AFRL-AFOSR-JP-TR-2024-0035

Robust data mining to learn meaningful patterns from numeric data which can be misleading

Aryal, Sunil
DEAKIN UNIVERSITY
1 GHERINGHAP STREET
GEELONG, VIC, 3220
AUS

01/01/2024
Final Technical Report

Air Force Research Laboratory
Air Force Office of Scientific Research
Asian Office of Aerospace Research and Development
Unit 45002, APO AP 96338-5002

REPORT DOCUMENTATION PAGE

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ORGANIZATION.

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED	
20240101	Final	START DATE 20200924	END DATE 20230923

4. TITLE AND SUBTITLE
Robust data mining to learn meaningful patterns from numeric data which can be misleading

5a. CONTRACT NUMBER	5b. GRANT NUMBER	5c. PROGRAM ELEMENT NUMBER
	FA2386-20-1-4005	61102F
5d. PROJECT NUMBER	**5e. TASK NUMBER**	**5f. WORK UNIT NUMBER**

6. AUTHOR(S)
Sunil Aryal

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
DEAKIN UNIVERSITY
1 GHERINGHAP STREET
GEELONG, VIC 3220
AUS

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)	11. SPONSOR/MONITOR'S REPORT NUMBER(S)
AOARD UNIT 45002 APO AP 96338-5002	AFRL/AFOSR IOA	AFRL-AFOSR-JP-TR-2024-0035

12. DISTRIBUTION/AVAILABILITY STATEMENT
A Distribution Unlimited: PB Public Release

13. SUPPLEMENTARY NOTES

14. ABSTRACT
This report summarizes the work from a very successful grant - the 6th we have awarded to this PI. The PI published 7 papers from this grant alone which is tremendous output and they have greatly advanced the understanding of coupling mechanisms between the solar wind and magnetosphere/ionosphere. We will not be supporting another grant with Dr. Horvarth but only because we have well exceeded our usual limit of 1-2 follow-on grants. However, at our recommendation, the Navy will be supporting her work going forward.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES
a. REPORT	b. ABSTRACT	c. THIS PAGE		
U	U	U	SAR	28

19a. NAME OF RESPONSIBLE PERSON	19b. PHONE NUMBER *(Include area code)*
GEOFFREY ANDERSEN	

Standard Form 298 (Rev.5/2020)
Prescribed by ANSI Std. Z39.18

Award Number

FA2386-20-1-4005

Report Type

Final

Reporting Periods

24 Sept 2020 – 23 Dec 2023

Distribution Statement

Distribution A – Approved for Public Release

Program Officer Name

Dr Geoffrey Andersen

Principal Investigator Name

Dr Sunil Aryal

Project Title

Developing robust framework for practical data mining

Abstract

In real-world applications, data can be measured and expressed/represented using different units or scales. For example, weight in kilograms or pounds and fuel-efficiency in kilometers per liter or liters per 100 kilometers. When data are given for pattern mining, information on how data are recorded is often not available, and only data values (numbers) are provided. Since most existing data mining algorithms are sensitive to data representation, they may yield different insights when the same data are represented differently. This project aims to develop data mining algorithms that are robust and not sensitive to data representations, providing consistent insights regardless of how are presented. We approach this project aim on two levels. First, at the algorithm level, by developing new algorithms that are robust to data representation. Second, at the data preprocessing level, by developing a robust data preprocessing technique so that existing algorithms can be used with the preprocessed data without modification.

The main contribution of the project is the development of a scale-invariant and data-dependent kernel similarity function, which enables the kernel-based methods to be robust to data representation. We have demonstrated the effectiveness of this proposed kernel function in clustering and Support Vector Machine (SVM) classification tasks. Additionally, building upon our preliminary studies preceding this project, we have developed: (i) a robust anomaly detection algorithm based on unsupervised stochastic random forest named 'usfAD' and demonstrated its effectiveness in cybersecurity problems; and (ii) a robust preprocessing technique called 'ARES' and demonstrated its effectiveness using three popular state-of-the-art clustering algorithms. The project has directly resulted in four publications: two journal articles and one conference paper have already been published, with another conference paper ready for submission.

1. Introduction and Background

In real-world applications, features of data objects can be represented in different units or scales [1, 2, 3, 4]. For example, weight in kilograms or pounds, temperature in °C or °F, sample variability in terms of standard deviation or variance, probability in terms of likelihood or log likelihood, fuel efficiency in kilometers per liter or liters per 100 kilometers. These different representations of data can involve linear or non-linear scaling of one another. Data representation can vary for various reasons including settings of devices or sensors used for measurement, domain or user requirements, and data compression to store or transmit using fewer bits. Such variation is common in application domains such as cyber-physical systems, networks and communications, the Internet-of-Things (IoT), and healthcare, where different features of data objects come from different sources and/or recorded by different sensors.

The variation in data representation due to non-linear scaling presents a challenge for automatic pattern mining using existing data mining algorithms, as it alters the distribution of data. Most existing data mining algorithms rely on pairwise distance between data objects or their density, making them sensitive to how data features are represented. Algorithms that perform well in one representation may struggle when the same data are represented differently. For instance, consider the red data point in Figure 1(a), which is evidently an anomaly. However, when the same data are represented using inverse scaling as $x' = x^{-1}$, as shown in Figure 1(b), the corresponding point appears to be normal data. Most existing anomaly detection algorithms fail to detect the anomaly in Figure 1(b).

When data are given for pattern mining, information on how data are recorded (units/scales) is often not available and only data values (numbers) are provided. Even if this information is known, we may not know whether the given form is appropriate for the task at hand. Most existing algorithms can provide misleading insights or inaccurate predictions if data are not given in the appropriate form, which can have severe consequences in critical applications like defence, healthcare, cybersecurity, and banking.

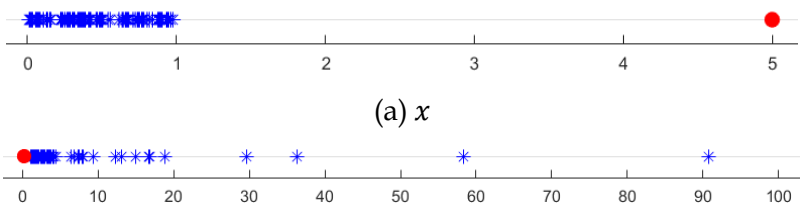


Figure 1. An example of the same data represented in two scales. The red data point in Case (a) looks like an anomaly, whereas the corresponding point in Case (b) is more like a normal data.

Data need to be transformed into an appropriate form before using existing algorithms. The most straightforward approach to identify the appropriate representation is to experiment with various transformations and assess which one yields the optimal task specific result. Because there are infinite number of possible transformations to try for each feature, it is not feasible to find the best combination from trial and error even in low-dimensional problems with tens of features, let alone high-dimensional ones with hundreds and thousands of features. Also, it requires expert judgement to identify appropriate form of data for unsupervised tasks such as clustering and anomaly detection.

Many psychologists have argued that raw numbers can be misleading as they may have been measured or presented in various ways. One should not conclude anything from a given set of numbers without understanding where they come from and the underlying process of generating them [5, 6, 7, 8]. Velleman & Wilkinson (1993) suggested that good data analysis does not make any assumption about data types/scales because data may not be what we see. Joiner (1981) provided some examples of 'lurking variables'–data appear to have one pattern, but in fact hide other information. However, in data mining, numeric data are assumed to be in 'interval scale' [2]–the meaning of a unit difference has the same meaning everywhere in the space. This assumption may not be true when data are represented as non-linear scaling such as logarithms and inverses.

Therefore, this project aims to develop data mining algorithms that are robust and insensitive to data representations, ensuring consistent insights regardless of how data are presented, particularly for clustering and anomaly detection. We plan to work in the project at two levels:

A. ***Algorithmic level***: developing robust algorithms capable of handling input data regardless of how they are measured or recorded (*Contributions 2.1 and 2.2*).

B. ***Data pre-processing level***: developing a robust data pre-processing technique ensuring that data expressed in different forms yield similar pre-processed data. This approach enables the use of existing algorithms without modification (*Contribution 2.3*).

2. Project Contributions

The main contribution of the project is in the following three folds.

2.1 PMK: A data-dependent and scale-invariant kernel similarity function

First, extending our previous work on general-purpose simple data-dependent similarity measure called $m_p - dissimilarity$ (with $p > 0$) [3], we develop a scale-invariant and data-dependent kernel similarity function called "*Probability Mass-based Kernel (PMK)*" and demonstrate its effectiveness in Clustering [9] and Support Vector Machine (SVM) classification [10] tasks. PMK estimates the similarity of two data objects based on the probability data mass between them in each feature/dimension. It has a probabilistic interpretation based on the naïve Bayesian assumption of feature independence. It differs from existing

widely used distance-based similarity measures (e.g., $\ell_p - norm$ with $p > 0$) and kernel functions (e.g., RBF and Laplacian) with the following two distinguishing characteristics:

a. *Data-dependent:* Unlike distance-based measures that are data distribution independent (i.e., the similarity of two objects is solely based on their spatial positions and it is not influenced by the distribution of other data), PMK is data distribution dependent—the similarity of two objects is influenced by the distribution of data around and between them.

b. *Scale-invariant:* Unlike distance-based measures that are sensitive to change in data representation due to non-scale scaling, PMK is invariant because the ordering is either preserved or reversed even in the case of non-linear scaling.

Because of the lack of the above-mentioned properties, the task specific performances of distance-based measures tend to vary across datasets and tasks. A similarity measure must be selected carefully for a given dataset and task. Also, the effectiveness of distance-based measures decreases as the dimensionality of data increases. Because PMK is both data-dependent and scale-invariant, it yields more consistent results than distance-based measures across a wide range of datasets and tasks. It also outperforms distance-based measures in high-dimensional problems. As the similarity between data objects adapts to the local density distribution, PMK overcomes the known limitation of distance/density-based clustering algorithms in detecting clusters of varying densities. For more technical details and experimental results in clustering and SVM classification, please refer to the attached full papers [9, 10] (**Attachment 1 and 2**).

2.2 usfAD: A scale-invariant anomaly detection method

Second, we extended our preliminary work on a robust anomaly detection method based on unsupervised stochastic forest, named 'usfAD'. We improved the method in terms of implementation, evaluated its performances in real-world datasets from the cybersecurity domain, and conducted sensitivity analysis of its parameters. It is inspired by the widely used very efficient anomaly detector called Isolation Forest (iForest) [11]. It is an attempt to make iForest invariant to data representation. It partitions the space using tree structures to define normal and anomalous regions in the training phase, using only data belonging to the normal class. During the testing phase, test data, comprising both normal and anomalous data, are ranked according to their anomaly scores based on their pathlengths in the trees. It is robust to the units/scales used to represent/measure input data and it can detect anomalies that might have been obscured due to inappropriate data representation. For more technical details and experimental results, please refer to the attached full paper [12] (**Attachment 3**).

2.3 ARES: A scale-invariant data preprocessing technique

Third, we continued working on robust data preprocessing method that is not sensitive to data representation. We extended our preliminary work on a variant of rank transformation called 'Average Rank over an Ensemble of Sub-samples (ARES)'. We demonstrated that it makes clustering robust to data representation and enables them to detect clusters with varying densities effectively. The idea is to use rank transformation [13] in multiple subsamples of data and aggregating the ranks. Unlike using rank transformation in the entire dataset, which may mask existing clusters, aggregating ranks from multiple subsamples helps preserve clusters to some extent and remains robust to data representation. Our empirical results, obtained from the three most widely used clustering algorithms—KMeans, DBSCAN and DP (Density Peak)—applied to real-world datasets, demonstrate that clustering after ARES transformation produces better and more consistent results. For more technical details and experimental results, please refer to the attached full paper [14] (**Attachment 4**).

3. Project Outcomes

In addition to the four publications (on bold below) discussed above resulted directly from the project contributions, the grant has also helped to work/contribute to other publications, grants, and

collaborations.

3.1 Journal Publications

- **Rasool, Z.*, Aryal, S.*, Bouadjenek, M. R. and Dazeley, R. (2023). "Overcoming Weaknesses of Density Peak Clustering Using a Data-dependent Similarity Measure",** ***Pattern Recognition, 137*** **(*Contributed equally)**
- **Aryal, S., Santosh, K. C., & Dazeley, R. (2021). usfAD: a robust anomaly detector based on unsupervised stochastic forest.** ***International Journal of Machine Learning and Cybernetics*****, 12(4), 1137-1150.**
- Baniya, A. A., Lee, T. K., Eklund, P. W., Aryal, S. & Robles-Kelly, A. (2023). "Online Video Super-resolution using Information Replenishing Unidirectional Recurrent Model", *Neurocomputing, 546*
- Ahmed, A. M., Abdekrazek, M., Aryal, S. & Nguyen, T. T. (2023). "An overview of Eulerian video motion magnification methods", *Computers and Graphics*, 117
- Baniya, A. A., Lee, T. K., Eklund, P. W. & Aryal, S. (2023). "Omnidirectional video super-resolution using deep learning", *IEEE Transactions on Multimedia*
- Aryal, S., Ting, K. M., Washio, T., & Haffari, G. (2020). A comparative study of data-dependent approaches without learning in measuring similarities of data objects. *Data mining and knowledge discovery*, *34*, 124-162.

3.2 Conference Publications

- **Malgi, V. V.*, Aryal, S.*, Rasool, Z. and Tay, D. (2023). Data-dependent and scale-invariant kernel for Support Vector Machine classification.** ***In PAKDD 2023: Proceedings of the 27th Pacific-Asia Conference on Knowledge Discovery and Data Mining,*** **pp. 171-182 (*Contributed equally)**
- **Aryal, S., Wells, J. R., Baniya, A. A. and Santosh, K. C. (2024), Enabling clustering algorithms to detect clusters of varying densities through scale-invariant data pre-processing,** ***arXiv*****. 2401.11402 (*finalising for submission*)**
- Aryal, S. & Wells, J. R. (2021). "Ensemble of Local Decision Trees for Anomaly Detection in Mixed Data". In ECML/PKDD 2021*: Proceedings of the 2021 European Conference on Machine Learning and Practice of Knowledge Discovery in Databases,* pp. 687-702.
- Samariya, D., Aryal, S., Ting, K. M., & Ma, J. (2020). A New Effective and Efficient Measure for Outlying Aspect Mining. In WISE 2020: Proceedings of the 2020 International Conference on Web Information Systems Engineering, Vol. 12343 (pp. 463-474).
- Aryal, S., Baniya, A. A., Razzak, I., & Santosh, K. C. (2021). SPAD+: An Improved Probabilistic Anomaly Detector based on One-dimensional Histograms. In IJCNN 2021: Proceedings of the 2021 International Joint Conference on Neural Networks (pp. 1-7)
- Samariya, D., Ma, J., & Aryal, S. (2022). "sGrid++: Revising Simple Grid Based Density Estimator for Mining Outlying Aspect". In WISE 2022: *Proceedings of the 2022 International Conference on Web Information Systems Engineering*, pp. 194-208.
- Ly, A., Dazeley, R., Vamplew, P., Naranjo, F. C. and Aryal, S. (2023). Elastic Step DDPG: A Novel Multi-Step Algorithm to Improve Sample Efficiency in Deep Deterministic Policy Gradient. *In IJCNN 2023: Proceedings of the International Joint Conference on Neural Networks*
- Huynh, N. D., Bouadjenek, M. R., Aryal, S., Razzak, I. & Hacid, H. (2024). "SimpsonsVQA: Enhancing Inquiry-Based Learning with a Tailored Dataset". In CVPR 2024: *Proceedings of the 2024 IEEE / CVF* Computer Vision and Pattern Recognition *Conference* (**under review**)

3.3 Research Visits and Collaboration

I visited USSF Space Operations Center Delta 2 at Dahlgren in April 2022 under the WoS Scientist visit

program for research collaboration aimed at utilizing our Anomaly Detection (AD) methods, including usfAD developed as part of this project, in their satellite object tracking problem. Since then, we have been actively working with their analysts and engineers to deploy and test AD methods in their development environment. Currently, we are focusing on adapting AD algorithms to detect abnormal patterns in satellite data based on orbital regimes and sensor type. Our collaboration has been features as a research translation success story as it progressed from TRL1 to TRL6 in a relative short period of time.

When I was in the US on the WoS visit, I visited Prof Danda B. Rawat at Howard University to establish research collaboration in AI for Cybersecurity and IoT. I visited Prof Takashi Washio at Osaka University when I was in Osaka to present our PMK paper in the Asia-Pacific Conference on Knowledge Discovery and Data Mining (PAKDD) 2023. This visit was supported by the Deakin University School of IT. I also attended the first Australia-India Critical Minerals Research Hub workshop at the Indian Institute of Technology (IIT) Bombay, India, where I presented our research and its potential applications in critical minerals research. This trip was funded by Monash and Deakin Universities.

During the project duration, we worked with agencies like Defence Science and Technology Group (DSTG) Australia, Table Tennis Australia, and Alcohol and Drug Foundation Australia, and collaborated with researchers in Australia, USA, UK, China, Denmark, India, and Nepal for joint publications, grants, and PhD supervision.

3.4 Talks

- Presented our research related to this project to multiple research groups at AFOSR and ONR [2020-2022, Online]
- Invited talks on my current research at the National Workshop on Machine Learning and Data Science (NWMLDS), Nepal [2020 and 2021, Online]
- Presented our research on usfAD at the Ubiquitous Robots (UR) Workshop on Robotic Swarms and Emergent Behavior [2021, Online]
- Presented our paper "*Ensemble of Local Decision Trees for Anomaly Detection in Mixed Data*" at the 2021 European Conference on Machine Leaning and Practice of Knowledge Discovery in Databases (ECML/PKDD) [2021, Online]
- Presented our current research on "*Practical Machine Learning*" at the Federation University [2021, In-person]
- Gave a talk at AFOSR Office in Arlington on "*Robust and Flexible Machine Learning for real-world problems*" while on the WoS visit [2022, In-person].
- Invited visit to the Department of Computer Science, Kathmandu University, Nepal and presented our current research [2023, In-person]
- Invited talk on "*AI in Critical Minerals Research*" at the Australia-India Critical Minerals Research Hub Workshop, Indian Institute of Technology Bombay, India [2023, In-person]
- Presented our paper titled "*Data-dependent and scale-invariant kernel for Support Vector Ma chine Classification*" at the 2023 Asia-Pacific Conference on Knowledge Discovery and Data Mining (PAKDD) [2023, In-person]
- Invited presentation on "*Our current research and collaboration with USSF Delta 2 Detachment*" to US Space Force and Australian Defence Science and Technology Group delegates at Defence Science Institute Melbourne [2023, In-person]

3.5 Other grant funding (awarded, approved, and submitted)

- Aryal, S. (2020) "Ensemble Learning for Outlier Detection", A$20K, *Office of National Intelligence (ONI)* Australia under the *AI for Decision Making Initiative (AI4DMI) Round 1* [Approved and Completed]
- Aryal, S. (2022), Window-on-Science (WoS) Travel Award, US$6,800, *AFOSR* to visit USSF Space Operations Center Delta 2 Dahlgren Virginia for research collaboration (visited in April 2022)

[Approved and Completed]

- Aryal. S. & Bouadjenek, M. R. (2022-2025), “Machine Learning in Heterogeneous Data from Multiple Sources”, US$225,000, *AFOSR* under the Director’s Research Initiative program 2022 [Approved and Ongoing]
- Aryal, S., Moghaddam, M. K, Dazeley, R. & Nakisa, B. (2023-2026), “Intelligent Table Tennis Match Analysis: Automated Insights Extraction through Machine Learning and Computer Vision”, A$62.5K, *Table Tennis Australia* [Approved and Ongoing]

3.6 Awards and Achievements

- Deakin School of IT Research Award for Excellence in Early Career Researcher performance, 2021
- Deakin School of IT Teaching Award for Excellence in Unit leadership and supporting student learning, 2021
- Promoted to Senior Lecturer at Deakin University from January 2022
- Deakin Faculty of Science, Engineering and Built Environment Teaching Award for Teaching Excellence, 2022
- Deakin University School of IT Research Award for Industry Engagement, 2022
- The Machine Learning-based solution that our team (collaborators and students from Monash and Deakin Universities) developed was the **third-place winner** out of 305 submissions in the outcome category of the George B. Moody PhysioNet Data Challenge 2022 to detect heart mumur
- Deakin School of IT Award for Outstanding Leadership, 2023
- Deakin School of IT Award for Excellence in Industry Collaboration and Partnerships, 2023
- Nominated by the school & faculty for VC Awards for Industry Research Engagement in 2021 & 2022, and International Collaboration in 2023
- Nominated by the Deakin School of IT leadership for the Rising Star VC Award, 2023
- Nominated by the Deakin School of IT leadership for a Victorian Tall Poppy Award 2023
- Appointed as the Deakin University School of IT Industry Practice Lead (Machine Learning) to lead school’s industry research initiatives and supporting colleagues, particularly ECRs, for industry engagement.
- Since 2021, our team is organizing Annual Deakin Simpson’s AI Challenge to encourage students to get involved in AI and Machine Learning. We have been able to secure funding for prizes every year from the Bendigo Community Bank. All three editions were well-attended by students from the schools of IT, Engineering and Business
- Late 2022, we founded the Machine Learning of Decision Support Research Group, which now has five academic staff, and over 10 research students working in different areas of AI and Machine Learning.

References

[1] S. S. Stevens, "On the theory of scales of measurement," *Science,* vol. 103, no. 2684, pp. 677-680,, 1946.

[2] T. L. Fernando and G. I. Webb, "SimUSF: An efficient and effective similarity measure that is invariant to violations of the interval scale assumption," *Data Mining and Knowledge Discovery,* vol. 31, no. 1, pp. 264-286, 2017.

[3] S. Aryal, K. M. Ting, T. Washio and G. Haffari, "Data-dependent dissimilarity measure: an effective alternative to geometric distance measures," *Knowledge and Information Systems,* vol. 53, no. 2, pp. 479-506, 2017.

[4] S. Aryal, K. M. Ting, T. Washio and G. Haffari, "A comparative study of data-dependent approaches without learning in measuring similarities of data objects," *Data Mining and Knowledge Discovery,* vol. 34, no. 1, pp. 124-162, 2020.

[5] F. M. Lord, "On the statistical treatment of football numbers," *American Psychologist,* vol. 8, no. 12, pp. 750-

751, 1953.

[6] J. T. Townsend and F. G. Ashby, "Measurement scales and statistics: The misconception misconceived," *Psychological Bulletin,* vol. 96, no. 2, pp. 394-401,, 1984.

[7] P. F. Velleman and L. Wilkinson, "Nominal, ordinal, interval, and ratio typologies are misleading," *American Statistician,* vol. 47, no. 1, pp. 65-72,, 1993.

[8] B. L. Joiner, "Lurking variables: Some examples," *The American Statistician,* vol. 35, no. 4, pp. 227-233, 1981.

[9] Z. Rasool, S. Aryal, M. R. Bouadjenek and R. Dazeley, "Overcoming Weaknesses of Density Peak Clustering Using a Data-dependent Similarity Measure," *Pattern Recognition,* vol. 137, p. 109287, 2023.

[10] V. V. Malgi, S. Aryal, Z. Rasool and D. Tay, "Data-dependent and Scale-Invariant Kernel for Support Vector Machine Classification," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Osaka, 2023.

[11] F. Liu, K. M. Ting and Z.-H. Zhou., "Isolation forest," in *Proceedings of the Eighth IEEE International Conference on Data Mining*, 2008.

[12] S. Aryal, K. C. Santosh and R. Dazeley, "usfAD: A robust Anomaly Detector based on Unsupervised Stochastic Forest," *International Journal of Machine Learning and cybernetics,* vol. 12, no. 4, pp. 1137-1150, 2021.

[13] W. J. Conover and R. L. Iman., "Rank transformations as a bridge between parametric and nonparametric statistics," *The American Statistician,* vol. 35, no. 3, pp. 124-129, 1981.

[14] S. Aryal, J. R. Wells, A. A. Baniya and K. Santosh, *Enabling clustering algorithms to detect clusters of varying densities through scale-invariant data preprocessing,* arXiv. 2401.11402, 2024.

Attachments:

Attachment 1: Pattern Recognition paper on MP-DPC [9]
Attachment 2: PAKDD 2023 paper on PMK [10]
Attachment 3: JMLC paper on usfAD [12]
Attachment 4: ArXiv paper on ARES-Clustering [14]

Overcoming Weaknesses of Density Peak Clustering Using a Data-dependent Similarity Measure

Zafaryab Rasool[1,*], Sunil Aryal[1], Mohamed Reda Bouadjenek, Richard Dazeley

School of Information Technology, Deakin University, Waurn Ponds Campus, Geelong, VIC 3216, Australia

Abstract

Density Peak Clustering (DPC) is a popular state-of-the-art clustering algorithm, which requires pairwise (dis)similarity of data objects to detect arbitrary shaped clusters. While it is shown to perform well for many applications, DPC remains: (i) not robust for datasets with clusters having different densities, and (ii) sensitive to the change in the units/scales used to represent data. These drawbacks are mainly due to the use of the data-independent similarity measure based on the Euclidean distance. In this paper, we address these issues by proposing an effective data-dependent similarity measure based on *Probability Mass*, which we call *MP-Similarity*, and by incorporating it in DPC to create MP-DPC, a data-dependent variant of DPC. We evaluate and compare MP-DPC against diverse baselines using several clustering metrics and datasets. Our experiments demonstrate that: (a) MP-DPC produces better clustering results than DPC using the Euclidean distance and existing data-dependent similarity measures; (b) MP-Similarity coupled with Shared-Nearest-Neighbor-based density metric in DPC further enhances the quality of clustering results; and (c) unlike DPC with existing data-independent and data-dependent similarity measures, MP-DPC is robust to the change in the units/scales used to represent data. Our findings suggest that MP-Similarity provides a more viable solution for DPC in datasets with unknown distribution or units/scales of features, which is often the case in many real-world applications.

Keywords: Clustering, Density Peak Clustering, Similarity Measure, Data-dependent Similarity

1. Introduction

Clustering is a fundamental task in data mining used to find groups (or clusters) of similar objects in a dataset based on a notion of similarity. Clustering has applications in areas such as pattern recognition [1], bioinformatics [2], image segmentation [3], information retrieval [4], etc. Over the last decades, a large number of clustering techniques following different mechanisms have been developed to support different applications. In particular, state-of-the-art Density Peak Clustering (DPC) technique [5] has gained growing

*Corresponding author

Email addresses: zafaryabrasool92@gmail.com (Zafaryab Rasool), sunil.aryal@deakin.edu.au (Sunil Aryal), reda.bouadjenek@deakin.edu.au (Mohamed Reda Bouadjenek), richard.dazeley@deakin.edu.au (Richard Dazeley)

[1]Zafaryab Rasool and Sunil Aryal contributed equally to this work.

attention over the last few years due to its simplicity with fewer initial parameters and ability to detect clusters of arbitrary shapes.

DPC can find arbitrary shaped clusters using the distance-based notion of (dis)similarity of data objects, i.e., objects with small distances are more similar than objects with larger distances. The basic idea of DPC is that the cluster centres are characterised as objects with higher local density than their neighbourhood and are separated by relatively large distances from other objects of higher density. Based on this idea, DPC first defines two measures for each object: (1) its *local density* ρ and (2) its *distance* δ *to the nearest higher density neighbor*, and then, it selects the objects with high ρ and large δ as the cluster centers. Finally, it assigns the rest of the objects to the clusters containing their nearest higher density neighbors. Because of its simplicity, DPC has been used for many applications such as neuroscience [6], remote sensing [7], biology [8], video-segmentation [9], traffic-network [10], computer vision [11], text mining [12], etc.

However, despite its popularity and simplicity, we note that DPC has two major weaknesses. First, DPC may fail to correctly identify clusters with highly variable densities. Second, the clustering results of DPC are very sensitive to units/scales used to measure/represent data features. Indeed, in real-world examples, data can be measured and expressed in different forms. For example, sample variability can be measured in standard deviation (σ) or variance (σ^2) and people's ability to borrow can be expressed in terms of debt-to-income ratio or income-to-debt ratio (inverse of each other). These issues are mainly due to the use of distance as the notion of similarity between data objects as we later discuss in Section 3.3.

In this paper, we argue that the above-mentioned limitations of DPC can be addressed using a data-dependent similarity measure that is robust to units/scales of data representation. In particular, we propose a similarity measure based on probability mass called *MP-Similarity* that uses the m_0-dissimilarity [13], and then, we incorporate it into DPC to get a variant that we call MP-DPC. This latter uses the same core procedures of the original DPC algorithm while replacing the distance-based (dis)similarity measure with MP-Similarity. We also use MP-Similarity with the Shared-Nearest-Neighbor-based Density Peak Clustering algorithm (SNNDPC) [14], a variant of DPC. We call this new variant of SNNDPC as MP-SNNDPC. We show through our extensive experiments on a wide range of real datasets that MP-DPC and MP-SNNDPC can detect clusters with varying densities and that their clustering results are robust to changing data units/scales. We also demonstrate that they significantly outperform the basic DPC and SNNDPC algorithms which are based on the Euclidean distance.

2. Background

Over the past few decades, several clustering techniques have been developed to address different problems in diverse applications. Below, we briefly review the main clustering methods based on the approaches they

adopt to grouping similar objects. For a comprehensive survey on the topic, we refer the reader to [15].

Partition-based methods construct several partitions of the data, where each partition is a cluster. These methods require pre-specifying the number of clusters such as k in k-means [16], a popular representative algorithm of partition-based clustering. The k-means algorithm randomly initialises k objects as centers (or cluster means), and assigns the rest of the objects to the nearest center. It then performs the following two steps: (i) updating center of each cluster based on the current assignment and (ii) assigning objects to clusters containing the nearest centers. k-means alternate between these steps until it finds the best centers i.e., no further change occurs. A major limitation of this family of algorithms is that they are not suitable to detect arbitrary shaped clusters. Moreover, the clustering results are sensitive to the initial selection of the cluster centers.

Hierarchical-based methods produce a tree-based hierarchical representation of clusters with the root representing the entire set of objects as one cluster and pairs of clusters or singletons are located at the lower levels of the tree. Tree partitions are constructed based on a proximity/similarity matrix and a partition of this tree generates the clustering results. Single linkage hierarchical clustering [17] is a representative example of this family. It determines the distance between two closest objects in different clusters and merge the two clusters if they contain objects separated by shortest distance. Single linkage follows a bottom-up approach such that it starts with singletons and continues merging two clusters until all the objects belong to one cluster.

Density-based clustering methods assume that the clusters are located in dense regions in the space separated by regions of lower density and are suitable for detecting irregularly-shaped clusters. DBSCAN [18], a popular representative of density-based clustering, finds objects with density greater than a threshold, which are connected to form clusters. However, selecting an appropriate threshold is difficult. Density Peak Clustering (DPC) [5] is another popular algorithm of this family that we will detail in the next section.

In addition to those discussed above, there exist other clustering methods in the literature. For example, **Distribution-based clustering methods** assume that data objects in a cluster have more likelihood of belonging to the same distribution (e.g., Gaussian distribution). These algorithms work well when the distribution of the data is known beforehand. Expectation-maximization [19] is a popular example of this family. **Grid-based methods**, such as STING [20], partition the original data space into several grids which are linked to form clusters based on statistical information such as the mean and the variance of the data objects.

3. Density Peak Clustering (DPC)

In this section, we first present in details the DPC algorithm, then, we describe a few of its existing variants and extensions, and finally, we analyse its main drawbacks.

3.1. DPC Algorithm

DPC [5] is a clustering algorithm based on the observation that *cluster centres* are characterized by: (i) *locally higher density*, i.e., a cluster centre has a higher-density neighbourhood than its neighbouring objects, and (ii) *relatively large separation*, i.e., cluster centres are at relatively large distances from other objects with higher local densities. Based on the above idea, DPC defines two quantities for each object that are used to identify the cluster centers from the rest of the objects. Formally, given a dataset $\mathcal{D} = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \cdots, \mathbf{x}^{(m)}\}$ with m objects (i.e., instances), where each data object $\mathbf{x}^{(i)} \in \mathbb{R}^n$ is an n-dimensional input feature vector, DPC proceeds in the following steps:

1. Compute the local density $\rho_{\mathbf{x}}$ of each object $\mathbf{x}$ as follows:

$$\rho_{\mathbf{x}} = \sum_{\mathbf{y} \in \mathcal{D}} \chi(dist(\mathbf{x},\mathbf{y}) - d_c) \tag{1}$$

   where $dist(\mathbf{x},\mathbf{y})$ represents the distance between object $\mathbf{x}$ and $\mathbf{y}$, $\chi(z) = 1$ if $z < 0$ and 0 otherwise, and d_c is the cutoff distance. In other words, the local density $\rho_{\mathbf{x}}$ of $\mathbf{x}$ is the number of objects that lie within distance d_c from $\mathbf{x}$.

2. For each object $\mathbf{x}$, compute the distance $\delta_{\mathbf{x}}$ which is the minimum distance between $\mathbf{x}$ and any other object $\mathbf{y} \in \mathcal{D}$ with density higher than $\rho_{\mathbf{x}}$. Specifically, the distance $\delta_{\mathbf{x}}$ is computed as follows:

$$\delta_{\mathbf{x}} = \min_{\mathbf{y} \neq \mathbf{x} \wedge \rho_{\mathbf{y}} > \rho_{\mathbf{x}}} \{dist(\mathbf{x},\mathbf{y})\} \tag{2}$$

   Here, the object $\mathbf{y}$ is the *nearest neighbor of high density* for $\mathbf{x}$. However, for the object $\mathbf{x}$ with the highest density, the distance $\delta_{\mathbf{x}}$ is computed as follows: $\delta_{\mathbf{x}} = \max_{\mathbf{y}} \{dist(\mathbf{x},\mathbf{y})\}$.

3. Using the computed values ρ and δ values, DPC aims to distinguish the cluster centres (peaks) from the rest of the objects. In particular, an object with high local density has its nearest neighbour of higher density relatively far and therefore has a large δ. Based on this idea, cluster centres are set to objects with high ρ and anomalously large δ. For selecting the cluster centres, a quantity γ is computed for each object as follows:

$$\gamma_{\mathbf{x}} = \rho_{\mathbf{x}} \times \delta_{\mathbf{x}} \tag{3}$$

   Objects with large γ values are set to be cluster centres.

4. Finally, once the cluster centres have been identified, the rest of the objects are then assigned to the clusters containing their nearest neighbour of higher density. This step is performed directly as assignment uses the nearest neighbour of higher density information obtained during the computation of δ in Equation 2.

3.2. Variants of DPC

Despite being a state-of-the-art clustering algorithm, DPC has limitations in dealing with clusters with complex shapes/structures such as varying densities [14]. To improve the performance of DPC algorithms in

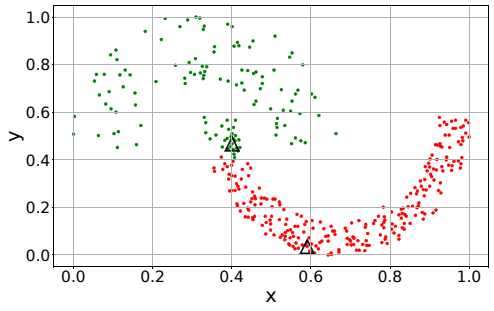
various applications, different variants of DPC have been proposed in the literature. Most of these employ measures that rely on the K-Nearest Neighbour (KNN) information of objects to capture the neighbourhood information. Shared-Nearest-Neighbour-based density peak clustering (SNNDPC) [14] uses Shared Nearest Neighbours (SNN) as a means to compute the similarities between objects and it selects the cluster centers based on the nearest neighbours and shared neighbours information. Furthermore, a two-step allocation strategy is introduced to reduce possible errors in cluster assignment. Another work based on fuzzy weighted K-nearest neighbour (FKNN-DPC) proposed by Xie et al. [21], focused on the problem of using different measures for local density computation in DPC. They proposed a uniform density metric based on KNN. Du et al. [22] proposed DPC-KNN, which initially introduced the idea of KNN to DPC and provided another option to compute local density. For handling high-dimensional datasets, principal component analysis (PCA) was used during preprocessing for dimensionality reduction. Recently, Hou et al. [23] proposed a density peak clustering algorithm which identifies the cluster centres using a relative density relationship approach and computes the local density using KNN. Wang et al [24] proposed McDPC for identifying clusters having multiple density peaks and low density clusters. McDPC improves the performance of DPC, however, relies on many additional parameters. In [25], authors have proposed a fuzzy kernel based on KNN for computing the local density and introduced a different method to select the cluster centers. Another recent work [26] uses mutual nearest neighbor information (based on KNN) to propose a new clustering algorithm based on DPC to detect arbitrary shaped clusters. The major issue with the above works is that they change the original semantics of DPC.

There exist other variants of DPC that focus on improving its speed on large datasets by reducing the distance computations [27, 28], however, they are out of the scope of the study of this paper.

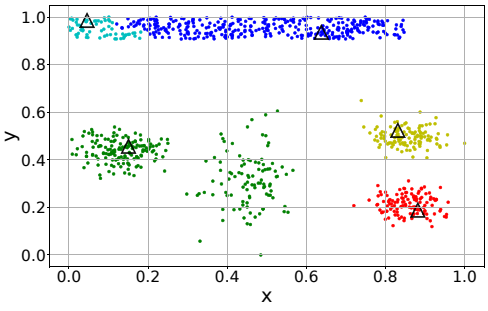
3.3. Limitations of DPC

The first limitation of DPC is the fact that it cannot identify clusters with highly varied densities. Indeed, because DPC selects cluster centers based on high ρ and large δ (or large γ), a cluster in a region in which none of the objects have sufficiently large γ may not be properly identified. Therefore, each object of this cluster will belong to the cluster of its nearest neighbor of higher density. Consequently, either some of the objects of this cluster or the entire cluster may become associated with another cluster. This issue is illustrated in Figures 1a and 1b, where in both examples, two cluster centers are incorrectly assigned to the same true cluster, leading to misclassification of many objects.

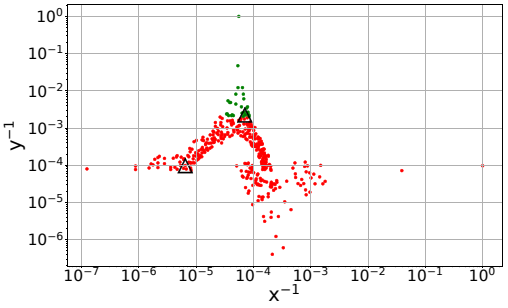
The second issue of DPC is that its clustering results are sensitive to the change in the units/scales used to measure/represent the features. Indeed, DPC requires the computation of the pairwise distance of objects for finding clusters. Hence, given that the distance between two objects can vary when the units/scales used to measure/represent the feature(s) changes, the clustering results can be significantly different from the results that can be obtained using the data in the original scale. Since the data of real-world applications are obtained from different sources, the units/scales of the features used may be different. For example, feature values can be represented in either log or *inverse* scale (e.g., likelihood can be given as log likelihood, fuel



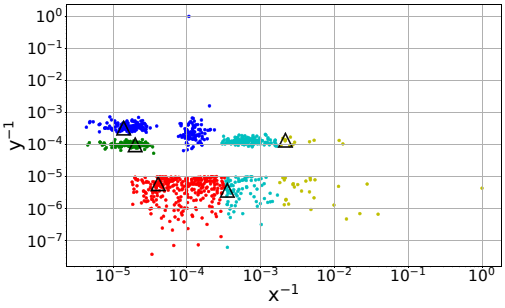
(a) Jain dataset.



(b) Synthetic Dataset.



(c) Jain dataset with scaled features.



(d) Synthetic dataset with scaled features.

Figure 1: Clustering results of the traditional DPC algorithm [First Row] Jain and Synthetic datasets with diverse densities shown in Subfigures (a) and (b) and [Second Row] Jain and Synthetic datasets with features transformed to inverse scale shown in Subfigures (c) and (d). Note that x^{-1} and y^{-1} are plotted on the logarithmic scale. Each Δ represents a cluster center.

efficiency of vehicles in KM/L or L/100KM). In such cases, it is natural to expect that DPC clustering results will not be affected by the change in units/scales.

While linearly scaled features are easily handled when the data is normalised to a unit scale, the non-linearly scaled features may still be problematic and difficult to resolve. For example, if a, b, c and d are real values in ascending order with equal interval, and a', b', c' and d' are their corresponding values in the logarithmic scale, then we have $b - a = d - c$ but $b' - a' \neq d' - c'$. Consequently, the similarity between two objects captured in different scales can be different, which can significantly affect the clustering results of DPC. The effect of data transformation on the results of DPC is illustrated in Figures 1c and 1d. We can observe that the clustering results obtained by DPC on these datasets are significantly different from those obtained on the datasets with the original scales.

One reason for the above-mentioned issues with DPC is because it uses the Euclidean distance for computing the similarity between two objects. A recent study by Aryal et al. [13] has mentioned two issues of using distance as a measure of (dis)similarity between objects in data mining: (i) the similarity of two objects is data distribution independent (i.e., it is not affected by the distribution of other data objects),

and (ii) the similarity of two objects is sensitive to the data representation (i.e., units/scales used to measure data features).

Previous work has tried to improve the performance of DPC by incorporating diverse similarity measures [14, 21], but most of these methods remain sensitive to units/scales of data. For example, SNNDPC [14] attempts to alleviate the issue of varying density in DPC to some extent by replacing the Euclidean distance-based similarity measure with a new measure based on Shared Nearest Neighbours (SNNs). However, it remains sensitive to the change in units/scales used to express data because SNNs are searched using the Euclidean distance. In addition, SNNDPC modifies some procedures in the original DPC algorithm, which causes the loss of the true semantic of DPC. This necessitates the development/usage of data-dependent similarity measures for DPC that capture the intrinsic structure of the data, thus reflecting the true similarity between objects and helping to achieve better clustering results.

4. MP-DPC

In this section, we propose a data-dependent similarity measure, which we call MP-Similarity. Then, using MP-Similarity we present MP-DPC, which is a data-dependent variant of DPC.

4.1. The Idea of Data-dependent Similarity

We have previously argued that the use of distance-based Euclidean similarity measures is not appropriate for estimating similarities between objects. The similarity estimated by a data-independent measure (such as the Euclidean distance) between two objects $\mathbf{x}$ and $\mathbf{y}$ in a dense region is the same as that of two equidistant objects in a sparse region. However, psychologists argue that the human-judged similarity between two objects is data-dependent [29, 30]. In other words, two objects in a sparse region should be more similar than two objects with equal distance in a dense region. For example, consider two groups of dogs such that the first group has only one breed of dogs (e.g., German Shepherd), while the second group has a mix of breeds. A human would judge two dogs in the first group to be less similar (as all dogs are of the same breed) than the two German Shepherd dogs in the second group. Thus, the similarity measure must take into account the distribution of data.

The similarity between two objects in a multidimensional space is estimated by aggregating their similarities in each dimension. If two objects $\mathbf{x}$ and $\mathbf{y}$ have the same value in a dimension i (i.e., $x_i = y_i$), a data-independent similarity measure assigns the similarity of 1 regardless of the distribution of data in the dimension i. However, as mentioned earlier, it should depend on the data distribution (i.e., the likelihood of x_i). For example, consider the data distribution shown in Table 1, where the objects are represented in the columns and the values of the features/dimensions in the rows. In particular, let us consider the two instances Inst1 and Inst2, that take the same values in both the i^{th} and j^{th} dimension, however their matching value in the i^{th} dimension is rare (only these two instances have the value of 2) and that in the j^{th} dimension is frequent (these two and six other instances have the value of 2). When estimating the (dis)similarity

Table 1: Example of a sample data distribution to show the importance of data-dependent similarity measure [13]

Dim.	*Inst1*	*Inst2*	*Inst3*	*Inst4*	*Inst5*	*Inst6*	*Inst7*	*Inst8*	*Inst9*	*Inst10*
.	.	.	.	.	.	.	.	.	.	.
i	2	2	1	1	1	1	1	1	1	1
j	2	2	2	2	2	2	2	2	1	1
.	.	.	.	.	.	.	.	.	.	.

between *Inst1* and *Inst2* using the Euclidean distance (data-independent measure), the matching values in dimensions i and j contribute equally to the overall similarity score. However, psychologists argue that they do not provide the same amount of information about the similarity of *Inst1* and *Inst2* and they should contribute differently. The matching value in dimension i is rare (low probability) and it provides more information than the matching value in dimension j which is frequent (high probability). This is especially true in the case of high-dimensional datasets in which the objects mostly lie in low-dimensional manifolds. Therefore, from the above discussion, similarity measures that take into account the data distribution both when $x_i \neq y_i$ and $x_i = y_i$ are useful to represent the intrinsic structure of the data.

4.2. Mass-based Probabilistic (MP) Similarity Measure

In this section, we first discuss the data-dependent dissimilarity measure known as m_0-dissimilarity, and then, we introduce our Mass-based Probabilistic Similarity (MP-Similarity) measure.

4.2.1. m_0-dissimilarity

Aryal et al. [13] proposed m_0-dissimilarity as a fully data-dependent dissimilarity measure and used it in the context of KNN classification. It estimates the dissimilarity of two objects $\mathbf{x}$ and $\mathbf{y}$ by using the probability mass of the region covering x_i and y_i in each dimension i instead of the spatial distance $|x_i - y_i|$. These dissimilarities are then aggregated to give the m_0 dissimilarity. The idea is that $\mathbf{x}$ and $\mathbf{y}$ are more dissimilar with respect to the dimension i if many objects in the dataset have the values of feature i between x_i and y_i. Let m be the number of objects in the dataset $\mathcal{D}$, n be the number of dimensions, $R_i(\mathbf{x},\mathbf{y}) = [min(x_i, y_i), max(x_i, y_i)]$ be the region covering objects $\mathbf{x}$ and $\mathbf{y}$ in dimension i and $|R_i(\mathbf{x}, \mathbf{y})| = |\{\mathbf{z} \in P : min(x_i, y_i) \leq z_i \leq max(x_i, y_i)\}|$, then m_0-dissimilarity of objects $\mathbf{x}$ and $\mathbf{y}$ is defined as:

$$m_0(\mathbf{x},\mathbf{y}) = \left(\frac{1}{n} \sum_{i=1}^{n} \log \frac{|R_i(\mathbf{x}, \mathbf{y})|}{m} \right) \tag{4}$$

In the above equation, $\frac{|R_i(\mathbf{x}, \mathbf{y})|}{m}$ represents the probability mass of the region in which x_i and y_i lie. Aryal et al. [13] provided a probabilistic interpretation of m_0-dissimilarity based on the Naive Bayesian assumption that attributes (dimensions) are independent of each other.

In the above equation, $|R_i(\mathbf{x}, \mathbf{y})|$ incorporates the region information in which x_i and y_i lie for each dimension i, which makes m_0 a data-dependent measure. In the example shown in Table 1, the same values

in dimensions i and j contribute differently in the overall dissimilarity of *Inst1* and *Inst2*. Moreover, m_0 is not affected when the values x_is along feature i are scaled (linearly or nonlinearly) to x'_is (a different scale) as the number of instances in $|R_i(\mathbf{x}, \mathbf{y})|$ and $|R_i(\mathbf{x}', \mathbf{y}')|$ remains the same because scaling either preserves or reverses the ordering of data values. Thus, m_0 captures the data-distribution as well as remain invariant to any change in the scale of data.

Computing $|R_i(\mathbf{x}, \mathbf{y})|$ requires a range search which makes it computationally expensive. To efficiently compute $|R_i(\mathbf{x}, \mathbf{y})|$, Aryal et al. [13] proposed to divide the range of continuous valued domain in the i^{th} dimension into b equal-frequency intervals or bins. For each bin, the proposed algorithm stores the frequency or the number of objects in that bin. Equal-frequency binning makes it invariant to linear or non-linear scaling of data. Because many data objects can have the same value in feature i (i.e., there can be duplicate values), it may be impossible to have perfectly equal-frequency bins, i.e., bins may not always have the same frequency. With bin frequencies, $|R_i(\mathbf{x}, \mathbf{y})|$ can be approximated quickly by aggregating the frequencies of the bins in which x_i and y_i lie and the bins in between. Data mass between each pair of bins in dimension i can be precomputed and stored in a $b \times b$ matrix so that $|R_i(\mathbf{x}, \mathbf{y})|$ can be calculated directly by a matrix lookup.

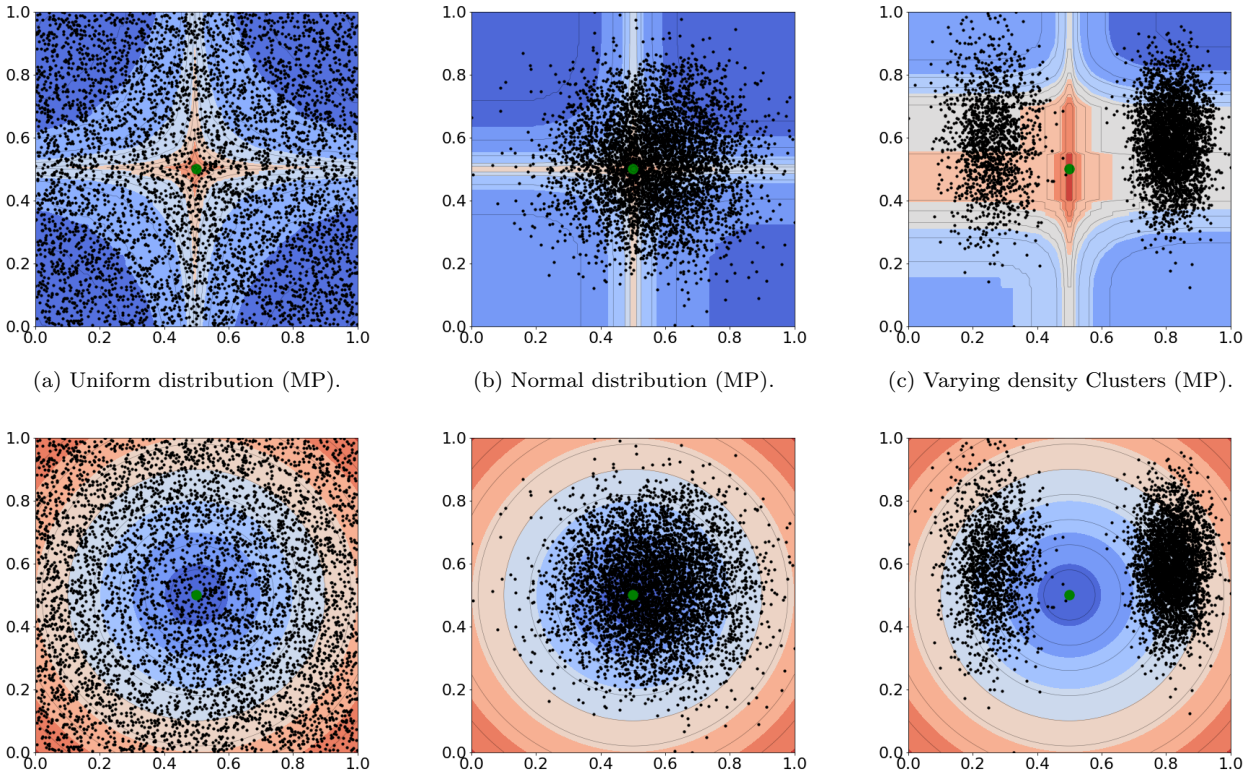
4.2.2. MP-Similarity

We observe that m_0 is not a valid metric as: (i) the dissimilarity $m_0(\mathbf{x},\mathbf{y})$ of objects when $\mathbf{x} = \mathbf{y}$ is non-zero, and (ii) the dissimilarities $m_0(\mathbf{x},\mathbf{x})$ and $m_0(\mathbf{y},\mathbf{y})$ may not be similar. Therefore, m_0 cannot be used as a similarity measure in applications where metric properties are required/assumed. Motivated by this, we propose a new similarity measure called Mass-based Probabilistic Similarity (MP-Similarity or MP), which is obtained by normalising m_0 using "$m_0(\mathbf{x},\mathbf{x}) + m_0(\mathbf{y},\mathbf{y})$" where $m_0(\mathbf{x},\mathbf{x})$ represents the self-dissimilarity. It is defined as:

$$MP(\mathbf{x}, \mathbf{y}) = \frac{2 * m_0(\mathbf{x},\mathbf{y})}{m_0(\mathbf{x},\mathbf{x}) + m_0(\mathbf{y},\mathbf{y})} \tag{5}$$

Thus, MP retains the merits of m_0, and like m_0-dissimilarity, MP satisfies the symmetric and triangle inequality assumptions – given x_i, y_i, and z_i in dimension i, $|R(x_i, y_i)| = |R(y_i, x_i)|$ and $|R(x_i, z_i)| \leq |R(x_i, y_i)| + |R(y_i, z_i)|$, which can be generalised in a multidimensional space. Also, from Equation 5, we have that $\forall \mathbf{x}, MP(\mathbf{x},\mathbf{x}) = 1$, which shows that MP satisfies the identity assumption. The range of similarity assigned by MP lies in the range of [0,1] with $MP(\mathbf{x}, \mathbf{x}) = MP(\mathbf{y}, \mathbf{y}) = 1$.

Next, we analyze the behaviour of MP-Similarity under different data distributions including varying density clusters. To do so, we generate three datasets of 5,000 instances with 2 dimensions each from the following distributions: (i) uniform distribution, (ii) normal distribution, and (iii) varying density clusters. Figure 2 shows the contour plots of similarity of objects in the space to the center (0.5, 0.5) using MP-Similarity (Figures 2a, 2b, 2c) and the Euclidean distance (Figures 2d, 2e, 2f). In the contour plots, we represent the region of high similarity with dark orange color and the region of low similarity with dark blue color. From Figure 2, we make the following observations:



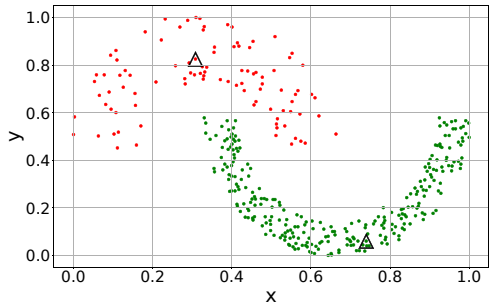
(a) Uniform distribution (MP). (b) Normal distribution (MP). (c) Varying density Clusters (MP).

(d) Uniform distribution (Euclidean). (e) Normal distribution (Euclidean). (f) Varying density Clusters (Euclidean).

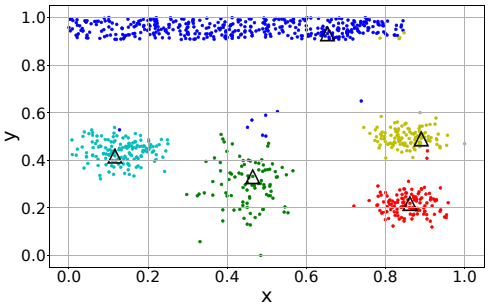
Figure 2: Contour plots of dissimilarity of objects in the 2-dimensional space with reference to object (0.5, 0.5), based on MP similarity (first row) and Euclidean distance (second row), in different distributions. The darker the color, the higher the dissimilarity.

- **MP-Similarity adapts to different data distributions.** Figures 2a and 2b show the contours of MP-Similarity on datasets with uniform distribution and normal distribution respectively. As it can be observed, the contours are different for the different data distributions which demonstrates the data-dependent behaviour of MP-Similarity as it adapts the contours to underlying data distribution. However, the data-independent similarity measures (e.g., Euclidean distance) produce the same contours irrespective of the different distributions.

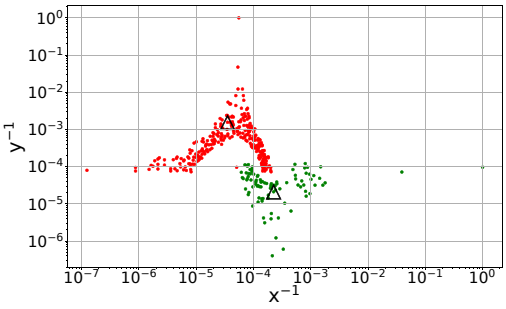
- **MP-Similarity adapts to different densities.** Figure 2c shows dataset containing two clusters with varied densities. As shown, from the center, the contour decreases slower in the region with sparse concentration of objects than in the region with dense concentration. Thus, MP-Similarity adapts its contours to the local data distribution which is not the case with a data-independent similarity measure, where contour vary at the same rate on either side regardless of the varying data concentration. This reflects the characteristic of a data-dependent measure discussed earlier: two objects in a sparse region are more similar than two objects with equal distance in a dense region.



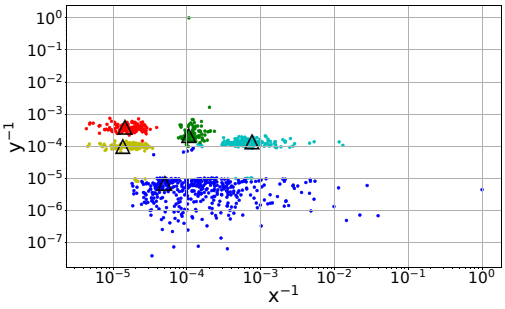
(a) Jain dataset.



(b) Synthetic Dataset.



(c) Jain dataset with scaled features.



(d) Synthetic Dataset with scaled features.

Figure 3: Clustering results of the MP-DPC algorithm on (i) Jain and Synthetic datasets with diverse densities shown in Figures (a) and (b) and (ii) Jain and Synthetic datasets with features transformed to inverse scale. Note that x^{-1} and y^{-1} are plotted on the logarithmic scale. Each Δ represents a cluster center.

The data-dependent characteristic of MP-Similarity makes it a suitable candidate for finding similarity between objects regardless of the underlying data distribution. Next, we show that MP-Similarity can be incorporated with DPC to improve its clustering results on datasets with varying density and scales.

4.3. MP-DPC

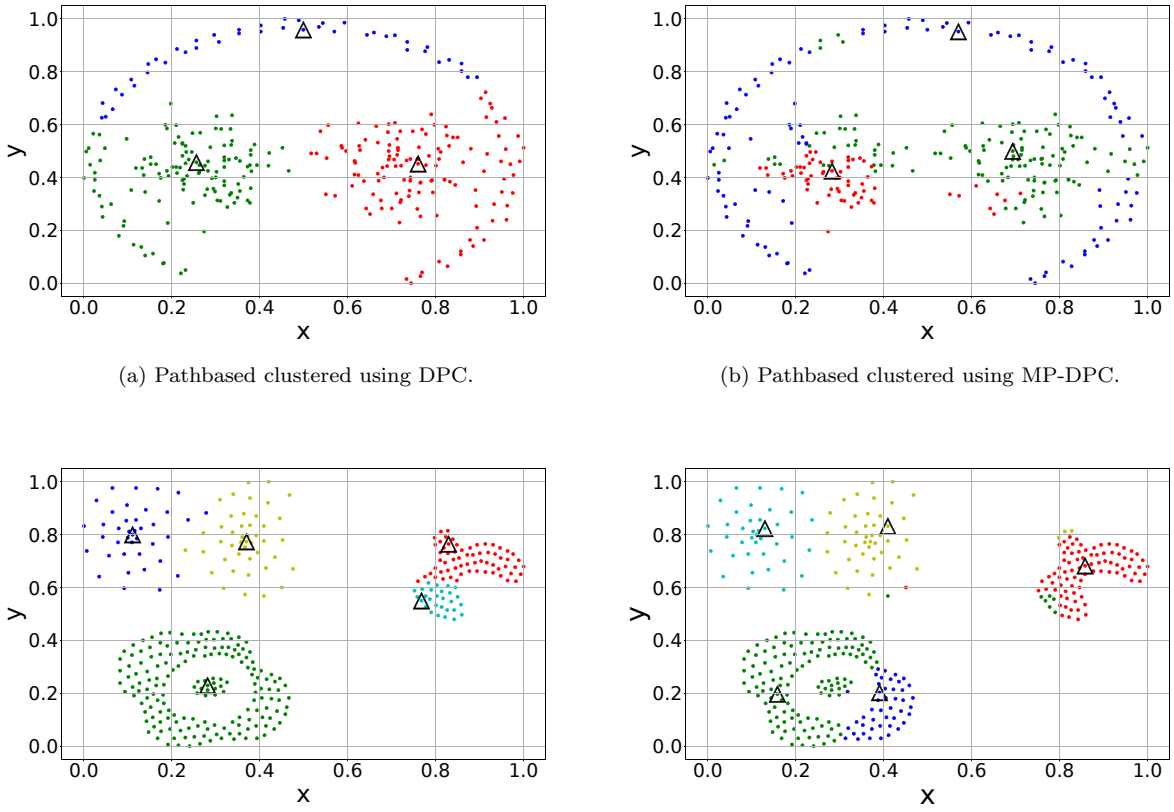
In this section, we present MP-DPC, which incorporates the MP-Similarity measure in DPC to create a data-dependent version of DPC.

DPC requires pairwise distances/dissimilarities between objects to compute ρ and δ as given in Equations 1 and 2. To do so, DPC uses the default Euclidean distance as the distance measure. We replace the default distance measure in DPC by MP-Similarity and we use it as follows:

$$dist(\mathbf{x},\mathbf{y}) = 1 - \mathrm{MP}(\mathbf{x},\mathbf{y}) \tag{6}$$

This version of DPC is called MP-DPC, and we argue that it does not require modifying the local density computation (or any other steps) in the procedure of the original DPC algorithm.

Figure 3 shows the performance of MP-DPC on the Jain dataset and a Synthetic dataset that have various densities. Unlike DPC, we observe in Figures 3a and 3b that MP-DPC is clearly able to identify the



(a) Pathbased clustered using DPC.

(b) Pathbased clustered using MP-DPC.

(c) Compound clustered using DPC.

(d) Compound clustered using MP-DPC.

Figure 4: Performance of DPC and MP-DPC algorithm on multi-density datasets Pathbased and Compound.

dense and sparse clusters in both datasets as it is able to adapt to the local data distribution. Similarly, when the features of these datasets are converted to inverse scale as shown in Figures 3c and 3d, MP-DPC is still able to identify the same clusters as in the original scale of the features. This is because MP-Similarity is dependent on the number of instances falling between two objects which do not vary when the scale is varied linearly or non-linearly. Thus, MP-DPC is suitable for varying data distributions, densities and scale.

Next, we show in Figure 4 the performance of MP-DPC on two complex shape datasets with multi-density clusters: the Pathbased dataset, which consists of three clusters and the Compound dataset, which consists of five clusters. We observe that on the Pathbased dataset, MP-DPC identifies the ring cluster better than DPC but shows some misclassifications on the other two clusters inside the ring. For the Compound dataset, we observe that both DPC and MP-DPC did not perform well: they both correctly identified the two sparse clusters on the upper left corner, but they both failed to correctly identify the other clusters. This performance of DPC and MP-DPC could be an issue of the method of selection of cluster centers, which we need to further investigate.

We also incorporate MP-Similarity in SNNDPC [14] to improve its performance and we call this new version MP-SNNDPC. The remaining steps of SNNDPC do not need to be modified as in MP-DPC.

Finally, we note that the time complexity of MP-DPC and MP-SNNDPC remains the same as their

original counterparts based on the Euclidean distance because $m_0(\mathbf{x},\mathbf{y})$ can be estimated in $O(n)$ time.

5. Experimental Evaluation

5.1. Experimental Setup

In this section, we describe the experimental setup we used in our evaluations, including a description of the baselines, details of our implementation, the datasets, and the metrics used.

5.1.1. Baselines

We demonstrate the performance of our MP-DPC and MP-SNNDPC algorithms by comparing them against the following state-of-the-art algorithms:

- DPC: the original DPC algorithm using the Euclidean distance;
- SNNDPC: the original SNNDPC algorithm using the Euclidean distance;
- IK-DPC: DPC algorithm using a similarity measure based on Isolation Kernel (IK) [31];
- IK-SNNDPC: SNNDPC algorithm using the IK measure;
- Lin-DPC: based on Lin's similarity measure [32];
- Lin-SNNDPC: SNNDPC using Lin's similarity measure;
- k-means [16]: which is used as a simple baseline.

We note that there exist other data-dependent measures, which can be categorized into one-dimensional and tree-based measures [13]. We have selected IK as it is a popular tree-based measure [31] and Lin as it is based on a similar idea as MP.

Specifically, IK [31] is a data-dependent similarity measure based on an ensemble of t random trees called Isolation Forest [33]. Each tree $H_i(i = 1, 2, ...t)$ partitions the space using a small subsample of data $\mathcal{D}_i \subset \mathcal{D}$ ($|\mathcal{D}_i| = \psi$). Partitions are created such that sparse regions comprise of larger partitions than the dense regions adapting to local data distribution. The similarity of two objects is estimated as the number of trees in which the two objects fall in the same partition.

Regarding the Lin similarity measure [32], it introduces a notion of similarity based on information theory for ordinal data. Aryal et al. [13] presented its multidimensional version to measure the similarity of two data objects $\mathbf{x}$ and $\mathbf{y}$ in multidimensional continuous domain, which aggregates the Lin's similarity in each dimension as shown below:

$$Lin(\mathbf{x},\mathbf{y}) = \frac{1}{n}\sum_{i=1}^{n}\frac{2\times\log\sum_{z_i=min(x_i,y_i)}^{max(x_i,y_i)}P(z_i)}{\log P(x_i)+\log P(y_i)} \tag{7}$$

We use the idea of equal-frequency bins and data mass in bins as in the case of MP-Similarity to compute $P(z_i)$, $P(x_i)$ and $P(y_i)$ efficiently. The fundamental difference between these two measures is the ability

of MP-Similarity to capture data-dependent information in features where $x_i = y_i$. In Lin's measure, the similarity of $\mathbf{x}$ and $\mathbf{y}$ in a dimension i is 1 regardless of the likelihood of x_i. In the example discussed in Table 1, both features i and j contribute equally to the overall similarity of $Inst1$ and $Inst2$ though they share a frequent value in the dimension j and a rare value in the dimension i. But, they contribute differently in MP-Similarity.

5.1.2. Implementation Details

The implementations of SNNDPC and k-means we used are based on [14] and [34] respectively. The implementation of IK is based on the tree-partition mechanism of Isolation Forest as described in [31]. The implementation of Lin is similar to the implementation of MP as both require $|R_i(\mathbf{x},\mathbf{y})|$ for the computation of similarity of two objects x and y in dimension i.

All algorithms requires parameters tuning to obtain optimal performance. Specifically, DPC requires the parameter d_c to be prespecified. Authors of the original DPC paper [5] suggested d_c to be such that the total number of neighbours are between 1-2% of the total number of points. Extensions of DPC [14, 21, 23] have modified the percentage to obtain the best results. We modify and extend this range from 1% to 3% with step size of 0.1% and report the best results. SNNDPC requires tuning the parameter k for obtaining the best results [14], which we selected in the range of [5, 50] with a step size of 5. For consistency, we use the same setting of d_c and k with the data-dependent and data-independent variants of DPC and SNNDPC.

For MP-DPC and Lin-DPC, MP-Similarity and Lin's measure require setting an appropriate number of bins b. We vary b from the set [20, 40, 60, 80, 100, $log_2(m)$], where $log_2(m)$ is the default value used in [13]. IK has two parameters, the *sampling size* (ψ) and the *number of trees* (t). For ψ, we select a value from $\{2^a | a = 2, 3, ..., 8\}$, while the number of trees is fixed to the default value of 100, as suggested in [31, 33]. Since the algorithm is random, we run it 5 times for each sampling size and report the mean score for each evaluation metric discussed in Section 5.1.4.

In k-means, the initial centres are selected using the k-means++ [35] method. Additionally, all algorithms require specifying the number of cluster centers. To be consistent, we fixed the number of clusters in these algorithms to be equal to the number of classes in the ground truth.

5.1.3. Datasets

The above algorithms are evaluated on 17 real-world datasets, which are described in Table 2. Many of these datasets are widely used in existing DPC literature, such as [14, 21, 23], and are obtained from UCI Machine Learning Repository [36]. These datasets are different in terms of number of objects, dimensionality, and the number of clusters. The varying properties of these datasets help to comprehensively evaluate the performance of the clustering algorithms, as they can represent different situations.

To eliminate large differences in the range of dimensions, we normalise the feature values of the datasets to be in the range of [0,1] using min-max normalisation. Note that normalisation is not required for MP-Similarity as it does not use the data values in the similarity calculation, but it is important for distance-based

Table 2: Summary of the Datasets.

Dataset	#Instances	#Attributes	#Clusters
Iris	150	4	3
Wine	178	11	3
Parkinson	197	23	2
Thyroid	215	4	3
Libra	360	90	15
Dermatology	366	33	6
WDBC	569	30	2
Balance Scale	625	4	3
Statlog (Vehicle)	846	18	4
Gtzan	1000	230	10
Hba	1500	187	15
Wap	1560	8460	20
Cardiotocography	2126	23	10
Fbis	2463	2000	17
Spambase	4601	58	2
Satimage	6435	36	6
Corel	10000	67	100

measures such as Euclidean distance.

5.1.4. Evaluation Metrics

We evaluate the performance of all clustering algorithms using the following popular metrics: Adjusted Mutual Information, Adjusted Rand Score and Fowlkes-Mallows Index. Specifically, let's U and V be respectively the ground truth cluster labels and cluster assignments by a clustering algorithm for m data objects, a be the number of object pairs that belong to the same cluster in U and V, b be the number of object pairs that belong to the same cluster in U but not in V, c be the number of object pairs that belong to the same cluster in V but not in U, d be the number of object pairs belonging to different clusters in U and V. The metrics are defined as follows:

- Adjusted Mutual Information (AMI): AMI is used to measure the similarity between two clusterings of the same data. It is a variation of Mutual Information (MI) that informs the reduction in the Entropy (H) of class labels when the class labels are known. Unlike mutual information, AMI is adjusted against chance which means that the similarity is only governed by the structure of the dataset appearing in two clustering and not by chance. AMI is calculated as follows [34]:

$$AMI = \frac{MI(U,V) - \mathbb{E}[MI(U,V)]}{\frac{1}{2}(H(U) + H(V)) - \mathbb{E}[MI(U,V)]} \tag{8}$$

  where $\mathbb{E}[MI(U,V)]$ is the expectation of the mutual information.

Table 3: AMI: Performance of Clustering algorithms and their data-dependent variants.

Datasets	k-means	DPC				k-means	SNNDPC			
		Euclidean	IK	Lin	**MP**		Euclidean	IK	Lin	**MP**
Iris	0.7387	0.7810	0.8286	**0.8479**	**0.8479**	0.7387	**0.9133**	0.7696	0.8968	0.8625
Wine	**0.8514**	0.7847	0.7702	0.6752	0.7070	0.8514	0.8769	0.8101	**0.9209**	0.9099
Parkinson	0.2318	0.1916	**0.2833**	0.2769	0.2115	0.2318	0.2637	0.2193	**0.2873**	0.2821
Thyroid	0.5909	0.3097	0.7530	0.7858	**0.7947**	0.5909	0.5858	0.6318	**0.8497**	0.8076
Libra	0.5399	0.5706	0.5234	0.5210	**0.5264**	0.5399	**0.6283**	0.5407	0.5237	0.5299
Dermatology	0.8811	0.8096	0.8121	0.8820	**0.9370**	0.8811	0.8990	0.8509	0.9108	**0.9301**
WDBC	0.6226	0.2447	0.6342	**0.7182**	0.6614	0.6226	**0.7335**	0.5872	0.6983	0.6933
Balance Scale	0.1116	0.1763	0.1002	**0.2144**	**0.2144**	0.1116	0.1726	0.1502	**0.1876**	0.1809
Statlog	0.0966	0.1735	0.2129	0.3368	**0.3413**	0.0966	0.2050	0.2457	**0.3405**	0.3230
Gtzan	**0.3288**	0.2738	0.1667	0.2785	0.2813	**0.3288**	0.3179	0.1969	0.2965	0.2944
Hba	0.3018	0.1977	0.2280	0.3024	**0.3097**	0.3018	0.2424	0.2743	**0.3653**	0.3556
Wap	**0.3143**	0.1147	0.1269	0.1415	0.2908	0.3143	0.1203	0.1383	0.2119	**0.3798**
Cardiotocography	0.2763	0.2241	0.2764	0.2940	**0.3200**	0.2763	0.2391	0.3157	**0.3571**	0.3510
Fbis	0.1762	0.3163	0.1189	0.2589	**0.3442**	0.1762	**0.2319**	0.0222	0.0280	0.2095
Spambase	0.0098	0.0950	0.0758	0.2186	**0.3815**	0.0098	0.0737	0.1568	0.2065	**0.3633**
Satimage	0.6119	**0.6377**	0.6194	0.5891	0.5958	0.6119	0.6475	0.5807	0.6816	**0.7107**
Corel	0.1973	0.1473	0.1753	0.2018	**0.2079**	0.1973	0.1743	0.2162	0.2493	**0.2514**
Avg.	0.4048	0.3558	0.3944	0.4437	0.4690	0.4048	0.4309	0.3945	0.4713	0.4962

- Adjusted Rand Index (ARI): ARI [37] is another metric to evaluate the similarity of the clusterings. It considers the number of objects existing in the same cluster and in different clusters, and measures the fraction of pairs of points that are correctly clustered to the same or different clusters.

$$ARI = \frac{a - (a+c)(a+b)/d}{(a+c) + (a+b)/2 - (a+c)(a+b)/d} \tag{9}$$

- Fowlkes Mallows Index (FMI): FMI [38] is used to measure the similarity of clusters obtained through different clustering algorithms. It is computed as follows:

$$FMI = \frac{a}{\sqrt{(a+b)(a+c)}} \tag{10}$$

A perfect clustering will result in achieving a maximum score of 1 for the above metrics.

5.2. Results

In this section, we report the results of our experiments comparing the performance of our proposed MP-Similarity measure in DPC and SNNDPC algorithms with other contenders.

Table 4: ARI: Performance of Clustering algorithms and their data-dependent variants.

Datasets	k-means	DPC				k-means	SNNDPC			
		Euclidean	IK	Lin	**MP**		Euclidean	IK	Lin	**MP**
Iris	0.7163	0.7196	0.8266	**0.8681**	**0.8681**	0.7163	**0.9222**	0.7498	**0.9222**	0.8857
Wine	**0.8685**	0.8191	0.7627	0.6661	0.6999	0.8685	0.8922	0.8226	0.9284	**0.9295**
Parkinson	0.0520	0.3363	0.3758	**0.3811**	0.2879	0.0520	0.2916	0.2172	0.2148	**0.2985**
Thyroid	0.6283	0.1822	0.8256	0.8601	**0.8623**	0.6283	0.6823	0.6638	**0.9064**	0.8618
Libra	0.3207	**0.3503**	0.3055	0.3024	0.3018	0.3207	**0.4134**	0.3141	0.3012	0.2914
Dermatology	0.7426	0.8159	0.7912	0.8785	**0.9409**	0.7426	0.8434	0.8216	0.8637	**0.9302**
WDBC	0.7302	0.2712	0.7327	**0.8183**	0.7741	0.7302	**0.8308**	0.6724	0.8052	0.8055
Balance Scale	0.1351	0.1348	0.1138	**0.2934**	**0.2934**	0.1351	**0.2203**	0.1885	0.1871	0.1997
Statlog	0.0757	0.1092	0.1531	0.2286	**0.2346**	0.0757	0.1509	0.1762	**0.2813**	0.2653
Gtzan	**0.1876**	0.1580	0.0915	0.1700	0.1670	**0.1876**	0.1513	0.0673	0.1391	0.1391
Hba	**0.1471**	0.1087	0.1087	0.1396	0.1454	0.1471	0.079	0.0897	**0.2016**	0.1828
Wap	0.1485	0.0555	0.0699	0.0638	**0.2368**	0.1485	0.0551	0.0479	0.1447	**0.1831**
Cardiotocography	0.1317	0.0820	0.1320	0.1651	**0.1889**	0.1317	0.1248	0.1622	**0.2315**	0.1964
Fbis	0.0493	0.1227	0.0401	0.129	**0.2437**	0.0493	**0.1111**	-0.0014	-0.0031	0.0942
Spambase	-0.0048	0.1514	0.0654	0.2987	**0.4344**	-0.0048	-0.003	0.1297	0.2427	**0.4492**
Satimage	**0.5263**	0.5120	0.5139	0.5016	0.5198	0.5263	0.5824	0.4937	0.6660	**0.7021**
Corel	0.0462	0.0307	0.0297	**0.0493**	0.0489	0.0462	0.0253	0.0361	0.0553	**0.0619**
Avg	0.3236	0.2917	0.3493	0.4008	0.4263	0.3236	0.3749	0.3324	0.4169	0.4398

5.2.1. Performance analysis

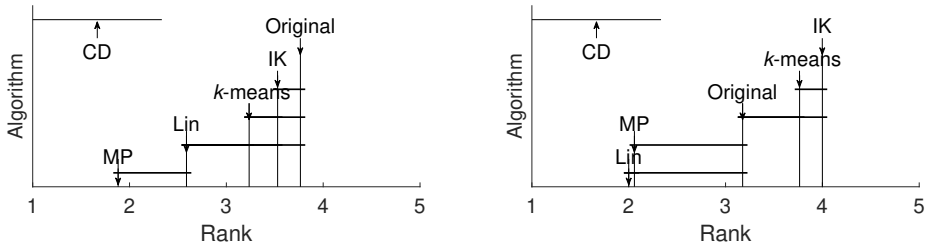
The obtained clustering results are shown in Tables 3, 4, and 5, for respectively AMI, ARI, and FMI. The different variants of DPC and SNNDPC are denoted by the name of the similarity measure they use. The average score over all the datasets is presented in the last row of each table.

Table 3 shows the AMI score obtained by the DPC and the SNNDPC algorithm using the different similarity measures (discussed earlier) and the k-means algorithm on different datasets. The results show that MP-DPC outperforms other variants of DPC and k-means on most datasets. An interesting observation is that all the data-dependent variants have generally better performance than that of the Euclidean. While IK could not achieve the best score on any dataset, it shows better results than the Euclidean on many datasets. k-means obtained best score on three datasets but has less overall average score than the MP and Lin. On the other hand, for SNNDPC, Lin has the best score on seven datasets, MP on five datasets and Euclidean on four datasets. However, MP achieves better average score than any of the contendors including Lin, demonstrating its superior performance.

In terms of ARI (Table 4) and FMI score (Table 5), similar behaviour is observed. While the performance of algorithms varied for some datasets, MP-DPC and MP-SNNDPC still outperform their other counterparts as well as k-means algorithm on most datasets. The close competitor to MP in all the three metrics is Lin

Table 5: FMI: DPC algorithm with data-independent Euclidean distance and data-dependent (dis)similarity measures

Datasets	k-means	DPC				k-means	SNNDPC			
		Euclidean	IK	Lin	**MP**		Euclidean	IK	Lin	**MP**
Iris	0.8112	0.8159	0.8846	**0.9115**	**0.9115**	0.8112	**0.9479**	0.8350	0.9478	0.9233
Wine	**0.9126**	0.8801	0.8452	0.7800	0.7952	0.9126	0.9330	0.8825	0.9525	**0.9532**
Parkinson	0.5957	**0.8140**	0.8059	0.7433	0.6982	0.5957	**0.8167**	0.6932	0.7584	0.7036
Thyroid	0.8546	0.5697	0.9206	**0.9356**	0.9351	0.8546	0.8611	0.8525	**0.9572**	0.9349
Libra	0.3726	**0.4009**	0.3627	0.3564	0.3547	0.3726	**0.4598**	0.3649	0.3605	0.3505
Dermatology	0.7947	0.8519	0.8325	0.9034	**0.9526**	0.7947	0.8824	0.8572	0.8944	**0.9441**
WDBC	0.8770	0.6595	0.8778	**0.9142**	0.8938	0.8770	**0.9217**	0.8467	0.9084	0.9085
Balance Scale	0.4666	0.5478	0.5042	**0.6025**	**0.6025**	0.4666	0.5251	0.5105	**0.6552**	0.6533
Statlog	0.3070	0.4209	0.3872	0.4540	**0.4610**	0.3070	0.3881	0.4281	**0.4793**	0.4713
Gtzan	0.2942	0.2971	0.2326	0.3020	**0.3028**	0.2942	0.3226	0.3005	**0.3343**	**0.3343**
Hba	0.2351	0.2015	0.2084	0.2350	**0.2410**	0.2351	0.2359	0.2086	**0.2772**	0.2713
Wap	**0.3642**	0.1961	0.2706	0.2615	0.3410	0.3642	0.3122	0.3056	0.3122	**0.3747**
Cardiotocography	0.2498	0.2310	0.2773	0.3265	**0.3335**	0.2498	0.2531	0.2814	**0.3689**	0.3233
Fbis	0.1916	0.2779	0.2588	0.2693	**0.3309**	0.1916	0.2917	0.3176	**0.3239**	0.2886
Spambase	0.7160	0.7179	0.6931	0.7221	**0.7537**	0.7160	0.7186	0.7213	0.7224	**0.7551**
Satimage	0.6142	**0.6299**	0.6145	0.6032	0.6095	0.6142	0.6934	0.5986	0.7382	**0.7677**
Corel	0.0564	0.0514	0.0622	**0.0739**	0.0728	0.0564	0.0791	0.0803	0.0929	**0.0939**
Avg	0.5126	0.5037	0.5317	0.5526	0.5641	0.5126	0.5672	0.5344	0.5932	0.5913

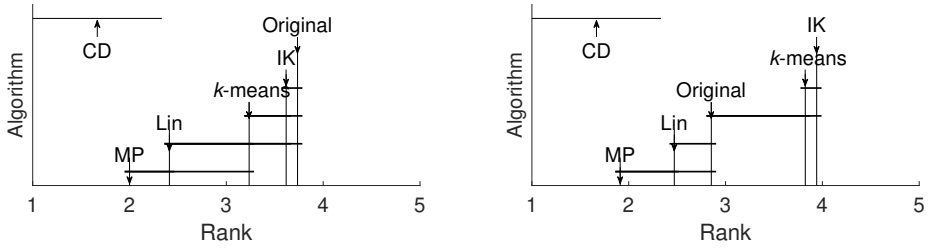


(a) DPC (b) SNNDPC

Figure 5: Post-hoc Nemenyi test ($\alpha = 0.10$) based on AMI scores shown in Table 4.

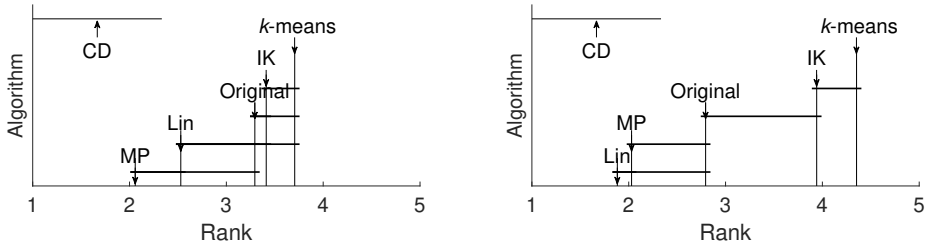
as they both consider the probability mass of the objects in each dimension.

Further, we conduct the Friedman test with the post-hoc Nemenyi test [39] to evaluate whether the performance difference between any two algorithms is significant based on the three metrics. For each metric, we show the results of significance test for DPC and SNNDPC clustering algorithms in Figures 5, 6, and 7. We note that two algorithms are significantly different if there is not a line linking them. The results show that the MP variants of the algorithms are better than at least two other clustering algorithms.



(a) DPC (b) SNNDPC

Figure 6: Post-hoc Nemenyi test ($\alpha = 0.10$) based on ARI scores in Table 4.



(a) DPC (b) SNNDPC

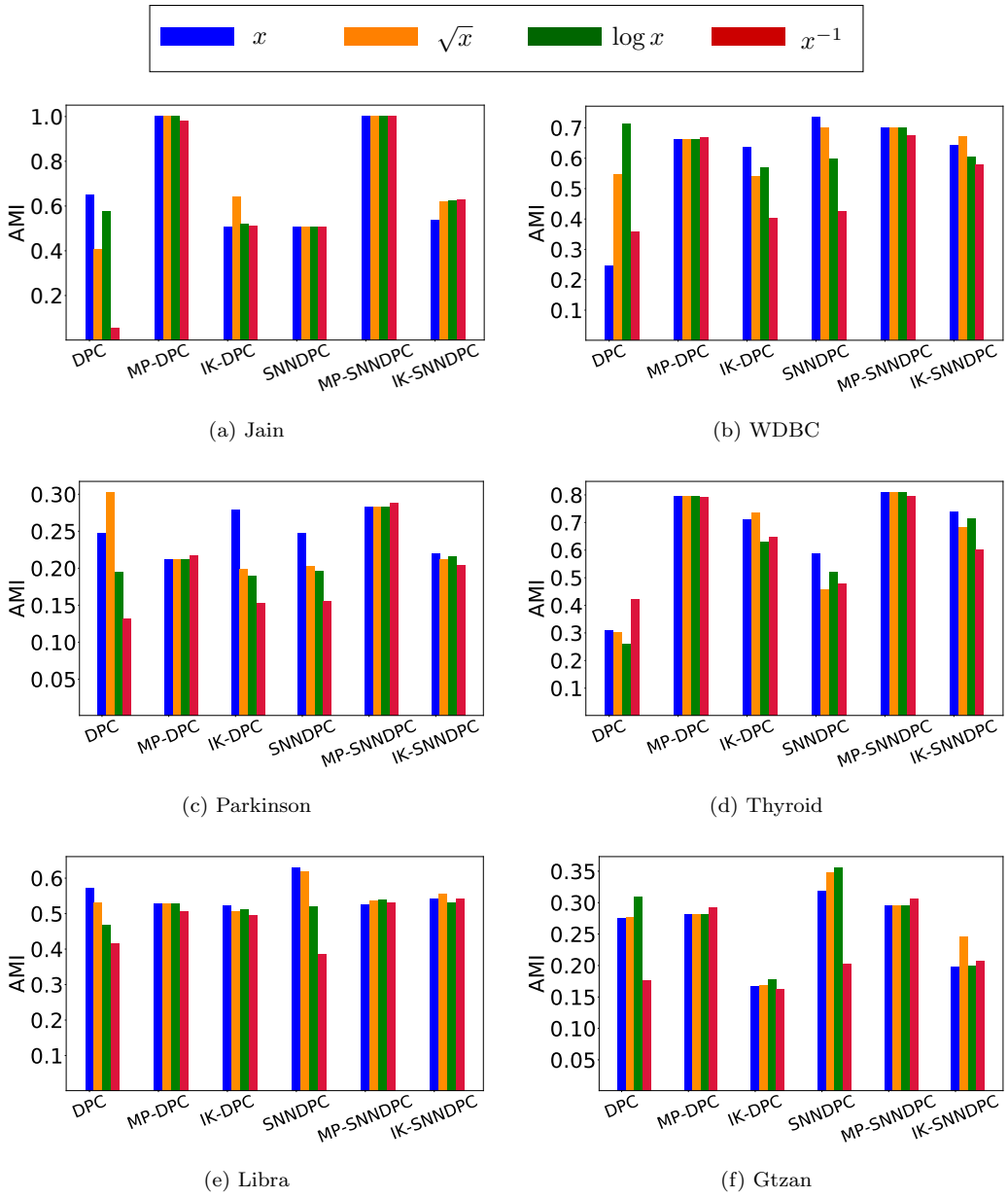
Figure 7: Post-hoc Nemenyi test ($\alpha = 0.10$) based on ARI scores in Table 5.

5.2.2. Sensitivity to Varying Feature Scales

In this section, we compare the performances of the MP variants of DPC and SNNDPC against the IK and Euclidean variants when the units/scales used to measure/represent the data features vary. As we have already shown that MP and Lin have similar performance, we do not include the results of Lin in this section. We use six datasets (Jain, WDBC, Parkinson, Thyroid, Libra and Gtzan) having different characteristics (size, dimensions, clusters and applications) for this experiment. We note that for the sake of brevity, we avoid presenting the results of the other datasets.

For the comparison, we transformed the data values of each feature x to: $\sqrt{x}$, $\log x$ and x^{-1}. Note that the transformations $\log x$ and $1/x$ are undefined for $x = 0$, Therefore, we apply the transformation to $c(x + \alpha)$, where $\alpha = 0.0001$ and $c = 100$ to obtain the results as done in [13]. The scaled data is again normalised in the range of [0,1].

Figure 8 shows the bar plots of the best AMI score achieved by contending algorithms in six different datasets. As can be observed from the plots of different datasets, the AMI scores of DPC, SNNDPC, IK-DPC and IK-SNNDPC varies greatly when the scale is changed. On the other hand, the results of MP-DPC and MP-SNNDPC are not affected on the $\sqrt{x}$ and $\log x$ scale. A slight variation is noticed in the *inverse* transformation for the MP variants of DPC and SNNDPC because of the binning involved in the preprocessing step where bin cut points falls in bins at different sides as inverse reverses the order. These results show that MP-DPC and MP-SNNDPC are robust to changes in the scale of the data. The above



(a) Jain (b) WDBC

(c) Parkinson (d) Thyroid

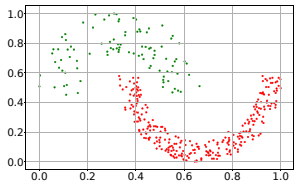
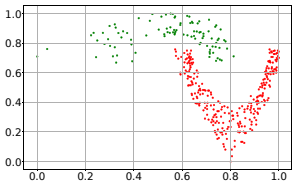
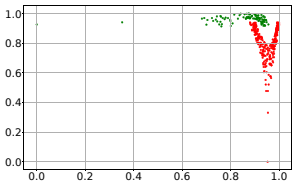
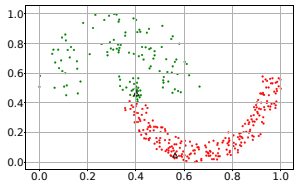
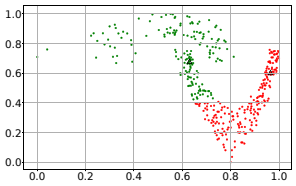
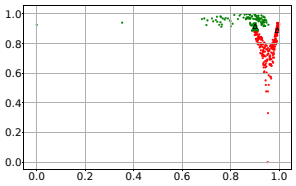
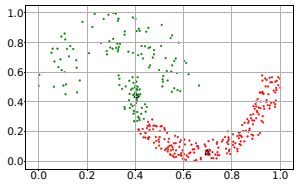
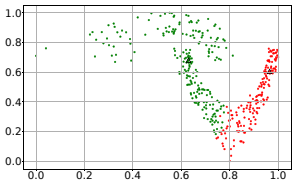
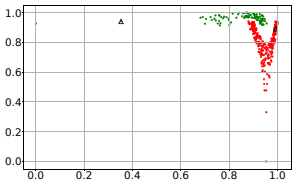
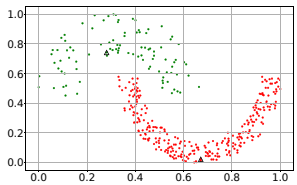
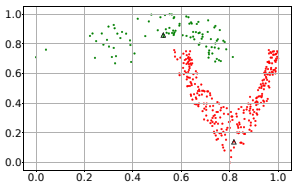
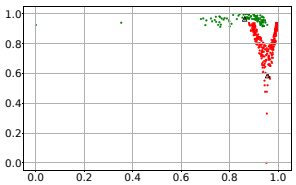
(e) Libra (f) Gtzan

Figure 8: Variation in AMI score of algorithms after scaling features as $\sqrt{x}$, $\log x$ and x^{-1}.

results have been previously hypothesized and shown in Section 4.

To further explain these results, we visualise the effect of varying scale on the results of the DPC, SNNDPC and MP-DPC using the Jain and Thyroid datasets. We plot the clusters obtained for these datasets using the above algorithms on the x, $\sqrt{x}$ and $\log x$ scale and check if there is a variation. Tables 6 and 7 show the plots of true clusters based on ground truth (in the first row) and clusters identified by DPC, SNNDPC and MP-DPC in the Jain and Thyroid datasets. To plot the results of the Thyroid dataset, we reduced its dimensionality to 2 using PCA. Note that we have not used all the algorithms (such as IK-based DPC or SNNPDC and MP-SNNDPC) or scales (x^{-1}), as we have already covered them in Figure 8.

Table 6: Clustering results with and without feature scaling in the Jain dataset.

	x	$\sqrt{x}$	$\log x$
Ground truth			
DPC			
SNNDPC			
MP-DPC			

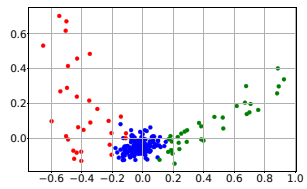
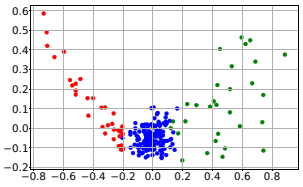
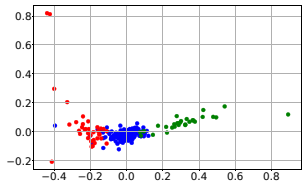
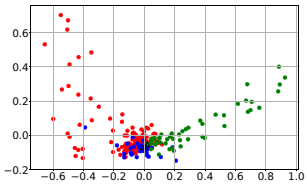
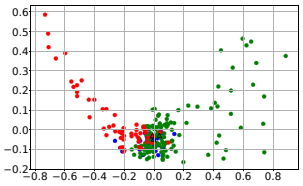
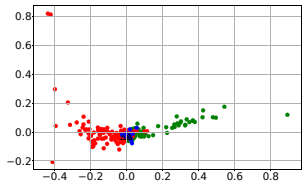
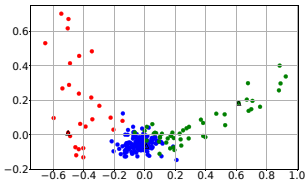
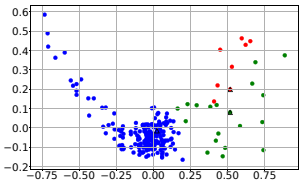
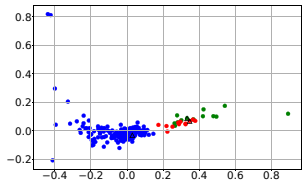
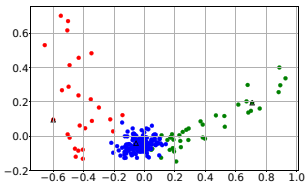
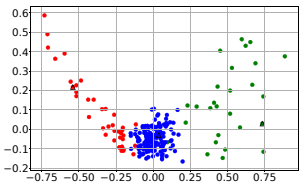
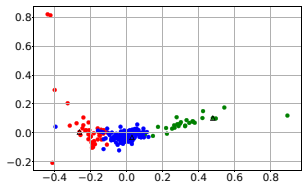
As observed in both Tables 6 and 7, MP-DPC produces clusters very similar to the ground truths in all cases, whereas those of DPC and SNNDPC vary significantly. In particular, DPC and SNNDPC results are worst on the Thyroid dataset with scaled features. These results of DPC and SNNDPC confirm our previous observations. The near perfect clustering results of MP-DPC are due to its ability to adapt to the local density distribution and robustness to the scaling of data features.

5.2.3. Sensitivity to parameter d_c

In this section, we analyse the sensitivity of the parameter d_c in DPC and MP-DPC. We used the Jain (synthetic) and Thyroid (real-world) datasets for this purpose and compare the effect of varying d_c on the clustering results of MP-DPC and DPC. Ten d_c values corresponding to different percentages in the range of 1% to 3% were chosen as shown in Figure 9. For the Jain dataset, MP-DPC showed perfectly consistent results for values of d_c from 2.2 to 2.6. DPC, on the other hand, had highly varying results in the set of selected values. Similarly for the Thyroid dataset, while MP-DPC achieved similar AMI score for many d_c values, the results of DPC varied for the different values of d_c as shown in the figure.

Overall, the results show that MP-DPC is less sensitive to the change in the value of d_c compared to

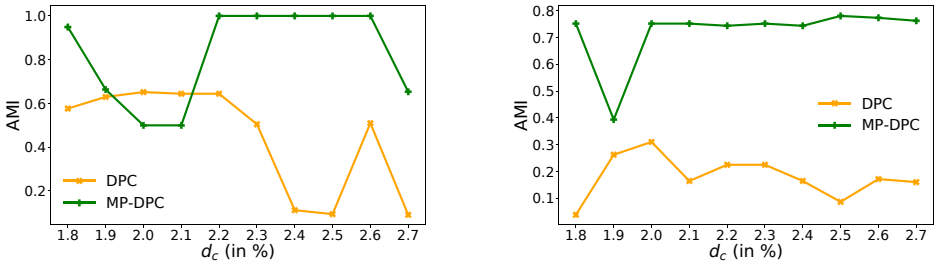
Table 7: Clustering results of the algorithms before and after feature scaling in the Thyroid dataset.

	x	$\sqrt{x}$	$\log x$
Ground truth			
DPC			
SNNDPC			
MP-DPC			

DPC with Euclidean distance.

6. Conclusion

In this paper, we focused on the popular Density Peak Clustering (DPC) algorithm, which is used in many applications. However, its inability to deal with varying density clusters and sensitivity to the representation of data limit its effectiveness for real-world problems where: (i) data have complex structure with varying density clusters; and (ii) how data features are expressed/represented may not be known. We overcome these two limitations of DPC by introducing a new data-dependent and scale-invariant similarity measure, which we call MP-Similarity. We show that MP-Similarity when incorporated into DPC (i) improves its performance on a range of datasets with varying characteristics, and (ii) provides consistent results even when the data is represented using different scales. Similar improvement in performance is also observed when MP-Similarity is used with SNNDPC algorithm, a variant of DPC algorithm, thus demonstrating the advantage of using MP-Similarity. Further, we show that another data-dependent and scale-invariant similarity measure (known as Lin's measure) also improves the performance of DPC and SNNDPC algorithm. This demonstrates the



(a) Jain dataset.

(b) Thyroid Dataset.

Figure 9: Effect of varying d_c.

effectiveness of such measures in clustering complex real-world data. The experimental results using the popular clustering metrics validate our claim.

Although MP-Similarity produces better results, it has a few limitations. Because MP-Similarity is based on m_0-dissimilarity, which assumes that the attributes are independent and computes the dissimilarity in each dimension separately, it may perform poorly when there is a strong correlation between the attributes. In such cases, measures such as IK may perform better. However, noting from the experimental insights, MP-DPC provides better performance compared to other contenders across various datasets of different dimensions and sizes, which shows that such effect may not be significant in practice.

Future work includes investigating the performance of MP-Similarity on other popular clustering algorithms such as DBSCAN and hierarchical clustering. These algorithms also use (dis)similarity measures based on data-independent Euclidean distance. We also plan to extend this work on mixed data consisting of both numerical and categorical attributes as these are common for many real-world datasets. It would also be of interest to study the behaviour of the proposed similarity in the context of classification and anomaly detection.

Acknowledgement

This research is funded by the US Air Force Office of Scientific Research (AFOSR) and Office of Naval Research (ONR) Global under grant number FA2386-20-1-4005.

References

[1] E. Diday, G. Govaert, Y. Lechevallier, J. Sidi, Clustering in pattern recognition, in: J. C. Simon, R. M. Haralick (Eds.), Digital Image Processing, Springer Netherlands, Dordrecht, 1981, pp. 19–58.

[2] Q. Zou, G. Lin, X. Jiang, X. Liu, X. Zeng, Sequence clustering in bioinformatics: an empirical study, Briefings in bioinformatics 21 (1) (2020) 1–10.

[3] J. Hou, W. Liu, E. Xu, H. Cui, Towards parameter-independent data clustering and image segmentation, Pattern Recognition 60 (2016) 25–36.

[4] M. R. Bouadjenek, S. Sanner, Y. Du, Relevance-and interface-driven clustering for visual information retrieval, Information Systems 94 (2020) 101592.

[5] A. Rodriguez, A. Laio, Clustering by fast search and find of density peaks, science 344 (6191) (2014) 1492–1496.

[6] D. Kobak, W. Brendel, C. Constantinidis, C. E. Feierstein, A. Kepecs, Z. F. Mainen, X.-L. Qi, R. Romo, N. Uchida, C. K. Machens, Demixed principal component analysis of neural population data, Elife 5 (2016) e10989.

[7] K. Sun, X. Geng, L. Ji, Exemplar component analysis: A fast band selection method for hyperspectral imagery, IEEE Geoscience and Remote Sensing Letters 12 (5) (2014) 998–1002.

[8] S. Zamuner, A. Rodriguez, F. Seno, A. Trovato, An efficient algorithm to perform local concerted movements of a chain molecule, PloS one 10 (3) (2015) e0118342.

[9] W. Wang, J. Shen, F. Porikli, R. Yang, Semi-supervised video object segmentation with super-trajectories, IEEE transactions on pattern analysis and machine intelligence 41 (4) (2018) 985–998.

[10] T. Anwar, C. Liu, H. L. Vu, C. Leckie, Partitioning road networks using density peak graphs: Efficiency vs. accuracy, Information Systems 64 (2017) 22–40.

[11] K. M. Dean, L. M. Davis, J. L. Lubbeck, P. Manna, P. Friis, A. E. Palmer, R. Jimenez, High-speed multiparameter photophysical analyses of fluorophore libraries, Analytical chemistry 87 (10) (2015) 5026–5030.

[12] Y. Zhang, Y. Xia, Y. Liu, W. Wang, Clustering sentences with density peaks for multi-document summarization, in: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2015, pp. 1262–1267.

[13] S. Aryal, K. M. Ting, T. Washio, G. Haffari, A comparative study of data-dependent approaches without learning in measuring similarities of data objects, Data mining and knowledge discovery 34 (1) (2020) 124–162.

[14] R. Liu, H. Wang, X. Yu, Shared-nearest-neighbor-based clustering by fast search and find of density peaks, Information Sciences 450 (2018) 200–226.

[15] D. Xu, Y. Tian, A comprehensive survey of clustering algorithms, Annals of Data Science 2 (2) (2015) 165–193.

[16] S. Lloyd, Least squares quantization in pcm, IEEE transactions on information theory 28 (2) (1982) 129–137.

[17] P. H. Sneath, R. R. Sokal, et al., Numerical taxonomy, Nature 193 (4818) (1962) 855–860.

[18] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: In Proceedings of KDD, 1996, pp. 226–231.

[19] G. J. McLachlan, T. Krishnan, The EM algorithm and extensions, Vol. 382, John Wiley & Sons, 2007.

[20] W. Wang, J. Yang, R. Muntz, et al., Sting: A statistical information grid approach to spatial data mining, in: VLDB, Vol. 97, Citeseer, 1997, pp. 186–195.

[21] J. Xie, H. Gao, W. Xie, X. Liu, P. W. Grant, Robust clustering by detecting density peaks and assigning points based on fuzzy weighted k-nearest neighbors, Information Sciences 354 (2016) 19–40.

[22] M. Du, S. Ding, H. Jia, Study on density peaks clustering based on k-nearest neighbors and principal component analysis, Knowledge-Based Systems 99 (2016) 135–145.

[23] J. Hou, A. Zhang, N. Qi, Density peak clustering based on relative density relationship, Pattern Recognition 108 (2020) 107554.

[24] Y. Wang, D. Wang, X. Zhang, W. Pang, C. Miao, A.-H. Tan, Y. Zhou, Mcdpc: multi-center density peak clustering, Neural Computing and Applications 32 (17) (2020) 13465–13478.

[25] A. Lotfi, P. Moradi, H. Beigy, Density peaks clustering based on density backbone and fuzzy neighborhood, Pattern Recognition 107 (2020) 107449.

[26] M. Abbas, A. El-Zoghabi, A. Shoukry, Denmune: Density peak based clustering using mutual nearest neighbors, Pattern Recognition 109 (2021) 107589.

[27] L. Bai, X. Cheng, J. Liang, H. Shen, Y. Guo, Fast density clustering strategies based on the k-means algorithm, Pattern Recognition 71 (2017) 375–386.

[28] Z. Rasool, R. Zhou, L. Chen, C. Liu, J. Xu, Index-based solutions for efficient density peak clustering, IEEE Transactions on Knowledge and Data Engineering (2020).

[29] A. Tversky, Features of similarity., Psychological review 84 (4) (1977) 327.

[30] C. L. Krumhansl, Concerning the applicability of geometric models to similarity data: The interrelationship between similarity and spatial density., Psychological Review (1978).

[31] K. M. Ting, Y. Zhu, Z.-H. Zhou, Isolation kernel and its effect on svm, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 2329–2337.

[32] D. Lin, An information-theoretic definition of similarity, in: Proceedings of the Fifteenth International Conference on Machine Learning, ICML '98, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998, pp. 296–304.

[33] F. T. Liu, K. M. Ting, Z.-H. Zhou, Isolation forest, in: 2008 eighth ieee international conference on data mining, IEEE, 2008, pp. 413–422.

[34] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research 12 (2011) 2825–2830.

[35] D. Arthur, S. Vassilvitskii, k-means++: The advantages of careful seeding, Tech. rep., Stanford (2006).

[36] D. Dua, C. Graff, UCI machine learning repository, http://archive.ics.uci.edu/ml (2019).

[37] P. Fränti, M. Rezaei, Q. Zhao, Centroid index: cluster level similarity measure, Pattern Recognition 47 (9) (2014) 3034–3045.

[38] E. B. Fowlkes, C. L. Mallows, A method for comparing two hierarchical clusterings, Journal of the American statistical association 78 (383) (1983) 553–569.

[39] J. Demšar, Statistical comparisons of classifiers over multiple data sets, The Journal of Machine learning research 7 (2006) 1–30.

Author Biographies

Zafaryab Rasool, PhD: Zafaryab is an Associate Research Fellow in the School of Information Technology, Deakin University, Australia. He completed his PhD in Computer Science from the Swinburne University of Technology, Australia in 2021. Previously, he earned his Master's degree from the Jamia Millia Islamia University, India in 2016 and Bachelor's degree from the Aligarh Muslim University, India in 2013. His research interest is in the areas of Data Mining, Machine Learning and Database, particularly focused on the clustering techniques.

Sunil Aryal, PhD: Dr Aryal is a Senior Lecturer in the School of Information Technology, Deakin University, Australia. Prior to joining Deakin in 2019, he worked as a lecturer at Federation University and worked in industry as a software developer and data analyst. He received his PhD from Monash University, Australia in 2017. His research interests are in the areas of Data Mining (DM), Machine Learning (ML), and Artificial Intelligence (AI), particularly in their applications to solve real-world problems. He has published more than 40 scientific papers in top tier DM/ML conferences/journals. His research is supported by US and Australian Defence and Intelligence agencies. He has been an investigator on research grants/contracts with funding over $2.3 millions.

Mohamed Reda Bouadjenek, PhD: Reda is a Lecturer in the School of Information Technology at Deakin University, Geelong, Australia. Previously, he was a Research Fellow at The University of Toronto (2017-2019) and at The University of Melbourne (2015-2017) and before that, he was a postdoc researcher

at INRIA France (2014-2015). Reda earned a Ph.D. and an MSc in Computer Science from the University of Paris-Saclay France respectively in 2013 and 2009, and a BSc in Computer Science from USTHB Algeria in 2008. His research spans a broad range of topics related to the data-driven fields of Machine Learning, Deep Learning, and Information Retrieval. He has applied analytic and algorithmic tools from these fields to solve real-world problems related to diverse applications such as recommender systems, interactive visual search interfaces, social network analysis, and data quality.

Richard Dazeley, PhD: Dr Dazeley is an Associate Professor of Computer Science at Deakin University (Geelong). He is a highly experienced researcher in multiobjective, interactive, deep, safe and explainable Reinforcement Learning (RL). He has published several of the most widely cited papers in multiobjective RL where his work significantly contributed to the foundation of this field of research. He been recognized as a nationally leading ICT Curriculum Designer - receiving the Australian National Award as ICT Educator of the Year (2016) from the ACS and was nominated for the International ICT Educator of the Year award from the South-East Asia Regional Computer Confederation (SEARCC) (2017). Prior to working at Deakin, he was employed at Federation University Australia where he was the Head of the IT Discipline and the co-Founder and Leader of the Federation Learning Agents Research Group (FLAG).

Data-dependent and scale-invariant kernel for Support Vector Machine classification

Vinayaka Vivekananda Malgi*, Sunil Aryal*, Zafaryab Rasool, and David Tay

Deakin University, Geelong, Waurn Ponds, VIC 3216, Australia

Abstract. Kernel similarity function allows a Support Vector Machine (SVM) classifier to learn the maximum margin hyperplane in a higher dimensional space where two classes are linearly separable without explicitly mapping the data. Most existing kernel functions (e.g., RBF) use spatial positions of two data instances in the input space to compute their similarity. These kernels are data distribution independent and sensitive to data representation (i.e., units/scales used to measure/express data). Since this can be unknown in many real-world applications, a careful selection of a suitable kernel is required for a given problem. In this paper, we present a new kernel function based on probability data mass that is both data-dependent and scale-invariant. Our empirical results show that the proposed SVM kernel outperforms popular existing kernels.

Keywords: SVM classification · Kernel functions · Data-dependent kernel · Scale-invariant kernel · Information-theoretic similarity.

1 Introduction

Support Vector Machine (SVM) classifier learns the separating hyperplane that maximises the margin between data points belonging to different classes [3]. Because data in many real-world applications have complex structures and classes are often not linearly separable in the input space, the idea of '*kernel trick*' allows learning the maximum margin hyperplane in a higher dimensional space where the classes are linearly separable without explicitly projecting the data. To achieve this, it requires special type of measures/functions to compute pairwise similarity of data. They are called kernel similarity functions, often referred to as 'kernels' in the SVM classification context. Most commonly used kernels such as Radial Basis Function (RBF) and Laplacian use spatial distance of two points in the input space to compute their kernel similarity. As discussed by Aryal et al. (2020) [2], such a distance-based notion of similarity may not be effective in real-world problems as it is:

1. *Data-independent*: The similarity of two instances solely depends on their spatial positions and it is not affected by the distribution of other data. Unit distance between two points has the same degree of similarity everywhere

* Corresponding authors and they contributed equally to this work, Email: vinayaka.malgi@outlook.com and sunil.aryal@deakin.edu.au

in the space regardless of the density distribution. Psychologists [6] have argued that the two points in a sparse region are considered more similar than the other two points with the same geometric distance but located in dense regions. For example, two Caucasian persons are judged as less similar in Europe than in Asia because there are many Caucasian people in Europe compared to those in Asia, i.e., density is higher.

2. *Sensitive to data representation*: Because geometric model relies on the spatial positions of data points in the input space, they are sensitive to how data are represented/expressed. The distance between the two points can change significantly if the same data is represented/expressed differently by non-linear scaling. In real-world applications, data come from various sources and can be measured/expressed in different forms. For example, sample variability can be measured as standard deviation (σ) or variance (σ^2) and credit risk of customers can be measured as Income-to-Debt ratio or Debt-to-Income ratio. When data are given for analysis, mostly we are given only data values (numbers) and we may not know how they are represented, let alone the most appropriate representation.

These are the reasons as to why a kernel function that works well in one problem or dataset does not work well for others. Thus, kernel function has to be selected carefully for a given problem.

Recently, some data-dependent (dis)similarity measures [1, 5, 9, 2] have been proposed. Among them, only the m_p-dissimilarity [2] is fully data-dependent and invariant to the change in data representation. It is data-dependent because the similarity of two objects in each attribute/feature is estimated as the probability data mass between them. Two instances in dense regions are less similar or more dissimilar to each other than the two instances with the same spatial distance but located in sparse (low-density) regions. It captures the essence of the human perceptual notion of similarity suggested by psychologists. It is robust to data representation as it does not use feature values directly in the similarity calculations, it just uses the number of data points between the two data points under consideration in each feature. As the ranking is preserved or reversed even in the case of non-linear scaling of data, data mass between any two points is not changed. It has been shown to produce better and more consistent results across a wide range of datasets in the k-nearest neighbors (k-NN) classification and content-based information retrieval problems (CBIR) [2]. However, in its current form, it cannot be used as a kernel function in the SVM classification because the self-similarity of data instances is not constant.

In this paper, we extend a variant of m_p-dissimilarity, namely m_0-dissimilarity, into a kernel function and use it in the SVM classification framework. We call the new kernel function as **P**robability **M**ass-based **K**ernel (PMK). Like m_0-dissimilarity, PMK is both data-dependent and scale-invariant. Our empirical results show that PMK consistently outperforms widely used state-of-the-art (SoTA) data-dependent and data-independent kernel functions, making it the optimal choice kernel function in practical real-world problems, particularly

in domains where data are captured from various sources using different devices/sensors.

2 PMK: Probability Mass-based Kernel

2.1 Notations and preliminaries

Let D be a collection of N labelled training instances, where each instance $\mathbf{x} = \langle x_1, x_2, \cdots, x_M \rangle$ is represented by an M-dimensional vector of its values of the M selected features; and L be a N-dimensional vector of their class labels. In this paper, we focus on numeric data, i.e., each $x_i \in \mathbb{R}$ ($\mathbb{R}$ is a real domain). In the training process, a SVM classifier learns the maximum margin hyperplane separating different classes using the similarity matrix of all $(\mathbf{x}, \mathbf{y})$ pairs in D based on a kernel function $K(\mathbf{x}, \mathbf{y})$ and label vector L. In the testing phase, to predict a class label of an unseen test instance $\mathbf{q}$, the vector of its kernel similarities with training instances in D is used. Thus, the choice of kernel function $K(\mathbf{x}, \mathbf{y})$ is central to learning a good SVM classifier.

2.2 m_0-dissimilarity

The m_0-dissimilarity of $\mathbf{x}$ and $\mathbf{y}$ is estimated as [2]:

$$m_0(\mathbf{x}, \mathbf{y}) = \frac{1}{M} \sum_{i=1}^{M} \log\left(\frac{|R_i(\mathbf{x}, \mathbf{y})|}{N}\right) \tag{1}$$

where, $R_i(\mathbf{x}, \mathbf{y}) = [\min(x_i, y_i), \max(x_i, y_i)]$ is a region covering $\mathbf{x}$ and $\mathbf{y}$ in dimension i; and $|R_i(\mathbf{x}, \mathbf{y})| = |\{\mathbf{z} \in D : \min(x_i, y_i) \leq z_i \leq \max(x_i, y_i)\}|$, where $|\cdot|$ represents cardinality.

It is data-dependent because $|R_i(\mathbf{x}, \mathbf{y})|$ depends on how many other instances fall between x_i and y_i. It is scale-invariant because the number of data points falling between x_i and y_i does not change due to linear or non-linear scaling of data as the ranking is either preserved or reversed. Because, (i) $|R_i(\mathbf{x}, \mathbf{x})|$ and $|R_i(\mathbf{y}, \mathbf{y})|$, and (ii) $|R_i(\mathbf{x}, \mathbf{x})|$ and $|R_j(\mathbf{x}, \mathbf{x})|$, can be different depending on the probability masses at x_i, y_i, and x_j; the self-dissimilarity of data instances based on m_0-dissimilarity are not constant, i.e., $m_0(\mathbf{x}, \mathbf{x})$ and $m_0(\mathbf{y}, \mathbf{y})$ can be different. However, self-dissimilarity is minimal for any instance $\mathbf{x}$, i.e., $m_0(\mathbf{x}, \mathbf{x}) \leq m_0(\mathbf{x}, \mathbf{y})$ for $\forall \mathbf{y} \neq \mathbf{x}$. Because of non-constant self-(dis)similarity, it cannot be used as a kernel function in SVM.

It is computationally expensive to compute $|R_i(\mathbf{x}, \mathbf{y})|$ as it requires a range search, especially in the case where either or both of them are unseen and N is large. However, as suggested in [2], it can be approximated quickly by converting a continuous-valued domain in each dimension i into an ordinal discrete domain by discretizing the range of data values into $b \ll N$ intervals/bins $(h_{i,1}, h_{i,2}, \cdots, h_{i,b})$. By storing the frequency of each bin from D in the preprocessing step, $|R_i(\mathbf{x}, \mathbf{y})|$ can be approximated quickly based on the frequencies

of the bins where $\mathbf{x}$ and $\mathbf{y}$ falls and bins in between as $|R_i(\mathbf{x},\mathbf{y})| = \sum_{a=l}^{u} |h_{i,a}|$, where $h_{i,l}$ and $h_{i,u}$ are the bins in which $\min(x_i, y_i)$ and $\max(x_i, y_i)$ fall. In each dimension i, data frequencies between all pairs of bins can be pre-computed and stored as a $b \times b$ matrix. Then, $|R_i(\mathbf{x},\mathbf{y})|$ can be computed as a table look-up by finding bins where they fall. Because Equal-Width Discretisation (EWD), where bins are of the same width, is sensitive to unit/scales of data and outliers, Equal-Frequency Discretisation (EFD), where bins have the same frequency where possible, is used. As it may not be possible in practice to have the same frequency in each bin because of duplicate values, bins may have different frequencies.

2.3 Proposed new kernel function

Based on m_0-dissimilarity (Eqn. 1), we define a new **P**robability **M**ass-based **K**ernel similarity function, referred to as 'PMK' in short, as:

$$K_{PMK}(\mathbf{x},\mathbf{y}) = \frac{2 * m_0(\mathbf{x},\mathbf{y})}{m_0(\mathbf{x},\mathbf{x}) + m_0(\mathbf{y},\mathbf{y})} \tag{2}$$

Unlike $m_0(\mathbf{x},\mathbf{y})$ which is a measure of dissimilarity, $K_{PMK}(\mathbf{x},\mathbf{y})$ is a similarity of $\mathbf{x}$ and $\mathbf{y}$ and it is in the range of [0, 1]. The self-similarity of data instances is always maximal and the constant of 1. Now it can be used as a kernel function to learn a SVM classifier. Like $m_0(\mathbf{x},\mathbf{y})$, it is both data-dependent and invariant to units/scales of data.

The proposed kernel function has a probabilistic interpretation. It can be viewed as the multi-dimensional extension of Lin's Information Theoretical Similarity Measure [7] for ordinal data, where the similarity of two one-dimensional ordinal values x and y is estimated as:

$$s_{lin}(x,y) = \frac{2 \times \log \sum_{z=\min(x,y)}^{\max(x,y)} P(z)}{\log P(x) + \log P(y)} \tag{3}$$

where, $P(x)$ is the probability of x and it is estimated using the frequency of value x, $f(x)$, as $P(x) = f(x)/N$; and $\sum_{z=\min(x,y)}^{\max(x,y)} P(z)$ is the probability mass in the range between and including x and y, $R(x,y)$.

In the literature, the similarity of two instance $\mathbf{x}$ and $\mathbf{y}$ in a multidimensional space using Lin's approach is estimated by aggregating their similarities in each dimension based on Eqn. 3:

$$s_{lin}(\mathbf{x},\mathbf{y}) = \frac{1}{M} \sum_{i=1}^{M} \frac{2 \times \log \sum_{z_i=\min(x_i,y_i)}^{\max(x_i,y_i)} P(z_i)}{\log P(x_i) + \log P(y_i)} \tag{4}$$

Aryal et al. (2020) [2] used Lin's similarity measure defined in Eqn. 4 in multidimensional continuous spaces using the same discretisation as done for m_0-dissimilarity discussed above. The similarity of $\mathbf{x}$ and $\mathbf{y}$ in each dimension i is estimated based on Eqn. 3 using probability masses in one-dimensional regions $R_i(\mathbf{x},\mathbf{y})$, $R_i(\mathbf{x},\mathbf{x})$, and $R_i(\mathbf{y},\mathbf{y})$. Lin's measure is applied in each dimension

Table 1: An example of data distribution in two dimensions of a multi-dimensional dataset [2]

Dim.	$Inst1$	$Inst2$	$Inst3$	$Inst4$	$Inst5$	$Inst6$	$Inst7$	$Inst8$	$Inst9$	$Inst10$
.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.
i	2	2	1	1	1	1	1	1	1	1
j	2	2	2	2	2	2	2	2	1	1
.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.

separately and the one-dimensional similarities are aggregated to compute the final similarity in the multidimensional space. A more natural extension of Lin's approach in multidimensional space would be to define an M-dimensional region $R(\mathbf{x}, \mathbf{y})$ that encloses $\mathbf{x}$ and $\mathbf{y}$ which has the length of $R_i(\mathbf{x}, \mathbf{y})$ in each dimension i, and estimate the probability mass in the region. The similarity of $\mathbf{x}$ and $\mathbf{y}$ can be estimated as:

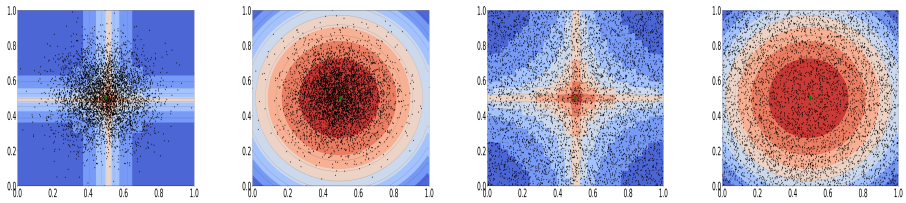
$$s'_{lin}(\mathbf{x}, \mathbf{y}) = \frac{2 \times \log P(R(\mathbf{x}, \mathbf{y}))}{\log P(R(\mathbf{x}, \mathbf{x})) + \log P(R(\mathbf{y}, \mathbf{y}))} \tag{5}$$

To have a reasonable estimate of $P(R(\mathbf{x}, \mathbf{y}))$ in a high-dimensional space, a large amount of data is required. It is not realistic in many application domains. However, we can get a good approximation of it with Naive Bayesian assumption that dimensions are independent as $P(R(\mathbf{x}, \mathbf{y})) \approx \prod_{i=1}^{M} P(R_i(\mathbf{x}, \mathbf{y}))$ and Eqn. 5 can be written as:

$$s'_{lin}(\mathbf{x}, \mathbf{y}) = \frac{2 \times \log \prod_{i=1}^{M} P(R_i(\mathbf{x}, \mathbf{y}))}{\log \prod_{i=1}^{M} P(R_i(\mathbf{x}, \mathbf{x})) + \log \prod_{i=1}^{M} P(R_i(\mathbf{y}, \mathbf{y}))} \tag{6}$$

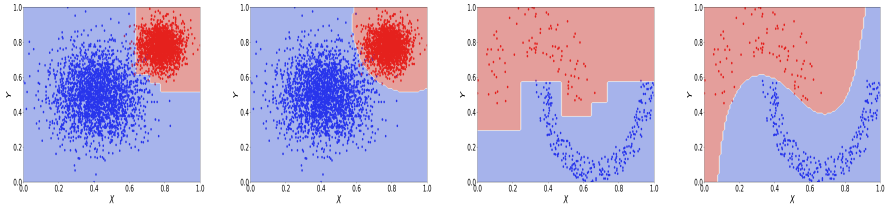
Replacing the log of products with the sum of logs, $P(R_i(\mathbf{x}, \mathbf{y})) = |R_i(\mathbf{x}, \mathbf{y})|/N$ and using Eqn. 1, Eqn. 6 results in the PMK defined in Eqn. 2.

One fundamental difference between PMK and the traditional Lin's approach (Eqn.4) is that the similarity of $\mathbf{x}$ and $\mathbf{y}$ in dimensions where $x_i = y_i$ is always 1 in the latter irrespective of $P(x_i)$, whereas that is not the case in PMK as it considers the $\log P(x_i)$ in the calculation. Considering $P(x_i)$ can be useful because sharing rare values can provide more information about the similarity of $\mathbf{x}$ and $\mathbf{y}$ than sharing a very frequent value. To understand this, let's look at an example dataset shown in Table 1 [2], $Inst1$ and $Inst2$ have the same values in dimensions i and j, but their value in dimension i is less common (has lower probability) than their value in dimension j. In the case of Lin's approach as used in the literature, their similarities in dimensions i and j contribute equally to the overall similarity. But, as suggested by psychologists, they provide different amount of information about the similarity of $Inst1$ and $Inst2$. Common rare value in dimension i provides more information compared to sharing frequent value in dimension j. Many instances can have the same value in many dimensions in high-dimensional problems as data often lies in a low-dimensional



(a) Normal-PMK (b) Normal-RBF (c) Uniform-PMK (d) Uniform-RBF

Fig. 1: Contour plots of kernel similarities of points in a 2-dimensional space with the centre (0.5, 0.5) under Normal and Uniform data distributions.



(a) 2Modes-PMK (b) 2Modes-RBF (c) Jain-PMK (d) Jain-RBF

Fig. 2: Decision boundaries of SVM classifiers using PMK and RBF kernel in two 2-dimensional two-class synthetic datasets with different class densities.

manifold. Therefore, measure like PMK that considers $P(x_i)$ can perform better than those which do not.

To demonstrate the difference between data-dependent and data-independent kernel similarity, we present the contour plots of kernel similarities of points in a 2-dimensional space to the centre (0.5, 0.5) using PMK and RBF kernels given two datasets of 5000 points generated from normal and uniform distributions in Fig. 1. Dark orange represents the regions of high similarity, while dark blue represents the region of low similarity. It shows the data-dependent behaviour of PMK as it adapts the contours to underlying data distribution. But, the most widely used RBF kernel produces exactly the same contour regardless of underlying data distribution.

To show the effect of PMK in SVM classification, we present decision boundaries learned by SVM using PMK and RBF kernels in two 2-dimensional two-class synthetic datasets with classes of different density distribution in Fig. 2. In both datasets, the decision boundaries of RBF are pushed towards sparse classes. As a result, it maximises correct predictions for the dense classes, more samples from sparse classes are misclassified. However, in the case of PMK, the decision boundaries are pushed towards dense classes in both cases. This result

suggests that PMK can better differentiate classes with varying densities rather than favouring dense class.

2.4 Positive Definiteness of the PMK kernel

The basic notion in non-linear SVM is the existence of a mapping of the input pattern into a higher dimensional space, that more readily allow for a shattering (separation) of the different classes. In practice the mapping need not be explicitly defined, but is implicit via the kernel function. For a given kernel function $K(\mathbf{x}, \mathbf{y})$ to correspond to a higher dimensional mapping, it must be positive definite and satisfies the Mercers condition, i.e. for any arbitrary function $f(\mathbf{x})$, the following condition must be satisfied

$$\int d\mathbf{x} \int d\mathbf{y}\ f(\mathbf{x})K(\mathbf{x}, \mathbf{y})f(\mathbf{y}) \geq 0 \tag{7}$$

Unless the kernel function is relatively simple, e.g. polynomial, Gaussian, it is not analytically tractable to show if (7) is satisfied or not. This is certainly the case with our proposed PMK kernel, which is dependent on the data distribution. We adopted an experimental approach to show that the kernel is positive definite by checking the eigenvalues of the pairwise kernel similarity matrix (aka Gram matrix). We randomly generated datasets with 5000 points from three types of distributions: (i) normal; (ii) uniform; and (ii) a mixture of five Gaussians. We considered spaces with two, five and ten dimensions. For each distribution and dimensionality, we generated three different datasets resulting in 27 synthetic datasets. We also tested for Gram matrices of real-world datasets as shown in Table 2. In all cases, with both synthetic and real-world datasets, eigenvalues are non-negative.

3 Empirical Evaluation

We used 21 real-world datasets sourced from the UCI Machine Learning Repository [4] having varying sizes (from 699 to 43680), varying numbers of classes (from 2 to 100) and varying numbers of attributes (10 to 6826). The properties of these datasets are provided in the first four columns of Table 2. For each dataset, a 10-fold cross-validation was conducted and reported the average accuracy over 10 folds. Support Vector Classifier (SVC) of the Scikit-Learn Machine Learning Library [8] was used as an SVM classifier.

3.1 Comparison of SVM with PMK and other kernels

We compared PMK with five state-of-the-art (SoTA) data-independent and data-dependent kernel functions: (i) Isolation Kernel (IK); (ii) Radial Basis Function (RBF); (iii) Laplacian (Lap.); (iv) Lin's measure as defined in the literature by aggregating Lin's similarity in each dimension (Lin); and (v) Laplacian kernel on the rank transformation of data (Lap.R). Lap.R is considered because

Table 2: Average classification accuracy of SVM classifier over a 10-fold cross-validation. The best average accuracy and best average rank are boldfaced. N:#Instances, M: #Dimensions, and K: #Classes in a dataset.

Data[Ref]	N	M	K	PMK	IK	RBF	Lap.	Lin	Lap.R
Breast Cancer[4]	699	10	2	**0.800**	0.727	0.799	0.768	0.675	0.767
Gtzan[2]	1000	230	10	0.782	0.697	**0.799**	0.783	0.650	0.774
Hba [2]	1500	187	15	**0.780**	0.665	0.720	0.761	0.710	0.765
Steel Plate[4]	1941	27	7	0.751	0.777	0.745	**0.778**	0.724	0.717
Rejafada[4]	1996	6826	2	**0.982**	0.958	0.956	0.966	0.952	0.954
Mfeat[4]	2000	649	10	**0.990**	0.980	0.986	0.983	0.927	0.937
Cardio[4]	2126	23	3	0.993	0.992	**0.995**	0.989	0.779	0.925
Hydraulic[4]	2205	43680	4	**1.000**	0.996	0.723	0.554	0.954	0.993
Segment[4]	2310	19	7	0.980	0.976	0.977	**0.986**	0.959	0.820
Fbis[2]	2463	2000	17	**0.890**	0.690	0.800	0.849	0.670	0.660
Madelon[4]	2600	500	2	**0.625**	0.555	0.590	0.606	0.520	0.622
Malware[4]	2955	1087	4	**0.995**	0.985	0.978	0.993	0.962	0.994
Page Blocks[4]	5473	10	5	**0.987**	0.977	0.971	0.974	0.900	0.888
First Order[4]	6118	51	5	0.565	**0.580**	0.562	0.535	0.425	0.520
Satimage[4]	6435	36	7	0.926	0.907	0.917	**0.927**	0.900	0.888
Musk[4]	6598	166	2	**0.998**	0.990	0.994	0.992	0.846	0.941
Taiwan Bank[4]	6819	96	2	0.967	0.969	0.970	**0.972**	0.934	0.950
Isolet[4]	7797	617	26	0.970	**0.977**	0.975	0.572	0.787	0.962
Corel[2]	10000	67	100	**0.503**	0.415	0.400	0.492	0.494	0.150
Ismis[4]	12495	191	6	0.938	0.921	**0.974**	0.972	0.939	0.221
Gas sensory[4]	13910	128	6	0.994	**0.996**	0.993	0.995	0.991	0.976
		Avg. Acc.		**0.876**	0.841	0.847	0.830	0.795	0.783
		Avg. Rank.		**1.799**	3.427	3.094	2.714	5.189	4.713

using kernels on the rank transformation of data is the simplest way of making them invariant to the change in data representation. As rank transformation of data can be computationally expensive, we discretised data as done in PMK and used bin ranks. Among the contending kernels: (i) Lin and Lap.R are both data-dependent and robust; (ii) IK is data-dependent but not robust; and (iii) RBF and Lap are neither data-dependent nor robust. It is interesting to note that though Lin and Lap.R are data-distribution dependent when $x_i \neq y_i$, they do not consider data distribution in the case of $x_i = y_i$. In a way, they are partially data-dependent.

The SVM cost parameter 'C' and parameters of kernel functions are tuned in each train-test fold through five-fold cross-validation: C in $\{0.01, 0.1, 10, 100\}$; Number of Bins (b) for PMK, Lin and Lap.R in $\{25, 50, 75, 100, (\log_2 N + 1)\}$; Subsample size ($\psi$) for IK in $\{2^m | m = 2, 3, 4, 5, 6, 7, 8\}$; and γ for RBF, Lap and Lap.R in $\{0.01, 0.1, 1, 10, 100\}$. The forest size parameter in IK was set to the default value of 100 as suggested by the authors of [9]. Because IK is a random method, for each fold, we did 10 runs and took the average accuracy.

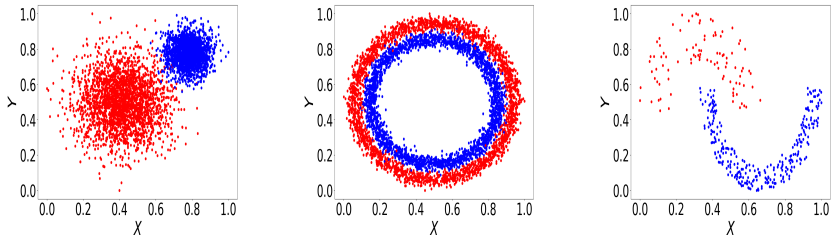
Table 3: Average classification accuracy of SVM with PMK and other SoTA classifiers. RF, XGB, LMNN and ITML did not complete due to the 'out of memory' errors (n/a) on three large/high-dimensional datasets.

Data	SVM_{pmk}	RF	XGB	$LMNN_{knn}$	$ITML_{svm}$	$LMNN_{svm}$
Breast Cancer	0.800	0.963	**0.986**	0.966	0.958	0.962
Gtzan	**0.782**	0.729	0.734	0.725	0.700	0.740
Hba	**0.780**	0.725	0.731	0.677	0.704	0.741
Steel Plate	0.751	**0.768**	0.763	0.721	0.720	0.752
REJAFADA	**0.982**	0.976	0.993	0.961	0.957	0.956
Mfeat	**0.990**	0.987	0.981	0.985	0.969	0.984
Cardio	**0.993**	0.905	0.908	0.775	0.910	0.920
Hydraulic	**1.000**	n/a	n/a	n/a	n/a	n/a
Segment	**0.980**	0.975	0.977	0.955	0.921	0.968
Fbis	**0.890**	0.823	0.847	0.710	0.560	0.557
Madelon	0.625	0.733	**0.843**	0.553	0.474	0.605
Malware	**0.995**	0.989	0.992	0.972	0.977	0.974
Page Blocks	**0.987**	0.980	0.972	0.966	0.958	0.965
First Order	0.565	**0.585**	0.583	0.538	0.542	0.500
Satimage	**0.926**	0.916	0.918	0.911	0.895	0.924
Musk	**0.998**	0.977	0.985	0.976	0.996	0.997
Taiwan Bank	**0.967**	0.925	0.822	0.811	0.923	0.930
Isolet	**0.970**	0.966	0.734	0.957	0.967	0.952
Corel	0.503	0.502	**0.611**	0.424	0.350	0.250
Ismis	**0.938**	n/a	n/a	n/a	n/a	n/a
Gas sensory	**0.996**	n/a	n/a	n/a	n/a	n/a

The average classification accuracies over a 10-fold cross-validation run of SVC classifiers with different kernels are provided in Table 2. PMK produced the best classification results in 11 datasets followed by Lap. in 4 datasets, and RBF and IK in 3 datasets each. Lin and Lap.R did not produce the best result in any dataset. The average accuracy and the average rank show that PMK produced better and more consistent results across different datasets with the best average rank of 1.799 followed by Lap. (2.714) and RBF (3.094).

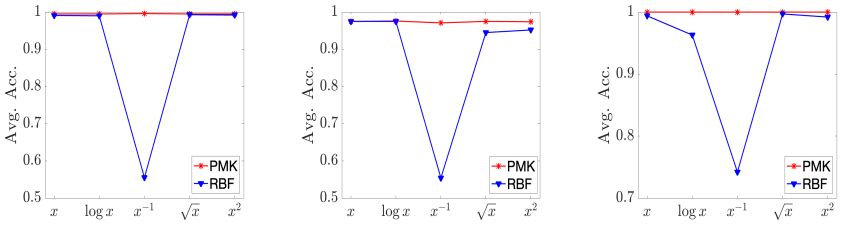
3.2 Comparison of SVM with PMK against other SoTA classifiers

Table 3 reports the classification results of SVM with PMK and the SoTA classification algorithms of Random Forest (RF), eXtreme Gradient Boosting (XGB), and Large Margin Nearest Neighbor ($LMNN_{knn}$). One way of making SVM with data-independent kernels (e.g., RBF kernel) data-dependent is to transform data into some latent space using data-dependent techniques like metric learning and use SVM in the latent space. We used SVM classification with RBF kernel on latent spaces resulted by Large Margin Nearest Neighbor (LMNN) and Information Theoretic Metric Learning (ITML), represented as $ITML_{svm}$ and $LMNN_{svm}$ in Table 3, respectively. The results show PMK had the best accuracy results in 16 datasets followed by XGB in 3 datasets and RF in 2 datasets. For SVM



(a) Gaussian (b) Circle (c) Jain

Fig. 3: Synthetic Datasets



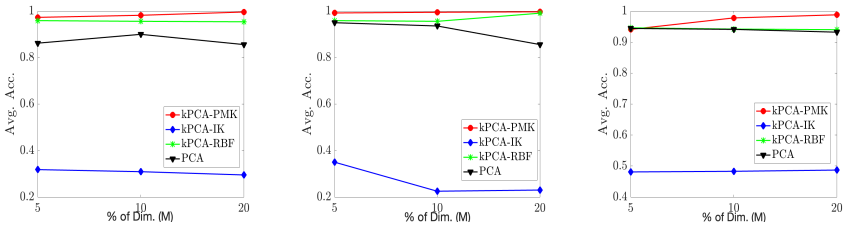
(a) Gaussians (b) Circles (c) Jain

Fig. 4: Average classification accuracy of PMK and RBF kernels over a 10-fold classification with different transformations of data.

using PMK and in cases of LMNN and ITML with SVM using RBF kernel, we tuned the cost and kernel parameters as mentioned earlier. For RF and XGB, we tuned the number of estimators and learning rate parameter in the range of {100, 200, 400, 600, 800, 1000} and {0.01, 0.08, 0.1, 0.2, 0.3}, respectively. For the metric learning method of LMNN (both SVM and k-NN), we tuned the number of neighbors (k) in the range of {5, 10, 15, 20, 25, 50}.

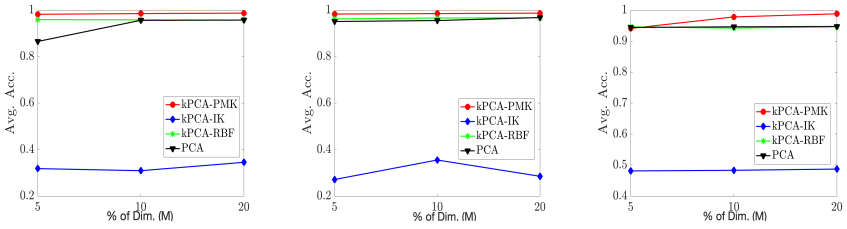
3.3 Robustness towards scales of measurement

To understand the robustness of PMK with respect to the change in units/scales used to represent data, we evaluated the performances of PMK and RBF in SVM classification in three two-class two-dimensional synthetic datasets namely Gaussian, Circle, and Jain, shown in Fig. 3. We created four variants of each dataset using non-linear scaling of data based on logarithm, inverse, square root and square, where each feature value x was transformed to $\log x$, x^{-1}, $\sqrt{x}$ and x^2 respectively. Since, x^{-1} and $\log x$ are not defined for $x = 0$, all the transformations were applied on $x' = c(x + \delta)$ as discussed in [5, 2], where $\delta = 0.0001$ and $c = 100$. Data values in both dimensions are normalised to be in the range of [0, 1] before and after the transformations. The average accuracy over a 10-fold cross-validation run of PMK and RBF on the three synthetic datasets and their



(a) RF (b) XGB (c) K-NN

Fig. 5: Average classification accuracy of RF, XGB and k-NN classifiers in Hydraulic dataset with reduced dimensions.



(a) RF (b) XGB (c) K-NN

Fig. 6: Average classification accuracy of RF, XGB and k-NN classifiers in Rejafada dataset with reduced dimensions.

scaled variants are shown in Fig. 4. As shown in the figure, PMK performed consistently (i.e., same average accuracy was obtained for all five variants) due to its scale-invariant characteristic, while the accuracies of RBF fluctuated when data representations were changed and it performed poorly for the case of x^{-1} in particular. Since RBF uses Euclidean distance, the similarity between two objects is affected by the non-linear scaling of the data.

3.4 Dimensionality reduction using Kernel PCA

We also evaluated the effectiveness of the proposed PMK kernel for dimensionality reduction using kernel PCA (kPCA). We used Hydraulic and Rejafada, two datasets with dimensionalities more than 5000. We reduced the number of dimensions to 5%, 10% and 20% of the original input dimensions and used three classifiers - RF, XGB and K-NN ($k = 10$). The average classification accuracies over a 10-fold cross-validation run of the three classifiers using reduced dimensions based on kPCA with PMK, IK and RBF and simple PCA are shown in Fig. 5 and Fig. 6. In all cases, dimensionality reduction based on kPCA using PMK produced better classification results and kPCA using IK produced the worst classification results.

4 Concluding Remarks

Most widely used kernel functions in SVM classifier learning such as RBF and Laplacian are data-independent and sensitive to units/scales of data. They may not produce good results in practical real-world problems, where data have complex structures and unknown units/scales of measurement. We may not know the units/scales of data when they are given for pattern extraction, often only values/numbers are provided. Therefore, kernels that are adaptive to data distribution (data-dependent) and robust to the variation in the units/scales used to represent data (scale-variant) are preferred. In this paper, we present one such kernel based on probability data mass called PMK which is both data-dependent and scale-invariant. We show that PMK produces better and more consistent results than existing data-independent and data-dependent kernels across a wide range of datasets from various real-world applications.

Acknowledgment

This material is based upon work supported by the U.S Air Force Office of Scientific Research under award number FA2386-20-1-4005.

References

1. Aryal, S., Ting, K.M., Haffari, G., Washio, T.: mp-dissimilarity: A data dependent dissimilarity measure. In: 2014 IEEE International Conference on Data Mining. pp. 707–712. IEEE (2014)
2. Aryal, S., Ting, K.M., Washio, T., Haffari, G.: A comparative study of data-dependent approaches without learning in measuring similarities of data objects. Data mining and knowledge discovery **34**(1), 124–162 (2020)
3. Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. Cambridge University Press (2000)
4. Dua, D., Graff, C.: Uci machine learning repository [http://archive. ics. uci. edu/ml]. irvine, ca: University of california. School of Information and Computer Science **25**, 27 (2019)
5. Fernando, T.L., Webb, G.I.: SimUSF: an efficient and effective similarity measure that is invariant to violations of the interval scale assumption. Data Mining and Knowledge Discovery **31**(1), 264–286 (2017)
6. Krumhansl, C.L.: Concerning the applicability of geometric models to similarity data: The interrelationship between similarity and spatial density. Psychological Review **85**(5), 445–463 (1978)
7. Lin, D., et al.: An information-theoretic definition of similarity. In: International Conference on Machine Learning (ICML). pp. 296–304 (1998)
8. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
9. Ting, K.M., Zhu, Y., Zhou, Z.H.: Isolation kernel and its effect on svm. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 2329–2337 (2018)

usfAD: A robust anomaly detector based on unsupervised stochastic forest

Sunil Aryal[1], KC Santosh[2] and Richard Dazeley[1]

[1]*School of Information Technology, Deakin University*
75 Pigdons Rd, Waurn Ponds VIC 3216, Australia
Email: {sunil.aryal, richard.dazeley}@deakin.edu.au

[2]*Department of Computer Science, University of South Dakota*
414 E Clark St, Vermillion, SD 57069
Email: santosh.kc@ieee.org

ABSTRACT

In real-world applications, data can be represented using different units/scales. For example, weight in kilograms or pounds and fuel-efficiency in km/liter or liter/100km. One unit can be a linear or non-linear scaling of another. The variation in metrics due to the non-linear scaling makes Anomaly Detection (AD) challenging. Most existing AD algorithms rely on distance- or density-based functions, which makes them sensitive to how data is expressed. This means that they are representation dependent. To avoid such a problem, we introduce a new anomaly detection method, which we call 'usfAD: Unsupervised Stochastic Forest-based Anomaly Detector'. Our empirical evaluation in synthetic and real-world cybersecurity (spam detection, malicious URL detection and intrusion detection) datasets shows that our approach is more robust to the variation in units/scales used to express data. It produces more consistent and better results than five state-of-the-art AD methods namely: local outlier factor; one-class support vector machine; isolation forest; nearest neighbor in a random subsample of data; and, simple histogram-based probabilistic method.

Keywords: Measurement scales and units, anomaly detection, outlier detection, robust anomaly detection, intrusion detection, spam detection, and cyber security.

1. INTRODUCTION

1.1 Background

Anomalies (also sometimes referred to as outliers) are data instances that are significantly different from most of the other data causing suspicions that they were generating from a different mechanism from the one that is normal or expected (Hawkins, 1980). Anomaly Detection (AD) is the task of detecting anomalies in a given dataset automatically using computers and algorithms (Chandola, Banerjee, & Kumar, 2009). It has many applications such as (Aggarwal, 2017):

- **Intrusion detection** – detecting unauthorised access requests and malicious activities in computer networks.
- **Fraud detection** – detecting fraudulent and suspicious credit card and other financial transactions in banking.
- **Spam detection** – detecting malicious and phishing emails in electronic communications.

Most existing anomaly detection algorithms (Breunig, Kriegel, Ng, & Sander, 2000; Liu, Ting, & Zhou., 2008; Aryal, Ting, & Haffari, 2016; Aryal, Baniya, & Santosh, 2019) assume that anomalies have feature values that are significantly different from those of normal instances. In other words, anomalies are few and different and they lie in low density regions.

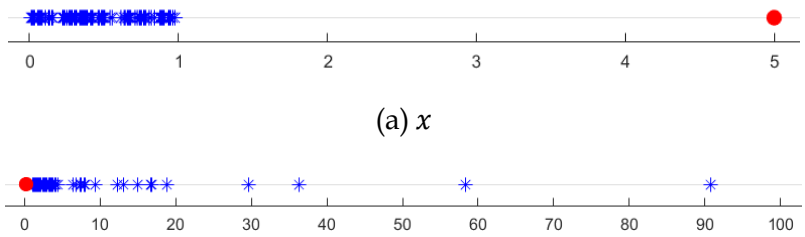
1.2 Motivation

In real-world applications, features of data objects can be measured in different units or recorded in different scales (Stevens, 1946; Fernando & Webb, 2017; Aryal S. , Ting, Washio, & Haffari, 2017; Aryal S. , Ting, Washio, & Haffari, 2020). For example: (i) patient's weight can be measured in kilograms or pounds and temperature in °C or °F; (ii) price of vehicles can be recorded in integer scale as $x = 100{,}000$ or logarithmic scale of base 10 like $x' = 5$, fuel efficiency in km/liter as $x = 9.0$ or ltr/100km as $x' = 11.11$; and (iii) sample variability can be measured in terms of standard deviation (std) or variance (var) ($std = \sqrt{var}$ or $var = std^2$). Note that the different

representations in Case (i) are linear scaling of one another whereas those in Cases (ii) and (iii) are non-linear scaling of each other.

Data representation can vary because of various reasons: (i) settings of devices used for measurement; (ii) domain and/or user requirements; and (iii) data compression to store and transmit using less bits. Such variation can be common in many application domains, such as cyber-physical systems, networks and communications, internet-of-things and healthcare, where different features of data objects are measured/recorded by different sensors.

The variation in metrics due to non-linear scaling makes automatic detection of anomalies a challenging task. Existing anomaly detection algorithms (Breunig, Kriegel, Ng, & Sander, 2000; Bay & Schwabacher, 2003; Aryal, Ting, & Haffari, 2016) rank instances in a database based on their densities or distances to nearest neighbors (NNs). The instances with low densities or large distances to their NNs are flagged as anomalies. Because they use feature values to compute density or NN distances, they are sensitive to how features are measured. For example, the red data point in Figure 1(a) clearly appears to be an anomaly, but when the same data is represented in the inverse scale as $x' = x^{-1}$, shown in Figure 1(b), the corresponding point looks like a normal data. Anomaly can be masked by the variation in data representation. Most existing anomaly detection algorithms cannot detect the anomaly in Figure 1(b).



(a) x

(b) $x' = x^{-1}$

Figure 1. An example of the same data represented in two scales. The red data point in Case (a) looks like an anomaly, whereas the corresponding point in Case (b) is more like a normal data.

When data is provided for anomaly detection only the magnitude (numbers) are available and the information about unit/scale used to measure/record them may not be available. Even if unit/scale used is known, we may not know whether the given form is appropriate for anomaly detection. Most existing algorithms do not consider such information and they can give misleading results

if data is not given in the appropriate form. In critical applications, such as in cybersecurity and banking, the consequences of true negatives and/or false positives can be severe.

Data needs to be transformed into an appropriate form before using existing algorithms. The simplest way to identify the appropriate form is to try different representations (scaling/transformations) and test which one produces the best task specific result. Because there are infinite number of possible transformations to try for each feature, it is not feasible to find the best combination from trial and error even in low-dimensional problems with tens of features, let alone high-dimensional ones with hundreds and thousands of features.

As discussed in Baniya et al. (2019), many psychologists argued that raw numbers can be misleading as they could have been measured/presented in different ways. One should not conclude anything from a given set of numbers without understanding where they come from and the underlying process of generating them (Lord, 1953; Townsend & Ashby, 1984; Velleman & Wilkinson, 1993; Joiner, 1981). Velleman & Wilkinson (1993) suggested that good data analysis does not make any assumption about data types/scales because data may not be what we see. Joiner (1981) provided some examples of 'lurking variables' – data appear to have one pattern, but in fact hide other information. However, in data mining, numeric data is assumed to be in 'interval scale' (Fernando & Webb, 2017) – the meaning of a unit difference is same everywhere in the space. This assumption may not be true when data is represented as non-linear scaling such as logarithm and inverse.

Recently, the impact of units/scales has been studied in the context of pairwise similarity measurement of data and robust (dis)similarity measures have been introduced (Aryal S. , Ting, Washio, & Haffari, 2017; Fernando & Webb, 2017; Aryal S. , Ting, Washio, & Haffari, 2020). These robust (dis)similarity measures have been shown to perform better than distance-based approaches (*e.g.*, Euclidean and Cosine distances) in content-based information retrieval and k-NN classification tasks. However, these robust (dis)similarity measures are not applicable in the anomaly detection task because of their implicit assumptions that are counter-intuitive for anomaly detection.

1.3 Contributions

In this research, we introduce an anomaly detection technique that is robust to data representation. Our work is motivated by ideas from 'Unsupervised Stochastic Forest' (USF)

(Fernando & Webb, 2017) and Isolation Forest (iforest) (Liu, Ting, & Zhou., 2008). USF is a variant of Unsupervised Random Forest (URF) (Shi & Horvath, 2006; Breiman, 2001) used by Fernando & Webb (2017) to develop a robust similarity measure. Iforest is a fast anomaly detection method based on a variant of random forest. It does not use distance/density and hence runs significantly faster than distance/density-based methods. However, its anomaly detection result is sensitive to the units/scales of data. This paper makes the following two main contributions:

i. Propose a new robust anomaly detection method by combining the ideas of USF and iforest. We extend USF to make it applicable in the anomaly detection task, by incorporating the iforest approach to rank data instances based on their anomaly scores. We call the proposed method as 'usfAD' (Unsupervised Stochastic Forest based Anomaly Detector).
ii. Compare the anomaly detection performance of usfAD against five state-of-the-art methods using synthetic and real-world cybersecurity (spam detection, phishing URL detection and intrusion detection) datasets. To evaluate the robustness of methods to different representations of the same data (x), we use the non-linear scaling functions: square (x^2); square root ($\sqrt{x}$); logarithm ($\log x$); and, inverse (x^{-1}).

1.4 Paper organization and notations

The rest of the paper is structured as follows. Prior works related to this paper are discussed in Section 2 and the proposed usfAD is discussed in Section 3. Experimental setup and results are presented in Section 4 followed by related discussion in Section 5. The last section concludes the paper with potential directions for future research.

The list of key notations and symbols used in this paper are provided in Table 1.

Table 1. Notations

$\boldsymbol{x} = \langle x_1, x_2, \cdots, x_d \rangle$	A vector representing a data instance in an d-dimensional continuous space ($\mathbb{R}^d$)
$x_i \in \mathbb{R}$	The value of $\boldsymbol{x}$ in the i^{th} dimension (feature)
D ($\lvert D \rvert = n$)	A training set of n normal instances

Q ($\|Q\| = q$)	A test set of q instances that is a mixture of normal and anomalous data
$\mathcal{D} \subset D$ ($\|\mathcal{D}\| = \psi \ll n$)	A small subset of training data D
$kNN(\cdot, D)$	The set of k nearest neighbors of a test instance in D using Euclidean distance
T_j ($j = 1, 2, \cdots, t$)	A tree in an ensemble of t trees
h	The parameter that defines the height of a tree
$\ell_j(\cdot)$	The pathlength of a test instance in tree T_j
$L_j(\cdot)$	The leaf node in tree T_j where a test instance reached
$m(\cdot)$	The data mass in a region (*e.g.*, a node of a tree)

2. RELATED WORKS

The anomaly detection problem can be solved using three approaches: supervised, unsupervised or semi-supervised learning (Chandola, Banerjee, & Kumar, 2009). In the **supervised approach**, a classification model is learned from labelled training instances from both normal and anomalous classes, which is then used to predict class labels for test data (a mixture of normal data and anomalies). Labelled training samples from anomalous class may not be available in many real-world applications (Chandola, Banerjee, & Kumar, 2009). In the **unsupervised approach**, instances in a database are ranked directly based on some outlier score. In the training process, a scoring model is learned (without using labels), which is used to compute anomaly score of test data to rank them in the testing phase. Anomalies are assumed to be few and different. This approach may not work well when the assumption does not hold, *i.e.*, when there are far too many anomalies (Boriah, Chandola, & Kumar, 2008; Chandola, Banerjee, & Kumar, 2009). In the **semi-supervised approach**, a profile of normal data is learned using training data from the normal class only but the label information is not used in the learning process. In this regard, it is different from the semi-supervised learning approach discussed in other machine learning problems, where a model is learned in the training phase using partially labelled training set (only a subset of training data is labeled). In the testing phase, instances are ranked based on how well

they comply with the learned profile of normal data. It makes no assumptions about anomalies nor does it require any training samples from the anomalous class (Boriah, Chandola, & Kumar, 2008; Chandola, Banerjee, & Kumar, 2009). Thus, the semi-supervised approach is more realistic in real-world applications.

In this paper, we focus on the semi-supervised approach where a model is learned from a training set D of n instances belonging to the normal class only and evaluated on a test set Q of q instances from a mixture of normal and anomalous data. In this section, we review some widely used existing anomaly detection methods that are applicable for the semi-supervised approach.

2.1 Nearest neighbor based methods

In Nearest Neighbor (NN)-based methods, the anomaly score of a test instance $\boldsymbol{x} \in Q$ is estimated based on the distances to its kNNs in D. Local Outlier Factor (LOF) (Breunig, Kriegel, Ng, & Sander, 2000) and k^{th} NN distance (Bay & Schwabacher, 2003) are the most widely used semi-supervised and unsupervised methods. Being different from normal instances, anomalies are expected to have larger distances to their kNNs than normal instances. To compute the anomaly score of $\boldsymbol{x} \in Q$, NN-based methods require to compute its distances with all instances in D. It is computationally expensive if the size of training set D is large. Sugiyama and Borgwardt (2013) showed that the nearest neighbor search in a small subset $\mathcal{D} \subset D$, $|\mathcal{D}| = \psi \ll n$ is enough for anomaly detection. They proposed a simple, but very fast, anomaly detector called Sp**.** The anomaly score of $\boldsymbol{x} \in Q$ is its distances to the nearest neighbor (1NN) in $\mathcal{D}$. It has been shown that Sp with ψ as small as 25 produces competitive results to LOF but runs several orders of magnitude faster (Sugiyama & Borgwardt, 2013). Ting et al. (2017) used an ensemble approach using multiple subsamples $\mathcal{D}_j \subset D$, $|\mathcal{D}_j| = \psi \ll n$ $(j = 1, 2, \cdots, t)$ and computed the anomaly score of $\boldsymbol{x} \in Q$ as the average distance to the nearest neighbor (1NN) in t subsamples (Ting, Washio, Wells, & Aryal, 2017).

2.2 Support vector based methods

These methods define the boundary around normal (expected) data and identify a set of data instances lying in the boundary called Support Vectors (SVs) using the kernel trick. They compute the pairwise similarities of instances in the training set D using a kernel function. Gaussian kernel that uses Euclidean distance is a popular choice. In the testing phase, the anomaly score of $\boldsymbol{x} \in Q$ is estimated based on its kernel similarities with the SVs. One-Class Support Vector Machine

(SVM) (Scholkopf, Platt, Shawe-Taylor, Smola, & Williamson, 2001) and Support Vector Data Description (SVDD) (Tax & Duin, 2004) are widely used methods in this class. The training process is computationally expensive in the case of large D because of the pairwise similarity calculations. In the testing phase, the runtime is linear to the size of SVs. Different approaches have been suggested to speed up the training and testing process (Rekha, 2015; Jiang, Wang, Hu, Kakde, & Chaudhuri, 2017)

2.3 Isolation based methods

This class of methods are based on the assumption that anomalies are more susceptible to isolation. Isolation Forest (iforest) (Liu, Ting, & Zhou., 2008) uses an ensemble of t random trees. Each tree T_j $(j = 1, 2, \cdots, t)$ is constructed from a small subsample of training data ($\mathcal{D}_j \subset D$, $|\mathcal{D}_j| = \psi \ll n$). The idea is to isolate every instance in $\mathcal{D}_j$ by random partitioning of data space into two non-empty sets in each node. Anomalies are expected to isolate early and have shorter average pathlengths than normal data. Aryal et al. (2014) used the average relative data mass in the leaf nodes and their immediate parents as the anomaly score of a test instance (Aryal S. , Ting, Wells, & Washio, 2014). These methods are very efficient as they do not require any distance/similarity calculations. Isolation using nearest neighbor ensemble (iNNE) (Bandaragoda, Ting, Albrecht, Liu, & Wells, 2014) uses a different mechanism of isolation in $\mathcal{D}_j$. It creates a hypersphere centered at each instance in $\mathcal{D}_j$ with the radius equal to the distance to its nearest neighbor (NN). Anomalies are expected to fall in hyperspheres with a large radius.

2.4 Histogram based methods

Methods based on one-dimensional histograms (Goldstein & Dengel, 2012; Aryal, Ting, & Haffari, 2016; Aryal, Baniya, & Santosh, 2019) are another type of efficient methods. In each dimension, a fixed number of equal-width bins are created and the data mass (frequency) in each bin is recorded. Because anomalies are few and different, they are expected to fall in bins with lower frequencies (*i.e.*, low density bins) in many dimensions. Simple Probabilistic Anomaly Detector (SPAD) (Aryal, Ting, & Haffari, 2016) is a histogram-based anomaly detector, where bin width in each dimension depends on the variance of data in that dimension.

3. usfAD: A ROBUST ANOMALY DETECTOR BASED ON UNSUPERVISED STOCHASTIC FOREST

All methods discussed in Section 2 are sensitive to how data is represented because they assume that anomalies have feature values significantly different from those of normal data. They often produce poor results in datasets where this assumption does not hold, such as when data is not given in the appropriate form (Figure 1(b)). However, we may not know the appropriate form of data in many real-world applications. In this section, we propose a new anomaly detection method that is robust to the units/scales of data used. Our proposed method is motivated by the ideas of *Unsupervised Stochastic Forest (USF)* and *Isolation Forest (iforest)*.

USF **is** a variant of random forest used by Fernando and Webb (2017) to define a similarity measure that is robust to units/scales of data. The similarity of two instances is defined as the number of shared leaves in the forest of t trees. With a user defined tree height parameter (h), each tree T_j in the ensemble is constructed from a small subsample of data, $\mathcal{D}_j \subset D$ $(j = 1, 2, \cdots, t)$, where $|\mathcal{D}_j| = 2^h$. The subsample of data is divided into two equal subsets at each internal node of a tree by splitting at the median of the sample values of an attribute selected at random. The process is repeated until nodes have one instance each. It creates balanced binary tress with all leaf nodes at the same height h. The tree structure is robust to how data is measured because of the median splits.

As discussed in Section 2.3, iforest is a variant of random forest used for anomaly detection. Each tree T_j is constructed from a small subsample of data, $\mathcal{D}_j \subset D$ $(j = 1, 2, \cdots, t)$, where $\left|\mathcal{D}_j\right| = \psi \ll n$. Instances in $\mathcal{D}_j$ are isolated using random partition at each internal node. Both attribute and split point are selected at random. Iforest is sensitive to how data is measured because of the random split. The probability of having a split between two consecutive data points is proportional to their distance. The random split results in unbalanced binary trees which are the core of iforest for anomaly detection. The average pathlengths of a test data instance in trees is used as its anomaly scores.

Because of the balanced binary trees, pathlength cannot be used as the anomaly score in USF. We propose the following extensions to USF so that pathlength can be used. When a tree T_j is constructed from $\mathcal{D}_j \subset D$, normal and anomalous regions are defined in each node using the entire training data D. Therefore, tree construction is a two-step process.

Algorithm 1: Build a tree from a given subsample of data

Input: $\mathcal{D}$ – subsamples of training data ($|\mathcal{D}| = 2^h$)

Output: a USF tree

`usfTree`($\mathcal{D}$)

1. `IF` $|\mathcal{D}| = 1$ `THEN` /* check if leaf node is reached */
2. `RETURN` /* return if leaf */
3. $self.a \leftarrow$ `select`$(1, 2, \cdots, d)$ /* randomly select an attribute */
4. $S \leftarrow$ `sort`$(\mathcal{D}_{self.a})$ /* sort sample values for the selected attribute */
5. $self.s \leftarrow \left(S\left[\frac{|\mathcal{D}|}{2}\right] + S\left[1 + \frac{|\mathcal{D}|}{2}\right]\right)/2$ /* median split */
6. $\mathcal{D}_L \leftarrow$ `filter`$(\mathcal{D}_{self.a} \leq self.s)$; $\mathcal{D}_R \leftarrow$ `filter`$(\mathcal{D}_{self.a} > self.s)$ /* filter samples */
7. $self.lNode \leftarrow$ `usfTree`$(\mathcal{D}_L)$; $self.rNode \leftarrow$ `usfTree`$(\mathcal{D}_R)$ /* build two subtrees */

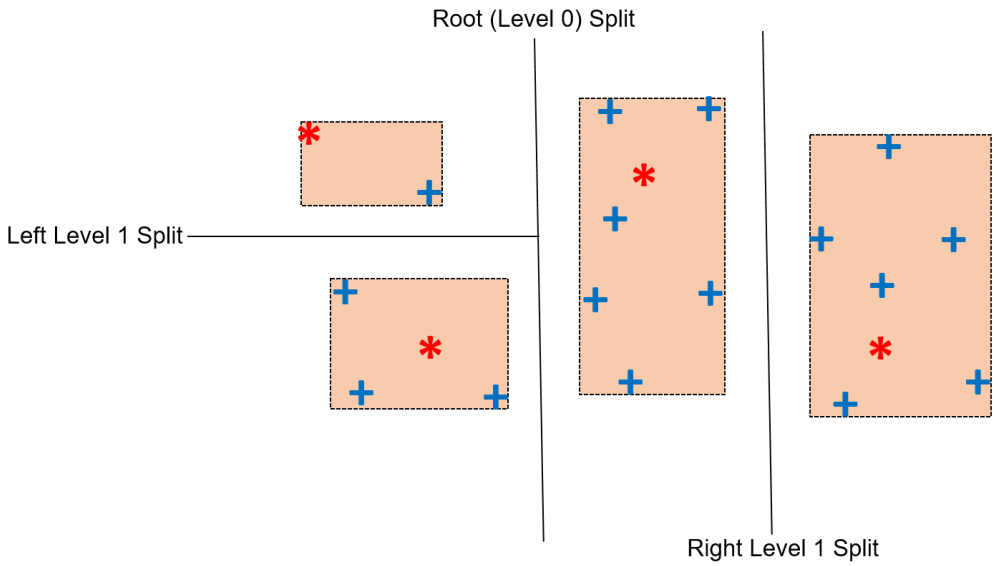


Figure 2. An example of partitioning and definition of normal regions in leaf nodes with the training data D of 20 normal instances and $h = 2$. Data point represented by red asterisks are subsamples $\mathcal{D} \subset D$ selected to build the tree ($|\mathcal{D}| = 2^2 = 4$). Orange rectangles represent normal regions in leaf nodes.

a. Building T_j from $\mathcal{D}_j$: This is the same as in USF where instances in the subsample $\mathcal{D}_j$ are isolated using median splits in each node. The detailed procedure is provided in Algorithm 1.

Algorithm 2: Update tree to define normal regions in each node

Input: D – the entire normal training data

`updateTree`(D)

1. `IF` $self.lNode =$ `NULL THEN` /* check if it is a leaf */
2. $self.m \leftarrow |D|$ /* record training data mass */
3. $self.range \leftarrow$ `rangeAll`(D) /* range of training data in all dimensions */
4. `RETURN` /* return */
5. $self.range \leftarrow$ `range`($D_{self.a}$) /* range of values in dimension $self.a$ */
6. $D_L \leftarrow$ `filter`$\left(D_{self.a} \leq self.s\right)$; $D_R \leftarrow$ `filter`$\left(D_{self.a} > self.s\right)$ /* filter data */
7. $self.lNode$.`updataTree`(D_L); $self.rNode$.`updataTree`(D_R) /* do it on subtrees */

Algorithm 3: Compute anomaly score of $\boldsymbol{x}$ in a tree

Input: $\boldsymbol{x}$ – A test data instance; ℓ – pathlength so far ($\ell = 0$ for root)

Output: Anomaly score of $\boldsymbol{x}$

`score`($\boldsymbol{x}$, p)

1. `IF` $self.lNode =$ `NULL THEN` /* check if it is a leaf */
2. `IF inNormalRegion`($\boldsymbol{x}$) `THEN` /* check if $\boldsymbol{x}$ in the normal region */
3. `RETURN` $\ell + \log_2(self.m)$ /* return augmented pathlength */
4. `RETURN` ℓ /* return pathlength (in anomaly region) */
5. `IF inNormalRange`($\boldsymbol{x}_{self.a}$) `THEN` /* check if it is in normal range */
6. $p \leftarrow \ell + 1$ /* increase pathlength */
7. `IF` $\boldsymbol{x}_{self.a} \leq self.s$ `THEN` /* check if $\boldsymbol{x}$ falls to left subtree */
8. `RETURN` $self.lNode$.`score`($\boldsymbol{x}$, ℓ) /* get score from the left subtree */
9. `RETURN` $self.rNode$.`score`($\boldsymbol{x}$, ℓ) /* else get score from the right subtree */
10. `RETURN` ℓ /* out of normal range in internal node, return pathlength */

b. Defining normal and anomalous regions in nodes of T_j using D: Once T_j is built, the entire training data D is passed to T_j. The region defined by the bounding hyper-rectangle that covers the training data within a leaf node is considered as the "normal region" and the rest is considered as the "anomalous region". An example of space partitioning and the definition of "normal regions" in leaf nodes is shown in Figure 2. The training data mass in each leaf node is stored. In internal nodes, only the attribute selected for partitioning the space is used

to define the normal data range. The algorithm to update T_j to define normal and anomalous regions, and store data mass in leaf nodes is provided in Algorithm 2.

In the testing phase, the anomaly score of a test instance $\boldsymbol{x}$ in each tree T_j is computed as the pathlength of the first node where it falls outside of the normal region. As shown in Algorithm 3, $\boldsymbol{x}$ is traversed down the T_j in each internal node only if its value of the attribute selected to split the node is within the normal data range. Otherwise the traversal is terminated, and the current height is returned as the score of $\boldsymbol{x}$ in T_j, $\ell_j(\boldsymbol{x})$. If $\boldsymbol{x}$ reaches a leaf node, $\ell_j(\boldsymbol{x})$ is estimated based on whether it lies in the normal or anomalous region: (i) if it lies in the anomalous region, $\ell_j(\boldsymbol{x}) = h$; and, (ii) if it lies in the normal region, the score is the height augmented by the training data mass in the leaf ($L_j(\boldsymbol{x})$), $\ell_j(\boldsymbol{x}) = h + \log_2(m(L_j(\boldsymbol{x})))$, where $m(L_j(\boldsymbol{x}))$ is the data mass in the leaf where $\boldsymbol{x}$ falls in tree T_j. The second term is to ensure that leaf nodes with higher densities have a larger score than those with low densities. Similar adjustment was done in iforest (Liu, Ting, & Zhou., 2008). The final anomaly score of $\boldsymbol{x}$ is the average score over the ensemble of t trees like in iforest (Liu, Ting, & Zhou., 2008):

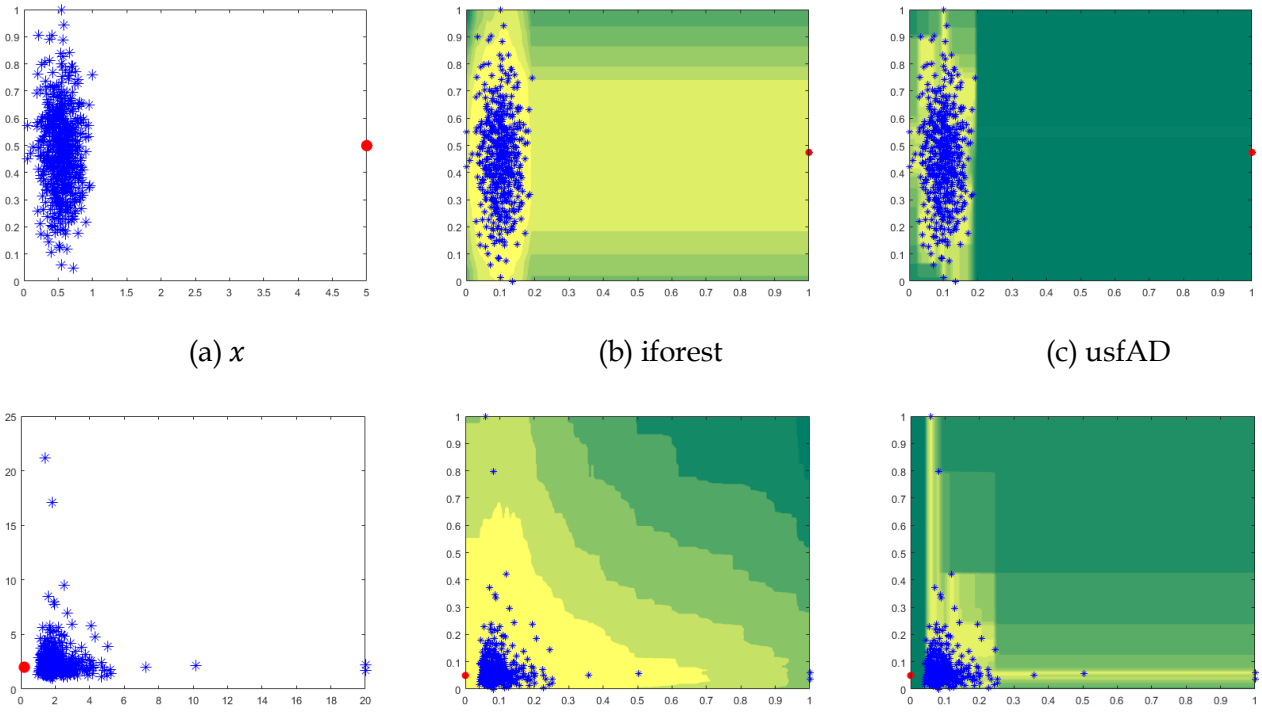
$$s(\boldsymbol{x}) = \frac{1}{t}\sum_{j=1}^{t} \ell_j(\boldsymbol{x}) \qquad \textit{Eq. 1}$$

We call the proposed method 'usfAD' as it is based on the Unsupervised Stochastic Forest (USF). It uses the same idea of isolating anomaly regions from normal regions as in iforest (Liu, Ting, & Zhou., 2008) but using different mechanism of isolation. Anomalies will have smaller scores than normal instances. Because of the median splits, usfAD is robust to the change in units/scales of data measurement. It is based on the orderings or ranks of data which is either preserved or reversed if the data is measured in different scales. If a data point u lies between $[x, y]$ in one scale, the corresponding point u' lies in between $[x', y']$ in another scale. As the median of even data samples is the mid-point of the two data in the middle, the definition of normal regions in leaf nodes can vary slightly if data is measured differently. It may cause small differences in the anomaly detection results.

Figure 3 shows the contour plots of anomaly scores of points in a two-dimensional space using iforest ($t = 100,\ \psi = 256$) and usfAD ($t = 100,\ h = 5$) in a dataset in two scales: x and $x' = x^{-1}$. It is clear that the anomaly represented by the red dot is not identified as a strong anomaly by

iforest in both representations. It appears to be an anomaly in the original scale (Subfigure (b)) but the actual score is not significantly different from those of some blue points. It clearly appears to be a normal point in the inverse scale (Subfigure (e)). The anomaly can be detected as a strong anomaly by usfAD in both scales (Subfigures (c) and (f)).

In terms of runtime, usfAD runs slightly slower than iforest. In the training phase, it requires to define normal regions in nodes passing the entire training data in each tree. During testing, for each test data, it requires an overhead to check if it falls in the normal or anomaly region in each node. Its training runtime complexity is $O(nth + t\psi d)$, where $\psi = 2^h$. The testing runtime complexity to rank a test instance is $(t(h + d))$. It needs $O(t\psi d)$ space to store the ensemble of trees. Like iforest, its testing time is independent of training data size n.



(a) x (b) iforest (c) usfAD

(d) $x' = x^{-1}$ (e) iforest (f) usfAD

Figure 3. An example dataset in two scales: x and $x' = x^{-1}$ and anomaly contours of iforest $(t = 100, \psi = 256)$ and usfAD $(t = 100,\ h = 5)$. Note that data is normalized to the unit range in both dimensions. The darker the color, the lower the anomaly score, i.e., higher the chances of being anomaly. Note that training data D or D' does not include the red dot, only normal data points represented by blue asterisks are included.

4. EMPIRICAL EVALUATION

This section presents the setup and results of experiments conducted to evaluate the performance of usfAD against existing anomaly detection methods. Five widely used state-of-the-art methods of LOF, one-class SVM, iforest, Sp and SPAD were used as contenders for performance evaluation. Six two-dimensional synthetic datasets and four benchmark cybersecurity datasets were used. All experiments were conducted in the semi-supervised setting. In each dataset, half of the normal instances were used as training data D and the remaining other half of normal data and all anomalies are considered as test data Q as done in (Boriah, Chandola, & Kumar, 2008). Anomaly detection model was learned from D and tested on Q.

The performance was evaluated in terms of the Area Under the ROC Curve (AUC). AUC is estimated using the rankings of test instances in Q based on their anomaly scores (anomalies are expected to have higher ranks than normal instances) and ground truth labels (Ting, Washio, Wells, & Aryal, 2017; Hand & Till, 2001). It is equivalent to the probability that a randomly chosen anomaly will be ranked below a randomly chosen normal instance (Hand & Till, 2001). For the random methods of iforest, Sp and usfAD, each experiment was repeated 10 times and reported the average AUC. The same training and test sets of a dataset were used for all experiments with the dataset.

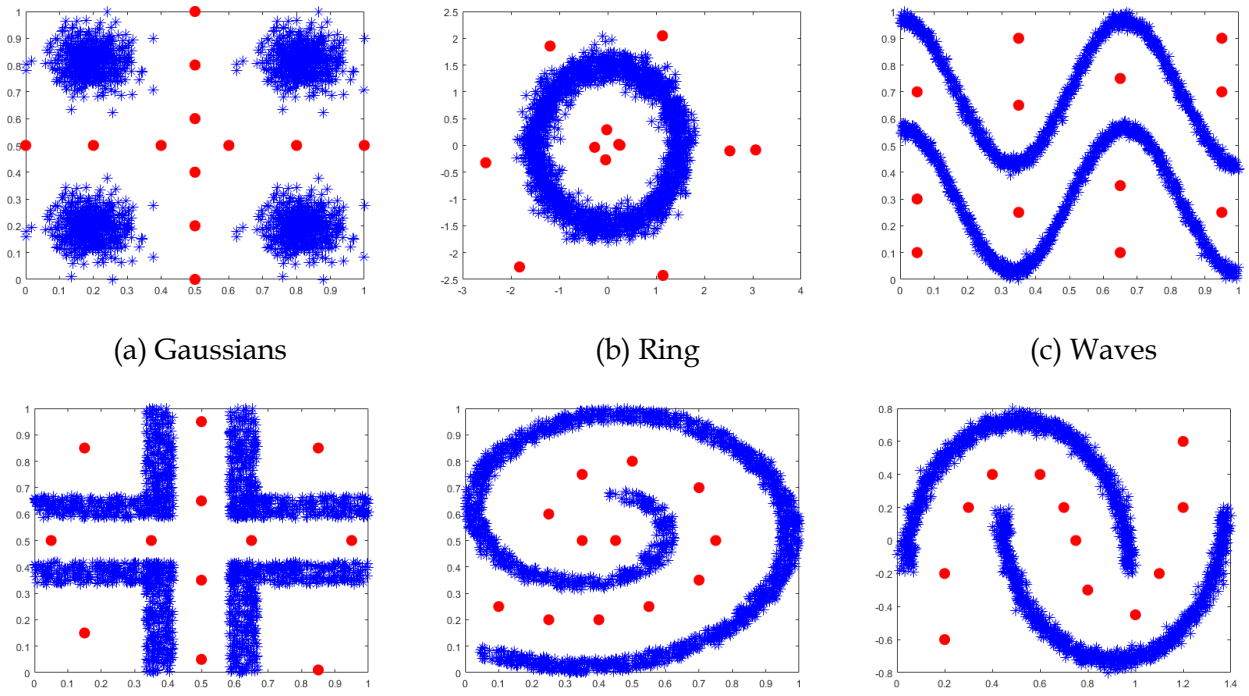
In terms of implementation, the python implementations of LOF and SVM included in the Scikit-learn Machine Learning Library (Pedregosa, et al., 2011) were used. Other methods and experimental setups were also implemented in Python. All the experiments were conducted in a Linux machine with 2.27 GHz processor and 8 GB memory. Parameters in algorithms were set to suggested default values by respective authors as:

- LOF: Nearest Neighbour parameter $k = \lfloor\sqrt{n}\rfloor$;
- Sp: Subsample size $\psi = 25$;
- SPAD: Number of bins $b = \lfloor\log_2 n\rfloor + 1$;
- iforest: Ensemble size $t = 100$, and Subsample size $\psi = 256$;
- SVM: Default settings of all parameters ($kernel = 'rbf', \gamma = 1/d, \nu = 0.5$); and
- usfAD: Tree height $h = 5$, and Ensemble size $t = 100$.

To evaluate the robustness of algorithms with different representations of the same data, non-linear scaling using square (x^2), square root ($\sqrt{x}$), logarithm ($\log x$) and inverse (x^{-1}) were used. To cater for $\log x$ and x^{-1} at $x = 0$, all transformations were applied on $\hat{x} = c(x + \delta)$, with $\delta = 0.0001$ and $c = 100$. Note that data was normalized to the unit range in each feature before rescaling to ensure the same effect of δ and c in all features. Once data was rescaled, it was renormalized to be in the unit range again. This study used the same procedure of rescaling as employed by Fernando & Webb (2017).

Results in synthetic datasets and real-world cybersecurity datasets are discussed separately in the following two subsections.

4.1 Synthetic datasets



(a) Gaussians (b) Ring (c) Waves

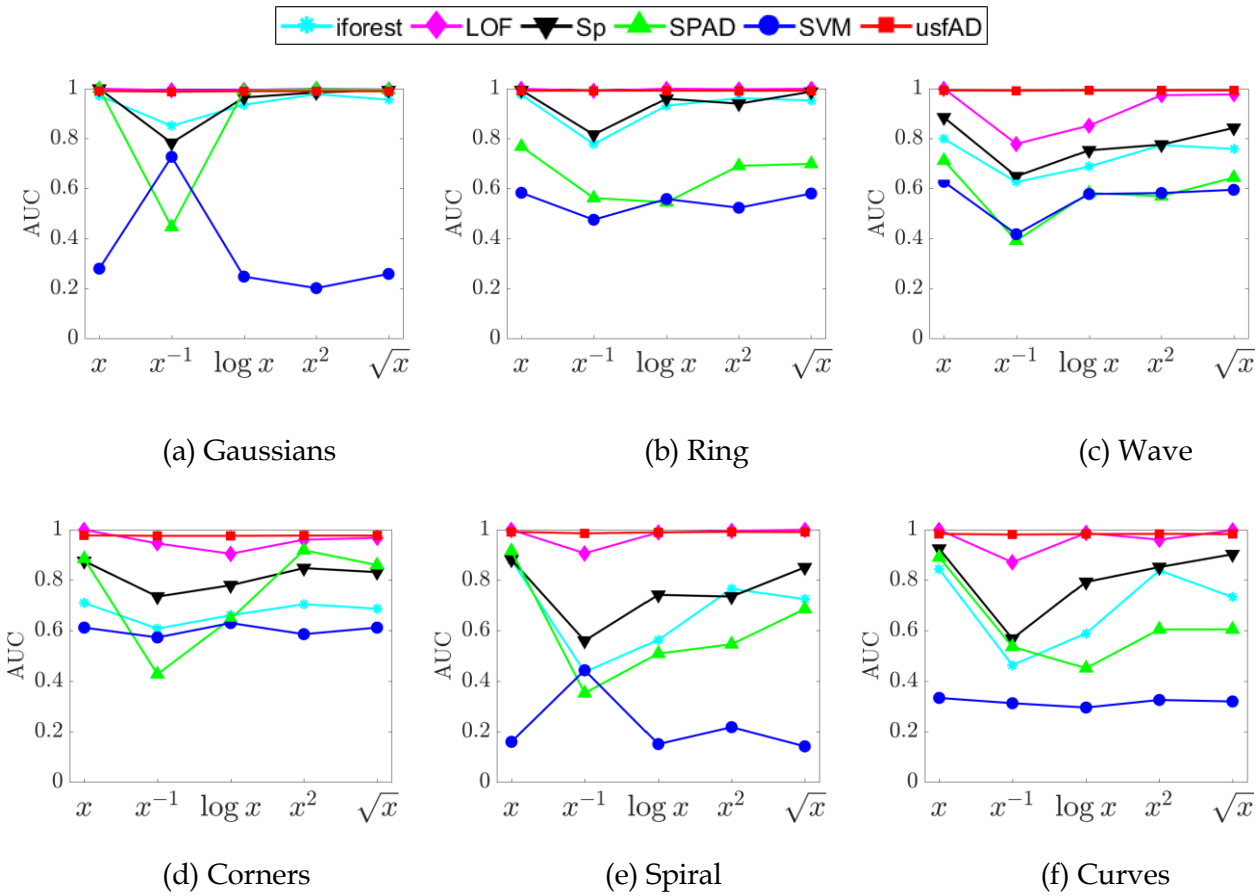
(d) Corners (e) Spiral (f) Curves

Figure 4. Six two-dimensional synthetic datasets. Each has 2000 normal instances represented by blue asterisks and 12 anomalies represented by red dots.

To evaluate the effectiveness and robustness of usfAD and other contenders in detecting anomalies in various data distributions, six two-dimensional datasets as shown in Figure 4 were used. These datasets represent good examples of scenarios where normal or expected data (represented by blue asterisks) form complex structures and anomalies (represented by red dots)

are added at various parts in the data space. In all cases, points represented by red dots clearly look like anomalies visually. We expect existing anomaly detection methods to detect them successfully.

AUCs of the contending methods in the six synthetic datasets in the scales of x, x^{-1}, $\log x$, x^2 and $\sqrt{x}$ are presented in Figure 5. It clearly shows that usfAD produced the best or equivalent to the best results in all datasets. It produced similar results with and without non-linear scaling of data. It shows that usfAD is robust to how data was measured. It is clear that all five existing anomaly detection methods were sensitive to how data is expressed. It shows LOF is the least sensitive existing method, but its performance dropped with some scaling in some datasets, *e.g.*, inverse and logarithmic scaling in Waves and Corners.



(a) Gaussians (b) Ring (c) Wave

(d) Corners (e) Spiral (f) Curves

Figure 5. AUC in the six synthetic datasets with and without non-linear scaling of data

It is interesting to note that four existing methods (SVM, iforest, Sp and SPAD) did not rank all anomalies before normal instances even in the original scale (x) in most datasets. These results demonstrate their limitations. In many of these examples, it is not possible to learn the perfect boundary covering the normal data only without any error. Thus, SVM ranked some normal instances before anomalies resulting in poor AUC. Because some anomalies in these examples lie between normal data in both axes, they cannot be isolated easily in trees. They have longer pathlengths than some instances at the fringe of data distribution in any axis that can be isolated early. Thus, iforest failed to detect these types of anomalies. Sp has high variance because of the use of a very small subsample of data resulting in poor performance in datasets with complex structures. Because SPAD computes anomaly scores in each axis separately as it assumes the dimensions are independent. This results in it being unable to detect anomalies that look normal when examined in any individual axis but would have appeared as anomalous when examined from both axes together.

Another interesting result to note is that some existing methods produced better results after rescaling than in the original space, *e.g.*, SVM produced best results with the inverse scaling in Gaussians and Spiral.

4.2 Real-world datasets: cybersecurity

In real-world datasets, four widely used benchmark cybersecurity datasets were used: Spambase[1] (a publicly available emails collection to detect spams), NSL-KDD[2] (a new version of the traditional KDD99 intrusion detection dataset created by resolving some inherent issues and limitations), ISCX-URL[3] (a dataset of benign and malicious URLs) and UNSW-NB15[4] (a widely used network intrusion detection dataset). The characteristics of the four datasets used are provided in Table 2.

Cybersecurity datasets were used because their feature values can be measured or expressed in different forms. For example, in the ISCX-URL dataset, URLs are represented by Lexical features (Mamun, Rathore, Lashkari, & Stakhanova, 2016). Several features are based on the ratios of the lengths of various components of URLs such as domain, path and arguments. Some of them are

[1] https://archive.ics.uci.edu/ml/datasets/spambase
[2] https://www.unb.ca/cic/datasets/nsl.html
[3] https://www.unb.ca/cic/datasets/url-2016.html
[4] https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/

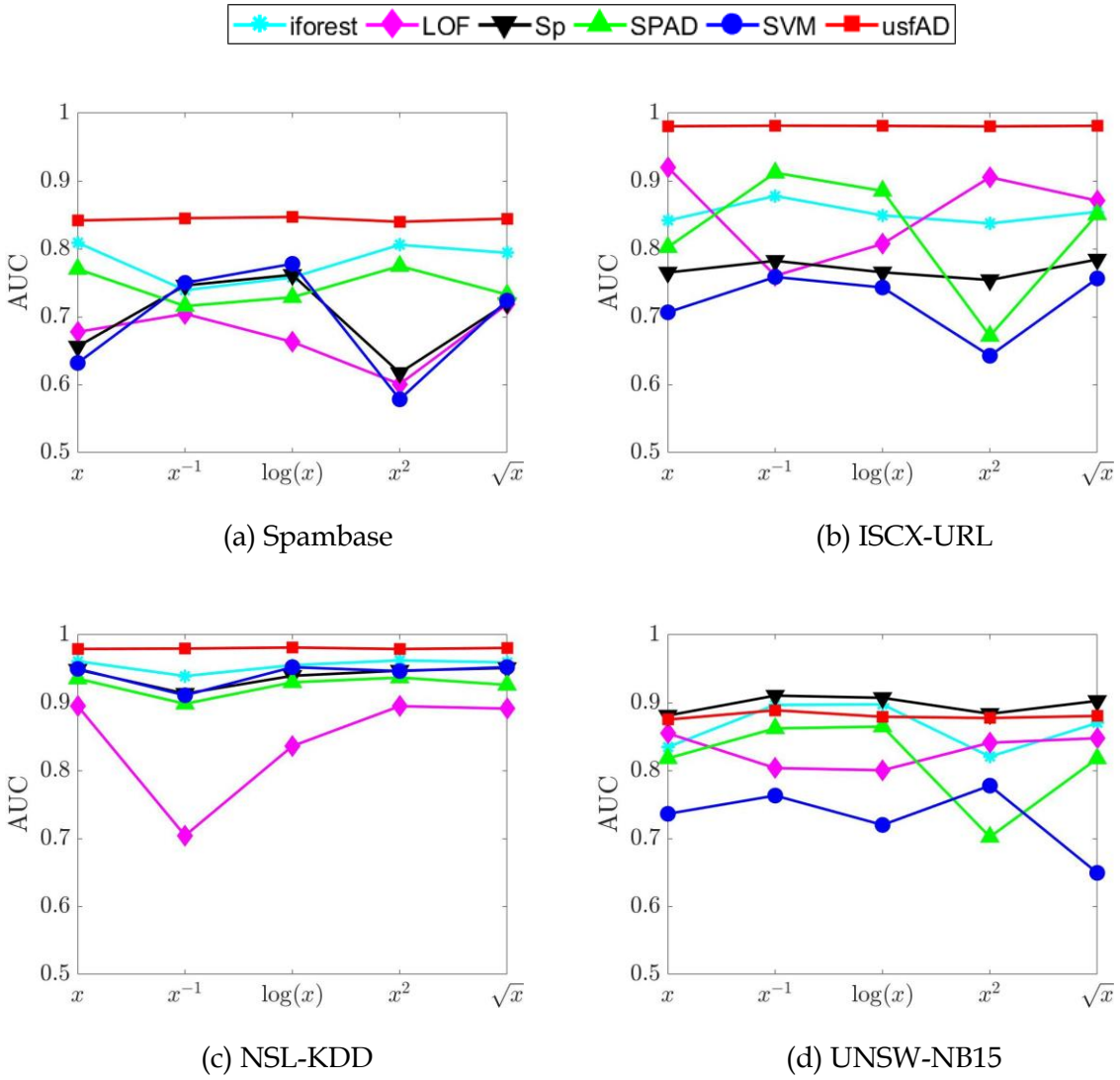
represented in such a way that the ratio is less than 1, whereas others are not. Length of a component (such as domain) is used in numerator in one ratio and as denominator in others, *e.g.*, domain to URL length ratio, path to domain length ratio, argument to path length ratio, *etc.* Depending on how such features are measured, existing anomaly detectors produce different results.

Table 2. Characteristics of cybersecurity datasets used

Dataset	#dimensions (d)	#Training normal data (n)	Test data	
			#Normal data	#Anomalies
Spambase	57	1394	1394	176
ISCX-URL	75	3889	3890	7570
NSL-KDD	38	38527	38527	71463
UNSW-NB15	39	82336	82337	93000

Anomaly detection results in terms of AUC of existing methods and the proposed usfAD in the four cybersecurity datasets using the given form of data (x) and their non-linear scalings using x^{-1}, $\log x$, x^2 and $\sqrt{x}$ are presented in Figure 6. As expected, all five existing methods were sensitive to the scales of data used, their performances varied significantly. Some existing methods produced better results with other scales. For example, in ISCX-URL, AUCs of all existing methods except LOF increased with the inverse scale. LOF produced better result with the inverse scaling than x in Spambase. The proposed method of usfAD produced the same results regardless of the scaling used in all four datasets. It produced the best results in three datasets except in UNSW-NB15, where Sp produced slightly better results than usfAD. However, Sp produced significantly worse results than usfAD in the other three datasets. The results of usfAD were more consistent than any other contender across the four datasets.

In terms of runtime, as expected from the time complexities discussed in Section 3, we observed that usfAD was faster than LOF and SVM, and slower than iforest and Sp. For example, in the largest dataset of UNSW-NB15, the total runtime (including training and testing) of usfAD was 437 seconds compared to 36 seconds (Sp), 60 seconds (SPAD), 84 seconds (iforest), 606 seconds (LOF), and 1562 seconds (SVM). Note that we used the implementations of LOF and SVM available in the Scikit-learn Library which are optimized to run fast using efficient data structures. For other methods (iforest, Sp, SPAD and usfAD), we implemented them without ensuring code was optimized.



(a) Spambase

(b) ISCX-URL

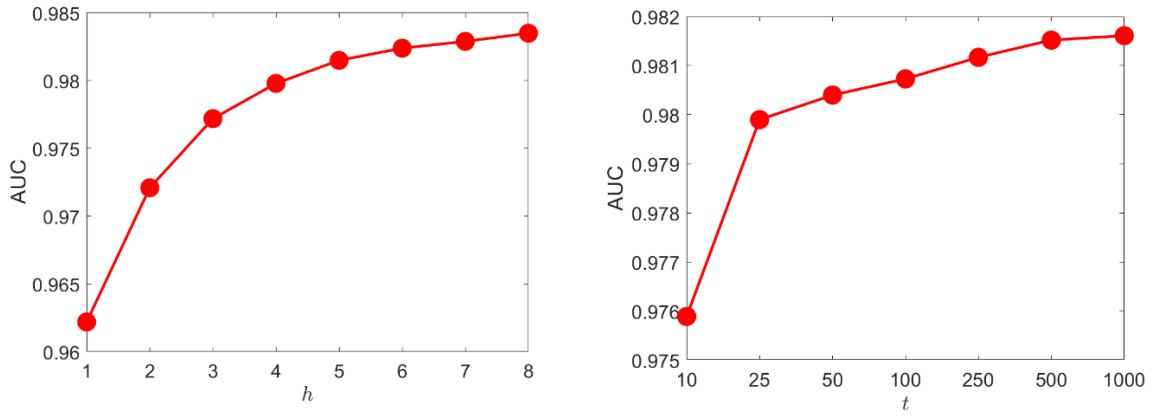
(c) NSL-KDD

(d) UNSW-NB15

Figure 6. AUC of contending methods in four widely used benchmark cybersecurity datasets with order preserving and order reversing scaling of data

4.3 Sensitivity of parameters

There are two parameters in usfAD: tree height (h) and number of trees (t). Note that h determines the subsample size to build each tree, *i.e.*, $\psi = 2^h$. The sensitivity analysis of the two parameters were conducted in the ISCX-URL dataset. To analyze the effect of one parameter, the other parameter was set to the default value. When h was varied from 1 to 8, t was set to the default value of 100; and when t was varied from 10 to 1000, h was set to the default value of 5.



(a) Tree Height (b) Number of trees

Figure 7. Sensitivity of two parameters in usfAD: tree height (h) and number of trees (t)

The AUCs of usfAD with different settings of h and t in the ISCX-URL dataset are provided in Figure 7. In both cases, with the increase in the parameter value, the AUC of usfAD improved significantly for the first few lower values and then gradually flatten out towards the higher values. These results indicate that the parameters are not very sensitive as long as they are set to sufficiently large values. Note that higher values of h and t will increase the runtime linearly. We need to trade-off performance and efficiency. For a right balance, h and t were set to 5 and 100, respectively, as default values. The default settings were used in all experiments conducted in this study. The performance of usfAD can be further improved by proper tuning of the two parameters.

5. DISCUSSION

Results in this research show that the given representation of data may not be appropriate for anomaly detection using existing distance or density-based methods. In the literature,

representation learning techniques (Bengio, Courville, & Vincent, 2013; Zhong, Wang, Ling, & Dong, 2016; Pang, Cao, Chen, & Liu, 2018) were used to map data from the given input space into a latent space that maximizes the task specific performance of a given data mining algorithm. Representation learning can be viewed as learning appropriate transformations that best suits the dataset and algorithm at hand. It is primarily used for unstructured data such as image, text, audio and video. Learning an appropriate representation for anomaly detection is very challenging because anomalies can be of different types and can appear anywhere in the data space, *i.e.*, anomalies do not have a particular set of characteristics. Furthermore, representation learning has some other issues too: (i) requires extensive learning which can be computationally expensive in large and/or high-dimensional data sets; (ii) learns representation appropriate for the given algorithm, representation learned for one algorithm may not be appropriate for others in the same data set; and (iii) one cannot interpret the meaning of new features and what type of information they capture.

Another alternative suggested in the literature to address the issue of units and scales of measurement of data is rank transformation (Conover & Iman., 1981). It is robust to how data is measured or represented because ranks/orders are either preserved or reversed when data is expressed differently. It has been shown that using ranks instead of actual values, distance or similarity based algorithms produces better results for tasks such as classification and clustering (Fernando & Webb, 2017; Aryal S. , Ting, Washio, & Haffari, 2017; Aryal S. , Ting, Washio, & Haffari, 2020). Rank transformation, however does not work for the task of anomaly detection. Because the rank differences of consecutive values are always one irrespective of the magnitude difference, the transformed data is uniformly distributed making anomalies difficult to detect.

To address the above-mentioned issue of rank transformation, Baniya et al. (2019) proposed a robust alternative to rank transformation that preserves differences between data instances in the transformed space to some extent. They use average ranks over multiple subsamples of data. Data transformation based on the Average Rank over an Ensemble of Subsamples (ARES) has been shown to improve classification accuracies of algorithms such as *k*NN, Artificial Neural Networks and Logistic Regression (Baniya, Aryal, & Santosh, 2019). However, the similar results could not be achieved in the anomaly detection task.

At the first glance, this work may appear to be related to multiscale data analysis (Bakshi, 1999; Weinan, 2011). However, they are fundamentally different concepts and they are addressing

different issues. In the literature, the term 'multiscale' is used in the context of applications/problems where data can be viewed at different hierarchical levels (scales). For example, (i) temporal data (*e.g.*, daily, monthly, yearly, *etc.*), images (pixels, shapes, objects, *etc.*), spatial data (various geographic resolutions). Multiscale data analysis examines data at multiple levels (scales in time, frequency or resolution) and integrate the information in the learning process. There are several works using multiscale data for anomaly detection in spatial and temporal domains (Cheng & Li, 2006; Siddiqui, Khan, & Ferens, 2017; Liu, et al., 2017; Gao, et al., 2019). Multiscale data analysis does not address the issue related to the form/format when data is given for analysis. The issue is also applicable while recording data at different levels. Learning algorithms may not perform well if the data at each level are not presented in an appropriate form.

6. CONCLUSIONS AND FUTURE WORK

In today's world, data is recorded by sensors everywhere around us. The sensors' readings are transferred to a server, and stored in a database before they are analyzed. In this process, the representation of data may have been changed for various reasons such as sensors' settings, compression to decrease communication and storage costs, and encoding for security reasons. When data is given for analysis, we may not know how they are measured and stored. This makes anomaly detection very challenging. Anomalies can be masked and look like normal data in the given representation and existing anomaly detection algorithms that are primarily based on distance or density may give true negative or false positive results.

This is the first work studying the issue of data representation in the context of anomaly detection. It demonstrates that the fundamental assumption made by almost all existing methods that *anomalies are few and different* can be counterproductive if data is represented inappropriately. Often in practice, we simply have a bunch of numbers and we may not know how they are represented. Therefore, we need anomaly detection algorithms that do not make such assumptions and are robust to data representation. We believe this work will change the way automatic anomaly detection problem has been studied in the past and lead to a potential paradigm shift in future anomaly detection research.

This paper introduces a simple robust anomaly detection algorithm based on unsupervised stochastic forest called usfAD. Experimental results in synthetic and real-world cyber security

datasets with different scaling of data show that it is robust to how data is measured. Because of the median split used to construct trees in the forest, its results are consistent and stable across different scaling in all datasets. The implementation is based on the ordering of data and it is robust to how data is presented as the variation in units/scales either preserves or reverses the orderings. It outperforms most existing state-of-the-art anomaly detection methods even when using the given form of the data. This result shows that some features of the given datasets may not be recorded in the appropriate form for anomaly detection using existing methods, which are sensitive to units/scales of data measurement.

In future, we would like to extend this idea to work in the clustering problem, where most existing state-of-the-art methods use distance or density. Also, we would like to explore a non-tree-based implementation of usfAD. Another potential avenue for future research would be to investigate on scale-invariant representation learning for anomaly detection. .

CONFLICTS OF INTEREST

Authors declare no conflict of interest.

ACKNOWLEDGMENT

This paper is an extension of a conference paper published in Proceedings of the 22nd Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD) 2018 (Aryal S. , 2018). Authors would like to thank Mr Arbind Agrahari Baniya for his help to run some experiments in this extended version of the paper. This material is based upon work supported by the Air Force Office of Scientific Research under award number FA2386-20-1-4005.

REFERENCES

Aggarwal, C. C. (2017). *Outlier Analysis.* Springer.

Aryal, S. (2018). Anomaly detection technique robust to units and scales of measurement. *Proceedings of the 22nd Pacific-Asia Conference on Knowledge Discovery and Data Mining*, (pp. 589-601).

Aryal, S., Baniya, A. A., & Santosh, K. (2019). Improved histogram-based anomaly detector with the extended principal component features. arxiv. Retrieved from https://arxiv.org/abs/1909.12702

Aryal, S., Ting, K. M., & Haffari, G. (2016). Revisiting Attribute Independence Assumption in Probabilistic Unsupervised Anomaly Detection. *Proceedings of the 11th Pacific Asia Workshop on Intelligence and Security Informatics*, (pp. 73-86).

Aryal, S., Ting, K. M., Washio, T., & Haffari, G. (2017). Data-dependent dissimilarity measure: an effective alternative to geometric distance measures. *Knowledge and Information Systems, 53*(2), 479-506.

Aryal, S., Ting, K. M., Washio, T., & Haffari, G. (2020). A comparative study of data-dependent approaches without learning in measuring similarities of data objects. *Data Mining and Knowledge Discovery, 34*(1), 124-162.

Aryal, S., Ting, K. M., Wells, J. R., & Washio, T. (2014). Improving iForest with Relative Mass. *Proceedings of the 18th Pacific Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, (pp. 510-521).

Bakshi, B. R. (1999). Multiscale Analysis and Modelling using Wavelets. *Journal of Chemometrics, 13*(1), 415-434.

Bandaragoda, T., Ting, K. M., Albrecht, D., Liu, F., & Wells, J. (2014). Efficient anomaly detection by isolation using nearest neighbour ensemble. *Proceedings of the IEEE International Conference on Data Mining Workshops*, (pp. 698-705).

Baniya, A. A., Aryal, S., & Santosh, K. C. (2019). A Novel Data Pre-processing Technique: Making Data Mining Robust to Different Units and Scales of Measurement. *Proceedings of the 26th International Conference on Neural Information Processing (ICONIP) of the Asia-Pacific Neural Network Society*, (p. Accepted).

Bay, S. D., & Schwabacher, M. (2003). Mining distance-based outliers in near linear time with randomization and a simple pruning rule. *Proceedings of the ninth ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, (pp. 29-38).

Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 35*(8), 1798-1828.

Boriah, S., Chandola, V., & Kumar, V. (2008). Similarity measures for categorical data: A comparative evaluation. *Proceedings of the eighth SIAM International Conference on Data Mining*, (pp. 243-254).

Breiman, L. (2001). Random Forests. *Machine Learning, 45*(1), 5-32.

Breunig, M. M., Kriegel, H.-P., Ng, R. T., & Sander, J. (2000). LOF: Identifying Density-Based Local Outliers. *Proceedings of ACM SIGMOD Conference on Management of Data*, (pp. 93-104).

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly Detection: A Survey. *ACM Computing Surveys, 41*(3), 15:1-15:58.

Cheng, T., & Li, Z. (2006). A Multiscale Approach for Spatio-Temporal Outlier Detection. *Transactions in GIS, 10*(2), 253-263.

Conover, W. J., & Iman., R. L. (1981). Rank transformations as a bridge between parametric and nonparametric statistics. *The American Statistician, 35*(3), 124-129.

Fernando, T. L., & Webb, G. I. (2017). SimUSF: An efficient and effective similarity measure that is invariant to violations of the interval scale assumption. *Data Mining and Knowledge Discovery, 31*(1), 264-286.

Gao, Z., Guo, L., Ma, C., Ma, X., Sun, K., Xiang, H., . . . Liu, X. (2019). AMAD: adversarial multiscale anomaly detection on high-dimensional and time-evolving categorical data. *Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data (DLP-KDD '19)*, (pp. 1-8).

Goldstein, M., & Dengel, A. (2012). Histogram-based Outlier Score (HBOS): A fast Unsupervised Anomaly Detection Algorithm. *Proceedings of the 35th German Conference on Artificial Intelligence*, (pp. 59-63).

Hand, D. J., & Till, R. J. (2001). A simple generalisation of the area under the roc curve for multiple class. *Machine Learning, 45*(2), 171–186.

Hawkins, D. M. (1980). *Identification of Outliers.* Chapman and Hall.

Jiang, H., Wang, H., Hu, W., Kakde, D., & Chaudhuri, A. (2017). Fast Incremental SVDD Learning Algorithm with the Gaussian Kernel. *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence (AAAI)*, (pp. 3991-3998).

Joiner, B. L. (1981). Lurking variables: Some examples. *The American Statistician, 35*(4), 227-233,.

Liu, F., Ting, K. M., & Zhou., Z.-H. (2008). Isolation forest. *Proceedings of the Eighth IEEE International Conference on Data Mining*, (pp. 413-422).

Liu, Q., Klucik, R., Chen, C., Grant, G., Gallaher, D., Lv, Q., & Shang, L. (2017). Unsupervised detection of contextual anomaly in remotely sensed data. *Remote Sensing of Environment, 202*(1), 75-87.

Lord, F. M. (1953). On the statistical treatment of football numbers. *American Psychologist, 8*(12), 750-751.

Mamun, M. S., Rathore, M. A., Lashkari, A. H., & Stakhanova, N. (2016). Detecting Malicious URLs Using Lexical Analysis. *Proceedings of the International Conference on Network and System Security (NSS 2016)*, (pp. 467-482).

Pang, G., Cao, L., Chen, L., & Liu, H. (2018). Learning Representations of Ultrahigh-dimensional Data for Random Distance-based Outlier Detection. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, (pp. 2041-2050).

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., . . . Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research, 12*, 2825-2830.

Rekha, A. G. (2015). A fast support vector data description system for anomaly detection using big data. *Proceedings of the 30th Annual ACM Symposium on Applied Computing (SAC)*, (pp. 931-932).

Scholkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., & Williamson, R. C. (2001). Estimating the Support of a High-Dimensional Distribution. *Neural Computation, 13*(7), 1443-1471.

Shi, T., & Horvath, S. (2006). Unsupervised learning with random forest predictors. *Journal of Computational and Graphical Statistics, 15*(1), 118-138.

Siddiqui, S., Khan, M. S., & Ferens, K. (2017). Multiscale Hebbian neural network for cyber threat detection. *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, (pp. 1427-1434).

Stevens, S. S. (1946). On the theory of scales of measurement. *Science, 103*(2684), 677-680,.

Sugiyama, M., & Borgwardt, K. M. (2013). Rapid Distance-Based Outlier Detection via Sampling. *Proceedings of the 27th Annual Conference on Neural Information Processing Systems*, (pp. 467-475).

Tax, D., & Duin, R. (2004). Support vector data description. *Machine Learning, 54*(1), 45-66.

Ting, K. M., Washio, T., Wells, J. R., & Aryal, S. (2017). Defying the gravity of learning curve: a characteristic of nearest neighbour anomaly detectors. *Machine Learning, 106*(1), 55-91.

Townsend, J. T., & Ashby, F. G. (1984). Measurement scales and statistics: The misconception misconceived. *Psychological Bulletin, 96*(2), 394-401,.

Velleman, P. F., & Wilkinson, L. (1993). Nominal, ordinal, interval, and ratio typologies are misleading. *American Statistician, 47*(1), 65-72,.

Weinan, E. (2011). *Principles of Multiscale Modeling* (Vol. 6). Cambridge University Pres.

Zhong, G., Wang, L.-N., Ling, X., & Dong, J. (2016). An overview on data representation learning: From traditional feature learning to recent deep learning. *The Journal of Finance and Data Science, 2*(4), 265-278,.

Authors' biographies:

Sunil Aryal, PhD: Dr Aryal is a lecturer at the School of Information Technology, Faculty of Science, Engineering and Built Environment, Deakin University Australia. Prior to joining Deakin University in 2019, he worked as a lecturer at Federation University and sessional teaching staff at various institutions. He also worked in industry as a software developer and data analyst. He received his PhD from Monash University, Australia. His research interests are in the areas of data mining, machine learning, artificial intelligence, health informatics, and information systems. He is particularly interested in applying machine learning to solve real-world problems in different domains, particularly in healthcare, cybersecurity, and IoT. He has published more than 25 scientific papers in top tier conferences and journals in the field of data mining and machine learning. He has served as a program committee member at various conferences such as AAAI, IJCAI, ECML/PKDD, PAKDD, and ICONIP.

K.C. Santosh, PhD: Dr. Santosh is an Associate Professor and Graduate Program Director of the Department of Computer Science at the University of South Dakota (USD). Also, Dr. Santosh serves School of Computing and IT, Taylor's University as a Visiting Associate Professor. Before joining USD, Dr. Santosh worked as a research fellow at the U.S. National Library of Medicine (NLM), National Institutes of Health (NIH). He worked as a postdoctoral research scientist at the LORIA research centre, Universite de Lorraine in direct collaboration with industrial partner ITESOFT, France. He also worked as a research scientist at the INRIA Nancy Grand Est research centre, France, where he has received his PhD diploma in Computer Science. Dr. Santosh demonstrated expertise in artificial intelligence, machine learning, pattern recognition, computer vision, image processing, data mining, and big data with various application domains, such as healthcare informatics and medical imaging, graphics recognition, document information content exploitation, biometrics, forensics, speech analysis, satellite imaging, robotics, and Internet of Things. He published more than 150 peer-reviewed research articles, authored 4 books, and edited more than 15 books, journal issues and conference proceedings. Dr. Santosh serves as an associate editor for multiple journals, such as Int. J. of Machine Learning & Cybernetics, and chaired more than 10 international conference events in the domain. Dr. Santosh is the proud recipient of the President's Research Excellence Award (USD, 2019) and Department of Health & Human Services (2014).

Richard Dazeley, PhD: Dr. Dazeley is an Associate Professor of Computer Science at Deakin University (Geelong). He is a highly experienced researcher in multiobjective, interactive, deep, safe and explainable Reinforcement Learning (RL). He has published several of the most widely cited papers in multiobjective RL where his work significantly contributed to the foundation of this field of research. He been recognised as an nationally leading ICT Curriculum Designer – receiving the Australian National Award as ICT Educator of the Year (2016) from the ACS and was nominated for the International ICT Educator of the Year award from the South-East Asia Regional Computer Confederation (SEARCC) (2017). Prior to working at Deakin, he was employed at Federation University Australia where he was the Head of the IT Discipline and the co- Founder and Leader of the Federation Learning Agents research Group (FLAG).