FAAD: A Self-Optimizing Algorithm for Anomaly Detection

Adeel Hashmi and Tanvir Ahmad
Department of Computer Engineering, Jamia Millia Islamia, India

Abstract: *Anomaly/Outlier detection is the process of finding abnormal data points in a dataset or data stream. Most of the anomaly detection algorithms require setting of some parameters which significantly affect the performance of the algorithm. These parameters are generally set by hit-and-trial; hence performance is compromised with default or random values. In this paper, the authors propose a self-optimizing algorithm for anomaly detection based on firefly meta-heuristic, and named as Firefly Algorithm for Anomaly Detection (FAAD). The proposed solution is a non-clustering unsupervised learning approach for anomaly detection. The algorithm is implemented on Apache Spark for scalability and hence the solution can handle big data as well. Experiments were conducted on various datasets, and the results show that the proposed solution is much accurate than the standard algorithms of anomaly detection.*

Keywords: *Anomaly detection, outliers, firefly algorithm, big data, parallel algorithms and apache spark.*

Received November 19, 2017; accepted April 28, 2019
https://doi.org/10.34028/iajit/17/2/16

1. Introduction

Anomaly detection is the process of identifying those patterns in data which do not conform to expected behaviour. These patterns are called as anomalies or outliers. According to [11], the definition of an outlier is: "An outlier is an observation that deviates so much from other observations (considered normal) as to arouse suspicion that it was generated by a different mechanism". For example, an abnormal traffic pattern in a computer network could mean that an unauthorized user is trying to compromise a system in the network. Anomaly detection finds application in a wide variety of domains such as intrusion detection in the field of networks and security; fraud detection in the field of banking and insurance, fault detection in wireless sensor networks, etc.

The outliers can be classified into different categories based on their occurrence; generally there are three kinds of outliers [10]:

1. *Point Outlier*: when a data point is different from remaining set of data points then this data point is termed as point outlier. For example, in credit card fraud detection, the outlier can be detected in terms of amount spent and location of transaction; if transaction amount is considerably higher compared to normal transactions and location is also different from usual then there is a high chance that it is a fraud transaction and an outlier.
2. *Collective Outlier*: when a collection of related data is different from rest of the data set then it is a case of collective outlier. For example, in some graph, there may be many low values but multiple low values in succession may lead to suspicion of an abnormal condition.
3. *Contextual Outlier*: when a data point is different from the rest with respect to some context then this data point is said to be a contextual outlier. For example, in context of age, a four feet child may be a normal person while four feet adult is an outlier.

Anomaly detection can be performed on textual as well as image data. The textual data can be spatial (related to the geographical conditions) or temporal (related to the time aspects). Images can also be used in anomaly detection, where abnormal patterns in the contents of the images can be identified (for example abnormal growth in a tissue). Basically anomaly detection is used for removing the noisy data and producing accurate data set. Various applications of outlier detection are listed below:

- *Fraud Detection*: fraud detection is one of the major applications of anomaly detection. Identifying transaction patterns which deviate from the normal transaction patterns of a particular customer may lead to suspicion of fraud.
- *Intrusion Detection*: intrusion detection identifies the suspicious patterns that may indicate a network or system attack from someone attempting to break into or compromise a system.
- *Sensor Networks*: a sensor network has multiple detection stations called sensor nodes. Reliability in wireless sensor networks is affected by various causes like environmental conditions, low quality sensors, etc., that lead to corrupted data generation by sensors. Through outlier detection techniques, faulty sensors are detected so that communication is not hindered.

- *Medical and public health outlier detection*: Anomaly detection in patient data helps to detect critical diseases at early stage for preventing it from becoming a severe and life-threatening disease. For example, detecting anomalous regions in images of heart or any other body part may lead to identification of a disease in early stages.

2. Anomaly Detection Algorithms

The anomaly detection techniques can be broadly divided into three categories: visual/graphical, statistical and machine learning. The visual techniques for anomaly detection includes scatter-plots which can help us discover anomalies in 2D datasets, but for large and multi-dimensional datasets we need some enhanced techniques like sparse traversal, etc., [5]. A statistics based definition of an outlier is: "an outlier is any data point more than 1.5 Inter Quartile Ranges (IQR) below the first quartile or above the third quartile"; based on this definition box-plots can be used for outlier detection [25]. Other statistical measures like frequencies [15], entropy [18], and correlation [24] have also been used in literature for anomaly detection.

The machine learning based algorithms for anomaly detection can be divided into three categories: supervised, semi-supervised and unsupervised. Supervised anomaly detection is analogous to a binary classification problem, in which a training dataset is provided containing points labelled as normal and outliers. All the classifiers like K Nearest Neighbours (kNN), Support Vector Machine (SVM), perceptron, naive bayes, random forest, etc., can be used for anomaly detection, however, supervised learning is not often used for anomaly detection as the ratio of outliers to normal points is quite low, hence training for outlier class is often not sufficient which leads to poor performance of the algorithms. Instead of supervised learning, semi-supervised learning is preferred for detecting anomalies. Semi-supervised learning is analogous to one-class supervised learning i.e., a model representing normal behaviour is constructed from a training data set. Those points in the test dataset which deviate from the normal are labelled as anomalies. The popular examples of semi-supervised algorithms are One-Class SVMs and replicator neural networks.

Most of the anomaly detection algorithms fall in the category of unsupervised learning. These unsupervised learning based algorithms can further be classified as distance-based and density-based. In distance-based approach, an object O in the dataset D is declared as an outlier if at least fraction *p* of the objects in D are farther than distance *d* from O [14]. Angiulli and Pizzuti [1, 2] and Angiulli *et al.*[3] calculated the sum of distances of the point under consideration with rest of the point in the dataset, and identified top *n* points which had highest cumulative sum (marked as outliers). Top-n kNN [22] is a simple approach for outlier detection where a distance measure like Euclidean distance can be used to identify *k* (user-defined value) nearest neighbours for each object, and declare *n* objects having highest cumulative/average value from its *k* neighbours as outliers. Distance-based clustering algorithms like k-means, Partition Around Medoids (PAM), etc., have also been used for outlier detection. In outlier detection by k-means algorithm, we initially make *k* clusters using k-means clustering algorithm, then we identify *n* points which have largest distance from their cluster centers to get *n* outliers.

Density Based Spatial Clustering of Applications with Noise (DBSCAN) [8] is one of the earliest algorithms to identify outliers, although its main objective is clustering. In DBSCAN, we set two parameters *eps* and *minPts*; we then start by picking a random point in the dataset and considering all the points in its neighbourhood which are at a distance less than *eps* away from it to form the cluster. Next we count the number of such points, if this count is more than *minPts* then this cluster is called as dense cluster. We expand the cluster by repeating the process for all the new points in the cluster. If we run out of new points and there are still points left to consider in the dataset then we again pick a random point and restart the process till all points in the dataset are considered. A point which has less than *minPts* in its cluster and is also not a part of any other cluster is declared as an outlier.

Local Outlier Factor (LOF) [4] is a non-clustering density-based outlier detection algorithm derived from DBSCAN algorithm, where a LOF score is calculated for each data point. In this method, number of neighbourhood points (*n*) to consider is set a priori; a reachability distance/density is computed (for the data point *p* under consideration) with *n* nearest neighbours. The reachability density of each of the *n* nearest neighbours is also calculated. The LOF score of the data point is ratio of the average density of the *n* nearest neighbour of the point and the density of the point itself. For a normal data point, the density of the point will be similar to that of its neighbours and the LOF value is low, whereas for an outlier LOF score will be high. Local Correlation Integral (LOCI) [20] addresses the difficulty of selecting the value of *n* in LOF by using different criteria for the neighbourhood. In LOCI, all the points within a value of *r* (radius) are considered as neighbours, and the reachability density is calculated w.r.t. to all these data points. In LOCI, a Multi-Granularity Deviation Factor (MDEF) value is calculated for each point; MDEF at radius *r* for a point is the relative deviation of its neighbourhood density from the average neighbourhood density in its *r* neighbourhood. Different values of *r* are taken and if the MDEF value deviates three times from the standard deviation of MDEFs in the neighbourhood then the point is flagged as an outlier.

Various strategies for anomaly detection in very large and high dimensional datasets have been discussed in literature. Koufakou *et al.* [15] implemented Attribute Value Frequency (AVF) in Map-Reduce for fast outlier detection in large categorical datasets. Liu *et al.* [18] proposed an Entropy-based Fast Detection algorithm for outlier detection in large datasets. Yan *et al.* [26] proposed a distributed outlier detection algorithm which employs compressive sensing for sampling high-dimensional data. Dimensionality Reduction for OUTlier detection (DROUT) can be used for anomaly detection in "very wide" datasets i.e., datasets with large number of features. Distributed Solving Sets (DSS) and Lazy Distributed Solving Sets (LDSS) are distributed strategies for anomaly detection in large datasets. For shorter turnaround time, GPU versions of parallel algorithms like DSS have also been implemented. Rapid Anomaly Detection (RAD) algorithm is used by NetFlix for outlier detection. RAD is able to handle high cardinality dimensions, non-normalized datasets, seasonality and minimizes false positives. RAD is based on Robust Principal Component Analysis (RPCA) which repeatedly calculates Singular Value Decomposition (SVD) and applies thresholds to singular values in each iteration. Recently, Twitter released an R package for anomaly detection in time-series data, which consists of Seasonal Hybrid Extreme Studentized Deviate (S-H-ESD) test for anomaly detection.

3. Swarm Intelligence

Swarm intelligence is a class of nature-inspired algorithms based on collective behaviour displayed by swarm of insects/organisms/animals in achieving some goal like foraging (act of wandering in search of food and other resources). These algorithms like other nature-inspired algorithms are used for optimization problems like minimizing a convex function, finding an optimal cluster, finding shortest route, etc.

3.1. Swarm-based Metaheuristics

Meta-heuristic term in the field of mathematical optimization refers to a high-level procedure which can be used for providing a good solution to an optimization problem, especially when exploration of whole search space is not possible. There are hundreds of swarm-based meta-heuristics available in the literature, and some of the popular ones are listed in Table 1.

Ant colony algorithm [6] is based on the ability of ants to find shortest path to the food source. Initially, the ants wander randomly in search for food. On locating food, they return to colony leaving a pheromone trail.

Table 1. Swarm Intelligence meta-heuristics.

Algorithm	Proposed by	Year
Ant Colony Optimization	Dorigo, Di Caro	1992
Particle Swarm Optimization	Kennedy, Eberhart, Shi	1995
Bacterial Foraging algorithm	Passino	2002
Artificial Fish Swarm	Li et. Al	2003
Glow-worm	Krishnanad and Ghose	2005
Artificial Bee Colony	Karaboga, Basturk	2007
Firefly algorithm	Xin-She Yang	2008
Cuckoo Search	Xin-She Yang, Deb	2009
Bat algorithm	Xin-She Yang	2010

When other ants come across such a trail, they quit wandering and follow the trail. The pheromones tend to evaporate with time; therefore pheromone density remains high on shorter paths compared to longer ones. So, the ants tend to follow the shortest path on which the pheromone density is highest.

Particle Swarm Optimization (PSO) [7] is based on bird flocking. The initialization in PSO is similar to that of genetic algorithm; a random population is generated to represent the members of the swarm. The swarm is updated throughout the generations in order to search for optimum. The particles fly through the solution space based on their own best value and the swarm's best value to obtain the global best.

Bacterial foraging algorithm [21] is based on the behaviour exhibited by bacteria while searching for food. The behaviour of bacteria depends upon the chemicals in its environment which may be secreted by other bacteria for communication. The bacteria move in swarms towards a location which has the optimal environment (in terms of food, etc.,).

The main concept in artificial fish swarm algorithm [17] is the visual scope of each fish. The visual scope of the fish may be empty, crowded, and normal. If the scope is normal, the fish moves towards the center of the scope. The swarming movement is activated if this central point has better fitness value; otherwise a random point within visual scope is selected (which also must have a better fitness value).

The glow worm algorithm [16] is based on the concept of bioluminescence/glow which is used by glow worms to communicate with each other. The glow worm with less light intensity move towards the glow worm with higher light intensity.

Artificial bee colony [13] algorithm is based on foraging behaviour of honey bees. Bee colonies send bees to collect nectar from flower patches in proportion to the amount of nectar available in the patch. These bees return and communicate with other bees at the hive via a waggle dance that informs other bees in the hive about the direction, distance, and quality of food sources.

Cuckoo search [31] algorithm is inspired by the breeding behaviour of cuckoos (to lay their eggs in the nests of other birds). Three basic operations associated with CSA are:

a. Every cuckoo lays one egg at a time, and dumps its egg in randomly selected nest in the environment.
b. The nests with good quality of eggs will remain for next generations.
c. The number of host bird nests is fixed, and the egg laid by a cuckoo is identified by the host bird depending on a probability in the range [0, 1] (under such situation, the host bird can either destroy the egg or destroy the present nest and build a new one).

Bat algorithm [27] is based on echolocation behaviour of bats. As the bat flies, it emits pulses which bounce back of the objects in its path and echo back to the bat. From these echoes, the bat can determine the distance of the object, size of the object, as well as its own speed of flight. The bat can adjust its velocity and pulse rates based on its proximity to the object/prey.

3.2. Anomaly Detection by Swarm Intelligence

According to Knorr's definition of an outlier, a data point is an outlier if it has a fraction ϕ or less of the total points within distance r (Figure 1). The values of ϕ and r are user-defined and very hard to decide. Instead of selecting values of ϕ and r randomly, an optimization algorithm can be used to obtain these values. The idea is: for each point p, consider a circle of radius r with p being the center, find the number of points being enclosed by this circle; let the number of points being enclosed is k, then the aim is to find the point p^* and radius r^* such that the value of k/r is minimized. This value of r^* is then used to compute the k/r ratio of other points and rank them. The points with low k/r ratio will be outliers. We can also set a value ϕ such that if $k/n<\phi$ (n is total number of points) for a point, then such a point will be an outlier.

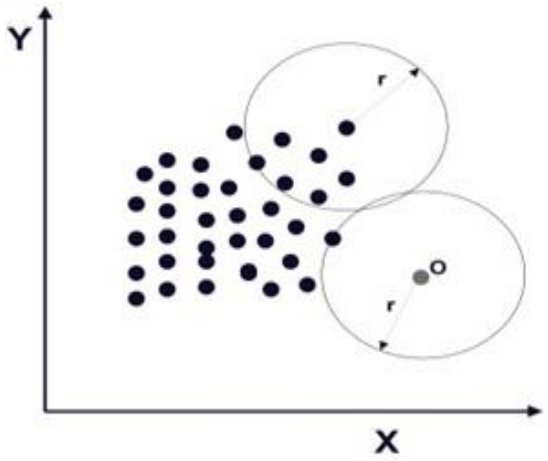

Figure 1. Knorr's definition of an outlier.

Swarm intelligence algorithms are used in many optimization problems and can also be used for this problem. The fitness function is k/r and we have to minimize this fitness function. However, if r is very small then k may be 0 giving the least possible value of k/r, and if r is very large then also $k/r \to 0$ when $r \to \infty$. Hence, the value of r should not be too high or too low. Therefore, we need to set a lower limit as well as an upper limit on the value of k. So, acc. to [19] the fitness function that could be used is:

$$\frac{\alpha}{r*k}+\frac{k}{r}+\frac{k}{n-k} \quad (1)$$

The first term is to limit the lower bound of r, and third term is to limit the upper bound of r. For a small value of r, the first term will be high so fitness function will return a high value (we will consider minimum value of k to be 1 i.e., we will consider the point itself in the value of k). If we set value of α=0.05n, then it will ensure that the value of r will be large enough. The third term becomes infinity for k=n (case when r is very large), so points for which third term comes out to be infinity will clearly not be outliers. This ensures that r is not too small which may lead to missing the outliers or too large to include outliers from the normal set of points.

4. Firefly Algorithm

Firefly algorithm developed by Yang [28, 29], is a meta-heuristic algorithm inspired by nature. Firefly algorithm is among those stochastic algorithms which follow randomization approach to search the solution. Some of the important fields of its application are optimization of dynamic and noisy environment and constraints, combinatorial and multi-objective optimization [33]. Apart from the field of optimization it is also capable of solving classification problems that we come across in the fields of neural network, data mining and machine learning [23, 32].

4.1. Biological Foundation

Fireflies are distinguished by their flashing light produced by a biochemical process known as bioluminescence. These bright flashing lights serve as a signal for mating as well as warning signals from potential predators. The fireflies move towards the brightest/fittest/optimal partner for mating and this behavior is exploited in the metaheuristic to find the optimal solution for a given problem. Firefly algorithm makes the following assumptions [9, 30]:

- The brightness/fitness of a firefly corresponds to the objective function.
- Each firefly is attracted to all other fireflies as they are unisex.
- A brighter firefly is more attractive, and a less bright firefly will move towards a firefly which is brighter. The attractiveness/brightness increases as the distance b/w the fireflies' decreases.

The objective function is the convex/fitness function which is to be minimized. Each firefly represents a candidate solution, and the aim is to find the solution

which gives the minima of the convex function. This can be done by calculating the fitness value of each solution, and moving the value of less fit solutions towards the fittest solution. This process may lead to a local best instead of global best, so the brightest firefly is needed to be moved randomly to a location where its brightness is increased even further. This random movement of brightest firefly ensures that we are able to reach a global best instead of local best. In next section, the mathematical formulation of the movements of the fireflies is presented.

4.2. Mathematical Foundation

From physics, we know that the light intensity $I(r)$ varies according to the inverse square law:

$$I(r) = I_s / r^2 \tag{2}$$

Where, I_s is the light intensity at source.

The light intensity I vary with distance r for a stated medium with absorption coefficient λ, acc. to the Equation:

$$I = I_0 e^{-\lambda r} \tag{3}$$

Where, I_o is the actual light intensity.

The Equations (2) and (3) can be combined to give the following Equation:

$$I = I_0 e^{-\lambda r^2} \tag{4}$$

Taking the above equations into consideration, the attractiveness of a firefly can be defined as:

$$\beta = \beta_0 e^{-\lambda r^2} \tag{5}$$

Where, β_0 is the attractiveness of the firefly at r=0.

In the real time environment, the attractiveness function of the firefly i.e., $\beta(r)$ can be any monotonically decreasing function described in the generalized form as:

$$\beta(r) = \beta_0 e^{-\lambda r^m} \tag{6}$$

The cartesian distance (r) is the distance between any two random fireflies i and j at location x_i and x_j.

$$r_{ij} = \left\| x_i - x_j \right\| = \sqrt{\sum_{k=1}^{d} (x_{i,k} - x_{j,k})^2} \tag{7}$$

In 2-dimensional case, we have

$$r_{ij} = \sqrt{(x_i - x_j)^2 - (y_i - y_j)^2} \tag{8}$$

The less bright firefly l is attracted to another more attractive firefly m according to the Equation:

$$x_l = x_l + \beta_0 e^{-\lambda r^2} (x_m - x_l) \tag{9}$$

$$x_l = \beta_0 e^{-\lambda r^2} x_m + (1 - \beta_0 e^{-\lambda r^2}) x_l \tag{10}$$

This equation (Equation (10)) gradually changes the value of the parameter (to be optimized) in each iteration so as to move towards the best value found so far until we reach convergence to obtain the global best. The value of the parameter λ plays a significant role in determining the speed of convergence and it follows the range of 0 to 10. Higher the value of λ, faster is the convergence to the solution but we should also not set it so high that it jumps the optimal solution (normally this value is set to 1).

4.3. The Algorithm

The pseudo-code for firefly meta-heuristic is given below.

begin
Identify objective function f(x);
Generate initial population x_i of fireflies;
Formulate light intensity I_i at x_i via f(x_i);
while t<maxGenerations do begin
for i := 1 to n do begin
for j := 1 to n do begin
if I_j>I_i then
Move firefly i towards j by using Eq. (10);
Evaluate new results; end;
Update light intensity; end;
Rank the fireflies and find the current best; end; end;
Display results;
end.

The algorithm starts with a population of fireflies where each firefly corresponds to a possible solution from the solution space. The solutions are evaluated based on a defined fitness function; the better the solution, brighter the firefly. The less bright fireflies are moved towards more bright fireflies i.e. the parameters of less promising solutions are changed so that they are closer to the parameters of more promising solutions (according to Equation (10)). To prevent the algorithm from converging towards local maxima/minima instead of global maxima/minima, the brightest firefly is moved randomly so as to increase its brightness. This process is repeated for a fixed number of iterations so as to converge to a global maxima/minima.

4.4. Firefly Algorithm for Anomaly Detection

Firefly algorithm can be quite easily used for detecting anomalies by optimizing the fitness function expressed in Equation (1) to obtain the optimal value of r which can be further utilized to detect the outlier points in the data. What we need to do is to start with a random population of fireflies (corresponding to the parameter to be optimized which in this case is the radius r) and find the brightest firefly (best value of r) in the current population and move remaining (less bright) fireflies i.e., towards this brightest firefly (optimal value of r) until convergence is reached. The pseudo-code for detecting outliers using firefly

algorithm is given below. In the firefly algorithm there are generic parameters like number of fireflies, gamma, beta0 as well as problem specific parameters like fitness function, etc., The pseudo-code for Firefly Algorithm for Anomaly Detection (FAAD) algorithm is given below.

Algorithm 1: (FAAD)

Input: Dataset D
Output: Anomalous data points in D
begin

Step-1. Initialization.
Generate a distance matrix dist_mat of D.
Define a list with N values of radius r, each value of r corresponds to a firefly.
Set objective function as F=k/r.

Step-2. Calculate brightness of each firefly.
For each r in list do
For each row in dist_mat do
i. Find number of elements less than r (set this value as k).
ii. Find fitness of rows acc to F.

Find row with minimum value of fitness function. The fitness of this row is the brightness of the firefly.

Step-3. Identify the brightest firefly.
In previous step, we get N fitness values corresponding to brightness of each firefly.
The firefly (value of r) with least fitness value is the brightest firefly, denoted by r.*

Step-4. Move fireflies towards the brightest firefly.
for each r in the list do
if(r!=r)*
update r acc. to Eq. (10)
else
update r by random walk*

Step-5. Repeat steps 3 and 4 for fixed number of iterations.
Return the value r (optimal value of r).*

Step-6. Rank the points for r.*
for each row in dist_mat do
i. Find number of elements less than r.*
ii. Find fitness of row acc. to F.

Rank all the rows (lower the fitness value, higher the rank).

Step-7. Return the k highest ranked points/rows as outliers and remaining as inliers.
end.

The algorithm starts with initializing the fireflies (a firefly corresponds to a value of *r*). Brightness of each firefly is calculated by identifying the point with lowest value of fitness function given in Equation 1. The less bright fireflies are moved towards brighter firefly i.e. the value of *r* for each firefly is changed so as to be near (by some fraction) to the best value of *r* in the current epoch. This process is repeated till convergence.

4.5. Parallelization in Spark MapReduce

MapReduce is a distributed/parallel programming model which runs on a cluster of commodity hardware, where a master node distributes the job to slave nodes for parallel processing. MapReduce model consists of two phases: map and reduce; in the map phase the job in divided into independent sub-tasks and assigned to slave nodes, and in the reduce phase the output returned by each slave is aggregated to give a final result. The two most popular tools which support MapReduce are Apache Hadoop and Apache Spark. Apache Spark is said to be faster than Apache Hadoop due to its in-memory computation capability and use of Resilient Distributed Database (RDD) on which two types of operations can be performed viz. transformation and action.

A Spark job typically contains sequential steps and those steps which contain independent sub-tasks can be parallelized by executing the sub-tasks in parallel as shown in Figure 2 below. In the MapReduce-based parallel implementation of the firefly algorithm for anomaly detection, the task of finding fitness of fireflies can be parallelized as all fireflies are independent of each other.

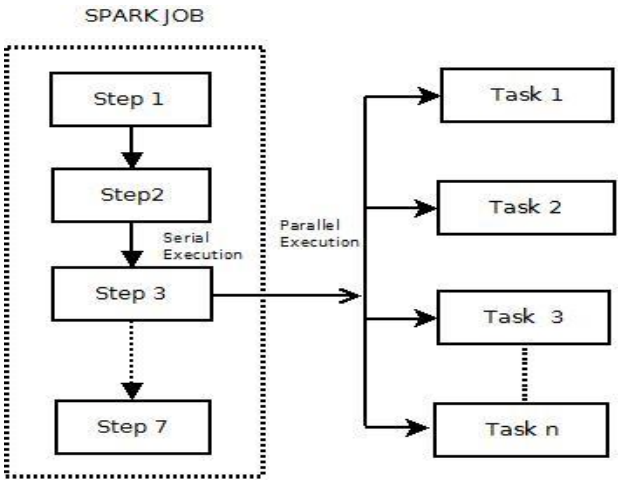

Figure 2. Spark Job.

A snapshot of parallelization in PySpark is given below.

```
sc = pyspark.SparkContext.getOrCreate()
data_rdd=sc.textFile(" dataset ")

# values of r
fireflies = [ 10, 100, 1000, 10000 ]

# convert the list to a Spark RDD
fireflies_rdd=sc.parallelize(fireflies)

def brightness(firefly):
# sequential code to find fitness
return fit

# code to parallelize a function
```

intensity = fireflies_rdd.map(lambda x: brightness(x))

5. Experiments

In this section, the performance of the proposed FAAD algorithm is compared with standard algorithms of anomaly detection viz. KMeans, DBSCAN, LOF, and Random Forest. Each of these algorithms require some parameters to be set, however the proposed FAAD algorithm doesn't require setting up of any parameters as it is self-optimizing. For k-means algorithm, the number of clusters was set to 30. For DBSCAN, we do not need to set the number of clusters, but we need to set the parameter *eps* i.e. the maximum distance between two samples for them to be considered as in the same neighborhood, and was given the value 0.8. LOF requires the number of neighbors to consider, and this value was set to 15. Random Forest requires the number of trees, which was set to 10. The algorithms were implemented in PySpark and the experiments were performed on i7 processor with 4 cores, 8GB RAM running Ubuntu 16.10. The performance was evaluated on the basis of accuracy of the algorithms (Table 2 and Figure 3). The metric used for accuracy is jaccard similarity between actual labels and predicted labels for each instance in the dataset. The experiments were conducted on openly available datasets: lymphography dataset, wisconsin breast cancer dataset, post-operative dataset, pageblocks dataset, credit card fraud detection dataset, forest cover dataset and kddcup99 dataset.

Lymphography dataset contains 148 instances and 18 attributes. There are 4 classes: normal find (label 1), metastasis (label 2), malign lymph (label 3), and fibrosis (label 4). Classes "normal find" and "fibrosis" have only 2 and 4 instances in the dataset, respectively and are treated as outliers. Hence, the dataset has 6 outliers and 142 normal points.

Post-Operative Patient dataset contains 90 instances with 9 attributes. The data points are classified into 3 classes: A, S and I. We will treat the points classified as A (64 in total) as normal points, whereas the points classified as I (2 in total) and S (24 in total) will be treated as outliers. So, the dataset consists of 26 outliers and 64 non-outliers.

Wisconsin breast cancer dataset has 483 instances with 10 attributes, containing 444 benign (label 2) instances and 39 malignant (label 4) instances. The original dataset has 699 instances, where 458 are benign and 241 are malignant. 16 instances have missing values, so we remove these instances, giving us 444 benign and 239 malignant instances. Following Hawkins [12], only one of every sixth malignant instance is kept, giving us 39 malignant instances, which are treated as outliers among 444 benign instances.

Page-blocks dataset has 5053 instances with 11 attributes, containing 4913 normal points and 140 outliers. The original page-blocks dataset has 5473 instances with 11 attributes. The dataset is mainly a classification problem where the blocks of documents are classified as text or non-text. Although it is a multi-class problem of 5 classes (text, horizontal line, picture, vertical line, and graphic), we will consider it as a two class problem: text and non-text, which gives us 4913 instances of text blocks and 560 instances of non-text blocks. We will take one of every four non-text instances i.e., we will keep only 140 instances which are not classified as text.

The credit card fraud detection dataset contains data of 284807 transactions with 31 attributes, out of which 492 transactions are labeled as fraud. These transactions are done with 172792 credit cards, hence there are multiple transactions for most of the cards.

The forest cover dataset has 581012 instances (with 54 attributes) which are classified into 7 classes (1-7), with class 4 having minimum number of instances (2747). We will consider instances of class 2 (283301 instances) and class 4 (2747 instances) only i.e., total 286048 instances. Moreover, we will consider only 10 attributes (ignoring 40 binary soil type attributes and 4 binary wilderness areas). Therefore, we will consider the instances from class 4 as outliers whereas the rest of the instances will be considered as inliers.

The original kddcup99 dataset (for intrusion detection) has 4898431 instances with 41 attributes. The original dataset has 3925651 attacks, but a smaller set with attribute "logged-in" as positive has 3377 attacks. Further, taking the "service" attribute, the dataset can be divided into http, smtp, ftp, ftp-data, others subsets. We consider only the http instances (567497 in total) with 2211 attacks and we will consider every type of attack as an outlier, and give same label to every type of attack.

Table 2. Accuracy of algorithms.

Dataset	K-Means	DBSCAN	LOF	Random Forest	FAAD
Lymphography	0.7942	0.8405	0.9256	0.9082	0.9729
Post-Operative	0.7461	0.7988	0.8666	0.8712	0.9111
Cancer	0.6438	0.7283	0.8033	0.8427	0.9751
Page-blocks	0.7159	0.8023	0.8750	0.8154	0.9703
Credit Card	0.6943	0.8217	0.8999	0.8545	0.9985
Forest Cover	0.5861	0.8096	0.8945	0.9148	0.9964
KDDCup99	0.5537	0.6158	0.6088	0.6859	0.9930

The FAAD algorithm doesn't assign labels to the instances, instead it ranks the instances in decreasing order of probability of being an outlier. So, the user needs to provide the count *k* of outliers to be detected, and the algorithm will label top *k* instances in the returned list as outliers, whereas rest of the instances will be labeled as normal. From the experimental results, it can be inferred that FAAD algorithm provides much accurate results than its counterparts.

6. Conclusions

In this paper, the authors proposed a firefly meta-heuristic based algorithm for anomaly detection. The major advantage of the proposed solution is that it is self-optimizing and is implemented on a big data tool in a parallel environment. Experiments were performed on multiple datasets to compare the performance of the proposed algorithm with existing algorithms. The results highlighted the drawbacks of existing algorithms which were run with default parameters, and therefore had poor accuracy. The proposed solution being self-optimizing doesn't suffer from such drawback and gives much accurate results.

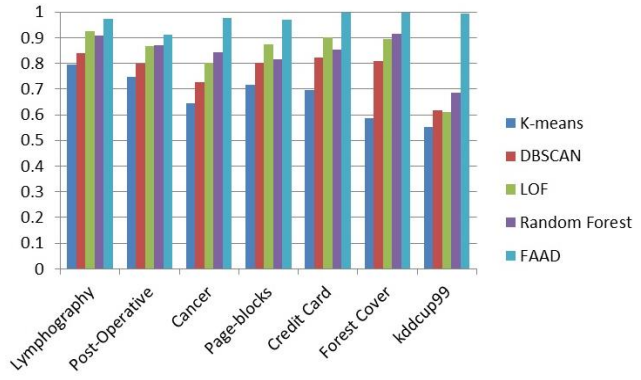

Figure 3. Comparative performance of algorithms.

References

[1] Angiulli F. and Pizzuti C., "Fast Outlier Detection in High Dimensional Spaces," *in Proceedings of European Conference on Principles of Data Mining and Knowledge Discovery*, Helsinki, pp. 15-27, 2002.

[2] Angiulli F. and Pizzuti C., "Outlier Mining In Large High-Dimensional Data Sets," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 2, pp. 203-215, 2005.

[3] Angiulli F., Basta S., and Pizzuti C., "Distance-Based Detection and Prediction of Outliers," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 2, pp. 145-160, 2006.

[4] Breunig M., Kriegel H., Ng R., and Sander J., "LOF: Identifying Density-Based Local Outliers," *in Proceedings of ACM SIGMOD Record*, vol. 29, no. 2, pp. 93-104, 2000.

[5] Bryson S., Kenwright D., Cox M., Ellsworth D., and Haimes R., "Visually Exploring Gigabyte Data Sets in Real Time," *Communications of the ACM*, vol. 42, no. 8, pp. 82-90, 1999.

[6] Dorigo M., Optimization, Learning and Natural Algorithms, Thesis, Politecnico di Milano, 1992.

[7] Eberhart R. and Kennedy J., "A New Optimizer Using Particle Swarm Theory," *in Proceedings of the 6th International Symposium on Micro Machine and Human Science*, Nagoya, pp. 39-43, 1995.

[8] Ester M., Kriegel H., Sander J., and Xu X., "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," *in Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, vol. 96, no. 34, pp. 226-231, 1996.

[9] Fister I., Yang X., and Brest J., "A Comprehensive Review of Firefly Algorithms," *Swarm and Evolutionary Computation*, vol. 13, no. 1, pp. 34-46, 2013.

[10] Han J., Pei J., and Kamber M., *Data Mining: Concepts and Techniques*, Elsevier, 2011.

[11] Hawkins D., *Identification of Outliers*, Chapman and Hall, 1980.

[12] Hawkins S., He H., Williams G., and Baxter R., "Outlier Detection Using Replicator Neural Networks," *in Proceedings of International Conference on Data Warehousing and Knowledge Discovery*, Aix-en-Provence, pp. 170-180, 2002.

[13] Karaboga D. and Basturk B., "A Powerful and Efficient Algorithm for Numerical Function Optimization: Artificial Bee Colony Algorithm," *Journal of Global Optimization*, vol. 39, no. 3, pp. 459-471, 2007.

[14] Knorr E. and Ng R., "Algorithms for Mining Distance Based Outliers in Large Datasets," *in Proceedings of the International Conference on Very Large Data Bases*, New York, pp. 392-403, 1998.

[15] Koufakou A., Secretan J., Reeder J., Cardona K., and Georgiopoulos M., "Fast Parallel Outlier 1 Detection for Categorical Datasets Using Mapreduce," *in Proceedings of IEEE International Joint Conference on Neural Networks*, Hong Kong, pp. 3298-3304, 2008.

[16] Krishnanand K. and Ghose D., "Glowworm Swarm Based Optimization Algorithm for Multimodal Functions with Collective Robotics Applications," *Multiagent and Grid Systems*, vol. 2, no. 3, pp. 209-222, 2006.

[17] Li X., A New Intelligent Optimization-Artificial Fish Swarm Algorithm, PhD Thesis, Zhejiang University, 2003.

[18] Liu B., Fan W., and Xiao T., "A Fast Outlier Detection Method for Big Data," *in Proceedings of Asian Simulation Conference*, Singapore, pp. 379-384, 2013.

[19] Mohemmed A., Zhang M., and Browne W., "Particle Swarm Optimisation for Outlier Detection," *in Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation*, Portland, pp. 83-84, 2010.

[20] Papadimitriou S., Kitagawa H., Gibbons P., and Faloutsos C., "Loci: Fast Outlier Detection Using the Local Correlation Integral," *in Proceedings of the 19th International Conference*

on Data Engineering, Bangalore, pp. 315-326, 2003.

[21] Passino K., "Biomimicry of Bacterial Foraging For Distributed Optimization and Control," *IEEE Control Systems Magazine*, vol. 22, no. 3, pp. 52-67, 2002.

[22] Ramaswamy S., Rastogi R., and Shim K., "Efficient Algorithms for Mining Outliers from Large Data Sets," *ACM SIGMOD Record*, vol. 29, no. 2, pp. 427-438, 2000.

[23] Sajwan M., Acharya K., and Bhargava S., "Swarm Intelligence Based Optimization for Web Usage Mining in Recommender System," *International Journal of Computer Applications Technology and Research*, vol. 3, no. 2, pp. 119-124, 2014.

[24] Sugumaran P., Ravi K., and Shanmugam T., "A Novel Algorithm for Enhancing Search Results By Detecting Dissimilar Patterns Based on Correlation Method," *The International Arab Journal of Information Technology*, vol. 14, no. 1, pp. 60-69, 2017.

[25] Tukey J., *Exploratory Data Analysis*, Addison-Wesley Publication Company, 1977.

[26] Yan Y., Zhang J., Huang B., Sun X., Mu J., Zhang Z., and Moscibroda T., "Distributed Outlier Detection Using Compressive Sensing," *in Proceedings of the ACM SIGMOD International Conference on Management of Data*, Melbourne, pp. 3-16, 2015.

[27] Yang X., *Nature Inspired Cooperative Strategies for Optimization*, Springer, 2010.

[28] Yang X., "Firefly Algorithm, Stochastic Test Functions and Design Optimization," *International Journal of Bio-Inspired Computation*, vol. 2, no. 2, pp. 78-84, 2010.

[29] Yang X., *Research and Development in Intelligent Systems*, Springer, 2010.

[30] Yang X., *Nature-Inspired Metaheuristic Algorithms*, Luniver Press, 2010.

[31] Yang X. and Deb S., "Cuckoo Search Via Lévy Flights," *in Proceedings of World Congress on Nature and Biologically Inspired Computing*, Coimbatore, pp. 210-214, 2009.

[32] Yang X. and He X., "Firefly Algorithm: Recent Advances and Applications," *International Journal Swarm Intelligence*, vol. 1, no. 1, pp. 36-50, 2013.

[33] Zang H., Zhang S., and Hapeshi K., "A Review of Nature-Inspired Algorithms," *Journal of Bionic Engineering*, vol. 7, no. 4, pp. 232-237, 2010.

Adeel Hashmi is a PhD scholar in Jamia Millia Islamia. He has done his BTech and MTech from IP University Delhi. He has over 8 years of teaching/research experience. His areas of interest in research are machine learning, data mining and big data.

Tanvir Ahmad is a Professor in Department of Computer Engineering at Faculty of Engineering and Technology. He has over 20 years of teaching and research experience. He has multiple publications in reputed international journals and conferences.