objects for outlier detection. These negative objects do not help to detect outliers; conversely, they have a negative influence on the algorithm's accuracy. Therefore, HDFOD has more chances for finding the correct outliers, that is, HDFOD has a high accuracy



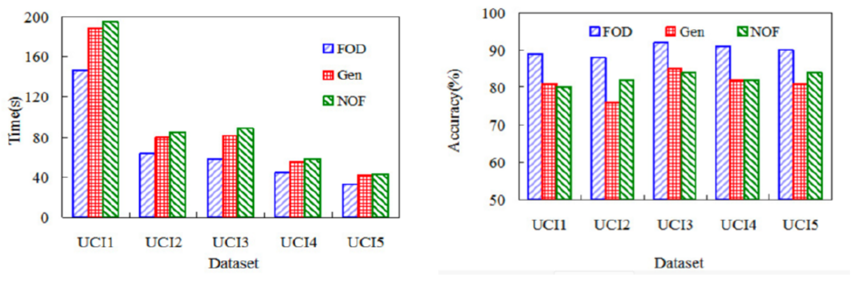
(a). Accuracy with various dimension numbers (b). Accuracy with various object numbers

Fig. 3. HDFOD's efficiency with increasing numbers of data attributes and objects. The nearness threshold is set to 0.35, and the sparsity threshold is set to -2



(a). Accuracy with various dimension numbers (b). Accuracy with various object numbers

Fig. 4. HDFOD's accuracy with increasing numbers of data attributes and objects. The nearness the sparsity threshold is set to 0.35, and threshold is set to -2



(a). Efficiency comparison (b). Accuracy comparison

Fig. 5. Comparisons of accuracy and efficiency among HDFOD, Gen and NOF. Five UCI datasets are tested.

7 Conclusion and Future Work

We have developed a fuzzy constraint-based outlier detection method called HDFOD, which improves not only the effectiveness and accuracy of outlier detection, but also the pertinence and understandability of mining results. To improve the pertinence, we used the nearness measure theory in fuzzy mathematics to describe a priori information and prune some objects that do not satisfy the constraint conditions. Such a constraint technique substantially reduces the sizes of datasets. HDFOD detects outlier by searching sparsity subspaces in reduction datasets. For searching optimized sparse subspaces, we extended and incorporated the GA into HDFOD. By using synthetic and UCI datasets, we were able to validate the efficiency of HDFOD in detecting constraint outlier. In addition, HDFOD's fuzzy constraint improves the pertinence of mining results. Performance tuning will be our future research direction. In particular, we will extend our approach to parallel and distributed computing environments, which enable HDFOD in processing large-scale high- dimensional datasets.

8 References

[1] E. Eskin, "Anomaly detection over noisy data using learned proba- bility distributions," in International Conference on Machine Learning, 2000, pp. 255–262.

[2] F. Chen, C. T. Lu, and A. P. Boedihardjo, "Gls-sod:a generalized local statistical approach for spatial outlier detection," in ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Wash- ington, Dc, Usa, July, 2010, pp. 1069–1078.

[3] S. Hido, Y. Tsuboi, H. Kashima, M. Sugiyama, and T. Kanamori, “Statistical outlier detection using direct density ratio estimation,” Knowledge and Information Systems, vol. 26, no. 2, pp. 309–336, 2011.

[4] S. M. Erfani, S. Rajasegarar, S. Karunasekera, and C. Leckie, “High- dimensional and large-scale anomaly detection using a linear one-class svm with deep learning,” Pattern Recognition, vol. 58, no. C, pp. 121–134, 2016.

[5] S. Y. Jiang and Q. B. An, “Clustering-based outlier detection method,” in Fifth International Conference on Fuzzy Systems and Knowledge Dis- covery, 2008, pp. 429–433.

[6] Y. Shi and L. Zhang, “Coid: A cluster-coutlier iterative detection approach to multi-dimensional data analysis,” Knowledge and Information Systems, vol. 28, no. 3, pp. 709–733, 2011.

[7] C. C. Aggarwal, “A segment-based framework for modeling and mining data streams,” Knowledge and information systems, vol. 30, no. 1, pp. 1–29, 2012.

[8] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, “A multi-step outlier-based anomaly detection approach to network-wide traffic,” Information Sciences, vol. 348, pp. 243–271, 2016.

[9] F. Jiang, G. Liu, J. Du, and Y. Sui, “Initialization of k-modes clustering using outlier detection techniques,” Information Sciences, vol. 332, pp. 167–183, 2016.

[10] J. Huang, Q. Zhu, L. Yang, D. D. Cheng, and Q. Wu, “A novel outlier cluster detection algorithm without top-n parameter,” Knowledge-Based Systems, vol. 121, pp. 32–40, 2017.

[11] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, “Lof: identifying density-based local outliers,” in ACM sigmod record, vol. 29, no. 2. ACM, 2000, pp. 93–104.

[12] B. Liu, Y. Xiao, P. S. Yu, Z. Hao, and L. Cao, “An efficient approach for outlier detection with imperfect data labels,” IEEE Transactions on Knowledge and Data Engineering, vol. 26, no. 7, pp. 1602–1616, 2014.

[13] J. Huang, Q. Zhu, L. Yang, and J. Feng, “A non-parameter outlier detection algorithm based on natural neighbor,” Knowledge-Based Systems, vol. 92, no. C, pp. 71–77, 2016.

[14] M. Salehi, C. Leckie, J. C. Bezdek, T. Vaithianathan, and X. Zhang, “Fast memory efficient local outlier detection in data streams,” IEEE Transactions on Knowledge and Data Engineering, vol. 28, no. 12, pp. 3246–3260, 2016.

[15] C. C. Aggarwal and S. Y. Philip, “An effective and efficient algorithm for high-dimensional outlier detection,” The International Journal on Very Large Data Bases, vol. 14, no. 2, pp. 211–221, 2005.

[16] H.-P. Kriegel, P. Kroger, E. Schubert, and A. Zimek, “Outlier detection ¨ in arbitrarily oriented subspaces,” in IEEE International Conference on Data Mining, 2012, pp. 379–388.

[17] E. Muller, M. Schiffer, and T. Seidl, “Statistical selection of relevant ¨ subspace projections for outlier ranking,” in 2011 IEEE 27th International Conference on Data Engineering (ICDE). IEEE, 2011, pp. 434–445.

[18] J. Zhang, Y. Jiang, K. H. Chang, S. Zhang, J. Cai, and L. Hu, “A concept lattice based outlier mining method in low-dimensional subspaces,” Pattern Recognition Letters, vol. 30, no. 15, pp. 1434–1439, 2009.

[19] J. Zhang, X. Yu, Y. Li, S. Zhang, Y. Xun, and X. Qin, “A relevant subspace based contextual outlier mining algorithm,” Knowledge-Based Systems, vol. 99, no. C, pp. 1–9, 2016.

[20] X. Zhao, J. Zhang, and X. Qin, “Loma: A local outlier mining algorithm based on attribute relevance analysis,” Expert Systems with Applications, vol. 84, pp. 272–280.

[21] L. A. Zadeh, “Fuzzy sets,” Information and Control, vol. 8, no. 3, pp. 338–353, 1965.

[22] Z. Ghahramani, “Probabilistic machine learning and artificial intelli- gence.” Nature, vol. 521, no. 7553, pp. 452–459, 2015.

[23] D. Conte, P. Foggia, C. Sansone, and M. Vento, “Graph matching applications in pattern recognition and image processing,” in International Conference on Image Processing, 2003. ICIP 2003. Proceedings, 2017, pp. II–21–4 vol.3.

[24] E. Cables, M. T. Lamata, and J. L. Verdegay, “Rim-reference ideal method in multicriteria decision making,” Information Sciences, vol. s 337-338, pp. 1–10, 2018.

[25] Y. Gurevich, “Logic and the challenge of computer science,” Trends in Theoretical Computer Science, pp. 1–57, 2017.

[26] W. Bandler and L. J. Kohout, “Special properties, closures and interiors of crisp and fuzzy relations,” Fuzzy Sets & Systems, vol. 26, no. 3, pp. 317–331, 1988.

[27] R. Hosseini and M. Maziani, “Type-2 fuzzy pattern recognition,” in Pattern recognition summer school, 2010, p. 3.

[28] H. B. Mitchell, “Pattern recognition using type-ii fuzzy sets,” Infor- mation Sciences, vol. 170, no. 2C4, pp. 409–418, 2005.

[29] D. E. Goldberg, “Genetic algorithm in search, optimization, and machine learning,” vol. xiii, no. 7, pp. 2104–2116, 1989.

[30] D. Wu and J. M. Mendel, “A vector similarity measure for linguistic approximation: Interval type-2 and type-1 fuzzy sets,” Information Sciences, vol. 178, no. 2, pp. 381–402, 2008.

[31] D. Wu and J. M. Mendel, “A comparative study of ranking methods, similarity measures and uncertainty measures for interval type-2 fuzzy sets,” Information Sciences, vol. 179, no. 8, pp. 1169– 1192, 2009.

[32] Ali and N. Senan, “The effect of normalization in violence video classification performance,” in International Conference on Advances in Computing and Intelligent System, 2017.

[33] C. Aggarwal, J. B. Orlin, and R. P. Tai, “Optimized crossover for the independent set problem,” Operations Research, vol. 45, no. 2, pp. 226–234, 1997.

[34] K. Kailing, H.-P. Kriegel, and P. Kroger, “Density-connected subspace ¨ clustering for high- dimensional data.” in Proceedings of the 2004 SIAM International Conference on Data Mining, pp. 246–257, 2004.

[35] W.N.Street,“UCI machine learning repository,” 1998. [Online]. Available: http://archive.ics.uci.edu/ml

[36] M. Young, The Technical Writer’s Handbook. Mill Valley, CA: University Science, 1989.

[37] V. Sharma, “A Fuzzy Constraint Based Outlier Detection Method,” in International Conference on Intelligent Computing, vol 11645, 2019