The International Journal Of Engineering And Science (IJES)
|| Volume || 4 || Issue || 11 || Pages || PP -70-77|| 2015 ||
ISSN (e): 2319 – 1813 ISSN (p): 2319 – 1805

An Efficient Unsupervised AdaptiveAntihub Technique for Outlier Detection in High Dimensional Data

[1]Mrs.R.Lakshmi Devi, [2]Dr.R.Amalraj
[1]*ReseachScholar, Mother Teresa Women's University, Kodaikanal, India*
[2]*Associate Professor, Dept. of Computer Science Sri Vasavi Arts College, Erode, India.*

----------ABSTRACT----------

Identification of unsupervised outliers in a high dimensional data becomes an emerging technique in today's research in the area of data mining. Increase of dimensionality leads to various challenges. Hubness especially Antihubs (points that infrequently occur in k nearest neighbor lists) is the recently known concept for the increase of dimensionality pertaining to nearest neighbors. Outlier detection using AntiHub method is refined as Antihub[2] to reevaluate the outlier scores of a point produced by the AntiHub method. However, it leads to increase the computation time of an algorithm with large number of computations. This paper establishes an approach called AdaptiveAntihub, which embeds an adaptive technique in Antihub[2] for unsupervised outlier detection mainly to reduce the number of computations and computation time of an algorithm and compares the results produced by Antihub[2] with AdaptiveAntihub. The experimental results illustrate that AdaptiveAntihub outperforms well and it also proves that there is a significant reduction in the number of computations and computation time when this proposed technique is applied to detect unsupervised outliers in high dimensional data.

Keywords: AdaptiveAntihub, Antihub, Antihub[2], Outliers, Unsupervised

Date of Submission: 14 November 2015 Date of Accepted: 25 November 2015

I. INTRODUCTION

An outlier is an observation which appears to be inconsistent with the remainder of that set of data. In data mining, detection of outliers is an important research area. Most of the applications which apply outlier detection are high dimensional. Increase of dimensionality leads to sparsity of data. Sparse data is difficult to handle. The sparsity of high dimensional data signifies that every point is an almost equally good outlier [1]. Outlier (anomaly) detection refers to the process of finding patterns that do not conform to standard behavior.

Outliers can be of three types such as point, contextual or collective outliers. Outlier detection techniques can be classified into three different categories such as supervised, semi supervised and unsupervised based on the existence of the labels for outliers. The unsupervised outlier detection is more applicable, where only a single data set without labels is given. The other techniques both require labelling data to produce the appropriate training set which is an expensive, time consuming and burdensome task [2]. In this paper, the proposed technique is applied to an unsupervised outlier detection. Hubness or k-hubness of an object x: $N_k(x)$ is the number of times a point x is counted as one of the k nearest neighbors of any other point in a data set [3].

The concept of hubness has recently become as an essential aspect of the increase of dimensionality related to nearest neighbors [4] and can be used in a standard methods used for detecting outliers. [5] Explores an impact of hubness on a various machine-learning tasks that utilizes distances between points, belonging to supervised, semi-supervised, and unsupervised learning families and by examining the origin of hubness, authors show that it is an essential property of data distributions in high-dimensional data.

The rest of the paper is organized as follows: Section 2 specifies an existing methods related to unsupervised outlier detection and Section 3 explains the proposed approach and its implications. Finally, Section 4 describes the experimental results with real datasets, and the conclusion is in Section 5.

II. RELATED WORK

In Recent research, various papers explored the influence of hubness in high-dimensional data on different data mining outlier detection tasks. Reverse nearest neighbors count is recognized in unsupervised distance-based outlier detection [4]. Outlier scoring based on N_k counts used in the ODIN method was reformulated and introduced here as an antihub which defines the outlier score of point x from data set D as a function of $N_k(x)$ and explores the interplay of hubness and data sparsity. Outlier detection methods are

implemented centered on the properties of antihubs. The relationship between dimensionality, neighborhood size, and reverse neighbors are taken into account for the effectiveness.

Unsupervised outlier detection is the process of detecting outliers in a given dataset without the need of labels in the training set. The paper [6] proposes method for finding distance based outliers based upon the k nearest neighbor points. To calculate the number of data points falling, two algorithms such as nested loop join and index join algorithms are used. Also partition-based algorithm is used. This algorithm divides the data set into different subsets and then cuts entire partitions rapidly as it is determined that they cannot contain outliers.

Distance based method to deal with the problem of finding outliers for k dimensional data sets where k >=5 is focused in paper [7]. Applying three algorithms such as index based, nested loop based, and cell based, authors come to the conclusion that cell based is for k<=4 and nested loop is the choice for k>=5 and also finds that there is no limit on the size of the dimensions.

A mostly used density based method is the local outlier factor (LOF) [8], which influenced many variations, e.g. LDOF (Local Distance-based Outlier Factor) approach [9], and LoOP (Local Outlier Probability) [10]. In many unsupervised outlier detection algorithms proposed, nearest-neighbor based algorithms appears to be the mostly used methods today [11, 12]. In this context, outliers are determined by their distances to their nearest neighbors.

The approach used in [13] is based on the relationship between k- Nearest Neighbor and Reverse Nearest Neighbor which involves two phases. First phase is dealing with the problem of finding a query point using kNN.The second step introduces Boolean Range Query which checks the existence of a point in a given region can be used for RNN queries problems.

Reverse nearest neighbor search [14] is used on high dimensional and multimedia data. [15] explores an important feature of the curse concerning to the distribution of k-occurrences (the number of times a point appears among the k nearest neighbors of additional points in a data set) and shows that, as dimensionality increases, this distribution of data is skewed and hub points arise (points with very high k-occurrences).

The concept of hubness (Antihub^2) presented in [4] motivated to implement the proposed approach AdaptiveAntihub to reduce the computation time by reducing computations.

III. PROPOSED APPROACH

I. Methodology

1.1 Antihubs

Hubness is derived from the idea of k occurrences. Different data points occur in k neighbor sets with increasingly unequal frequencies. When some points occur in many kNN sets, it is referred to as hubs, while others occur either very rarely or not at all, then they are referred to as antihubs. More specifically, hubness refers to an increasing skewness in the k occurrence distribution in high-dimensional data [16].

The concept of Antihub has recently become as an essential aspect of the increase of dimensionality related to nearest neighbors.In summary, the emergence of antihubs is closely interrelated with outliers in high-dimensional data suggests that antihubs can be used as an alternative to standard outlier-detection methods.

The basic structure of Antihub [4] is given below:

$AntiHub_{dist}$ (D, k)
Input:
- Distance measure dist.
- Ordered data set $D = (x_1, x_2. \ . \ . \ x_n)$, where $x_i \in R^d$, for $i \in \{1, 2 \ . \ . \ . \ n\}$
- No. of neighbors $k \in \{1, 2 \ . \ . \ .\}$

Output:
- Vector $s = (s_1, s_2. \ . \ . \ s_n) \in R^n$, where s_i is theoutlier scoreof x_i, for $i \in \{1, 2 \ . \ . \ . \ n\}$

Temporary variables:
- $t \in R$

Steps:
1) For each $i \in (1, 2 \ . \ . \ . \ n)$
2) $t := N_k(x_i)$ computed w.r.t. dist and data set $D \backslash x_i$
3) $s_i := f(t)$, where $f: R \rightarrow R$ is a monotone function.

Antihub method takes high dimensional data set as an input. For each point x, it finds out the $N_k(x)$ value (reverse k-nearest neighbor count of x) with respect to Euclidian distance. Outlier score is obtained based on the function f where it is $1/ (N_k(x) + 1)$, which assumes that the higher the score, the more the point is considered an outlier, and maps the scores to the (0, 1] range. If the outlier score crosses the limit of user defined threshold, then it is treated as an outlier.

1.2 Proposed Algorithm

An AdaptiveAntihub technique extending Antihub^2 is constructed to reevaluate the outlier score of a point by considering N_k scores of the neighbors of x in addition to $N_k(x)$ itself. This technique is explored in this paper especially to improve the efficiency and to reduce the computational time with reduced number of computations. The basic architecture of the proposed approach is given in Fig 1.

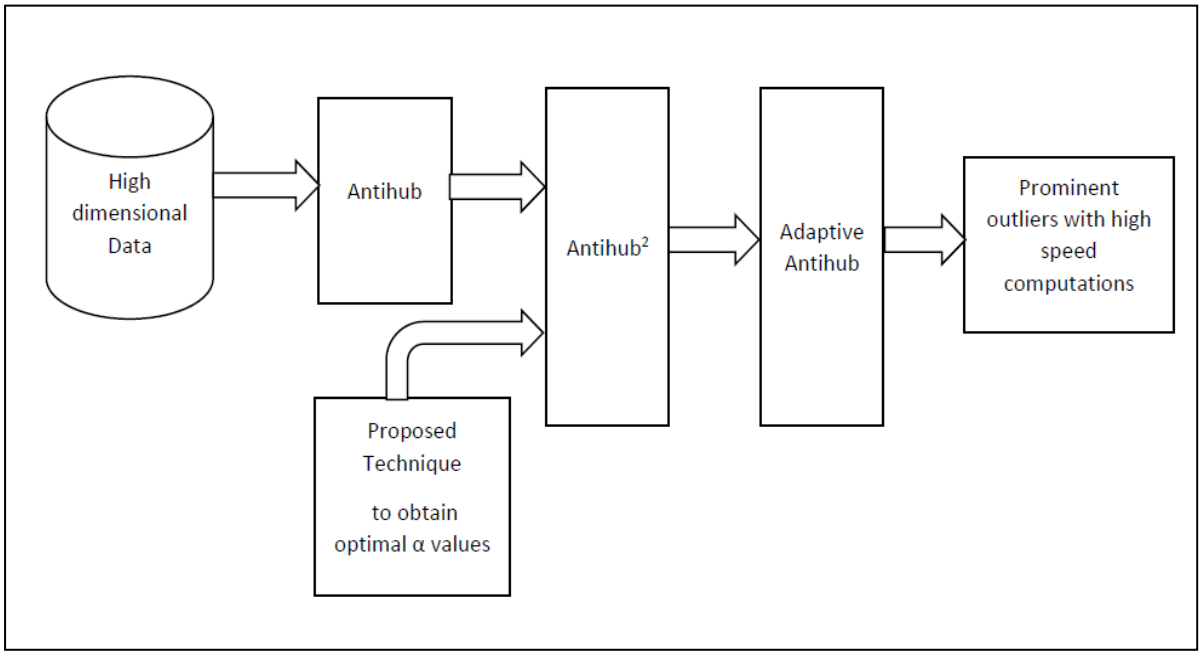

Fig 1. Basic architecture of the proposed approach.

This paper establishes an approach called AdaptiveAntihub which embeds an adaptive technique in Antihub^2 algorithm which is a refinement of Antihub where scores of $N_k(x)$ and N_k scores of the neighbors of x, are considered for unsupervised outlier detection. In high dimensional data, for each point x, AntiHub^2 adds $(1 - \alpha) \bullet N_k(x)$ to α times the sum of N_k scores of the k nearest neighbors of x, where $\alpha \in [0, 1]$. Since each α is applied for each point x and α is also measured by the step size where step $\in (0, 1)$, it may lead Antihub^2 to attain longer period of time mainly because of large number of iterations. In contrast to this concept, an AdaptiveAntihub is introduced where, it divides the α set into four sections. Instead of applying all α values for each point x, the proposed technique applies only the limited number of α. It omits moving into last section based on the concept that x depends its neighbors only on certain percentage. By comparing the corresponding cdisc values of first three sections it moves into the corresponding direction quickly so that it will reduce the number of computations and computation time of an algorithm.

The basic structure of the proposed algorithm AdaptiveAntihub is as follows:

AdaptiveAntiHub$_{dist}$ (x, k, p, step)

Input:
- Distance measure dist
- Ordered data set $D = (x_1, x_2. \ . \ . \ x_n)$, where $x_i \in R^d$, for $i \in \{1, 2 \ . \ . \ . \ n\}$
- No. of neighbors $k \in \{1, 2 \ . \ . \ .\}$
- Ratio of outliers to maximize discrimination $p \in (0, 1]$
- Search parameter step $\in (0, 1]$

Output:
- Vector $s = (s_1, s_2. \ . \ . \ s_n) \in R^n$, where s_i is the outlier score of x_i, for $i \in \{1, 2 \ . \ . \ . \ n\}$

Temporary variables:
- AntiHub scores $a \in R^n$
- Sums of nearest neighbors' AntiHub scores $ann \in R^n$
- Proportion $\alpha \in [0, 1]$
- (Current) discrimination score cdisc, disc $\in R$
- (Current) raw outlier scores ct, $t \in R^n$

Local functions:

discScore(y, p): for $y \in R^n$ and $p \in (0, 1]$ outputs thenumber of unique items among ⌈np⌉ smallest members of y, divided by ⌈np⌉.

1. a: = $AntiHub_{dist}$ (D, k)
2. For each $i \in (1, 2 \ldots n)$
3. ann_i: = $\sum NN_{dis}t$ (k, i) a_j, where NN_{dist} (k, i) is theset of indices of k nearest neighbors of x_i
4 .disc: = 0
5. For each $\alpha \in (0, step, 2 \cdot step \ldots 1)$
6. sbeg =1; send=no.of α values; divv=send/4;
   lmid=round (sbeg + divv);
   rmid=round (send – divv);
7. while lmid <= rmid && conc==0 &&lmid >= sbeg
   {
8. Find α (lmid) and α (rmid) as alphalmidv and alpharmidv respectively
9. For each $i \in (1, 2 \ldots n)$
10. Find the corresponding ct_i values (lctand rct) based on alphalmidv and alpharmidv respectively where ct_i: = $(1 - \alpha) \cdot a_i + \alpha \cdot ann_i$
11. Find lmcdisc and rmcdisc values based on lct and rct values respectively where cdisc: = discScore (ct, p)
12. If lmcdisc == rmcdisc
    t = lct; disc = lmcdisc; conc=1; break;
    elseif lmcdisc < rmcdisc
    t = rct; disc =rmcdisc; lmid = round (lmid + divv);
    elseif lmcdisc >= rmcdisc
    t = lct; disc = lmcdisc; rmid = lmid; mid = round (lmid – divv);
   }
13. For each $i \in (1, 2 \ldots n)$
14. s_i: = f (t_i), where f: $R \rightarrow R$ is a monotone function

IV. EXPERIMENTAL EVALUATION

An adaptive techniqueis applied in Antihub^{2}algorithm for the evaluation of computation complexity and statistical measures of accuracy is used for performance evaluation. In this section, the effectiveness and behavior of the proposed approach is examined in terms of number of computations, computation time and accuracy by applying it on three different real data sets obtained (wilt, aloi, and churn) against those algorithm. Wilt data set consists of image segments, generated by segmenting the pansharpened image with totally 4339 image segments. It involves 6 attributes. ALOI (Amsterdam Library of Object Images) dataset is a color image collection of 1, 00,000 small objects with 64 attributes. churn is a dataset with 1667 objects and 21 attributes. This section describes those experiments and their results.

Accuracy is the proportion of true results, either true positive or true negative, in a population. It measures the degree of veracity of a test on a condition. The terms that are used along with the description of accuracy are true positive (TP), true negative (TN), false negative (FN), and false positive (FP). Accuracy can be described as

Accuracy = (TN + TP)/ (TN+TP+FN+FP) = (No. of correct assessments)/No of all assessments)

TABLE I

NUMBER OF COMPUTATIONS ENCOUNTEREDFORANTIHUB^2 AND ADAPTIVEANTIHUB WHEN K=120

FOR ALOI, WILT AND CHURN DATA SETS

	Antihub^2	**AdaptiveAntihub**
ALOI	48279	4598
WILT	91119	8678
CHURN	35007	3334
AVERAGE	58135	5537

Table I shows the number of computations occurred for both the algorithms for all the three datasets when k=120. From this table, it is well understood that AdaptiveAntihub has a significant reduction (in an

average of 90.47% of reduction) in number of computations, occurs for all the three datasets when it is compared with the existing Antihub[2]. It proves that AdaptiveAntihub outperforms well than the other.

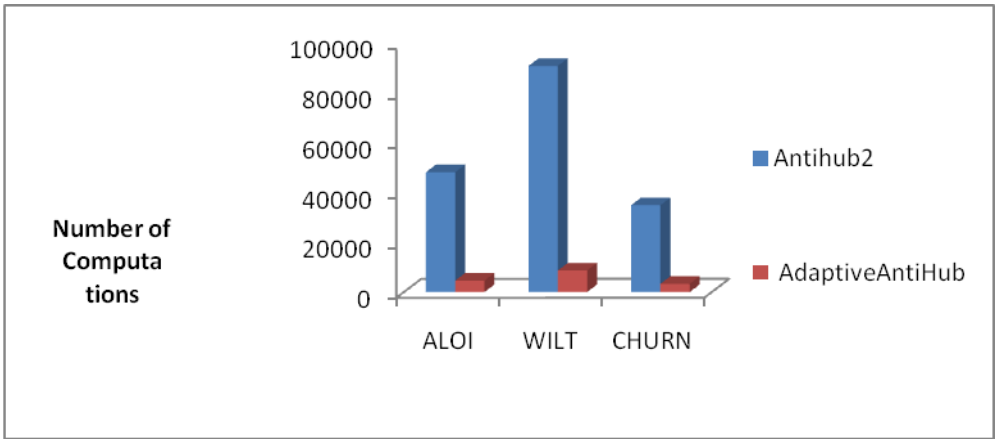

Fig 2 The number of computations encountered for Antihub[2] and AdaptiveAntihub for ALOI, WILT and CHURN data sets

Fig 2 demonstrates that, when we compare the number of computations for the Antihub[2] and AdaptiveAntihub for all the three datasets, AdaptiveAntihub has a significant reduction in number of computations and also proves that it outperforms well with the data set dimensionality than Antihub[2].

TABLE II

THE COMPUTATION TIME OF ANTIHUB2 AND ADAPTIVEANTIHUB WHEN K=120

FOR ALOI, WILT AND CHURN DATA SETS

	Antihub[2] (secs)	**AdaptiveAntihub (secs)**
ALOI	3.6426	3.1532
WILT	1.8552	1.376
CHURN	9.3879	7.1802
AVERAGE	4.9619	3.9031

Table II shows the computation time taken by Antihub[2] and AdaptiveAntihub for all the three datasets when k=120. From this table, it is well understood that the proposed AdaptiveAntihub has 21.33% of reduction in an average in computation time for all the three datasets when it is compared with the existing Antihub[2]. It proves that AdaptiveAntihub outperforms well than the other.

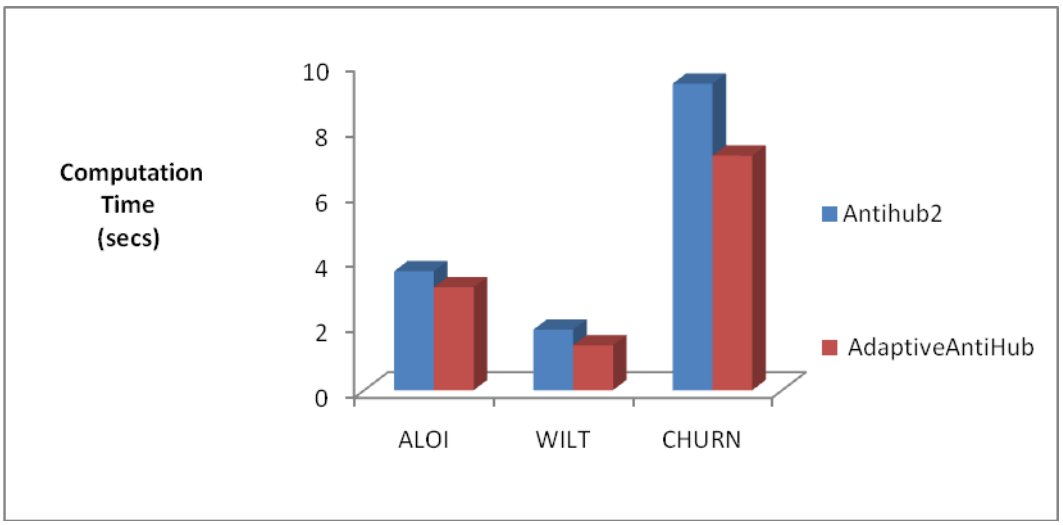

Fig 3 The computation time of Antihub[2]and AdaptiveAntihub for ALOI, WILT and CHURN data sets

Fig 3 illustrates that, when we compare the computation time of Antihub^2 with AdaptiveAntihub for all the three datasets, AdaptiveAntihub has a significant reduction in computation time and also proves that it outperforms well with the data set dimensionality than Antihub^2.

TABLE III

THE PERFORMANCE ACCURACY OF ANTIHUB^2 AND ADAPTIVEANTIHUB FOR ALOI, WILT AND CHURN DATA SETS WHEN K=10, 50, 100 AND 120

	k Value	**Antihub^2**	**AdaptiveAntihub**
ALOI	10	0.7703	0.7612
	50	0.7577	0.7603
	100	0.7586	0.759
	120	0.7595	0.7595
	Average	0.761525	0.76
WILT	10	0.9813	0.9825
	50	0.9829	0.9829
	100	0.9827	0.9827
	120	0.9829	0.9829
	Average	0.98245	0.98275
CHURN	10	0.9904	0.9994
	50	0.9988	0.9868
	100	0.9418	0.9418
	120	0.9328	0.9304
	Average	0.96595	0.9646

Table III shows the Performance Accuracy of Antihub^2 and AdaptiveAntihub, for ALOI, WILT and CHURN data sets when k=10, 50, 100 and 120. It also illustrates that accuracy in an average are at the same level for both the algorithms for all the three data sets even when the k value changes.

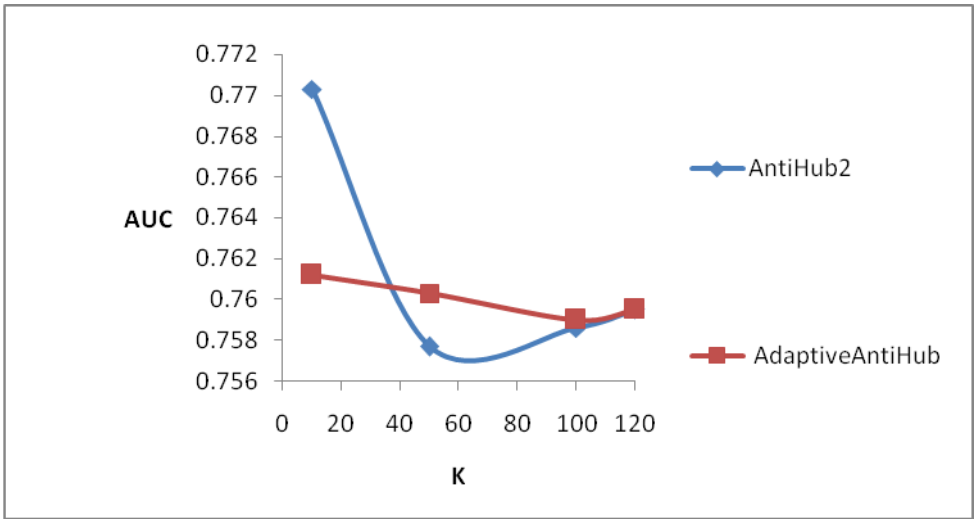

Fig 4 The performance accuracy of Antihub^2 and AdaptiveAntihub for ALOI dataset

Fig 4 shows that the performance accuracy of Antihub^2 and AdaptiveAntihub when using the dataset ALOI are almost at the same level when k=10, 50, 100, 120. Therefore while comparing the number of computations and computation timebetween the two algorithms in Table I as well as Table II, it is well understood that AdaptiveAntihub has obtained 90.47% of reduction in number of computations and 13.43% of reduction in computation time than Antihub2 with almost, the same level of accuracy for both the algorithms in ALOI dataset.

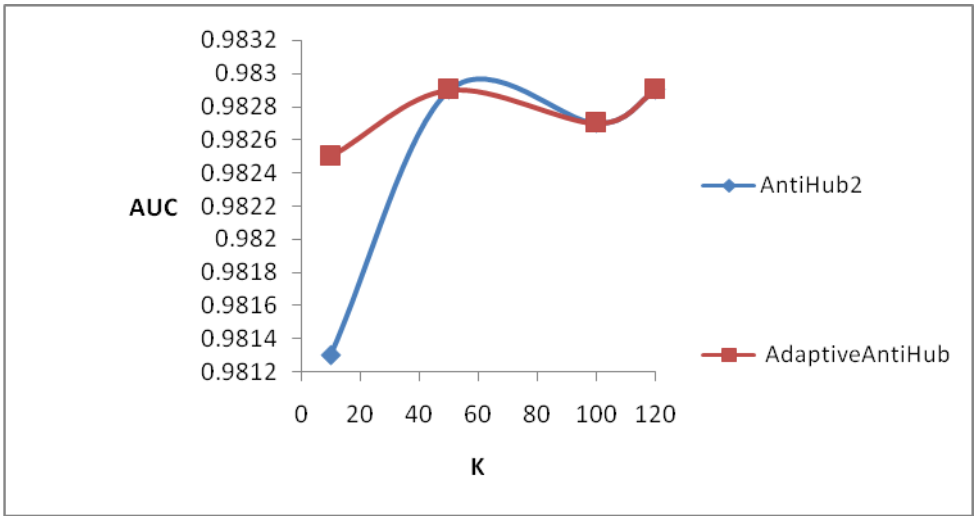

Fig 5 The performance accuracy of Antihub^2 and AdaptiveAntihub for WILT dataset

Fig 5 shows that the performance accuracy of Antihub^2 and AdaptiveAntihub when using the dataset WILT are almost at the same level when k=10, 50, 100, 120. Therefore while comparing the number of computations and computation time between the two algorithms in Table I as well as Table II, it is well understood that AdaptiveAntihub has obtained 90.47% of reduction in number of computations and 25.83% of reduction in computation time than Antihub2 with almost, the same level of accuracy for both the algorithms in WILT dataset.

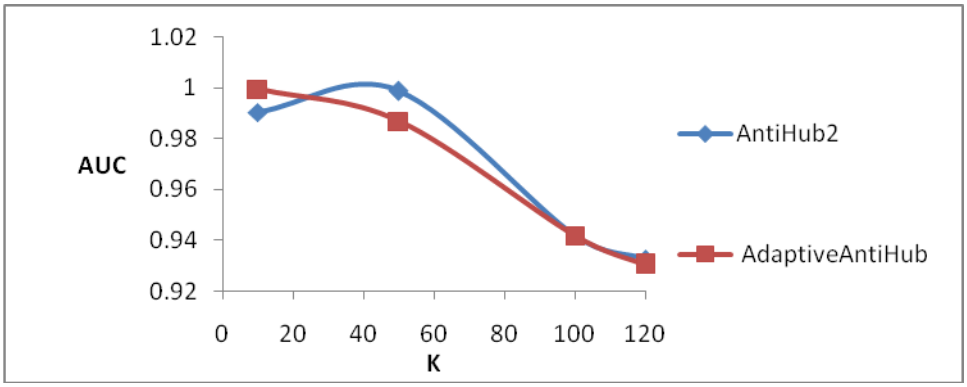

Fig 6 The performance accuracy of Antihub^2 and AdaptiveAntihub for CHURN dataset

Fig 6 shows that the performance accuracy of Antihub^2 and AdaptiveAntihub when using the dataset CHURN are almost at the same level when k=10, 50, 100, 120. Therefore while comparing the number of computations and computation time between the two algorithms in Table I as well as Table II, it is well understood that AdaptiveAntihub has obtained 90.47% of reduction in number of computations and 23.51% of reduction in computation time than Antihub2 with almost, the same level of accuracy for both the algorithms in CHURN dataset.

V. CONCLUSION

This paper, presents an approach AdaptiveAntihub where anadaptive technique is applied to hubness, especially Antihub^2 for unsupervised outlier detection to reduce the number of computations and computation time. The performance of Antihub^2 is empirically compared with AdaptiveAntihub in terms of number of computations, computation time and accuracy by applying it into three different data sets and found from the experimental results that AdaptiveAntihub provides a significant reduction in number of computations and computational time than Antihub^2 with same level of accuracy for both the algorithms for all the three datasets. From this analysis, it is well understood that when an AdaptiveAntihub technique is applied, itoutperforms well with the data set dimensionality than Antihub^2 algorithm on identifying meaningful and interesting outliers.

REFERENCES

[1] C. C. Aggarwal and P. S. Yu, "Outlier detection for high dimensional data" *ACM SIGMOD RECORD* February 2002.

[2] Amer, Mennatallah, and Slim Abdennadher. "Comparison of unsupervised anomaly detection techniques." *Bachelor's Thesis* (2011).

[3] A. Zimek, E. Schubert, and H.-P. Kriegel, "A survey on unsupervised outlier detection in high-dimensional numerical data," *Statistical Analysis and Data Mining, vol. 5*, no. 5, pp. 363–387, 2012.

[4] Miloˇs Radovanovi´c, Alexandros Nanopoulos, and Mirjana Ivanovi´ "Reverse Nearest Neighbors in Unsupervised Distance-Based Outlier Detection*" IEEE Transactions On Knowledge And Data Engineering,* October 2014.

[5] M. Radovanovi´c, A. Nanopoulos, and M. Ivanovi´c, "Hubs in space: Popular nearest neighbors in high-dimensional data," *J Mach Learn Res, vol. 11*, pp. 2487–2531, 2010.

[6] Ramaswamy, Sridhar, Rajeev Rastogi, and Kyuseok Shim. "Efficient algorithms for mining outliers from large data sets." *ACM SIGMOD Record. Vol. 29.* No. 2. ACM, 2000

[7] Knorr, Edwin M., Raymond T. Ng, and Vladimir Tucakov. "Distance-based outliers: algorithms and applications." *The VLDB Journal—the International Journal on Very Large Data Bases 8.3-4* (2000): 237-253.

[8] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," *SIGMOD Rec, vol. 29*, no. 2, pp. 93–104, 2000.

[9] Zhang, Ke, Marcus Hutter, and Huidong Jin. "A new local distance-based outlier detection approach for scattered real-world data." *Advances in Knowledge Discovery and Data Mining. Springer Berlin Heidelberg*, 2009. 813-822.

[10] Kriegel, Hans-Peter, et al. "LoOP: local outlier probabilities." *Proceedings of the 18th ACM conference on Information and knowledge management. ACM*, 2009

[11] Amer, Mennatallah, and Slim Abdennadher. "Comparison of unsupervised anomaly detection techniques." *Bachelor's Thesis* (2011).

[12] Amer, Mennatallah, and Markus Goldstein. "Nearest-neighbor and clustering based anomaly detection algorithms for rapidminer." *Proc. of the 3rd RapidMiner Community Meeting and Conference (RCOMM 2012).* 2012.

[13] Singh, Amit, Hakan Ferhatosmanoglu, and Ali Şaman Tosun. "High dimensional reverse nearest neighbor queries." *Proceedings of the twelfth international conference on Information and knowledge management. ACM*, 2003.

[14] J. Lin, D. Etter, and D. DeBarr, 2008 "Exact and approximate reverse nearest neighbor search for multimedia data," in *Proc 8th SIAM Int Conf on Data Mining (SDM)*, pp. 656–667.

[15] Radovanović, Miloš, Alexandros Nanopoulos, and Mirjana Ivanović. "Nearest neighbors in high-dimensional data: The emergence and influence of hubs." *Proceedings of the 26th Annual International Conference on Machine Learning. ACM*, 2009.

[16] N.Tomaˇsev, R. Brehar, D. Mladeni´c, and S. Nedevschi, "The influence of hubness on nearest-neighbor methods in object recognition," in *Proc. 7th IEEE Int. Conf. on Intelligent Computer Communication and Processing (ICCP)*, 2011, pp. 367–374.