Received: August 25, 2019

INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Efficient Outlier Detection for High Dimensional Data using Improved Monarch Butterfly Optimization and Mutual Nearest Neighbors Algorithm: IMBO-MNN

M. Rao Batchanaboyina[1]* **Nagaraju Devarakonda[2]**

[1]Department of Computer Science and Engineering,
Acharya Nagarjuna University, Guntur, Andhra Pradesh, India
[2]School of Computer Science and Engineering,
VIT- AP University, Amaravathi, Andhra Pradesh, India
*Corresponding author's Email: mallik.mit@gmail.com

Abstract: The hybrid Improved monarch butterfly optimization- mutual nearest neighbor (IMBO-MNN) is proposed for outlier detection in high dimensional data. It is a challenge to detect outliers in high dimensional information. The external behavior of the data points cannot be detected in high-dimensional data except in the locally relevant data sub-sets. Subsets of dimensions are called subspaces, and with an increase in data dimension, the number of those subspaces grows exponentially. In another subspace an information point that is an outlier can appear ordinary. It's essential to assess its outlier behavior according to the amount of subspaces in which it appears as an outermost part to characterize an outlier. Data is scarce in high-dimensional space and the concept of closeness does not preserve meaning. In fact, the sparsity of the high-dimensional data means that every point is nearly equal from the point of view of closeness-based finishes. As a result, for higher dimensional information finding is more complicated and non-obviously significant outliers. An enhanced MBO (IMBO) algorithm is offered for enhanced search precision and run time efficiency by a fresh adaptation provider. Statistical results indicate that the elevated local optimal prevention and quick convergence rate of the improved monarch butterfly optimization (IMBO) algorithm helps to exceed the basic MBOs in outlier detection. Comparatively, IMBO produces very competitive outcomes and tends to surpass present algorithms. Optimal value k remains a task, affecting the efficiency of kNN straightforwardly. We are presenting a fresh learning algorithm under kNN in this paper to alleviate this issue called mutual nearest neighbor (MNN). The main feature of our method is that the class marks of unknown instances are defined by mutually next to one another, instead of by closest neighbor. The advantage of mutual neighbors is that in the course of the prediction process pseudo close neighbors can be identified and taken not into account. The performance of the suggested algorithm has been examined with a number of studies. For 100 data, IMBO-MNN is in 4897 milliseconds, PSO is in 5239 milliseconds, random forest is 5347 milliseconds, PNN is 5278 milliseconds and KNN is in 5166 milliseconds. For 250 data, IMBO-MNN is in 5984 milliseconds, PSO is in 6132milliseconds, random forest is 6145 milliseconds, PNN is 6124 milliseconds and KNN is in 6152 milliseconds. For 500 data, IMBO-MNN is in 6416 milliseconds, PSO is in 6636 milliseconds, random forest is 6634 milliseconds, PNN is 6719 milliseconds and KNN is in 6710 milliseconds. For 1000 data, IMBO-MNN is in 6913 milliseconds, PSO is in 7111 milliseconds, random forest is 7019 milliseconds, PNN is 7134 milliseconds and KNN is in 7162 milliseconds. The proposed IMBO-MNN performs better with minimum time taken. The findings indicate that the technique can identify outliers in high dimensional data efficiently in a decreased calculation moment.

Keywords: Improved monarch butterfly optimization (IMPO), Mutual nearest neighbor (MNN), Outlier Detection, Subspaces and High dimensional data.

1. Introduction

An outlier is the event that is so distinct from the other findings that it is suspected of a distinct system [Hawkins-1980]. Most of these apps are high-dimensional fields where information may

have several hundred sizes [1]. Outliers are need to be detected in sectors like medical, public health, scamming, and sports statistics etc. The outliers can be detected in many forms. However, our job is to detect outliers with greater dimensional data [2]. Most of the latest research on the outliers' technique suggests that the information is comparatively small [3]. Many latest algorithms use nearness ideas to figure out outliers depending on their relation to the other information. In a large area however, the data are sparse and the idea of proximity cannot be retained. The sparseness of large-scale information in reality means that every item is nearly equal from the point of view of near-dimensional definitions [4]. As a result, the idea of discovering significant outliers is much more complicated and unclear for high dimensional data [5]. The outlier detection of novelty, error and interference is also known [6], and in reality, these techniques have many common traits, all of which are designed to identify outer observations instead of representative models [7]. The algorithm LOF was recently successfully used in external detection [8]. The Local Outlier Factor (LOF) is an algorithm based on densities, which detects local surface areas of a date set by allocation to each object a degree of outliers called the LOF [9]. Data points of less density are recognized as outlines within the LOF algorithm than their adjacent points [10]. Outliers dominate many actual figures, and a big study focus has been put on creating solid, non-excessive Principal Component Analysis (PCA) algorithms [11]. Data outliers are of excellent concern to the machine learning and data mining communities as they are able to disclose extraordinary behaviors, exciting trends and outstanding information events [12]. Identification or removal of outliers in the assessment of information becomes a crucial preprocessing phase [13]. For instance, sound suppression can enhance design efficiency because noises can disrupt the discovery of significant data while anomalous access detection can assist us remove interference from network access by reviewing access documents within the firewall at a time [14].

In our proposed work we propose an Improved Monarch Butterfly Optimization (IMBO) [15] algorithm with Mutual Nearest Neighbors (MNN) [16] for efficient outlier detection in high-dimensional data. IMBO has very low standard deviation compared to the other optimizers, which shows the robustness and stability of the algorithm.The IMBO algorithm passes the position of butterfly to the MNN function to check accuracy and hence produces optimized results. For each optimizer based on test datasets, the average precision rates and their standard deviation are calculated. This MNN is key to the possibility of commonly closest neighbours, which is, through the mutual neighbours, it predicts the class name of the new case. The advantage of mutual neighbors additionally makes forecast increasingly valid. In particular, MNN recognizes and utilizes mutual neighbors to decide the class tag in the wake of getting the nearest neighbors. In IMBO, all butterfly individuals were created by the migration operator and passed on to the individuals to come. This application works through the inquiry room in IMBO-MNN application to discover predictions with negative sparsity coefficients, in spite of the fact that at a much scaled down cost. The proposed IMBO-MNN performs better with minimum time taken to compute data when compared with other techniques.

The remaining article is structured as defies: Section 2 illustrates the existing works related to the proposed method. Section 3 describes the methods which are needed for this paper. Section 4 describes the proposed methodology on Hybrid IMBO-MM for outlier detection in multi-dimensional date. Section 5 presents the simulation results. Section 6 defines the conclusion.

2. Related work: a brief review

A method for detecting outliers in both mixed and single type was introduced by Mohamed Bouguessa [17]. They introduced bivariate beta mixture model to identify outliers in mixed-attribute data. However the accuracy of detecting outlier is low in this method. For the identification of outliers in heigh-dimensional sensor data, Xiaowu Deng et al. [18] developed a support high-order tensor data description (STDD) and kernel support high-order tensor data description (KSTDD). But this method did not guarantee the optimal result. Virgile Fritsch et al. [19] introduced the regularization in Minimum Covariance Determinant (MCD) estimator for detecting outlier in high dimensional data. The main disadvantage in this method is the computational cost is too high. Ayadi, et al [20] in 2019 described an outlier detection technique based on prediction approach and classification approach for WSN. But the main drawback in this approach is the slow convergence rate and hardware redundancy.

The Randomized strategy to the autonomous model and the Randomized Robust PCA to the sparse column outlier system showed the right subspace with computing complexity and sample complexity dependent only on the information volume by the parameters of coherency [21]. But

developing the optimal algorithms was too difficult in this method due to its underlying nonconvexity. Shu Wu and Shengrui Wang [22] collaborated to investigate alternatives to detect outlying materials. They provided step-by-step (ITB-SS) and single-pass (ITB-SP) techniques centered on information theory. The experiment demonstrated that the algorithms used can handle information sets with many artifacts and characteristics. However the defined optimal factor was need to be updated. Therefore it required more computational time. For the identification of outlier, the writers [23] employed hybrid technology for evolution. Their experiment resulted in the submitted technique of finding outbound by monitoring the density ranges of the information. Sparsity issues in elevated dimensionality were efficiently solved. But this method failed when applied to the high dimensional date. Chakraborty, et al [24] has introduced a novel outlier detection technique using deep stacked autoencoders and probabilistic neural networks for outlier identification. It degrades the performance of the outlier detection is the main disadvantage in this method. A novel MBO algorithm based on random local perturbation and opposition-based learning to create the opposition based population from the original population and also to improve the migration operator. This MBO algorithm easily fall into the local optima is the main disadvantage [25].

From the literature survey, they mainly focused on the identification of the outlier of high dimension of data and managing the outlier with single attribute information. The main challenges of the existing methods such as accuracy and computational time are achieved by the proposed technique. An improved MBO (IMBO) algorithm enhances the search operation and MNN method recognizes the outlier efficiently.

3. Preliminaries

MBO is a population meta-heuristic influenced by monarch butterflies ' migration behavior. The primary method of the MBO is to update butterflies generated by two primary providers: a migration operator and an adjusting operator (continuing solutions or people). These two carriers can be run separately and at the same time in MBO. MBO can be used in parallel processing with this function. MBO uses an iterative process to generate and update its people, like other swarm based and developmental algorithms. The following procedures can be defined as:

Step 1: Initialization: MBO initiates by producing randomly a pre-defined amount (population) of butterflies, which each butterfly can solve the issue.
Step 2: Fitness evaluations: The objective function of all butterflies is used to assess and sorted according to their fitness.
Step 3: Division: The classified population is split into two sub-populations: L1, and L2. L1 and L2 dimensions are governed by a set proportion *p*.
Step 4: Migration: This manufacturer uses randomly chosen L1 and L2 butterflies to build the first part of the fresh generation. The volume of the newly generated portion is equal to that of L1, in which each H^{r+1}in this phase is built as follows: suppose that $H_{i,k}{}^{r+1}$is the value of position *x* of individual *i*. Then, $H^{r+1}$is generated using the migration operator, as given in Eq. (1).

$$H_{i,k}^{r+1} = \begin{cases} H_{a_1,k}^{r} & a \le p \\ H_{a_2,k}^{r} & a > p \end{cases} \tag{1}$$

In Eq. (1), $H^{r}_{a1,k}$ and $H^{r}_{a2,k}$ are two random individuals selected from L1 and L2, at iteration *r*. *ra* is a random number obtained using (2), where *rand* is a random number generated from the uniform distribution and *pri* is a constant value equals 1.2. *pri* indicates the migration period.

$$ra = rand \times pri \tag{2}$$

Step 5: Adjustment: The second portion of the fresh generation is built by the carrier. This part is equal to L2 in volume. In contrast to the migration operator, the adjustment operator produces a fresh portion from L2 on the basis of the finest people and other random people. Assuming that $H_{i,k}{}^{r+1}$ is the value of the element *k* of the individual number *j*, then $H_{i,k}{}^{r+1}$is generated using Eq. (3) and it is further updated as shown in Eq. (4)

$$H_{j,k}^{r+1} = \begin{cases} H_{best,k}^{r} & rand \le p \\ H_{a_3,k}^{r} & rand > p \end{cases} \tag{3}$$

$$H_{j,k}^{r+1} = H_{j,k}^{r+1} + \alpha(dk_x - 0.5) \tag{4}$$

$$dx = levy(H_j^r) \tag{5}$$

Where α is stepping factor to control *dx* in the updation process, *dx* is a local walk by levy flight. Finally the two generated sub populations are combined together to create a new population. This process is repeated until the best solution is reached.

The pseudocode of the MBO is given in Algorithm 1.

Algorithm 1 Pseudocode of MBO

```
Initialize W=2, MaximumStepSize (mSS)=1.0,
Adjusting rate (part) =5/12, Period (pd)=1.2, n=6,
set maximum generation =MItr, Generation counter
=Itr
Initialize random population (pop) and find the best
Procedure IMBO (Pop, W, mSS, MItr, n,….)
For Itr=1 to MItr do
 Divide the population into L1 and L2
   For i=1 to numBtFly1 do
                                  *Migration Operator
     For j=1 to P do
        a1=rand * pd
        if a1 ≤ part then
         a2=round (numBtFly1 * rand+0.5)
           L1(i, j)=L1(a2,j)
        Else
         a3=round (numBtFly2 * rand+0.5)
             L1(i, j)=L1(a3,j)
        End if
     End for
   End for
   For i=1 to numBtFly2 do
                                  *Adjusting Operator
      Scale=mSS/(Itr2) // scale determination
      SS=exrnd(2*MItr) // step size
      delata H=Levy F(SS,P)
     For j=1 to P do
        if rand ≥ part then
              L2(i,j)=best(j)
        Else
            a4=round (numBtFly2 * rand+0.5)
                L2(i,j)=Pop2(a4,j)
          if rand > BR then
           L2(i, j)= L2(i, j)+scale*(delataH(j)-0.5)
           End if
         End if
     End for
   End for
End for
Pop= combine (L1, L2)
Elite best two individuals
Evaluate (Pop)
Return best
End procedure
```

4. Proposed methodology

4.1 Hybrid IMBO-MNN

It proposes a fresh algorithm for classification, known as Mutual nearest neighbor (MNN). In contrast to the traditional approaches used by k neighbors to forecast the fresh example class label x_0, MNN uses the notion of the closest neighbor, x_0, as the basis for the definition of its label. MNN defines first its closest shared neighbors and takes a choice in order to evaluate the tag ofx_0. The benefit is that the tag you are predicting is more credible because it originates from the close neighbor. In addition, during the forecasting operation certain "false" neighbors or outliers are also excluded. In certain actual apps, setting a globally optimal value of k in kNN may not be a great choice. In reality, example$x_1$in fig. 1, where information is always supposed to be of a good quality, is often seen as an outlier in bank fraud, network assessment or intrusion detection or loud classification and object recognition information. Noisy data such as$x_1$should therefore be removed because the efficiency of classifiers can substantially degrade. A concept call from the mutual nearest neighbor is implemented to eliminate this type of noise.

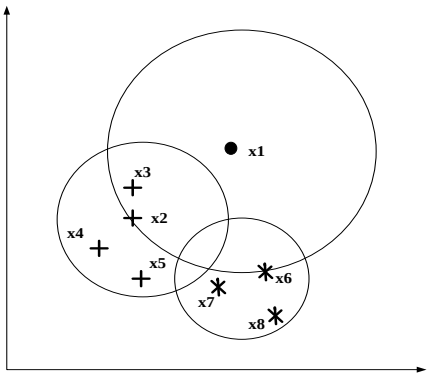

Figure. 1 Instance x_1 and its nearest neighbor

We can use the concept of MNN to assess the pseudo-neighbors in kNN to a certain extent. An example x locates intuitively in the middle of its neighbors when it is also seen as one of its neighbors by all of its neighbors. In other terms, x is not an isolated item (or sound information) if the closest vicinity of x takes it as its nearest adjacent. This sends a tip that x is not a noisy data if the closest partners have each other and the bigger $MN_k(x)$ of x, the center among their neighbors. Conversely, if x has not any closest shared neighbors, x will be regarded as sound information, i.e. $MN_k(x) = \{\}$.

In the light of the number of mutually close neighbors in the data set we are able to achieve the goal of data reduction. Another benefit of MNN is that the anticipated x tag is more credible with the use of $MN_k(x)$, because its pseudo-neighbors are removed and the remainder is nearer to them. The connection between personal buddies always

resembles that of our nearest mates, while we doubt other friends by those we do not know about. It is therefore more sensible if the prediction is made of a different number of shared neighbors than k-nearest neighbors. We propose the hybrid IMBO-MNN to pass the position of the butterfly in IMBO into the MNN function to optimize the complexity efficiency. The new adjustment officer has the same job as the ancient operator, but utilizes a distinct strategy, to build the second portion of the population. The objective of the fresh operator is to improve the equilibrium between research and use in the MBO algorithm. The proposed $H^{r+1}$of the fresh generation, in comparison to the ancient adaptation provider, produce the highest person H^r_{best} and the *r-th* edition of the individual H^r from past generation. Every item *k* is made using the mechanism shown in Eq. (6) in the fresh person H_j^{r+1}.

$$H_j^{r+1} = \begin{cases} H^r_{j,k} & s_1 > D \\ H^r_{best,k} - G \times rand\ s_2 \le 0.5\ s_1 \le D \\ H^r_{best,k} + G \times rand\ s_2 > 0.5 \end{cases} \quad (6)$$

Here $s_1$changing variable is used in this system to determine whether a fresh component k should take the highest person H^r_{best} or H^r_j of the previous generation into account. The value of $H^{r+1}{}_j$,*k* is copied to *j*, *k*, i.e. the individual inherits element *j* from his previous version, when $s_1$is greater than a Threshold *D*. In comparison, switcher $s_2$is used when building *Xt*+1 *j*, *k* depending on $H^r_{best,k}$ to determine the direction. s_2 ands_1 switches are both random digits taken in the interval from a uniform allocation [0,1]. The *D* threshold value is vibrant and linearly rises over the algorithm's iterations. The value of *D* is determined with (7) where *N* represents the complete amount of iterations, *f* is the present number and b is the cross-section of the line equation with a tiny value in the interval [0, 0.5]. The likelihood that s_1 will be less than limit *D*, is obviously proportionate to the number of iterations, which implies the MBO Algorithm will use $H^r_{best,k}$ to increase exploitation when the development is greater, than it was in the initial iterations. Note also that the likelihood of exploitation increases in the original iterations by increasing the significance of *y*. However, the setting *y* to a tiny value of around 0.2 has been found to be highly effective forest issues in this research because the change from exploration to exploitation is smoothed down.

$$D = y + \left(\frac{1-y}{N}\right) f \quad (7)$$

Variable *G* is used in the second instance of Eq. (4) to determine the magnitude of the steps in the operational method. *G* is computed using (8) in which n is an integer constant.

$$G = 1 - \left(\frac{f}{N}\right)^{\frac{1}{n}} \quad (8)$$

The pseudocode of IMBO-MNN is shown in Algorithm 2.

Algorithm 2 Pseudocode of Hybrid IMBO-MNN

Input: A training dataset *T*, an instance *x* and the number of nearest neighbors *k*;
Output: the predicted class label d(*x*) of *x*;
Initialize the label set *D*(*x*) and MNN $MN_k(x)$ of *x* as empty set, respectively, i.e. *D*(*x*) =Ø and $MN_k(x)$ = Ø
Initialize W=2, mSS=1.0, part=5/12, pd=1.2, n=6, rand, best;
Obtain the *k* nearest neighbors $N_k(x)$ of *x* from *T*;
For each neighbors $x_i \in N(x)$ of *x* **do**
Obtain the *k* nearest neighbors $N_k(x_i)$ of x_i from *T*;
If $x \in N_k(x_i)$ **then** // x_i is a MNN of *x*
Add x_i into $MN_k(x)$, i.e., $M_k(x) = M_k(x) \cup \{x_i\}$;
Add the class label d_i of x_i into *D*(*x*), i.e., $D(x) = D(x) \cup \{d_i\}$;
End if
IMBO (Pop, W, mSS, MItr, n,….)
For Itr=1 to MItr **do**
Divide the population into L1 and L2
For i=1 to numBtFly1 **do**
*Migration Operator
For j=1 to P **do**
a_1=rand * pd
if $a_1 \le$ part **then**
a_2=round (numBtFly1 * rand+0.5)
L1(i, j)=L1(a_2,j)
Else
a_3=round (numBtFly2 * rand+0.5)
L1(i, j)=L1(a_3,j)
End if
End for
End for
*Adjusting Operator
D=y + Itr * ((1-y)/MItr)
G=1-(Itr/MItr)$^{(1/n)}$
For i=1 to numBtFly2 **do**
For j=1 to P **do**
a_2=rand
if a_2 < D **then**
a_3=rand
if $a_3 \le$ D **then**
L2(i, j)=best(j)-G * rand
Else

L2(i, j)=best(j)+G * rand
End if
End if
End for
End for
End for
Pop= combine (L1, L2)
Elite best two individuals
Evaluate (Pop)
Return best
If $MN_k(x) = Ø$ then x is a noisy data; otherwise the class label $d(x)$ of x is determined with the majority voting in Eq. (9):
End procedure

$$d(x) = arg_{d_i}^{mx} \sum_{d_{i\in D(x)}} \sum_{x_{i\in MN_{k(x)}}} I(d_j = d_i) \qquad (9)$$

This algorithm operates easily and is simple. It initializes the group of applicant category mark s$d(x)$ and mutually close neighbors $MN_k(x)$ of x first of all before the forecast method begins. Secondly seeks to achieve the k nearest x neighbors using MNN search method. The algorithm will then start to define the closest $N_k(x)$ neighbors. The k values of each neighbor of x are also discovered by D to determine if x_i refers to the mutually exclusive neighbors of x. If applicable, then $MN_k(x)$ is inserted and its tag is converted to the x category tag of candidates. Finally, if a majority voting approach is used for the category tag $d(x)$ of x where it has shared neighbors, it will be determined; otherwise it will be a single point and not considered.

4.2 Efficient outlier detection for high dimensional data using IMBO-MNN

In order to assess outlier conduct on a rank-based rating, the detection of outliers is essential and helpful. The interest of an outlier can be measured in a data analyst's rating. The majority of external algorithms for identification are labeling processes to make a binary choice whether or not an information point is an outlier. Scoring and classification of outliers could contribute to a stronger comprehension of outliers ' behavior, as regards the remainder of the information. We offer an effective outlier detection algorithm using all information sub-spaces in this paper. A large number of real data sets are very high dimensional. Real data sets can include hundreds or thousands of aspects in some scenarios. Many standard surface detection techniques are not working very efficiently with growing dimensionality. This is an object of the famous dimension curse. In high-dimensional space, the information is sparse and, when fully analyzed, the real outlines are obscured by the sound impacts of several insignificant measurements. The purpose of this document is to classify information points according to their outlier conduct in a high-dimensional dataset. We do not take advantage of previous understanding of the basic distribution of data and are focused on unmonitored methods. Because of the dimensionality, each point is probably equidistant from each other in a full data space. But information points behave differently in the fundamental information sub-spaces. Some subspaces can be filled tightly with a data point while it can appear in the rest of the sub-spaces as an outline. It is therefore essential to explore the fundamental subspaces of the data set in order to discover significant outlines and also to assess the relative strength of outer behavior.

Let X be a data matrix of n rows (data points) and k columns (dimensions). An element X_{ij} of this data matrix represents a measurement of i-th data point in j-th dimension. The i-th row is a k-dimensional vector denoted by $X_i = X_{i1}, X_{i2},\ldots,X_{ik}$. A S subspace is a k size subgroup with dimensions $1,2,\ldots,k$. An information room k-dimensional has 2^k-1axis-parallel subspace. As the amount of subspaces increases exponentially and sizes decrease, our objective of classification meets significant difficulties at two stages. Firstly, it is computationally costly to examine such a big amount of information subspaces for outliers. The second problem is presented when the outliers are characterized. As the number of subspaces increases, the quantification of outside behaviour is hard. It is because the highly dimensional data need to be exponentially searched in order to consolidate the outer behaviour of all things.

Our data distribution and important sizes for detection of outliers are not known before. It is more essential to measure the outward behaviour of an information point than simply identify it as an outlier or inlier. We strive to better classify outliers in separate subspaces according to their attitude. We

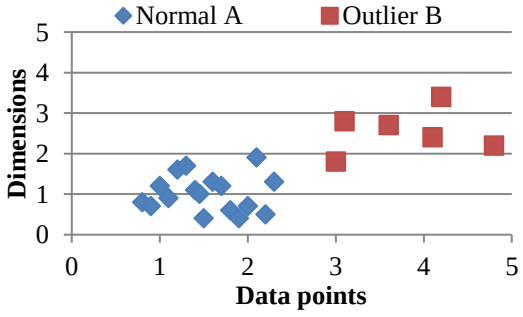

Figure. 2 Nearest neighbor **A** and **B** where k=3

are committed to supporting the process for enhancing performance via the outer score of each data point, taking into consideration the use of outlier detection for information washing. The dimensions of the subspaces must be adapted to our outlier detection technology. With increasing dimensionality our closest neighbors population reduces too, our parameters should be adjusted appropriately.

Given a note x, it usually includes the closest x neighbors with neighborhood data $N(x)$. The off-the-shelf learning algorithms such as MNN, e.g. $N(x) = \{x_1, x_2, \ldots, x_k\}$ can be obtained by x_i is the i^{th} closest neighbors. The closest A and B neighbors in fig.2 for instance is labelled with strong squares and dashed squares, if k=3 is taken into account in the kNN respectively. It should be emphasized that their neighbors are closely closer to each other for ordinary purposes, while the outliners are far away. We adopt the nuclear neighborhood rule as our anomalous level to delineate the characteristics of data distribution.

$$lps(x) = \|N(x)\|^* \tag{10}$$

where the larger is the $lps(x)$, the sparser the neighborhood of x.

In algorithm 3 we summarize the execution information of our method of identification. It consists of two main phases: 1) LPS estimates and 2) scale determination. In the former phase, the neighbors of x are obtained first by MNN and are then planned into a small subspace. The abnormal rating $lps(x)$ of x is predicted once the singular values are available. Assume that the information set Q shows n findings displayed by m. The cost of moment is $O(kn^2)$ the standard kNN algorithm. It needs $O(kn\log n)$ time to handle optimization and projection issues. Normally k is lower than m. Thus, $O(max(kn^3, knm^2))$ total is the time complexity of the suggested method.

Algorithm 3 Outlier Detection using IMBO-MNN

Input: The data collection **Q**, the number of neighbors k, and the number of outlier candidates s;
Output: The s desired outliers;
Pre-processing the data collection **Q**;
For each observation $\mathbf{x} \in \mathbf{Q}$
Obtaining the nearest neighbors $N(x)$ of **x** via IMPO-MNN; (*Algorithm 2)
Solving $\min_{\overline{Q}} \frac{1}{2}\left\|Q = \overline{Q}\right\|_F^2 + \lambda\left\|\overline{Q}\right\|*$ on $N(x)$ to extract principle components;
Projecting $N(x)$ into the desired low-dimensional subspace;
Calculating $lps(\mathbf{x})$ for **x** according to Eq. (5);
End For
Sorting local projection scores in a descending order;
Returning top s observations as desired outliers;

5. Experimental Results

5.1 Dataset description

Classification is one of the most efficient and straightforward guidelines for validating algorithms. Our simulation tests have no exception. In our simulation studies, we have selected 8 benchmark datasets of various kinds and sizes. All these data sets can be accessed from the UCI Machine Learning Repository and are often used to validate the classifier performance in literature. A few general data description for these datasets is summarized in Table 1.

5.2 Performance evaluation

The performance of the proposed system is compared with the existing systems like PSO and KNN. In fig. 3 proposed IMBO-MNN, PSO [26], KNN [27], random forest [24] and PNN [24] are compared for numbers of data against the time in milliseconds. For 100 data, IMBO-MNN is in 5003 milliseconds, PSO is in 5115 milliseconds, random forest in 5134 milliseconds, PNN in 5016 milliseconds and KNN is in 5164 milliseconds. For 250 data, IMBO-MNN is in 12515 milliseconds, PSO is in 12797 milliseconds, random forest in 12578 milliseconds, PNN in 12436 milliseconds and KNN is in 12966 milliseconds. For 500 data, IMBO-MNN is in 25038 milliseconds, PSO is in 25603 milliseconds, random forest in 25723 milliseconds, PNN in 25275 milliseconds and KNN is in 25851 milliseconds. For 1000 data, IMBO-MNN is in 50062 milliseconds, PSO is in 51232 milliseconds, random forest in 51346 milliseconds, PNN in 51497 milliseconds and KNN is in 51640 milliseconds.

Table 1. Dataset description

No	Datasets	Instances	Attributes	Classes	Outliers
1	Clean1	481	178	4	280
2	Glass	216	16	9	144
3	Ionosphere	381	48	4	218
4	Letter	22000	22	29	1143
5	Machine	246	14	9	126
6	Segment	2548	18	9	1406
7	Sonar	264	79	4	112
8	Wine	194	23	6	86

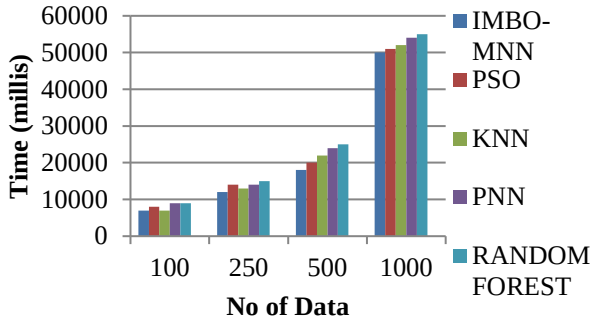

Figure. 3 Comparison between numbers of data against the time in milliseconds

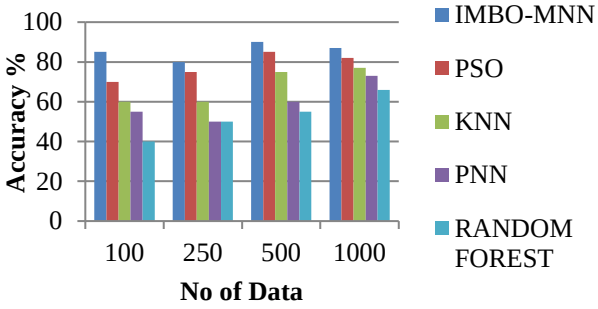

Figure. 4 Comparison between numbers of data against accuracy in percentage

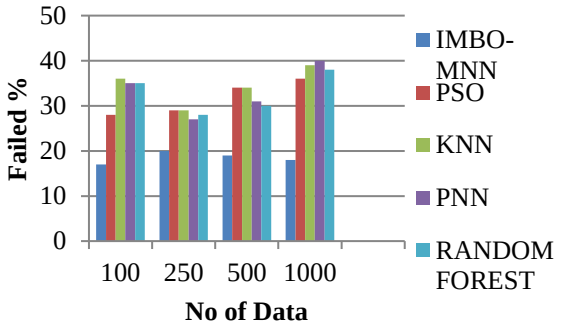

Figure. 5 Comparison between numbers of data against failed data in percentage

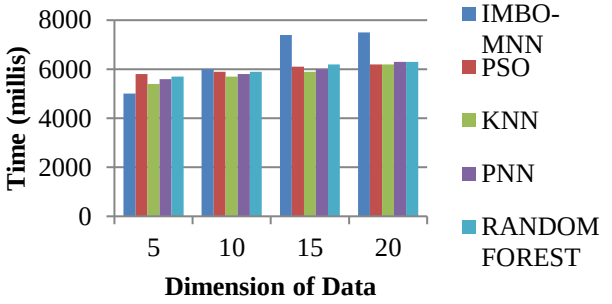

Figure. 6 Comparison between dimensions of data against the time in milliseconds

The proposed IMBO-MNN performs better with minimum time taken to compute data.

Proposed IMBO-MNN, PSO, random forest, PNN and KNN are compared for number of data against accuracy in percentage in fig. 4. For 100 data, IMBO-MNN has 82.83% accuracy, PSO has 68.43% accuracy, random forest has 75% accuracy, PNN has 67.63% and KNN has 61.12% accuracy. For 250 data, IMBO-MNN has 79.71% accuracy; PSO has 70.78% accuracy, random forest has 69.45%, PNN has 65.1% accuracyand KNN has60.03% accuracy. For 500 data, IMBO-MNN has 81.75% accuracy, PSO has70.63%accuracy, random forest has 73.67%,PNN has 63.9% accuracy and KNN has60.81% accuracy. For 1000 data, IMBO-MNN has 82.11% accuracy, PSO has 61.75% accuracy, random forest has 64.38% accuracy, PNN has 67.49% and KNN has 61.2% accuracy. The proposed IMBO-MNN performs better with high accuracy rate.

In fig. 5 proposed IMBO-MNN, PSO, random forest, PNN and KNN are compared for number of data against Failure in percentage. For 100 data, IMBO-MNN is 17.17% failed, PSO is 31.59% failed, random forest is 35.5% failed, PNN is 37.87%failed and KNN is 38.88%failed. For 250 data, IMBO-MNN is 20.29%failed, PSO is 29.22%failed, random forest is 31.24% failed, PNN is 35.39% failed and KNN is 39.97%failed. For 500 data, IMBO-MNN is 18.25%failed, PSO is 29.37% failed, random forest is 34.6% failed, PNN is 36.98% failed and KNN is 39.19%failed. For 1000 data, IMBO-MNN is 17.89%failed, PSO is 38.25%failed, random forest is 34.89% failed, PNN is 38.7% failed and KNN is 38.8%failed. The proposed IMBO-MNN performs better with minimum failed rate.

Proposed IMBO-MNN, PSO, random forest, PNN and KNN are compared for dimension of data against the time in milliseconds which is shown in fig. 6. For 100 data, IMBO-MNN is in 4897 milliseconds, PSO is in 5239 milliseconds, random forest is 5347 ms, PNN is 5278 ms and KNN is in 5166 milliseconds. For 250 data, IMBO-MNN is in 5984 milliseconds, PSO is in 6132milliseconds, random forest is 6145 ms, PNN is 6124 ms and KNN is in 6152 milliseconds. For 500 data, IMBO-MNN is in 6416 milliseconds, PSO is in 6636 milliseconds, random forest is 6634 ms, PNN is 6719 ms and KNN is in 6710 milliseconds. For 1000 data, IMBO-MNN is in 6913 milliseconds, PSO is in 7111 milliseconds, random forest is 7019 ms, PNN is 7134 ms and KNN is in 7162 milliseconds. The proposed IMBO-MNN performs better with minimum time taken.

In fig. 7 proposed IMBO-MNN, PSO, random forest, PNN and KNN are compared for dimension

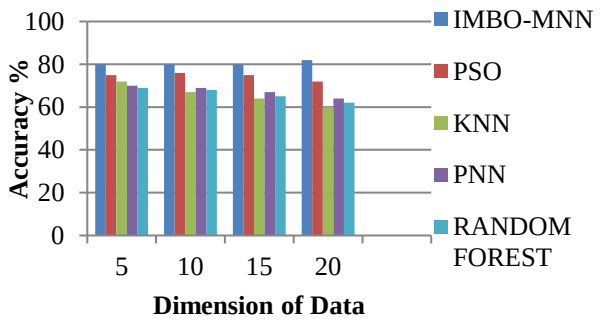

Figure. 7 Comparison between dimensions of data against accuracy in percentage

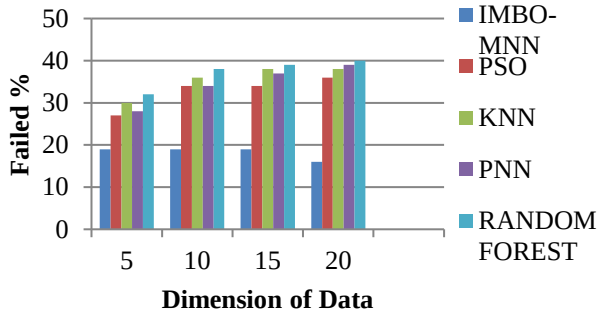

Figure. 8 Comparison between dimensions of data against failed data in percentage

of data against accuracy in percentage. For 100 data, IMBO-MNN has 80.69% accuracy, PSO has 64.98% accuracy, random forest has 67.8%, PNN has 64.28% accuracy and KNN has 60.48% accuracy. For 250 data, IMBO-MNN has 80.36% accuracy, PSO has 68.19% accuracy, random forest has 67.90% accuracy, PNN has 65.28% accuracy and KNN has 62.81% accuracy. For 500 data, IMBO-MNN has 80.7% accuracy, PSO has 65.04% accuracy, random forest has 64.9%, PNN has 63.42% accuracy and KNN has 62.19% accuracy. For 1000 data, IMBO-MNN has 82.98% accuracy, PSO has 63.16% accuracy, random forest has 70%, PNN has 68.47% accuracy and KNN has 60.13% accuracy. The proposed IMBO-MNN performs better with high accuracy rate.

Proposed IMBO-MNN, PSO, random forest, PNN and KNN are compared for dimension of data against Failure in percentage which is illustrated in fig. 8. For 100 data, IMBO-MNN is 19.31% failed, PSO is 35.02% failed, random forest has 39.4%failed, PNN is 35.67% failed and KNN is 39.52% failed. For 250 data, IMBO-MNN is 19.64% failed, PSO is 31.81% failed, random forest is 35.9% failed, PNN is 32.5% failed and KNN is 37.19% failed. For 500 data, IMBO-MNN is 19.3% failed, PSO is 34.96% failed, random forest is 29.57% failed, PNN is 31.78% failed and KNN is 39.19% failed. For 1000 data, IMBO-MNN is 17.02% failed, PSO is 36.84% failed, random forest is 37.45% failed, PNN is 34.21% failed and KNN is 39.87% failed. The proposed IMBO-MNN performs better with minimum failed rate.

6. Conclusion

In this document, we suggested a unique classification algorithm MNN for outlier detection by hybridizing IMBO with MNN. The MNN concept is central to the idea of mutually closest neighbors, which is, through the mutual neighbors, it predicts the class label of the new instance. In addition to distinguishing noisy data from the dataset, the benefit of mutual neighbors also makes forecast more credible. Specifically, MNN identifies and uses mutual neighbors to determine the class tag after obtaining the closest neighbors. In IMBO, all butterfly individuals were generated by the migration operator and passed on to the next generation. This application works through the search room in IMBO-MNN application to find predictions with very negative sparsity coefficients, although at a much reduced price. The findings have been analysed and discussed in terms of local optimum prevention, convergence speed and running time. By evaluating various data with the improved algorithm for outlier detection, the result shows the proposed IMBO-MNN method works more effective with minimum time consumption such as for 100 data, the IMBO-MNN has 80.69% accuracy whereas PSO has 64.98% accuracy, random forest has 64.9% accuracy, PNN has 63.42% accuracy and KNN has 60.48% accuracy.In future scope, the outlier detection in a high dimensional data with an improved IMBO algorithm will forecast the class label and also to achieve better performance in the proposed system, several optimization techniques may also be incorporated.

References

[1] M. Pardo and T. Hobza, “Outlier detection method in GEEs”, *Biometrical Journal*, Vol.56, No.5, pp. 838-850, 2014.

[2] P. Raña, G. Aneiros, and J. Vilar, “Detection of outliers in functional time series”, *Environmetrics*, Vol.26, No.3, pp. 178-191, 2015.

[3] J. Zhang and M. Zulkernine, “Anomaly Based Network Intrusion Detection with Unsupervised Outlier Detection”, In: *Proc. of the IEEE International Conference on Communications*, pp. 2388-2393, 2006.

[4] S. Barua and R. Alhajj, “High performance computing for spatial outliers detection using

parallel wavelet transform", *Intelligent Data Analysis*, Vol. 11, No. 6, pp. 707-730, 2007.

[5] A. Ghoting, S. Parthasarathy, and M. Otey, "Fast mining of distance-based outliers in high-dimensional datasets", *Data Mining and Knowledge Discovery*, Vol.16, No.3, pp. 349-364, 2008.

[6] N. Nigam and T. Saxena, "Global High Dimension Outlier Algorithm for Efficient Clustering and Outlier Detection", *International Journal of Computer Applications*, Vol.131, No.18, pp. 1-4, 2015.

[7] A. Kaur and A. Datta, "Detecting and ranking outliers in high-dimensional data", *International Journal of Advances in Engineering Sciences and Applied Mathematics*, Vol.11, No.1, pp.75-87,2018.

[8] F. Angiulli and C. Pizzuti, "Outlier mining in large high-dimensional data sets", *IEEE Transactions on Knowledge and Data Engineering*, Vol.17, No.2, pp. 203-215, 2005.

[9] G. Kollios, D. Gunopulos, N. Koudas, and S. Berchtold, "Efficient biased sampling for approximate clustering and outlier detection in large data sets", *IEEE Transactions on Knowledge and Data Engineering*, Vol.15, No.5, pp. 1170-1187, 2003.

[10] B. Liu and E. Fokoué, "Random Subspace Learning Approach to High-Dimensional Outliers Detection", *Open Journal of Statistics*, Vol. 5, No. 6, pp. 618-630, 2015.

[11] D. Gervini, "Outlier detection and trimmed estimation for general functional data", *Statistica Sinica*, Vol.22, No.4, pp.1639-1660, 2012.

[12] P. Gogoi, B. Borah, D. Bhattacharyya, and J. Kalita, "Outlier Identification using Symmetric Neighborhoods", *Procedia Technology*, Vol.6, No.1, pp. 239-246, 2012.

[13] X. Ru, Z. Liu, Z. Huang, and W. Jiang, "Normalized residual-based constant false-alarm rate outlier detection", *Pattern Recognition Letters*, Vol.69, No.1, pp. 1-7, 2016.

[14] J. Liu and J. Lian, "Outliers Detection of Dam Displacement Monitoring Data Based on Wavelet Transform", *Applied Mechanics and Materials*, Vol.71-78, No.1, pp. 4590-4595, 2011.

[15] G. Wang, S. Deb, X. Zhao, and Z. Cui, "A new monarch butterfly optimization with an improved crossover operator", *Operational Research*, Vol.18, No.3, pp. 731-755, 2018.

[16] H. Liu, S. Zhang, J. Zhao, X. Zhao, and Y. Mo, "A New Classification Algorithm Using Mutual Nearest Neighbors", In: *Proc. of 9th International Conference on Grid and Cloud Computing*, pp. 52-57, 2010.

[17] M. Bouguessa, "A practical outlier detection approach for mixed-attribute data. Expert Systems with Applications", *Expert Systems with Applications*, Vol.42, No.22, pp.8637-8649, 2015.

[18] X. Deng, P. Jiang, X. Peng, and C. Mi, "Support high-order tensor data description for outlier detection in high-dimensional big sensor data", *Future Generation Computer Systems*, Vol.81, No.4, pp.177-187.

[19] V. Fritsch, G. Varoquaux, B. Thyreau, J. Poline, and B. Thirion, "Detecting outliers in high-dimensional neuroimaging datasets with robust covariance estimators", *Medical Image Analysis*, Vol. 16, No. 7, pp.1359-1370, 2012.

[20] A. Ayadi, O. Ghorbel, M.S. BenSalah, and M. Abid, "Kernelized technique for outliers detection to monitoring water pipeline based on WSNs", *Computer Networks*, Vol.150, No.2, pp. 179-189, 2019.

[21] M. Rahmani and G. Atia, "Randomized Robust Subspace Recovery and Outlier Detection for High Dimensional Data Matrices", *IEEE Transactions on Signal Processing*, Vol.65, No.6, pp. 1580-1594, 2017.

[22] S. Wu and S. Wang, "Information-Theoretic Outlier Detection for Large- Scale Categorical Data", *IEEE Transactions on Knowledge and Data Engineering*, Vol.5, No.3, pp.589-602, 2013.

[23] A. Rao, D. Somayajulu, H. Banka, and R. Chaturvedi, "Outlier Detection in Microarray Data Using Hybrid Evolutionary Algorithm", *Procedia Technology*, Vol. 6, No.1, pp.291-298, 2012.

[24] D. Chakraborty, V. Narayanan, and A. Ghosh, "Integration of deep feature extraction and ensemble learning for outlier detection", *Pattern Recognition*, Vol.89, No.5, pp.161-171, 2019.

[25] L. Sun, S. Chen, J. Xu, and Y. Tian, "Improved Monarch Butterfly Optimization Algorithm Based on Opposition-Based Learning and Random Local Perturbation", *Complexity*, Vol.2019, No.4, pp.1-20, 2019.

[26] A.Wahid and A.C.S. Rao, "A distance-Based Outlier Detection Using Particle Swarm Optimization Technique", In: *Proc. of the Information and Communication Technology for Competitive Strategies*, pp. 633-643, 2019.

[27] Y. Pan, Z. Pan, Y. Wang, and W. Wang, "A new fast search algorithm for exact k-nearest

neighbors based on optimal triangle-inequality-based check strategy", *Knowledge-Based Systems*, 105088, 2019 doi: https://doi.org/10.1016/j.knosys.2019.105088