The 33rd International Conference on Industrial, Engineering &
Other Applications of Applied Intelligent Systems (IEA AIE 2020)
22-25th September 2020
Kitakyushu, Japan

POSTER PROCEEDINGS

Editors:
Hamido Fujita
Philippe Fournier-Viger
Jun Sasaki
Moonis Ali

ISBN: 978-4-901195-48-5

Preface

In recent decades, the society has entered a digital era where computers have become ubiquitous in all aspects of life, including education, governance, science, healthcare and the industry. Computers have become smaller, faster and the cost of data storage and communication have greatly decreased. As a result, more and more data is being collected and stored in databases. Besides, novel and improved computing architectures have been designed for efficient large-scale data processing such as big data frameworks, FPGAs and GPUs. Thanks to these advancements and recent breakthroughs in artificial intelligence, researchers and practitioners have developed more complex and effective artificial intelligence-based systems. This has led to a greater interest in artificial intelligence to solve real-world complex problems, and the proposal of many innovative applications.

This volume contains the **poster proceedings** of the 33rd edition of the International Conference on Industrial, Engineering, and other Applications of Applied Intelligent Systems (IEA AIE 2020), which was held from the 22nd to the 25th September in Kitakyushu, Japan. IEA AIE conference is an annual event that emphasizes applications of applied intelligent systems to solve real-life problems in all areas including engineering, science, industry, automation and robotics, business and finance, medicine and biomedicine, bioinformatics, cyberspace, and human-machine interactions. This year, 119 submissions were received. Each paper was evaluated using a double-blind peer review by at least three reviewers from an international Program Committee consisting of 82 members from 36 countries. Based on the evaluation, a total of 62 papers were selected as full papers, 17 as short papers, and 9 as poster papers. The full and short papers are published in the LNCS proceedings, while the poster papers are presented in this book.

In the program of IEA AIE 2020, two special sessions were organized named Collective Intelligence in Social Media (CISM 2020) and Intelligent Knowledge Engineering in Decision Making Systems (IKEDS 2020). Moreover, three keynote talks were given by distinguished researchers, one by Prof. Tao Wu from the Shanghai Jiao Tong University School of Medicine (China), one by Enrique Herrera Viedma from the University of Granada (Spain) and another by Ee-Peng Lim from the Singapore Management University (Singapore). Lastly, we would like to thank everyone who has contributed to the success of this year's edition of IEA-AIE, that is authors, Program Committee members, reviewers, keynote speakers and organizers.

September 2020

Hamido Fujita
Philippe Fournier-Viger
Moonis Ali
Jun Sasaki

Organization

General Chair

Hamido Fujita	Iwate Prefectural University, Japan

General Co-Chairs

Moonis Ali	Texas State University, USA
Franz Wotawa	Graz University of Technology, Austria

Organizing Chair

Jun Sasaki	Iwate Prefectural University, Japan

Program Chairs

Philippe Fournier-Viger	Harbin Inst. of Technology, China
Hideyuki Takagi	Kyushu University, Japan

Special Session Chairs

Yinglin Wang	Shanghai University of Finance and Economic, China
Ali Selamat	Universiti Teknologi Malaysia, Malaysia
Prima O.D.A.	Iwate Prefectural University, Japan

Special Session Organizers

Jerry Chun-Wei Lin	Western Norway University of Applied Sciences, Norway
Philippe Fournier-Viger	Harbin Inst. of Technology, China
Rage Uday Kiran	University of Aizu, Japan
Ngoc-Thanh Nguyen	Wroclaw University of Science and Technology, Poland
Van Du Nguyen	Nong Lam University, Vietnam

Publicity Chair

Toshitaka Hayashi	Iwate Prefectural University, Japan

Program Committee

Rui Abreu	University of Lisbon, Portugal

Otmane Ait Mohamed	Corcordia University, Canada
Hadjali Allel	ENSMA, France
Xiangdong An	University of Tennessee, USA
Artur Andrzejak	Heidelberg University, Germany
Farshad Badie	Aalborg University, Denmark
Ladjel Bellatreche	ENSMA, France
Fevzi Belli	Paderborn University, Germany
Adel Bouhoula	University of Carthage, Tunisia
Ivan Bratko	University of Ljubljana, Slovenia
João Paulo Carvalho	University of Lisbon, Portugal
Chun-Hao Chen	National taipei university of technology, Taiwan
Shyi-Ming Chen	National Taiwan University of Science and Technology, Taiwan
Flávio Soares Corrêa da Silva	University of Sao Paulo, Brazil
Giorgos Dounias	University of the Aegean, Greece
Alexander Ferrein	Aachen Univesity of Applied Science, Germany
Philippe Fournier-Viger	Harbin Inst. of Technology, China
Hamido Fujita	Iwate Prefectural University, Japan
Vicente García Díaz	University of Oviedo, Spain
Alban Grastien	Australian National University, Australia
Maciej Grzenda	Warsaw University of Technology, Poland
Jun Hakura	Iwate Prefectural University, Japan
Tim Hendtlass	School of Biophysical Sciences and Electrical Engineering. Australia
Dinh Tuyen Hoang	Yeungnam University, South Korea
Tzung-Pei Hong	National University of Kaohsiung, Taiwan
Wen-Juan Hou	National Central University, Taiwan
Ko-Wei Huang	National Kaohsiung University of Science and Technology, Taiwan
Quoc Bao Huynh	Ho Chi Minh City University of Technology, Vietnam
Said Jabbour	University of Artois, France
He Jiang	Dalian University of Technology, China
Rage Uday Kiran	University of Aizu, Japan
Yun Sing Koh	University of Auckland, New Zealand
Adrianna Kozierkiewicz	Wroclaw University of Science and Technology, Poland
Dariusz Krol	Wroclaw University of Science and Technology, Poland
Philippe Leray	University of Nantes, France
Mark Levin	Russian Academy of Sciences, Russia
Jerry Chun-Wei Lin	Western Norway University of Applied Sciences, Norway
Yu-Chen Lin	feng chia university, Taiwan
Jose Maria-Luna	University of Cordoba, Spain
Wolfgang Mayer	University of South Australia, Australia
Joao Mendes-Moreira	University of Porto, Portugal

Engelbert Mephu Nguifo	Université Clermont Auvergne, France
Mercedes Merayo	Universidad Complutense de Madrid, Spain
Abidalrahman Moh'D	Eastern Illinois University, USA
Anirban Mondal	Ashoka University, India
Saqib Nawaz	Harbin Inst. of Technology, China
Roger Nkambou	Université du Québec à Montréal, Canada
Ngoc-Thanh Nguyen	Wroclaw University of Science and Technology, Poland
Quang Vu Nguyen	Vietnam-Korea Friendship Information Technology College, Vietnam
Van Du Nguyen	Nong Lam University, Vietnam
Ayahiko Niimi	Future University Hakodate, Japan
Xinzheng Niu	University of Electronic Science and Technology of China, China
Farid Nouioua	Aix-Marseille Université, France
Mourad Nouioua	Harbin Inst. of Technology, China
Barbara Pes	University of Cagliari, Italy
Marcin Pietranik	Wroclaw University of Science and Technology, Poland
Ingo Pill	TU Graz, Austria
Matin Pirouz	California State University, USA
Krishna P. Reddy	International Institute of Information Technology, Hyderabad, India
Gregorio Sainz-Palmero	University of Valladolid, Spain
Eugene Santos Jr.	Dartmouth College, USA
Jun Sasaki	Iwate Prefectural University, Japan
Ali Selamat	Universiti Teknologi Malaysia, Malaysia
Nazha Selmaoui-Folcher	University of New Caledonia, New Caledonia
Sabrina Senatore	University of Salerno, Italy
Neal Snooke	Aberystwyth University, UK
Gerald Steinbauer	TU Graz, Austria
Ahmed Tawfik	Microsoft Research, USA
Trong Hieu Tran	Hanoi University of Engineering and Technology, Vietnam
Van Cuong Tran	Quang Binh University, Vietnam
Chun-Wei Tsai	National Sun Yat-sen University, Taiwan
Alexander Vazhenin	University of Aizu, Japan
Bay Vo	HCM City University of Technology, Vietnam
Toby Walsh	NICTA, Australia
Yutaka Watanobe	University of Aizu, Japan
Tomasz Wiktorski	University of Stavanger, Norway
Cheng Wei Wu	National Ilan University, Taiwan
Franz Wotawa	Graz University of Technology, Austria
Jimmy Ming-Tai Wu	Shandong University of Science and Technology, China
Mu-En Wu	National taipei university of technology, Taiwan
Unil Yun	Sejong University, South Korea
Wei Zhang	Adobe Systems, USA

Table of Contents

Empirical Study of K-means Clustering on Usability Performance Metrics in MAR-learning

Lim Kok Cheng[1,3], Ali Selamat[2,3,4], Mohd Hazli Mohamed Zabil[1], Md Hafiz Selamat[3], Rose Alinda Alias[3], Fatimah Puteh[3], Farhan Mohamed[3] and Ondrej Krejcar[4]

[1]*Universiti Tenaga Nasional, Selangor, Malaysia*
[2]*Malaysia-Japan International Institute of Technology, UTM KL, Malaysia*
[3]*Faculty of Computing, Universiti Teknologi Malaysia, Johor Bahru, Malaysia*
[4]*Center for Basic and Applied Research, Faculty of Informatics and Management, University of Hradec Kralove, Rokitanskeho 62, 500 03 Hradec Kralove, Czech Republic*
kokcheng@uniten.edu.my

Abstract. This paper presents empirical findings on machine learning clustering results from a bigger project relating to several related works embarking on performing machine learning algorithms on usability data. In this paper, results from applying K-means clustering on usability performance data is presented. This paper will highlight several related works followed the methodology formulated in this research. After that, the results and discussion were presented showing the feasibility and accuracy of K-means clustering on usability performance data relating to Mobile Augmented Reality (MAR) learning application. This paper contributes in proposing a model confirming the feasibility of K-means algorithm in prioritizing usability issues in MAR-learning.

Keywords: Usability, Mobile Augmented Reality Learning, English Language Teaching, Hierarchical Agglomerative Clustering, Performance Metrics.

1 Introduction

Performance metrics in usability engineering has been a common neglected measure in human computer interaction related studies. Furthermore, usability analysis often times relies on conventional comparative studies utilizing only descriptive statistical analysis without the ability of measuring usability factors independently. Due to that particular reason, the application of machine learning algorithms, specifically the clustering methods on usability data has been in its infancy. While researchers like [1] analyses the fundamentals of clustering algorithms, just very few researchers like [2] implemented machine learning analysis on usability improvements. It has been an area of interest on why has usability and machine learning have not been studied together. This project

[1] Lim Kok Cheng, Department of Computer Science and Information Technology, Universiti Tenaga Nasional, Jalan IKRAM-UNITEN, 43000, Selangor, Malaysia; E-mail: kokcheng@uniten.edu.my

therefore has embark on a quest tackling this very issues. Several works has been done in this project coupling with an emerging technology of Mobile Augmented Reality learning (MAR-learning) on English language. The several works can be referred from the content of [3] prior to this paper. The content of this article is to explore the validity of applying K-means algorithm on usability data collected through performance and self-reported metrics [4]. The focus is to re-replicate non significantly different results on 2 different datasets that goes though similar methodological procedures. The effort in this research is to strengthen confirmation on the feasibility of applying K-means on usability performance (quantitative based on users ability) and self-reported data (users qualitative and subjective satisfaction)

2 Related Works

2.1 Usability Issues in Augmented Reality Research

A study done by [5] shows that in Augmented Reality Learning Environment (ARLE) systems, usability evaluation emphasizes in producing better ease of use, satisfaction, immersion, motivation and performance. Santos et al. did an extensive systematic reviews on 43 different ARLEs with the concluded findings [5]. Santos et. al. in [5] reported that common tools used among their findings consists of observation, interviews and expert reviews among others. The methods suggested by the findings in [5] agrees with several techniques and methods earlier coined by Albert and Tullis in [4]. According to the research done by [4], most usability studies uses more self-reported metrics as compared to performance metrics. As a continuation of works done by [5], this research has further conducted studies as reported in [3]. The findings shows that in MAR-learning more than 80% of surveyed works preferred self-reported metrics over performance metrics. Despite for the fact that, there has been many arguments on the lesser reliable techniques of self-reported metrics, many researchers still cling to the comfort usage of self-reported metrics due to acceptable conveniences [6]–[8]. Both usability metrics of performance and self-reported can be combined is usability studies [9]. However, performance metrics has long been underrated in usability measures due to its technicality in implementation, even though it is much reliable. Many arguments has taken place in this matter by many researchers like [2], [4], [5], [10], [11], however it has been a never ending debate. Since the work involving performance and self-reported data has been reported in [3], [9], [12], this paper will present results on performance data only without possible interference of self-reported data.

2.2 K-means Clustering

Several clustering algorithms are used across many research areas in recent times. While K-means are generally used, partitioning based algorithms work in different fashion to obtained data analysis [13], [14]. K-means is one of the simplest unsupervised machine learning algorithms used to partition data based on locations and distance between data points [15]. K-means algorithm is reported to have worked better for large

dataset, while hierarchical clustering works better for smaller datasets [7], [16]. While K-means has been used in many research domains, the feasibility of K-means to be performed on usability performance data is still within infancy. The one reference of such research has been conducted is based on our prior work presented in [3]. In [3] however, only HAC has been performed only on combination of usability performance and self-reported data, without comparing with the performance of K-means and clustering performance on respective datasets comparatively. While works in [9] shows comparative result of HAC and K-means on usability data, individual clustering performance of either method has yet to be explored on separate usability features based on the collection method of either performance or self-reported metrics.

3 Methodology

As far as the methodology is concern, the generation of methods and techniques for this paper is to support the results presented in earlier publication of the same research team in [3], [12]. The works in [17] has layout the preliminary studies of this research project as a pre-operational framework clarification of the MAR-learning application called "InterviewME". This is succeeded by the works done in [9], deciding on the suitable clustering algorithm, where Hierarchical Agglomerative Clustering (HAC) is chosen as compared to K-means Clustering (K-means) based on the size of the datasets and several primary variables. Then, real time datasets were collected in [3], where usability prioritization were carried out for analysis of performance data based on three Mobile Augmented Reality (MAR) interaction parameters namely, Object Tracking (OT), Selecting (S) and Navigating (N). In [3], two performance metrics were used (Time-on-tasks and Error registration), while a self-reported metrics was used (5 Likert Scale) to measure participants' satisfaction level. The research work presented in this paper is objectified to explore the results attained from [3] with a different clustering algorithm, mainly K-means. The proposed methodology (Figure 1) with 3 phases is the entire process where the finding of this paper is extracted from the research activities done in Phase 3 of the project. The initial methodology has been designed to achieve these 2 objectives for phase 3:

- To evaluate the performance of K-means algorithm on usability performance data obtained through MAR object tracking.
- To get insight on the quality of K-means algorithm in clustering usability performance data obtained through MAR learning application.

The work of phase 1 has actually been presented in [17] where a repetitive pilot study has been carried out validating the usability and interactivity of the testable application called InterviewME, an MAR-learning application helping Malaysian tertiary educationist to master interview skills in English language. The development of the application has been guided by suggestions and recommendation discussed in [17]. The works of Phase 2 has generally been presented in [3], [12]. In [12], 168 students with equal gender representation has been selected, and after profiling 102 students were selected for the evaluation protocols. The selected samples will then go through a usability experimentation by carrying out object tracking activity (adapted from Real World Annotation Interaction in [5]). Usability metrics used is the standard ISO 9241-11, incorporating effectiveness, efficiency and satisfaction. Effectiveness were measured through time-on-

tasks, effectiveness were measured through error(s) registration and satisfaction were measured through engagement time (subsequent play)[12]. All students were selected using similar profiling techniques and requirements presented in [3]. Similar criteria were also used namely, years of smartphone usage to equalize device familiarity, frequency in engaging in social media applications, hourly usage of smartphone per day and has taken a specific English language course in their institution. Similar Android operating device is given to all 102 students, eliminating any possibilities of device familiarity and operational biases.

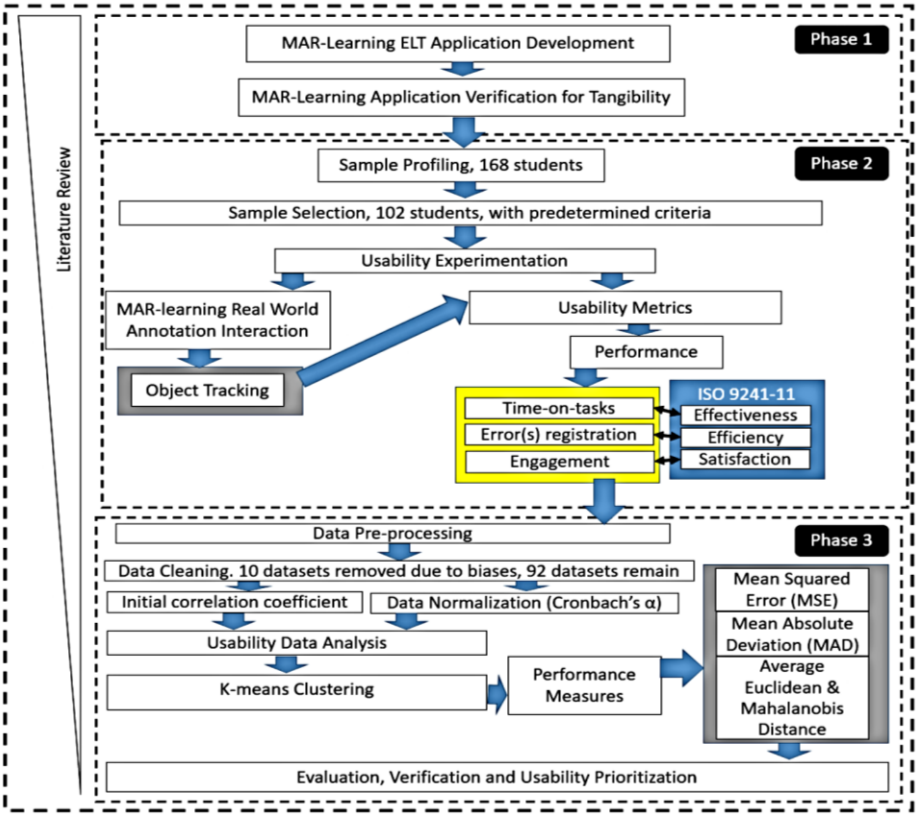

Fig. 1. Proposed methodology

The third phase involves data pre-processing. Since the initial correlation coefficient and data normalization [1] has been presented in [3], the focus of this paper is to compare only the performance measure of Correlation Coefficient (CC) [2]. Both formula of normalization and CC were explained in [1] and [2] respectively. Data cleaning were performed to prepare a low noise dataset where ambiguous rows and incomplete columns were eliminated. In this phase 10 set of results were removed due to anomalies not complying with expected criteria. Data normalization will then be performed using rescaling and evaluated through Cronbach's Alpha, where results were expected to be more than 0.70 (where reliabilities were deemed to be acceptable) [18], [19]. Correla-

tional values will also be measured through the usability metrics using correlation coefficient where results should be more than 0.30 (well-corelated) in behavioral sciences [2]. After the normalization process, Mean Squared Error (MSE), Mean Absolute Deviation (MAD) and Average Euclidean Distances (AED) (which will also be triangulated with Mahalanobis Distance Measure (MDM) [20]) were applied.

3.1 Hypothesis

Considering different discussions of how comparative results can be significantly different discussed by [1], [2], [21], this research believe that through the properly executed procedures from phase 1 to phase 3, these two hypotheses can be obtained.

- H1 – K-means clustering will achieve acceptable accuracy with low MSE, MAD, AED and MDM
- H2 – K-means clustering is proven feasible for usability data clustering by highlighting usability areas to be re-engineered.

4 Results and Discussion

Normalization is first performed on the original data set before the clustering process. With Cronbach's Alpha (α), before normalization, the α value is 0.640206 and after normalization the α value improved to 0.814953 showing better reliabilities between the 3 data arrays (Table 1). The α value for the original dataset also does not surpassed the benchmark suggested by [18] which is 0.70. After normalization, each combination of metrics were measured using correlation coefficient. All three metrics' data were more than 0.30 as proposed by [2]. For better reading comprehension, this paper will represent effectiveness metric with A, efficiency with B and engagement with C. The metric combination of AB scores a correlation coefficient value of 0.818583, AC scores a correlation coefficient value of 0.541752 and BC scores a correlation coefficient value of 0.451739. Even though the correlation value of BC is the lowest among the 3 combination, it is still above 0.30 which qualifies the combination to be analyzed through K-means clustering in the next processes. Next, the dataset combinations were clustered using Scikitlearn runnned by Python [22]. The K value of K-means needs to be determined before applying the Machine Learning techniques on them. The elbow method is applied here [15] to find the right amount of clusters for respective datasets. All usability metrics combination yields the value of optimal K to be 5, representing 5 clusters for each datasets respectively (Figure 4)

Table 1 summarizes the number of clusters and total population in each cluster within each combination. Cluster 1 shows the best performance while cluster 5 shows the worse performance in terms of metric combination.

Table 2 and table 3 respectively measure the MSE and MAD of each metric combination. For AB metric combination, the obtained MSE for vector x is 0.00827 and vector y is 0.00504, while the MAD for vector x is 0.07103 and vector y is 0.05512. While in AC metric combination, the obtained MSE for vector x is 0.08037 and vector y is 0.06025, while the MAD for vector x is 0.21219 and vector y is 0.15255. For BC metric combination, the obtained MSE for vector x is 0.01426 and vector y is 0.00949, while

the MAD for vector X is 0.09090 and vector y is 0.07954. Finally, for ABC metric combination, the obtained MSE for vector x is 0.01367, vector y is 0.00949 and vector z is 0.01598, while the MAD for vector X is 0.09385, vector y is 0.08400 and vector z is 0.10324. The results shown that the squared errors and absolute deviation from respective centroids vectors are generally low with all 4 K-means clustering indicating good performance accuracy in a nutshell. While the best performing clusters is the effectiveness/ efficiency metric combination (AB), higher errors were seen in the effectiveness and engagement metric combination (AC). Similar conclusion can be made from measuring the average Euclidean and Mahalanobis distances (Table 4), where the lowest (smaller distance means better clustering accuracy) average Euclidean and Mahalanobis scores were obtained by the effectiveness/ efficiency metric combination (AB), stating averages at 0.098179 and 0.096772 respectively. On the other hand the highest distances were obtained through the effectiveness and engagement metric combination (AC) scoring 0.288787 and 0.307562 respectively. The efficiency/ engagement (BC) scores 0.139947 and 0.139856 on both average distance measures, while the effectiveness, efficiency and engagement (ABC) metric combination scores 0.182280 and 0.139856 respectively. While AC scores the biggest distance measures, the overall distance measures were still acceptable in conforming the accuracy of the K-means clustering performance in this study. The results therefore accept the first hypothesis indicating the performance of K-means on usability performance data obtained through MAR learning application. On the other hand, referring back to Table 3, showing different percentages of clusters population, we can summarize that after the clustering process, data set from Group AC reveals the biggest population in the weakest usability cluster (Cluster 5), which stands at 25 samples (25.2%). Although the cluster differences are less significant as compared to Hierarchical Agglomerative Clustering reported in [3], [12], prioritization is still possible indicating that usability problems existed in the effectiveness and engagement metric combination, which can provide insights to design engineer to re-engineer the highlighted problems. With the ability of highlighting weak usability areas, the second hypothesis can be confirmed.

Table 1: Clusters distribution for usability metric combination

Datasets	**Number of Distribution/ Centroids Coordinates**				
	Cluster 1	**Cluster 2**	**Cluster 3**	**Cluster 4**	**Cluster 5**
Group AB	8 (8.6%)	28 (30.4%)	15 (16.3%)	21 (22.8%)	20 (21.7%)
Group AC	21 (22.8%)	12 (13.0%)	12 (13.0%)	22 (23.9%)	25 (27.2%)
Group BC	27 (29.3%)	12 (13.0%)	23 (25.0%)	19 (20.7%)	11 (12.0%)
Group ABC	10 (10.9%)	24 (26.1%)	21 (22.8%)	20 (21.7%)	17 (18.5%)

Table 2: Mean Squared Error for Metric Combinations

Metric Combination	**Mean Squared Error (MSE)**		
Vector	**x**	**y**	**z**
Effectiveness/ Efficiency (AB)	0.00827	0.00504	-
Effectiveness/ Engagement (AC)	0.08037	0.06025	-
Efficiency/ Engagement (BC)	0.01426	0.00949	-
Effectiveness/ Efficiency/ Engagement (ABC)	0.01367	0.01133	0.01598

Table 3: Mean Absolute Deviation for Metric Combinations

Metric Combination	**Mean Absolute Deviation (MAD)**		
Vector	**x**	**y**	**z**
Effectiveness/ Efficiency (AB)	0.07103	0.05512	-
Effectiveness/ Engagement (AC)	0.21219	0.15255	-
Efficiency/ Engagement (BC)	0.09090	0.07954	-
Effectiveness/ Efficiency/ Engagement (ABC)	0.09385	0.08400	0.10324

Table 4: Average Euclidean and Mahalanobis Distances

Metric Combination	**Euclidean Distance Mean**	**Mahalabonis Distance Mean**
Effectiveness/ Efficiency (AB)	0.098179	0.096772
Effectiveness/ Engagement (AC)	0.288787	0.307562
Efficiency/ Engagement (BC)	0.139947	0.139856
Effectiveness/ Efficiency/ Engagement (ABC)	0.182280	0.139856

5 Conclusion and Future Works

The results and discussion in the previous section re-affirms the implementation, feasibility and performance of K-means clustering technique on usability performance data obtained through MAR learning application. The presented outcome also shows that K-means is an acceptable algorithm to be implemented on usability data through empirical measures. Moving forward, this research will look into more clustering techniques to be performed on similar dataset as a comparison searching for the most and moderately suitable clustering techniques specifically used for usability performance data within the MAR learning domain.

References

[1] A. Bouguettaya, Q. Yu, X. Liu, X. Zhou, and A. Song, "Efficient agglomerative hierarchical clustering," *Expert Syst. Appl.*, vol. 42, no. 5, pp. 2785–2797, Apr. 2015, doi: 10.1016/j.eswa.2014.09.054.

[2] A. Oztekin, D. Delen, A. Turkyilmaz, and S. Zaim, "A machine learning-based usability evaluation method for eLearning systems," *Decis. Support Syst.*, vol. 56, pp. 63–73, Dec. 2013, doi: 10.1016/j.dss.2013.05.003.

[3] L. K. Cheng *et al.*, "Usability Prioritization Using Performance Metrics and Hierarchical Agglomerative Clustering in MAR-Learning Application.," in *SoMeT*, 2017, vol. 297, pp. 731–744, [Online]. Available: http://dblp.uni-trier.de/db/conf/somet/somet2017.html#ChengSZSAPMK17.

[4] W. Albert and T. Tullis, *Measuring the user experience: collecting, analyzing, and presenting usability metrics*. Newnes, 2013.

[5] M. E. C. Santos, A. Chen, T. Taketomi, G. Yamamoto, J. Miyazaki, and H. Kato, "Augmented reality learning experiences: Survey of prototype design and evaluation," *IEEE Trans. Learn. Technol.*, vol. 7, no. 1, pp. 38–56, 2014.

[6] J. Sandberg, M. Maris, and K. de Geus, "Mobile English learning: An evidence-based study with fifth graders," *Comput. Educ.*, vol. 57, no. 1, pp. 1334–1347, 2011.
[7] O. Tanaseichuk *et al.*, "An efficient hierarchical clustering algorithm for large datasets," *Austin J. Proteomics Bioinforma. Genomics*, vol. 2, no. 1, 2015.
[8] J. Boase and R. Ling, "Measuring Mobile Phone Use: Self-Report Versus Log Data," *J. Comput.-Mediat. Commun.*, vol. 18, no. 4, pp. 508–519, Jul. 2013, doi: 10.1111/jcc4.12021.
[9] K. C. Lim, A. Selamat, R. A. Alias, M. H. M. Zabil, F. Puteh, and F. Mohamed, "Measuring the Feasibility of Clustering Techniques on Usability Performance Data," *Indian J. Sci. Technol.*, vol. 11, no. 4, Jan. 2018, doi: 10.17485/ijst/2018/v11i4/121089.
[10] A. Dünser and M. Billinghurst, "Evaluating augmented reality systems," in *Handbook of augmented reality*, Springer, 2011, pp. 289–307.
[11] A. Dix, *Human-computer interaction*. Springer, 2009.
[12] L. K. Cheng *et al.*, "Feasibility Comparison of HAC Algorithm on Usability Performance and Self-Reported Metric Features for MAR Learning," in *SoMeT*, 2018.
[13] M. E. Celebi, H. A. Kingravi, and P. A. Vela, "A comparative study of efficient initialization methods for the k-means clustering algorithm," *Expert Syst. Appl.*, vol. 40, no. 1, pp. 200–210, Jan. 2013, doi: 10.1016/j.eswa.2012.07.021.
[14] A. Fahad *et al.*, "A Survey of Clustering Algorithms for Big Data: Taxonomy and Empirical Analysis," *IEEE Trans. Emerg. Top. Comput.*, vol. 2, no. 3, pp. 267–279, Sep. 2014, doi: 10.1109/TETC.2014.2330519.
[15] V. T., "Performance based analysis between k-Means and Fuzzy C-Means clustering algorithms for connection oriented telecommunication data," *Appl. Soft Comput.*, vol. 19, pp. 134–146, Jun. 2014, doi: 10.1016/j.asoc.2014.02.011.
[16] M. Kaur, "Comparison between k-means and hierarchical algorithm using query redirection," *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol. 3, no. 7, pp. 1454–1459, Jul. 2013.
[17] L. K. Cheng, A. Selamat, R. A. Alias, F. Puteh, and F. bin Mohamed, "InterviewME: A Comparative Pilot Study on M-Learning and MAR-Learning Prototypes in Malaysian English Language Teaching," in *Computational Intelligence in Information Systems*, 2016, pp. 219–234, doi: 10.1007/978-3-319-48517-1_20.
[18] S. Martin *et al.*, "Compensating for Memory Losses throughout aging: Validation and Normalization of the Memory Compensation Questionnaire for Non-Clinical French Populations," *Arch. Gerontol. Geriatr.*, vol. 60, no. 1, pp. 28–38, 2015, doi: 10.1016/j.archger.2014.10.013.
[19] A. Bouguettaya, Q. Yu, X. Liu, X. Zhou, and A. Song, "Efficient agglomerative hierarchical clustering," *Expert Syst. Appl.*, vol. 42, no. 5, pp. 2785–2797, Apr. 2015, doi: 10.1016/j.eswa.2014.09.054.
[20] A. Chokniwal and M. Singh, "Faster Mahalanobis K-means clustering for Gaussian distributions," in *2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2016, pp. 947–952, doi: 10.1109/ICACCI.2016.7732167.
[21] A. Joshi and R. Kaur, "A Review: Comparative Study of Various Clustering Techniques in Data Mining," *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol. 3, no. 3, pp. 55–57, Mar. 2013.
[22] M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter, "Efficient and Robust Automated Machine Learning," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 2962–2970.

A multi-agent model for countering terrorism

Oussama Kebir[1], Issam Nouaouri[2], Mouna Belhadj[3], and Lamjed Bensaid[1]

[1] Smart Lab, Institut Supérieur de Gestion de Tunis, Université de Tunis, Tunisia http://www.isg.rnu.tn/
[2] LGI2A, Univ. Artois, EA 3926, Béthune 62400, France http://www.univ-artois.fr/
[3] Institut Supérieur des Etudes Technologiques de Béja , Béja, Tunisia

Abstract. The rise of terrorism over the past decade did not only hinder the development of some countries, but also it continues to destroy humanity. To face this concept of an emerging crisis, every country and every citizen is responsible for the fight against terrorism. As conventional plans have become unusual against terrorism, governments are required to establish innovative concepts and technologies to support units in this asymmetric war. In this paper, we propose a multi-agent model for counter-terrorism characterized by a methodical steps and the flexibility to handle different contingency scenarios. The division of labour in our multi-agent model improves decision making and the structuring of organisational plans.

Keywords: Multi-agent model · counter-terrorism · asymmetric warfare · modeling.

1 Introduction

Counter terrorism decisions are based on real time reactions, in such a way that assumptions and projections of similar situations are replaced by actual conditions and facts that follow the chronological course of the attack. Peacetime planning facilitates crises' anticipating and refines the fast selection of a course of action. There are many forms of plans with differences in scope, complexity, and perspective. We mention that model describes how different agents react to the attack and how they organize their interactions with each other. While the plan translates these reactions into a series of strategic actions to be implemented and quantifies their impact. Actually, modelling is considered as one of the key concepts for driving the integration of the knowledge regarding the ensemble of agents.

Undoubtedly, not only bad modelling could lead to a military failure, but also the bad execution and the lack of military technology present major causes of loss. Modern war is becoming more and more complex and thus requires high-quality technologies. Besides, face to the high costs of major weapon systems, simulation of combats brings the solution for a reduced army budget. By saving material losses and reducing the need for costly prototypes and hazardous testing, simulation is considered the key factors for achieving military goals with the lowest costs.

The mismatch between capabilities of intervenants, types and goals of mission is generally due to the failure to provide an adequate model rather than to recognize and understand the enemy doctrine. Hence, we aim to create a coherent organizational intervention model that abides by the strategic, operational and tactical rules. This model supports decision makers with an integrated tool utilizing different modelling techniques and handling different planning levels on various spatial and temporal scales.

To do, we suggest, in this paper, a multi-agent model that outlines the role of every actor in order to manage the huge number of components and information of the multidimensional counter terrorism scenarios. Our model enriches the decision makers goals in order to provide fruitful cooperation among agents. On the basis of this model, we design a strategy for supporting cooperation of agents operating in a multi-agent network.

The paper is structured as follows. The next section presents our Multi-Agent Model for Countering Terrorism (MAMCT). During the second section, we analyze agents of our multi-agent model. Finally, we end with a discussion and a conclusion.

2 Description of the MAMCT model

Modelling is an important level that specifies the system. Hence, our proposed model will be executed in a virtual environment providing detailed results within an agent-based architecture. During this section, we aim to highlight our model' architecture and the mechanism of our plan for countering terrorism. Based on the activity diagram, we describe the behavior of the MAMCT model, presented in figure 1, showing the different decision paths that exist during its execution. In addition, we describe the interactive logic between the agents over time using the sequence diagram.

2.1 Model description

Our objective, in this paper, is to model the organization plan to counter a terrorist attack. Depending on the time of the attack, as well as location and the population present, the leader choice agent will analyze in a political point of view the severity of the attack. At the strategic level, those parameters can have a more optimal effect than other ones, thus a quick reaction produces an effective gain, and can sustain confidence in the state functioning and authority.

In our case, the leader choice agent sends to the operative agent the time proposed for this mission. Concurrently, a predefined plan will be executed at the tactical level; the attack location is the only input parameter that runs this tactical process. This plan organized a crack anti-terrorist reaction unit which could reportedly be assembled and deployed anywhere in the field without strategic or operational directives.

The operative agent classifies the level of emergency of the terrorist attack after providing a global vision on the situation in the operational areas. Activities

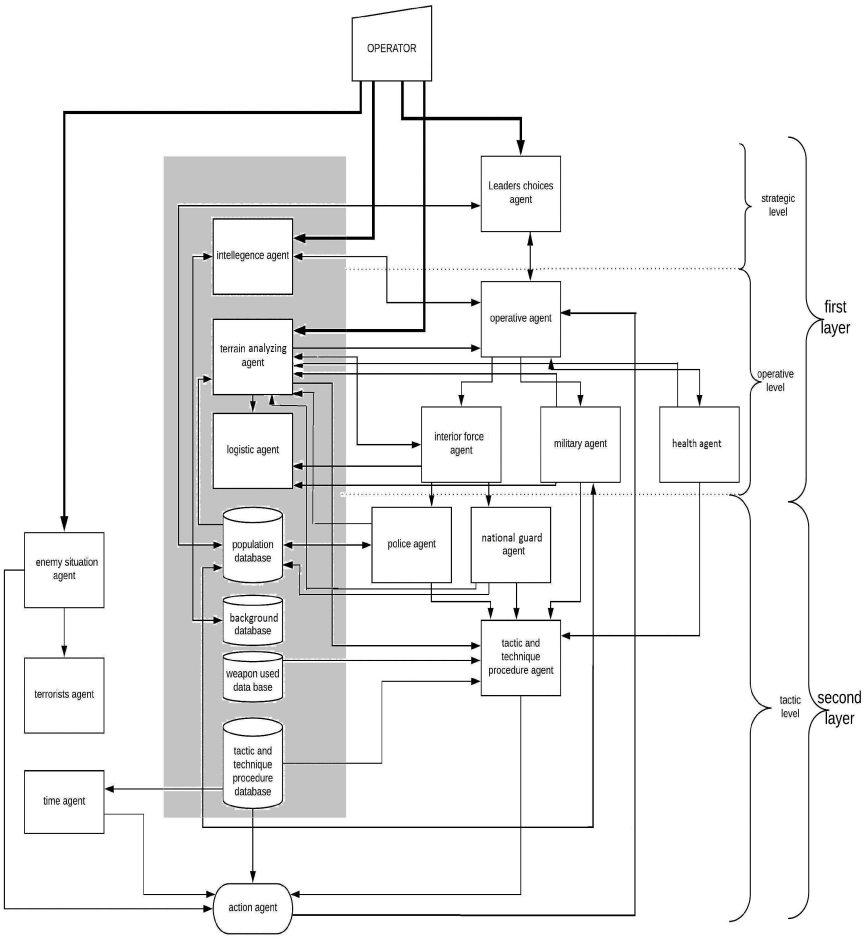

Fig. 1. MAMCT Model

of this agent match the results obtained from the intelligence agent and the terrain analyzing agent to establish a conventional vocabulary about the attack emergency level to achieve the tactical objectives, initiating actions, and applying resources to bring about and sustain potent countering actions.

The classification of the level of emergency will be sent to the military, health and interior force agents, which helps them to choose the optimal number of interveners for the mission. After sending automatically the number of soldiers selected by the predefined plan, the military agent focuses on filling the gap of the number needed for the mission, which reinforces of the predefined plan choices. Moreover, the same process will be carried out by the health agent to contribute to the exact number of ambulances needed.

The distinct character of the mission needs a territorial distribution of the operation area. The interior force agent, based on territorial distribution from the terrain analyzing agent, will select the main agent to neutralize terrorists and all interior forces bases concerned by the operation. Furthermore, it allots the number needed of policemen and National Guard men to each agent concerned. Those data will be sent to the police agent and the National Guard agent, to create a tempo-dispatching of intervenient based on the number of soldiers available in the bases.

The tactic and technique procedure agent collects the results from the police, National Guard, health and military agents. Then, it consists at dispatching every intervenient to his position in the field. Finally, this agent makes the tactic choices in terms of geographical positioning to give us the optimal solution to contain the situation and limit the terrorists' ability.

The action agent summarizes the model, he deals with how engagements are conducted using tactic and techniques database; we suppose that he combines the terrorist's deployment plan provided by the enemy situation agent and the soldiers' geographical positioning from the technique and tactic procedure agent in order to neutralize the terrorists with minimum losses.

2.2 Agents description

We mentioned that MAMCT is composed by fifteen agents and four databases. Throughout this section, we enumerate and discuss the roles of each agent.

Intelligence agent Intelligence agent gives estimation about the number of victims in the future based on:

- Number of victims in the real time
- Number of terrorists
- Location of the attack
- Population in the place of attack

This agent provides the operative agent with a classification of terrorists based on the subsequent information:

- Type of attack: isolated or organized (operator)

- Possible weapons used in the attack (operator)
- Backgrounds of terrorists and religion affiliation (background database)

Enemy situation agent The enemy situation agent is responsible for distributing terrorists on the field during the action; to be more sophisticated he takes in charge the tactical phase of the enemy; these tactic reactions depend on the terrorists' type and professionalism (intelligence agent) and the way of access to the battle field (terrain analyzing agent).

Leaders choices agent Leaders choices agent represents the strategic level of the mission: decisions of the political leaders are based on:

- Period: The period of the attack has an important effect on the number of civilians in the targeted place
- Location: The impact on the media and strategic decisions are related with the location of the attack

In our model, this agent intervenes only to impose the deadline of the mission based on the previous three factors.

Terrain analyzing agent The Terrain analysis agent receives input data from the operator, then gives us information about:

- The best access ways from and to the place of the attack (based on time): decisional and organizational aspect.
- Key points in the field, Concept of the operation, Avenues of approach, obstacles.

Logistic agent The purpose of this agent is to provide logistical support to units at the right time and place. It must therefore make decisions on the level of logistics required based on :

- Duration of the mission
- Number of soldiers needed for the mission

After that, it has to choose the right course of actions for the execution using information about location from the terrain analyzing agent. Outputs are sent to the military and interior forces agents. The logistic agent is not always used because generally missions against terrorist attacks are limited in time, so they do not need a lot of logistic sources.

Operative agent The aim of this agent is to identify a categorization of the level of emergency creating a common language to produce a system that allows a quick reaction between all intervening actors. The classification is based on:

- Classification of terrorists

- Estimation of the evolution of the number of victims
- Estimated time for the mission
- Location

This categorization provides the military, health, and interior forces agent with a summary of information about the attack.

Interior Force agent The interior force agent's responsibility is to assign the duties of the Ministry of the Interior according to the geographic location of the attack and the time factor based on the strategy of the action plan. Based on the attack classification, this agent will specify the

- Number of police men and national guard needed
- Bases concerned with the intervention

Military agent In urban areas, military intervention acts are well determined; soldiers usually build a support belt whose goal is to help interior forces to neutralize the enemies. The military agent receives the classification of the attack from the operative agent then designates the number of soldiers needed from every base. This agent gives the best distribution and planning of soldiers during the mission in accordance with the terrain analyzing agent and the population database.

Health agent In case of a terrorist attack, hospitals must be able to receive injured individuals for medical and surgical treatments. For these reasons, different medical resources optimization is fundamental to save human lives. This agent represents the distributions of available ambulance in time and the capacity of hospitals near the place of attack. The health agent is a wide topic of research, so we are limited to the management of ambulance and their distribution in the area of the terrorist act.

Police agent In accordance with the terrain analyzing agent, this agent's purpose is to provide the sufficient police force requested by the interior force agent from the police force available in the bases concerned with the intervention. The available number of policemen will be extracted from the population database.

National Guard agent This agent has the reel number of National Guard forces available in bases; his work consists in providing the number needed by the interior force agent.

Time agent The time agent is responsible for the time organization; it assigns the time needed for every step at the model' tactic level. This agent replicates the Mission time scheduling agent that was used in the ACOMSIM model.

Action agent As the last agent, the Action agent output's cover optimal decisions about the

- Planned execution (tactic plan)
- Concept of operation
- Loss probability

Tactic and technique procedure agent This agent manages the deployment of forces to the theatre of operations in order to gain an advantage over the enemy based on inputs from the terrain analyzing, military, National Guard and health agents.

Terrorists agent This agent focuses upon the notion of action and reaction in terrorist's behavior and modeling what terrorists do in combat referring to instructions from the enemy situation agent.

3 Conclusion and future work

Studying terrorist organizations' behaviour presents a topic of widespread interest within academic research and practice. Due to the sensibility of this kind of missions and the inability to make some experiments in real-life systems, we proposed, in this paper, a new multi-agent model for counting terrorism called MAMCT, encoding *Multi-Agent Model for Counting Terrorism*. This model allows leaders from all levels to understand the main purpose of the whole terrorist missions with adapting and using the guidelines to perform an efficient coordination along with making the best order decision.

The purpose behind designing our new MAMCT model is to predict and guide the planning of targeted actions and tasks in order to make the most accurate combination of effects in term of time, scope, and purpose.

Within terrorist acts, the relationship between actors is described by our MAMCT model through interaction analysis task assignment. The architecture of the stakeholders connection as an organization or person modelled is derived from the hierarchical structure of vertical interactions in the plan. Our MAMCT architecture provides a simplified reasoning architecture which obeys the basic military planning concepts. In fact, MAMCT architecture eases synchronization between commanders and their subordinate unit to link the concept of strategic level of operations with their operational design structure. Besides, it refers to different agents, where every one regards some specific role, along with a set of databases, which aim to store some important knowledge.

In a nutshell, our proposed model presents the first indispensable step to create a multi-agent system. In the future, we aim to simulate our proposal, using a multi-agent platform, and compare results with other models's simulations, from the literature, so as to provide more interest to the specificity of irregular conflicts. We opt, also, to highlight a better understanding of the future armed conflicts through achieving qualitative and quantitative analysis using statistical and artificial intelligence tools.

References

1. Ancker, C.J., Burke, M.D.: Doctrine for Asymmetric Warfare. Military Review (August) (2003)
2. Cil, I.: Mabsim : A multi agent based simulation model of military unit combat turkish general staff department of force development and resource management pp. 731–736 (2009)
3. Cil, I., Mala, M.: A multi-agent architecture for modelling and simulation of small military unit combat in asymmetric warfare. Expert Systems with Applications **37**(2), 1331–1343 (2010). https://doi.org/10.1016/j.eswa.2009.06.024
4. Deitchman, S.J.: Limited war and American defense policy. The MIT Press (1964)
5. Gowlett, P.: Moving forward with computational red teaming
6. Johnson, D.: Darwinian selection in asymmetric warfare: the natural advantage of insurgents and terrorists. Journal of the Washington Academy of Sciences pp. 89–112 (2009)
7. Johnson, D.D.P., Madin, J.S.: Population models and counterinsurgency strategies. Darwinian Security: Perspectives from Ecology and Evolution pp. 159–185 (2008)
8. Kress, M., Szechtman, R.: Why defeating insurgencies is hard: The effect of intelligence in counterinsurgency operations—a best-case scenario. Operations Research **57**(3), 578–585 (2009)
9. Lanchester, F.W.: Aircraft in warfare: The dawn of the fourth arm. Constable limited (1916)
10. Lepingwell, J.W.R.: The laws of combat? Lanchester reexamined. International Security pp. 89–134 (1987)
11. Makowsky, M.D., Rubin, J.: An agent-based model of centralized institutions, social network technology, and revolution. PloS one **8**(11), e80380 (2013)
12. Pechenkina, A.O., Bennett, D.S.: Violent and non-violent strategies of counterinsurgency. Jasss **20**(4) (2017). https://doi.org/10.18564/jasss.3540
13. Scheffran, J., Brzoska, M., Kominek, J., Link, P., Schilling, J.: Climate change and violent conflict. Science(Washington) **336**(6083), 869–871 (2012)
14. Sudhir, M.: Asymmetric War : A Conceptual Understanding. Journal of the Center for Land Warfare Studies pp. 58–66 (2008)
15. Taylor, J.G.: Force-on-force attrition modelling. Military Applications Section, Operations Research Society of America (1981)
16. Wooldridge, M., Street, C., Manchester, M., Jennings, N.R.: Intelligent Agents : Theory and Practice **10**(January), 1–62 (1995)
17. Yang, A., Abbass, H.A., Sarker, R.: Evolving agents for network centric warfare. In: Proceedings of the 7th annual workshop on Genetic and evolutionary computation. pp. 193–195. ACM (2005)

Recognition and Generation of Drawing Figures Using Neurodynamical Model Considering Pen Lifting

Yusuke Kubono, Shun Nishide, Xin Kang, and Fuji Ren

Faculty of Science and Technology, Tokushima University
2-1 Minami-Josanjima-cho, Tokushima, 770-8502 Tokushima, Japan

Abstract. In this paper, we propose a method to learn and generate drawing figures using a neurodynamical model with consideration of pen lifting. As the learning model, a recurrent neural network, namely Multiple Timescale Recurrent Neural Network (MTRNN) is used. The model is modified to adapt to pen lifting, where the drawing data could not be obtained. the proposed method predicts the pen state in addition to drawing sequence to self-organize the drawing dynamics. First, MTRNN is trained using drawing sequences, consisting of pen coordinate and pen state, obtained during drawing of several figures. Next, a drawing sequence is input into MTRNN for recognition. The recognition process is done by calculating a vector representing the drawing sequence in the self-organized dynamics space. The vector is then input into MTRNN to generate the sequence. Experiments were conducted with three types of drawing figures. The result of the experiment show that the model is capable of recognizing drawing motions with pen lifting, and generating the drawing sequences.

Keywords: Recurrent Neural Network, Drawing Sequence Learning

1 Introduction

Understanding of human development is one of the most desired goals in many research fields. One such field is cognitive developmental robotics [1], where a developmental model inspired by human development is incorporated into a robot. The development of the robot is analyzed to discover new findings about human development, while, on the other hand, such works also contribute to realization of developmental robots. Due to the development of deep learning methods, many ongoing researches are developing rapidly.

In our research, we take part in cognitive developmental robotics through development of drawing motions. Drawing is a dynamic skill with many knowledge about it's development in humans. For example, Luquet claims that human infants undergo a five stage development process in developing their drawing skills [2]. Based on this finding, we are currently creating a developmental humanoid drawing robot which starts from completely random motions, to develop it's drawing skill through human interaction [3]. A problem in the work was dealing

with pen lifting motions as data could not be acquired when the pen is lifted from the drawing canvas. Therefore, only basic figures that could be drawn with a single stroke were used as drawing targets.

In this paper, we propose a method to deal with pen lifting motions during drawing motions in order to develop the model to learn complex figures. The proposed method learns the pen state, whether it is lifted from the canvas or not, along with the drawing sequence. For the dynamics learning model, Multiple Timescale Recurrent Neural Network (MTRNN) is utilized to learn and imitate the drawing dynamics. Drawing sequences are trained using the model to self-organize the dynamics of each sequence. The pen state is included into the sequence to determine the timing to change the pen state from drawing to pen lifting or vise versa. Experiments with drawing sequences of three shapes have shown the model's capability to recognize and generate sequences that require pen lifting motions.

The rest of the paper is organized as follows. In Section 2, we present related works. In Section 3, the proposed method is presented. In Section 4, the setup of the experiment is described. In Section 5, the results of the experiment with discussion are presented. Conclusions and future work are presented in Section 6.

2 Related Works

In this section, we present related works based on two categories, works on imitation learning and works on drawing sequence learning.

2.1 Imitation Learning

Imitation learning is said to be one of the powerful techniques for robots to acquire new skills in the field of cognitive developmental robotics. A variety of models such as probabilistic models, reinforcement learning, Dynamic Movement Primitives (DMP), recurrent neural networks, etc. are used in imitation learning. This section introduces these works.

A typical probablistic model used in imitation learning is Hidden Markov Model (HMM). Sugiura et al. used HMMs to learn object manipulation motions of a robot such as rotating an object or placing an object on top of another by modifying the motion according to the reference point of the operation target [4]. The reference points were used to adapt to objects placed in different positions modifying the robot's manipulation motion based on these points.

Reinforcement learning has also been proven to be a powerful model for imitation learning. Kober et al. created various motions through development of robot motions by reinforcement learning, based on several predesigned motor primitives (basic motions) created by manually moving the robot [5]. Analysis of the model has shown that reinforcement learning could smoothly connect motor primitives to create complex motions, whereas simply connecting the motor primitives would result in accumulation of errors during the whole motion.

Recurrent neural networks are also popular models for learning dynamic sequences for imitation learning. Yokoya et al. proposed an imitation learning method between a human subject and robot, based on projection of a self-model, focusing on the triangular relation between robot, human, and object [6]. The generalization capability of neural networks has shown that the robot is capable of discovering imitable others through imitation of object manipulation motions.

2.2 Drawing Sequences Learning

Learning of drawing sequences is currently one of the hot topics due to development of deep learning models. The works can be mainly divided into two categories, learning of human drawings and creation of drawing robots.

Convolutional neural networks are one of the main models for learning sketch drawings. Simo-Serra et al. used fully convolutional networks composed of down-sample layers and up-sample layers to input rough sketches for outputting high quality vectorized simplifications [7]. The model focuses on the static structure of the sketch drawing to obtain vector representations of the drawing. A model focusing more on the dynamic structure of the drawing is proposed by David et al. [8]. The model is composed of a Long Short-Term Memory (LSTM) network to learn the handwriting dynamics of sketch. These work show the capability of deep learning models to learn static and dynamic structures of human drawings.

Researches on creating of drawing robots have also been conducted. Kulvicius et al. focused on Dynamic Movement Primitives (DMP) to reproduce handwritten characters by modifying and connecting DMP motions [9]. The work was evaluated in a simulation environment and not with an actual robot. Kudoh et al. implemented a drawing model into an actual arm robot to draw real objects [10]. The work used a stereo camera system to obtain the 3D model of the object, and draw the outline of the object using the robot.

3 Proposed Method

In this work, we utilize Multiple Timescale Recurrent Neural Network (MTRNN) as the learning model [11]. An overview of MTRNN and the modification of the model to adapt to pen lifting motions is presented in the following subsections.

3.1 Multiple Timescales Recurrent Neural Network (MTRNN)

MTRNN is a two-layered (input and output) recurrent neural network that inputs the neuron values of the current time step, calculating the neuron values of the next time step. The two layers of MTRNN form an identical structure representing the state or context values of each time step. Each layer of MTRNN consist of three types of neurons: IO neurons, C_f (Fast context) neurons, and C_s (Slow context) neurons. A parameter named time constant is set to each neuron set to determine the ratio of retaining the previous value of the neuron (similar to the forget gate in LSTM). A high time constant value will retain

a larger proportion of the previous value, while a time constant value close to zero would retain only a small proportion of the previous value. In this model, the time constant increases in the order of IO, C_f, C_s. MTRNN is capable of learning different levels of information within the model due to this hiearchical structure. The structure of MTRNN is shown in Fig. 1.

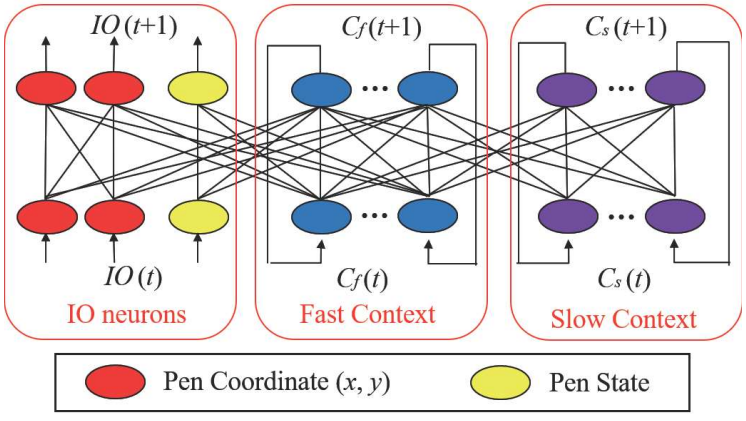

Fig. 1. Structure of MTRNN

There are three functions in MTRNN: training, recognition, and generation. In training, a training dataset is fed to the network through the IO neuron set. The output values of each time step is calculated through forward calculation. After the output values have been calculated, the Back Propagation Through Time (BPTT) algorithm is used to update the weight values of the network. In addition, the initial value of C_s ($C_s(0)$), which is independently set for each training sequence, is updated. Forward calculation and BPTT are recursively done until the output error converges. In recognition, a sequence data is input into MTRNN to calculate the $C_s(0)$ which represents the sequence. A similar calculation as training is conducted, but the weights are fixed and only the $C_s(0)$ is updated during the iteration. In generation, a $C_s(0)$ is input into MTRNN to calculate the sequence represented by $C_s(0)$. Forward calculation is used to calculate the output values which are recursively input into MTRNN to calculate the output values of the following steps. As such, recognition acts as an encoder of the sequence, while generation acts as a decoder. Detailed formulation of the calculation can be found in [11].

3.2 Proposed Method: Modification of MTRNN to Adapt to Pen Lifting Motions

In previous models, the IO neurons only consist of two pen coordinate neurons representing the x and y coordinates of the pen position. A pen tablet was used

to acquire the pen coordinates during drawing. Since the pen coordinate cannot be obtained when the pen is lifted, previous models could not adapt to pen lifting motions.

The proposed method attaches a pen state neuron in the IO neuron set to predict the state of the pen (whether it is lifted or drawing). The pen state neuron is trained to fire when the pen is lifted. To adapt to data deficiency of pen coordinate resulting from pen lifting, the output of the pen coordinate neurons in the previous time step is used as input of the pen coordinate neurons in the current time step. Thus, the model is capable of estimating the pen state along with the pen coordinate sequence.

Data deficiency due to pen lifting is dealt by the following ways in training, recognition, and generation. In training and recognition, the output value of the previous time step is used to interpolate the current input value in forward calculation. In BPTT, the output errors are calculated using the accumulated back propagated error as are done with context neurons. In generation, drawing is only conducted when the pen state is inactive.

4 Experimental Setup

To evaluate the proposed method, an experiment was conducted using data sequences obtained using a Liquid Crystal Display (LCD) Tablet Cintiq13HD (DTK-1301 / KO) developed by Wacom corporation. A human subject was asked to draw three types of figures, flower, candy, and butterfly, as shown in Fig. 3. Each figure was drawn 10 times in the order of the arrows shown in Fig. 3, for a total of 30 data. During drawing, the pen coordinate (x, y) and pen state (0 or 1) sequences are obtained at 10 frames/sec. The 30 sequences were used to train MTRNN.

Fig. 2. LCD tablet for drawing

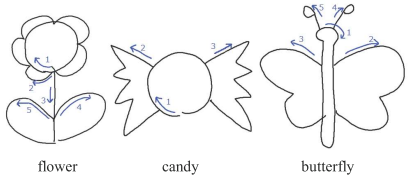

Fig. 3. Examples of Training Data

The number of neurons in MTRNN has been decided heuristically. Since the pen coordinate is two-dimensional (x and y), the number of IO neurons corresponding to pen coordinate was set to two. The pen coordinate normalized

to [0, 1] is input into each neuron. The number of IO neurons corresponding to pen state is one. The number of C_f neurons was set to 50 for each shape, and the number of C_s neurons was set to 10, 20, and 10, for flower, candy, and butterfly, respectively. The sigmoid function was used as the activation function for each neuron, and the time constants for IO neurons, C_f neurons, and C_s neurons were set to 2, 5, and 70, respectively.

In the experiment, MTRNN was trained using ten drawing sequences of each shape. After training, each sequence was used for recognition, calculating the $C_s(0)$ that represents the sequence. The $C_s(0)$ is input into MTRNN for generation, to calculate the drawing sequence that is represented by the $C_s(0)$. Evaluation is done based on how well the generated drawing sequence is reconstructed compared to the original sequence.

5 Experimental Results

In this section, some examples of the experimental results are presented. Out of the ten generated sequences for each shape, we present the best generated result (with the smallest error) and the worst generated result (with the largest error). Figure 4 presents the best generated sequence compared to the original sequence for each shape, while Fig. 5 presents the worst generated sequence.

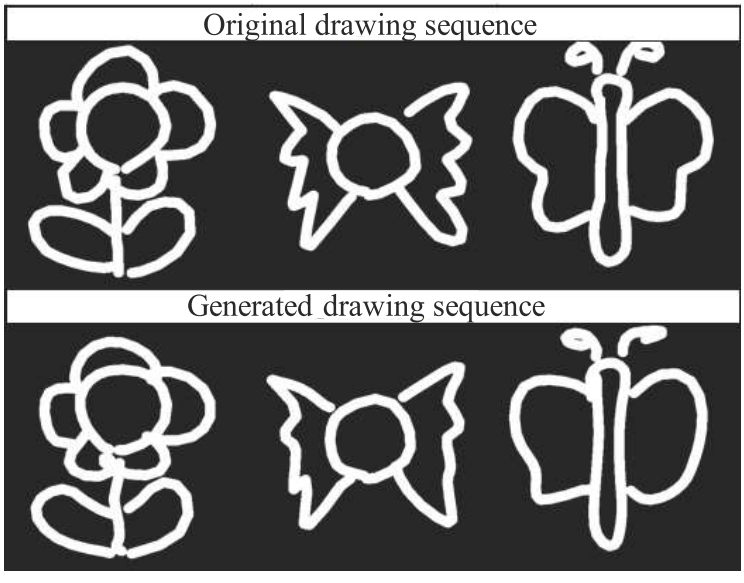

Fig. 4. Best Generated Sequences

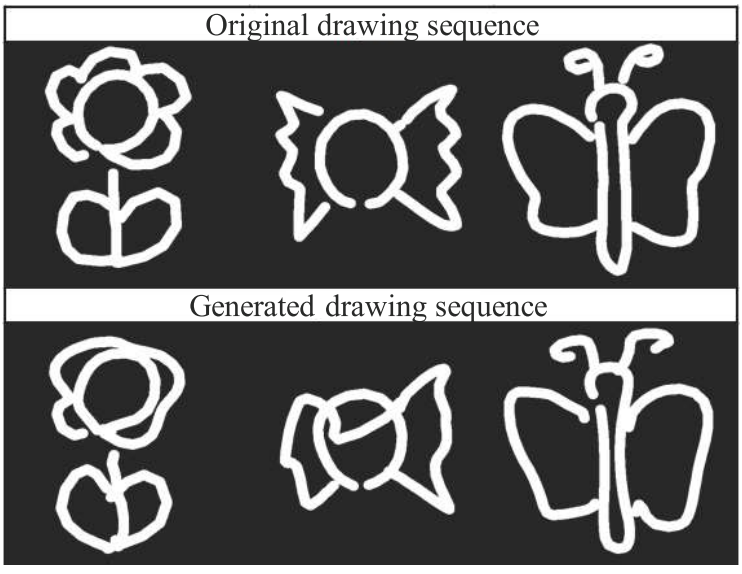

Fig. 5. Worst Generated Sequences

From Fig. 4 and Fig. 5, it can be seen that overall, the proposed method was capable of reconstructing the sequences. Looking closer to the results of each shape, there is not much difference between the best and worst generated sequences for butterfly, while the worst generated sequence for the flower and candy was quite different from the original sequence.

From the experimental results, it can be seen that the performance of generating the drawing sequence of butterfly was better compared to flower and candy. It can be considered that the difference in generation performance arises from the characteristic of MTRNN to learn the dynamics of sequences. Neurodynamics learning models are most suitable in learning cyclic motions, while motions that are static or have large flucuations are difficult to learn. Comparing the original sequence of the three shapes, it can be seen that the butterfly sequence has fewer sharp edges compared to flower and candy sequences. The motion direction changes greatly in sharp edges inducing difficulty in learning by MTRNN. In addition, the generation function recursively calculates the coordinate values from the initial position. Therefore, errors during the calculation accumulate along the sequence, resulting in large errors when drawing generation is completed.

6 Conclusion

In this paper, we presented a method to learn and generate drawing sequences consisting of multiple strokes using a neurodynamics learning model, namely

MTRNN. The proposed method adds a pen state neuron into the model to predict the state of the pen, whether it is drawing or lifted from the canvas. The training algorithm is also modified to adapt to deficient data when the pen does not have contact with the canvas. Experiments were conducted with three drawing shapes, each drawn ten times. The results of the experiments show that the model is capable of imitating drawing sequences using the recognition and generation functions of MTRNN.

As future work, we first plan to improve the performance of generating sharp edges of the shapes. An improvement of MTRNN model is required to deal with this issue. Further on, the model will be implemented into an actual developmental drawing robot system for creating an imitation drawing learning between a human and robot. We hope that our work will contribute to the progress of the field of cognitive developmental robotics.

Acknowledgments This research was supported by JSPS KAKENHI Grand Number 16H05877.

References

1. Asada M., MacDorman K. F., Ishiguro H., Kuniyoshi Y.: Cognitive Developmental Robotics as a New Paradigm for the Design of Humanoid Robots. Robotics and Autonomous systems, Vol.37, pp.185-193, 2001.
2. Luquet G. H.: Le Dessin Enfantin. 1927.
3. Nishide S., Ren F.: Improvement of Developmental Drawing Imitation Using Recurrent Neural Network Through Incorporation of AVITEWRITE Model. Proc. IEEE International Conference on Systems, Man, and Cybernetics, pp. 1-5, 2018.
4. Sugiura K., Iwahashi N., Kashioka H., Schaal S.: Learning, Generation and Recognition of Moitions by Reference-Point-Dependent Probablistic Models. Advanced Robotics, Vol.25, pp. 825-848, 2013.
5. Kober J., Peters J.: Imitation and Reinforcement Learning. IEEE Robotics and Automation Magazine, Vol. 17, No. 2, pp. 1-8, 2010.
6. Yokoya R., Ogata T., Tani J., Komatani K., Okuno H. G.: Discovery of Other Individuals by Projecting a Self-Model Through Imitation. Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1009-1014, 2007.
7. Simo-Serra E., Iizuka S., Sasaki K., Ishikawa H.: Learning to Simplify: Fully Convolutional Networks for Rough Sketch Cleanup. SIGGRAPH, 2016.
8. Ha. D, Eck. D: A Neural Representation of Sketch Drawings. in Proc. International Conference on Learning Representations, arXiv:1704.03477, 2017.
9. Kulvicius T., Ning K., Tamosiunaite M., Worgötter F.: Joining Movement Sequences: Modified Dynamic Movement Primitives for Robotics Applications Exemplified on Handwriting. IEEE Transactions on Robotics, Vol. 28, pp. 145-157, 2012.
10. Kudoh S., Ogawara K., Ruchanurucks M., Ikeuchi K.: Painting robot with multi-fingered hands and stereo vision. Robotics and Autonomous Systems, Vol. 57, No. 3, pp. 279-288, 2009.
11. Yamashita Y., Tani J.: Emergence of Functional Hierarchy in a Multiple Timescale Recurrent Neural Network Model: a Humanoid Robot Experiment. PLoS Computational Biology, Vol.4, No.11, e1000220, 2008.

DFS2P: An Efficient Deduplication Framework For SaaS Platforms

Quy H. Nguyen[2], Dac H. Nguyen[1,2,5], and Binh T. Nguyen[1,2,3,4]

[1] AISIA Research Lab
[2] Inspectorio Research Lab
[3] University of Science, Ho Chi Minh City
[4] Vietnam National University, Ho Chi Minh City
[5] John Von Neumann Institute, Ho Chi Minh City, Vietnam

Abstract. Filtering duplicated items or documents has been a ubiquitous problem in different industrial applications, especially for Software-as-a-service (SaaS) platforms. In this paper, we investigate the deduplication problem in a quality and compliance control platform, where users can assign multiple inspection bookings to different factories, and mistakenly create duplicated organizations. We collect two datasets where one dataset has 5600 existent organizations in one SaaS system, including both duplicated and non-duplicated records. The remaining dataset is created by crawling location data from OpenAddress (US Zone), combining new samples that are generated by different types of user's mistakes learning from the SaaS platform. We apply different methods toward the problem, including Bag-of-words (using Cosine Distance), Record Linkage toolkit, Hybrid methods, and the Siamese Neural Networks using the Triplet loss. The experimental results show that using Siamese Neural Networks achieves a better performance than other techniques.

Keywords: Duplicate Detection · Siamese Networks · Text Mining.

1 Introduction

There have been multiple studies during the last two decades related to the duplicate detection problem. By using the Word2Vec method and Hamming distances, Gao et al. [1] present a duplicate detection algorithm for light blogs and short comments and compare it with the unweighted Word2vec method and the traditional TF-IDF technique on the SICK corpus. Their experiments show that using weighted Word2Vec outperforms other technologies in terms of accuracy and recall. Also, by using a Siamese GRU neural network, Homma and co-workers [3] present another approach to encode each sentence to measure semantic equivalence between pairs of questions in one dataset released by Quora. Mudgal and colleagues [8] investigate a deep learning approach for entity matching and indicate that using deep learning approaches does not outperform the current solutions on structured entity matching. Still, it can significantly outperform them on textual and dirty entity matching with interesting experimental results. One can find more related works in [4,7,10].

In this paper, we investigate the duplicate detection problem in one SaaS platform related to quality and compliance management. It is one of the most popular issues in

numerous companies to retain the consistency and the cleanliness of users' data from different products. To study the problem, we first analyze all common mistakes extracted from existent duplicate items on the platform. Next, we collect the first dataset, having 5600 existent organizations (including duplicate cases). Also, by crawling location data from Open Address (US Zone), generating new samples by different types of users' mistakes, and combining these samples, we create the second dataset for extensive experiments. To learn an appropriate model for the problem, we apply different techniques, including Bag-of-words (using Cosine Distance), Record Linkage toolkit, Hybrid methods, and Siamese Neural Networks using the Triplet loss. The experimental results show that using Siamese Neural Networks can achieve a better performance than other techniques.

2 Our proposed methods

2.1 Duplication Analytics

The quality of both training and testing data is crucial for studying machine learning algorithms for the duplicate detection problem. Understanding all possible human mistakes and systems can give a better way to learning appropriate patterns and features of the problem. After collecting all organizations from the SaaS platform, we analyze all possible types of duplications created by users before studying a suitable prediction model. All addresses used in this paper are written in English. In general, an address may consist of either all or part of the following attributes: address number, building name or number, street name or road name with a street number or road number, ward, district, province or city, postal code, region, and country. Different countries have different formats for addresses. For example, an address in China is "30, Donggang 3rd Road, Quzhou, Zhejiang, China 324000" while another in Vietnam can be "227, Nguyen Van Cu street, Ward 4, District 5, Ho Chi Minh City, 700000, Vietnam". Besides, users from dozens of countries with different cultures and typing behaviors create more challenges for the duplicate prediction problem.

With 5600 organizations in the database of the SaaS platform, 131 organizations have duplicate uniquely determined records. The total number of samples related to these organizations is 5787; the ratio of organizations having duplicate items is 2.34%. Typically, there are different mistakes from humans and systems: spelling, wrong wordings, system error, synonyms, acronyms, and others. The most popular reason comes from spelling errors (42.14%) of users, while the corresponding percentages for wrong wordings, system errors, acronyms, synonyms, and others are 26.43%, 15.71%, 11.43%, 19.29%, and 12.34%, respectively.

2.2 Data Processing and Augmentation

There is no doubt that two addresses can be considered as duplicated ones if they represent a unique entity located at the same location on the world map. Therefore, two addresses only have some difference in the letter case (upper-case or lower-case) as well as additional punctuation marks can be considered as the same thing. In this paper,

we apply the following preprocessing steps: (1) Lower case; (2) Removing all punctuation marks; (3) Removing all simple mistakes (by human or system) such as, e.g., multiple spaces, characters "Null" or "'NaN".

As the percentage of duplications is pretty small, we aim at doing augmentation techniques to generate more duplicate records in the training dataset for learning appropriate algorithms. There are two types of data augmentation in this paper. First, we apply all possible reasons from users for producing duplicate records in the database, as mentioned previously, for generating more samples. To diversify duplicate cases in the experiments, we use further techniques (including random insertion, swap, and deletion) by using a Python library, namely **nlpaug**[6].

2.3 Model Learning

	Accuracy	**F1**	**Precision**	**Recall**
TS-Lab	0.9736	**0.8396**	**0.9082**	**0.7807**
TS-OpenAddress	0.9527	0.5323	0.44	0.6735
Jaro Winkler	0.9624	0.5106	0.5333	0.4898
Levenshtein	0.9706	0.5714	0.6857	0.4898
Damerau Levenshtein	0.9706	0.5714	0.6857	0.4898
Q-gram	0.9665	0.6496	0.5588	0.7755
Cosine Distance	0.9747	0.6667	0.7045	0.6327
Smith-Waterman	0.9747	0.6931	0.6731	0.7143
LCS	0.9731	0.6733	0.6538	0.6939
Hybrid Model	0.9552	0.6415	0.5965	0.6939
Cosine Distance	0.9673	0.5918	0.5918	0.5918

Table 1: The best performance of each approach in the validation dataset.

In this section, we present how we construct a suitable machine learning for the duplicate prediction problem. First, we consider seven different distances and similarity methods, popularly used for measuring the similarity of texts. They are Jaro Winkler distance, Levenshtein distance, Damerau Levenshtein distance, Q-Gram, Longest Common Subsequence (LCS), Smith-Waterman, and the cosine distance. Importantly, these distances can be easily found in a Record Linkage System [6,7] or Record Linkage Toolkit [7].

It is worth noting that Jaro distance is the weighted sum of the percentage of matched characters, while Jaro Winkler improves this measure for matching prefix characters [13]. Also, Levenshtein and Damerau Levenshtein determine the distance as the minimum number of single-character edits (such as, e.g., insertion, deletion, substitution) [6,7]. Moreover, Q-Gram approximates the string-matching score by finding common Q-grams, which are substrings of length Q [12]. Longest Common Subsequence (LCS)

[6] https://github.com/makcedward/nlpaug

[7] https://recordlinkage.readthedocs.io/en/latest/

finds the longest common subsequence in two strings or sequences [6,7]. Subsequently, the cosine distance can measure the distance between two sentences, which is precisely the distance of two addresses in our problem.

Triplet Siamese Model We apply the Siamese neural networks [5] for estimating the similarity between the two addresses. Given the input data as two addresses A_1 and A_2, the proposed neural network aims to measure the difference and conclude whether they are matching under one threshold. We assume that after the preprocessing step, the lengths of these addresses A_1 and A_2 are L_1 and L_2, consecutively. As L_1 and L_2 may be different, we insert "$< pad >$" into each address to ensure they have the same length M, where M is the maximum length among all addresses in one batch. By using character tokenization, each character can be represented by a unique number. Here, we only use 58 characters, including "space," 26 alphabet characters, ten digits, and 31 special characters in ASCII. It turns out that one can convert each address to a form such that a neural network can easily understand and perform well instead of using the human language context directly [14]. After this step, one can obtain a vector of numbers as the input data of a Siamese model. Next, the input data go into an embedding layer, and it embeds the corresponding input values into a H-dimensional space [9]. Typically, the embedding layer can cast a tokenized vector with length M into a matrix with size $(M \times H)$. Before passing this matrix to the next layer, the "unpadding" step is necessary to optimize the value of M. It is a crucial step to ensure the calculation time of Recurrent Neural Network (RNN), as well as Gated Recurrent Unit (GRU), is as good as possible without affecting too much the performance of the prediction model. Typically, this step can remove all padded elements at the embedding layer, to restore their original length before inserting into the GRU layer. After this step, we can get two matrices of sizes $L_1 \times H$ and $L_2 \times H$, corresponding to two original addresses, A_1 and A_2, consecutively. It is worth emphasizing that both L_1 and L_2 are much smaller than M. As a consequence, one can not only decrease the calculation time but also does not affect the final results [2]. Next, outputs of the GRU layer are two matrices of sizes $L_1 \times P$ and $L_2 \times P$, where P is the hidden dimension of the GRU layer. Note that one can tune the value of P during training the Siamese model. After that, these matrices can be padded to have the same size of $M \times P$ and then continue going through one fully-connected layer to achieve the final two-dimensional vectors for the original addresses A_1 and A_2, sequentially.

Triplet Loss To enhance the performance of the proposed Siamese model for the duplicate detection problem, we decide using the triplet loss [11] when training the network. Figure 1 depicts the architecture of training a Siamese model using the triplet loss. Here, the "anchor", w_{anchor} represents a unique address, "positive", $w_{positive}$, stands for a duplicate case, and "negative", $w_{negative}$, is an address, which is different from the anchor. Here, the triple loss can be defined as follows:

$$\text{Loss} = sim_{negative} - sim_{positive} + margin, \tag{1}$$

where

$$sim_{positive} = 1 - \frac{w_{anchor} \cdot w_{positive}}{\|w_{anchor}\|_2 \left\|w_{positive}\right\|_2}, sim_{negative} = 1 - \frac{w_{anchor} \cdot w_{negative}}{\|w_{anchor}\|_2 \left\|w_{negative}\right\|_2}$$

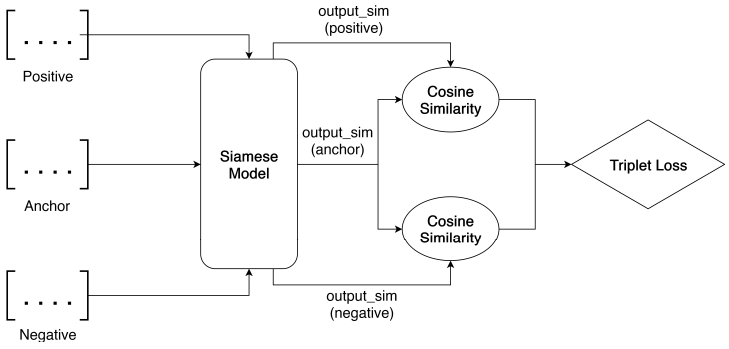

Fig. 1: The architecture for enhancing the performance of our proposed Siamese model using the triplet loss.

and "$\cdot$" is the outer product of two vectors. During training models, we learn the optimal model parameters by minimizing the corresponding triplet loss.

3 Experiments

In this paper, we run all experiments on a computer with Intel(R) Core(TM) i7 2 CPUs running at 2.4GHz with 128GB of RAM and an Nvidia GeForce RTX 2080Ti GPU. We measure the performance of seven methods by using the Record Linkage Toolkit[8] (RLT) and our Triplet Siamese model, and consider the following metrics: precision, recall, and F1-score.

3.1 Data set

Triplet Siamese model requires three inputs (anchor, positive, and negative) consumed once during training. Here, the anchor represents a unique address, and the positive is the duplicate information. In contrast, the negative is a sample different from the anchor. In our experiments, we consider two different training datasets for studying duplicate prediction models. The first training dataset comes from the SaaS platform, with 5600 unique organizations. Initially, 131 organizations have duplicate records, but 22 triplets (a) and 109 positive pairs (b) are available. To populate (b), we randomly select samples that are not duplicate records to become negative components. When having enough triplets, we apply two augmentation techniques, as mentioned in the preceding section. We use a characteristic learned from the analytics phase by using pairs of addresses having different street numbers (c). The total number of samples when we complete all (a), (b), and (c) is 31424.

Next, we consider one OpenAddress dataset containing 26,871,004 addresses (US-EN only). Among these millions of addresses, we randomly collect about 2000 ones.

[8] https://recordlinkage.readthedocs.io/en/latest/about.html

Importantly, no duplication exists here. By applying the proposed augmentation techniques, we can achieve a set of 43978 locations. We then pick 2000 addresses as anchors, use the corresponding augmented samples as positives, and select any of the remaining unique addresses as negatives.

3.2 Experimental Results

To measure the performance of our proposed method and other techniques, we consider the following experimental scheme. We train the proposed Siamese model by studying two different training datasets: one collected from the SaaS platform (namely **TS-Lab**) and the remaining one from Open Address data (namely **TS-OpenAddress**). During the training process, we use one validation dataset, including 49 positive pairs (anchor and positive) of duplicate addresses collected from our platform. Remarkably, these addresses are completely different from ones in training datasets. Subsequently, we add other 1176 negative pairs (anchor, negative) along with 49 positive pairs to obtain 1225 pairs for the validation dataset. Also, we consider other 114 positive pairs (anchor and positive) of duplicate addresses as well as 5000 negative pairs (anchor, negative) to generate the testing dataset.

For seven different distance and similarity methods (Jaro Winkler distance, Levenshtein distance, Damerau Levenshtein distance, Q-Gram, Longest Common Subsequence (LCS), Smith-Waterman, and the cosine distance), we do necessary preprocessing steps for the input data. Importantly, when tuning learning parameters and thresholds for each learning model or similarity method, we consider different thresholds in the set $\{0.10, 0.11, \ldots, 0.98, 0.99\}$ and choose the best one by measuring the corresponding performance in the validation dataset in terms of the F1-score. Afterward, we apply these learning thresholds to measure the performance of each method in the testing dataset.

Table 1 describes the best performance of different techniques in the validation dataset. One can see that using the triplet Siamese model with the TS-Lab dataset outperforms all other classifiers in terms of all F1-score, precision, and recall. The best thresholds for all approaches (TS-Lab, TS-OpenAddress, Jaro Winkler distance, Levenshtein distance, Damerau Levenshtein distance, Q-Gram, Longest Common Subsequence (LCS), Smith-Waterman, and the cosine distance) are 0.57, 0.72, 0.71, 0.44, 0.44, 0.46, 0.65, 0.36, 0.7, respectively.

Additionally, we study one hybrid approach by combining seven distance-based models and considering the predicted similarity of each distance-based model as input data of the hybrid model. In experiments, we apply the logistic regression method for learning the hybrid model. It is worth noting that our proposed methods using both TS-Lab and TS-OpenAddress outperform all other techniques in terms of the F1-score in the testing dataset. More detailed, using TS-Lab achieves the highest F1-score (68.36%) while TS-OpenAddress gets 64.25%. The hybrid model can obtain the best recall (91.84%), while the traditional method using the cosine distance has the most considerable precision (96.08%).

	Accuracy	F1	Precision	Recall
TS-Lab	0.983	**0.6836**	0.5839	0.8246
TS-OpenAddress	0.9806	0.6426	0.546	0.7807
Jaro Winkler	0.9814	0.4444	0.3115	0.7755
Levenshtein	0.9793	0.4362	0.295	0.8367
Damerau Levenshtein	0.9793	0.4362	0.295	0.8367
Q-gram	0.9797	0.4583	0.3077	0.898
Cosine	0.9877	0.5468	0.4222	0.7755
Smith-Waterman	0.9795	0.456	0.3056	0.898
LCS	0.9844	0.5	0.3604	0.8163
Hybrid Model	0.9779	0.4433	0.2922	**0.9184**
Cosine Distance	0.9869	0.5939	**0.9608**	0.4298

Table 2: The best performance of each approach in the testing dataset.

4 Conclusion and further work

We have investigated the duplicate detection problem in our SaaS platform by considering addresses of existent organizations. We have proposed a triplet Siamese model by using character tokenization, one embedding layer, one GRU layer, and one fully-connected layer as well as the triplet loss to calculate the similarity between two given addresses. We have compared our method with other distance-based models in text mining. We have collected two different training datasets from the SaaS platform and the Open Address website. The experimental results show that the proposed methods outperform other models in terms of F1-score. The proposed methods can be integrated into various applications for automatically detecting the existent organizations when users insert new bookings or removing all duplicated records to maintain the consistency of databases across different micro-services. In the future, we plan to continue improving our methods by applying new algorithms related to the problem.

Acknowledgement

We would like to thank The National Foundation for Science and Technology Development (NAFOSTED), University of Science, Inspectorio Research Lab, and AISIA Research Lab for supporting us throughout this paper.

References

1. Gao, J., He, Y., Zhang, X., Xia, Y.: Duplicate short text detection based on word2vec. In: 2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS). pp. 33–37 (Nov 2017). https://doi.org/10.1109/ICSESS.2017.8342858
2. Giménez, M., Palanca, J., Botti, V.: Semantic-based padding in convolutional neural networks for improving the performance in natural language processing. a case of study in sentiment analysis. Neurocomputing (2019). https://doi.org/https://doi.org/10.1016/j.neucom.2019.08.096, http://www.sciencedirect.com/science/article/pii/S0925231219314687

3. Homma, Y.: Detecting duplicate questions with deep learning (2017)
4. Hoogeveen, D., Bennett, A., Li, Y., Verspoor, K.M., Baldwin, T.: Detecting misflagged duplicate questions in community question-answering archives. In: ICWSM (2018)
5. Jane, B., Isabelle, G., Yann, L., Eduard, S., Roopak, S.: Signature verification using a "siamese" time delay neural network. In: Proceedings of the 6th International Conference on Neural Information Processing Systems. pp. 737–744. NIPS'93, Morgan Kaufmann Publishers Inc., San Francisco CA USA (1993), http://dl.acm.org/citation.cfm?id=2987189.2987282
6. Jaro, M.A.: Unimatch: A record linkage system: User's manual. Tech. rep., U.S. Bureau of the Census, Washington, D.C. (1976)
7. Jaro, M.A.: Advances in record-linkage methodology as applied to matching the 1985 census of tampa, florida. Journal of the American Statistical Association **84**(406), 414–420 (1989). https://doi.org/10.1080/01621459.1989.10478785, https://www.tandfonline.com/doi/abs/10.1080/01621459.1989.10478785
8. Mudgal, S., Li, H., Rekatsinas, T., Doan, A., Park, Y., Krishnan, G., Deep, R., Arcaute, E., Raghavendra, V.: Deep learning for entity matching: A design space exploration. In: Proceedings of the 2018 International Conference on Management of Data. pp. 19–34. SIGMOD '18, ACM, New York, NY, USA (2018). https://doi.org/10.1145/3183713.3196926, http://doi.acm.org/10.1145/3183713.3196926
9. Noraset, T., Liang, C., Birnbaum, L., Downey, D.: Definition modeling: Learning to define word embeddings in natural language. CoRR **abs/1612.00394** (2016), http://arxiv.org/abs/1612.00394
10. Papenbrock, T., Heise, A., Naumann, F.: Progressive duplicate detection. Knowledge and Data Engineering, IEEE Transactions on **27**, 1316–1329 (05 2015). https://doi.org/10.1109/TKDE.2014.2359666
11. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: A unified embedding for face recognition and clustering. CoRR **abs/1503.03832** (2015), http://arxiv.org/abs/1503.03832
12. Ukkonen, E.: Approximate string-matching with q-grams and maximal matches. Theoretical Computer Science **92**(1), 191 – 211 (1992). https://doi.org/https://doi.org/10.1016/0304-3975(92)90143-4, http://www.sciencedirect.com/science/article/pii/0304397592901434
13. Winkler, W.E., Thibaudeau, Y.: An application of the fellegi-sunter model of record linkage to the 1990 u.s. decennial census. Tech. Rep. Statistical Research Report Series RR91/09, U.S. Bureau of the Census, Washington, D.C. (1991)
14. Zhang, X., LeCun, Y.: Text understanding from scratch. CoRR **abs/1502.01710** (2015), http://arxiv.org/abs/1502.01710

Opinion Mining Classifier for Jordanian Dialect

Hani D. Hejazi [1[0000-1111-2222-3333]], Ahmed A. Khamees [1[0000-0002-9324-464X]], Said A. Salloum [1,2[0000-0002-6073-3981]], and Khaled Shaalan [1[0000-0003-0823-8390]]

[1] Faculty of Engineering &IT, The British University in Dubai, UAE
[2] Research Institute of Sciences & Engineering, University of Sharjah
ssalloum@sharjah.ac.ae

Abstract. The faster the growing the Arabic content in Social Media and Internet, the more the demanding for implementing Artificial Intelligence technologies to automate the sentiment analysis of the users reviews in Social Media. In this paper, we have implemented a solution using three classifiers: Decision Tree, Naive Bayes, and Maximum entropy to predict the analysis of sentiments in reviews of Jordanian Dialect (JD). JD is chosen because it is widely used in Gulf and Levantine countries. The significant results of experiments is attained using the Naive Bayes classifier with ISRIStemmer stemmer. The highest performance results in terms of standard evaluation measures obtained 0.829 accuracy, 0.84658 Positive F-score, and 0.80524 Negative F-score.

Keywords: Sentiment analysis, Decision Tree, Naive Bayes.

1 Introduction and Background

1.1 Jordanian Arabic Dialect

JAD is a continuum of common spoken Levantine Arabic of the people of the Hashemite Kingdom of Jordan HKJ as well as across the Levantine countries. JAD is a language that has a Semitic structure [1]. JAD has mutual varieties that are spoken by an estimated population of around 10 million people and understood by near Arabic speaking countries. Modern Standard Arabic (MSA) is the considered as the official language of HKJ and used mainly by most of the media, written articles, books and newspapers. But the daily spoken conversations are communicated by the different local variations of JAD. According to [17] JAD is subdivided into five dialect variations:

1. Hybrid (Modern) Dialect (HD): it has been a presented since the announcement of Amman as a capital of Jordan in the early 20th century. Thus, people started moving towards the capital from northern and southern regions, and later on from the Palestinian Territories.
2. Northern Dialect (ND): it is spoken mostly in the area from Amman up to the north city of Irbid. Where the pronunciation of the letter (Q) is pro-

nounced as (G). For instance, “Qamr”, which means “Moon”, becomes “Gamr” ND is part of the southern Levantine region.

3. Southern (Moab) Dialect SD: is spoken in the area at the south of Amman, such as the cities of Ma’an, Tafileh, Karakm and Shoubak. One of the differences in this dialect is the pronunciation of the final letter (ta-marbouta) is spelled as (e). For instance, “Madrasah”, which means “School”, becomes “Madrase”. This SD belongs to the outer regions of southern Levantine.
4. Bedouin Dialect (BD): is mostly spoken in the east area of Jordan consists mainly of Badia region, like Alazraq.
5. Aqaba Dialect (AD): is spoken in only the access area of Jordan by the sea in the Gulf of Aqaba.

On the other hand, [2] subdivided the JAD into three types: Urban, Rural, and Bedouin. The Urban Dialect UD has been emerged from the migrations towards the major cities. The Rural Dialect RD is spoken often in the small cities and villages. The author divided RD into two subcategories (Horan, and Moab) where Horan Dialect is spoken in the northern region and west of Amman, while Moab Dialect is spoken in the area south to Amman. The Bedouins Dialect, however, is spoken in the desert areas by Bedouins.

1.2 JAD Grammar

Different types and variations of JAD influence the grammar, pronunciation and vocabulary. Moreover, as [17] declared, JAD speakers must deal with the other differences when addressing females, males and groups. Verb conjunctions and plurals are irregular and hard to determine their stem or root letters. This makes it difficult for non-Arabic speakers to pronounce it. JAD grammar is mixture and considered as a Semitic language that developed and influenced by many people over years. For example, the article definitions in JAD has many forms, like in (el) this prefix means (the) like in (“el korsi” meaning the chair). In addition, JAD continually adds suffixes and clitics to generate new contrary words and negation words (e.g. the ‘sh’ suffix in “bednash” means “we do not want”).

1.3 Opinion Mining (OM)

OM has been studied at three categories of text: Document, Sentence, and Aspect Entity. As per [3], OM can be classified using two methods: Statistical and Lexical methods. The Statistical method uses Machine Learning (ML) algorithms, e.g. Naïve Bayesian (NB), Support Virtual Machine (SVM), Decision Trees (DT), Maximum Entropy (ME), etc. The Lexical method is described as unsupervised kind of approach that exploits dictionaries, lexicons, and some other linguistic rules of polarity classification [4]–[8]. As [9] stated, OM is defined as a part of Natural Language Processing (NLP) and Machine Learning algorithms that deals with the people’s opinion or emotion regards a particular topic they wrote about or posted on the web through different forms of text, like in social media and blogs [10]–[12]. Furthermore, opinions are classified to positive, negative, or neutral depending on the users implicit or explicit

point of view that requires analysis to determine the text polarity [13]. Likewise, [14] defined OM as the process of finding reviewers opinion in specific topic or matter. The author also focused on his definition of OM on analyzing the text and sentences and text polarity into three classifications: positive, negative, or neutral. [15] mentioned that OM has become very important and popular in the recent research, due to its significance in several fields (e.g. education, political, healthcare, marketing, etc.) Moreover, OM is important to make decisions especially for individuals who holds critical job roles in governments and organizations. OM has some challenging in Arabic languages, [3] highlighted some major ones such as the existence of many dialects, the lack of tools and resources of Arabic dialects, Arabic lexicons limitations of Arabic dialects, and uses of idioms and compound phrases, etc. Depending on the previous facts, the need to do more research on OM is raised, more specifically in JAD. In this paper, the authors focus on analyzing corpus polarity of Arabic text dataset found on [16] research written in JAD. This is achieved by first making normalization on the Arabic text to omit the short vowels, remove duplicate letters, stem the words to omit any concatenated suffixes or prefixes, special character removal, and possible replacement of synonyms [17]. All this is made to enhance the accuracy of polarity classification. Then, using Python programming, in particular NLTK package, to implement three different types of machine learning classifiers (NB, DT, and ME). Results shown that the NB outperformed the other classifiers [18].

This paper divided into six sections. In the first section, the authors introduced a brief history, JAD different variations, OM definition and challenges of Arabic language analysis. In section two, we introduced some of the previous related work been done regarding to JAD. In Section three, we discussed the used methodology to do our research. In Section four, the authors argued the results depending on an evaluation between the three different classifiers been used in this research. In Section five, the authors presented the results of the analysis that were applied on the target JAD dataset. Finally, in section six, the authors summarized the conclusion, challenges and future work would be done.

2 Related work

Several methods been presented to do OM on JAD. Each method applied some recognized classifiers in order to distinguish the best one among them in terms of accuracy. [16] investigated how Arabic dialectal reviews is classified into two types of sentiment; positive or negative ones based on the selection of several method features using SVM algorithm. To train the SVM classifier, an annotated dataset of 2400 Jordanian dialect reviews was used considering only positive and negative polarities (1200 positive, 1200 negative). The author has chosen a hybrid approach in features selection trying to improve the performance. The author used RapidMiner machine learning tool to conduct his experiment on vector model type of represented text suitable for machine learning experiments. Unigrams, Bigrams and merged features of

both have been experienced in order to select the most relevant feature to improve the performance of the classifier, including the informative aspects and excluding the less descriptive ones. The experiment employed N-fold cross-validation technique to evaluate the performance of the classifier on the first four stages and the fifth one based on 5-fold and 10-fold cross-validation weights on sequence. It has been shown that selecting SVM feature method gave the highest results in terms of Recall, Precision and Accuracy in comparison with the other five experienced methods, with Accuracy of 92.12%. In addition, results shown that selecting a combination of feature methods can improve the performance of the classifier. [3] presented OM on Jordanian dialect dataset that has 2500 reviews collected manually from a Jordanian website called Jeeran. The author implemented a semi-supervised approach for OM (positive, negative) analysis on this dataset. Learning classifiers then combined with dialectal lexicons and combined with some algorithms of machine learning (i.e. SVM, NB, K-NN and Random Forest) to determine the best model of classification. In addition, the author evaluated nine different feature methods to improve the performance process of classifiers by selecting the top three evaluation features selection methods (i.e. SVM-based, Correlation based, and Principle Components Analysis) that tend to result in best accuracy. The author used LIBSVM package and WEKA software that introduces algorithms of machine learning and data mining. Classification using SVM algorithm obtained the highest accuracy of 92.3%. [9] Presented a hybrid approach for Arabic OM analysis, a combination of Multilayered Perceptron (MLP), NB algorithm, JRip, Decision Tree, ZeroR, OneR, and PART to classify the dataset into a positive and negative polarities. The Arabic dataset represent written opinions in MSA and Jordanian dialect tweets, which consists of 2000 Twitter posts in different topics (e.g. arts and politics) 1000 labeled as positive tweets and 1000 as negative tweets. The comparison of several classifiers were tested over the collected dataset of tweets. As a result, the combined MLP performance outperformed over the rest tested classifiers with accuracy of 99.5%. [18] Analyzed OM of customers' comments on the public pages on Facebook of the telecommunication companies in Jordan (e.g. Zain, Orange and Umniah). These comments are collected into a dataset using Netvizz software in a form of page data module. In [12] and [13], the authors collected 14332 instances to form the dataset of the three companies. 3862 of the collected dataset labeled as negative and 365 as positive ones. On the other hand, the author used RapidMiner software to construct the intended model by performing a supervised approach including SVM, NB, Decision Trees, and K Nearest Neighbors classifiers. Then, a comparison on results between these approaches are given in terms of accuracy. The results of SVM gave the highest performance over all of the other classifiers in terms of F-measure and accuracy. [14] Proposed an OM model on several Arabic tweets that are presented in Jordanian dialect. Total number of 3550 of tweets were classified into a positive, negative and neutral classes by SVM and NB classifiers. The author concluded that SVM classifier as better performance than the NB one as SVM gets an accuracy equals to 82.1%.

[21] introduced a corpus of Arabic tweets written in a Jordanian dialect, and labeled manually as positive and negative. The corpus has 1800 tweets 900 annotated as posi-

tive and 900 as negative ones. The author investigated several supervised machine learning OM approaches using RapidMiner software that allowed the author to perform evaluation on the corpus using SVM and NB algorithms based on different scenarios. The author concluded that SVM classifier using weighting scheme of Term Frequency-Inverse Document Frequency (TF-IDF) with the Bigrams achieved the better performance over the NB classifier with accuracy of F-score 88.27% and 88.72%. [22] presented an Arabic polarized lexicons of Jordanian dialect by first extracting 28000 entries from analyzing 15100 reviews posted in Jeraan website and expanding the lexicon using Google Translate. Polarity given to each entry and score of confidence. The author verified the process of polarization and precision by selecting 800 of the total tagged entries. Then evaluated by two Arabic mother tongue speakers. Results shown that polarization precision was 90% as long as there are high enough score of confidences. [19] tackled the problem of OM on reviews written as dialects in Arabic language, in which there polarity (positive or negative) can be examined by using different supervised machine learning classifiers and features. The author classified 28576 dataset of reviews collected from Jeeran website by using SAMCAL model that first performs a web crawler, which is followed by filtering out the Jordanian and Saudi reviews in order to finally to extract some features from each review. The author applied SVM and NB and Maximum entropy algorithms to evaluate their performance in OM. Results shown that Maximum Entropy has the best performance over the other two algorithms with accuracy of 83.83%. [23] presented Jordanian Arabic dialect under investigation to figure out OM classification for the Arabic reviews posted in the Jeeran website. The author introduced an approach that combines two categories of lexicons, include entries for dialectal words and compound phrases. One lexicon holds reviewers' opinions, and the other lexicon contains objective words with opinion tendency. These two lexicons are fed into a classification process using SVM classifier to differentiate polarity of positive reviews from negative ones. The author built a dataset to train the SVM classifier by manually collecting reviews from Jeeran website. The author collected 2730 different reviews (1527 of them were positive and 1203 were a negative ones). Furthermore, the author investigated how the performance of the classifier is affected by combining opinion features with different models of N-gram. After evaluating the two concerns under investigation results show that the performance of OM classification can be improved in terms of Accuracy, Precision, and Recall for both used models of the two set of lexicons and the combination of N-gram with opinion features by 3.9%, 0.82%, and 7.4%, and by 5.7%, 4.3%, and 7.4%, respectively, on comparison with the two baseline models, with accuracy of 95.6%.

3 Methodology

In order to predict sentiments from reviews written in Jordanian Dialect, and as per literature review and the related work, we propose to use Naïve Bayes, Decision Tree, SVM, and Maximum entropy classifiers. First, we prepared one unified dataset of

sentiments that are provided by [16], regardless of their polarities, and transform it to comply with our format requirements. The reason is that we used the 5-folded cross-validation technique, which requires the whole data sets to be merged and divided at runtime to generate the training and testing datasets. We report the average of the five runs to reduce the bias and randomness effect on the results. Stemming, stop words removal and data cleansing are also used and the results with/without them were compared. Tokenization is performed with unigram, bigram, trigram, and mixgram, mixgram models. The mixgram model presents a combination of tokens extracted using unigram, bigram, and trigram together such that more discriminating features from each grams were considered. The results are also compared to show the best feature extraction and selection schema. Accuracy, F-score, Precision, and Recall are the standard measures used to evaluate the results of each. We used NLTK package that are built on top of python to implement the classifiers, namely: DecisionTreeClassifier, NaiveBayesClassifier, and MaximumEntropyClassifier, for features training and testing. The details of the methodology are discussed in the following subsections.

3.1 Dataset

We used the dataset introduced by [16], which is built from user reviews extracted from Jeeran website. These reviews are related to restaurants, movies, places, among others. The dataset contains 2730 labeled user reviews. Labels are either Positive (PO) or Negative (NG). These reviews are divided into 409 reviews as testing dataset, consisting of 208 positive and 201 negative reviews, and 2321 training dataset, consisting of 1319 positive and 1002 negative reviews. This dataset are mostly written in JD, but few English words might appear. Moreover, some MSA, Gulf, Syrian, and Egyptian dialects also appeared. This dataset is not clean nor well-formed, which should be handled. It might contain miss-spelled words, repeated characters, parentheses and special characters. Nevertheless, some reviews were assigned incorrect labels, which we have fixed, e.g. “prices are good” is labeled as negative but we made it positive. Both testing and training datasets were consolidated in one dataset. We applied the 5-folded cross-validation technique to the whole dataset, where 20% is considered as testing dataset and 80% is considered training dataset. The measures are calculated and the average of each measure is taken. After updating the mislabeled reviews, the positive reviews is increased by one and the negative reviews are decreased by one. So, the dataset is divided between two lists each one contains original training and test sets, each of which include positive and negative set

3.2 Preprocessing

In a preprocessing stage, data normalization and cleaning are applied. Firstly, short Arabic vowels, punctuation, and single letter words are removed from the text. Then, repeated characters are replaced by a single occurrence. Next, words are passed to ISRIStemmer stemmer to remove prefixes and suffixes, initial “Hamza” is normalized to “bare alif”, connectives are removed, and stop words are removed. Afterwards, common words are mapped to their closely related synonym in order to have less

number of words passed to the classifier. For example, words like “salty”, “noisy”, “boring”, “badly”, “worst”, “disgusting” are simplified to “bad” because they identified as a bad feature. Tokenization was done based on words separated by white spaces. To simplify the process, the unigram, bigram, trigram, mixgram are used to extract the features that identify the sentiment of a review. English words are eliminated since we concentrate on Arabic text sentiment analysis.

3.3 Features Extraction and Selection

In order to represent a model for the sentiments, we need to identify the best classification features that represent the user review, which we can use to predict the polarity of the sentiment. These features are to be passed to the classification algorithm for training and building the sentiment classification model. We have several ways to choose from the features using N-gram language model. We have implemented Unigram, Bigram, Trigram, and Mixgram feature prediction. The frequency of the feature and the most informative feature property are used to indicate the highest features that decide on the polarity of the sentiment. The number of features were about 3000 and the prediction depends on the N-gram that is used. In mixgram, we mix between the most informative features that appear in unigram, bigram, and trigrams so we get a unified feature set that includes the most informative features among all grams.

3.4 Experiments Design

Our purpose from using a machine leaning classifier is to classify the data into two predefined categories: positive and negative reviews. Firstly, we analyzed the dataset and modified it as required to build the classifier, where it is necessary to have two lists; one for positive sentiments and another one for negative sentiments. The ISRIStemmer stemmer is used because it is simple and can remove two to three prefixes and suffixes, stop words, and short vowels. Identification and replacement of a word with its common synonyms is useful with rare sentiment words and led to a better frequency, for example that occurrence of Salty or Noisy was changed to Bad, because these rare words are not represented enough and would disappear on frequency selection. Three Classification algorithms were chosen DecisionTreeClassifier, NaiveBayesClassifier, and MaxentClassifier, all from NLTK package, since they are most used algorithm in sentiment analysis and have proven good results as per the related work. The evaluation uses Accuracy, Precision, Recall, and F-score measures to decide the best algorithm and the stemming effect.

4 Evaluation

The dataset provided by [9] was used. It contains a user reviews for different experiences like food, movies, and trips. It contains 1202 negative reviews and 1528 positive reviews. In our evaluation, experiments are either general or specific to polarities. So, we have used Accuracy, Positive Precision, Negative precision, Positive Recall,

Negative Recall, Positive F-score, Negative F-score, where the positive targets the positive test data and negative targets the negative test data. The results are divided into four parts; the first three parts report the performance of each algorithm while the last one shows the overall results.

4.1 Decision Tree Classifier

In this experiment, we investigate building a predictive model using Decision Tree machine learning algorithm that decides on the label (polarity). The results for Decision Tree classifies is shown in Figure 1.

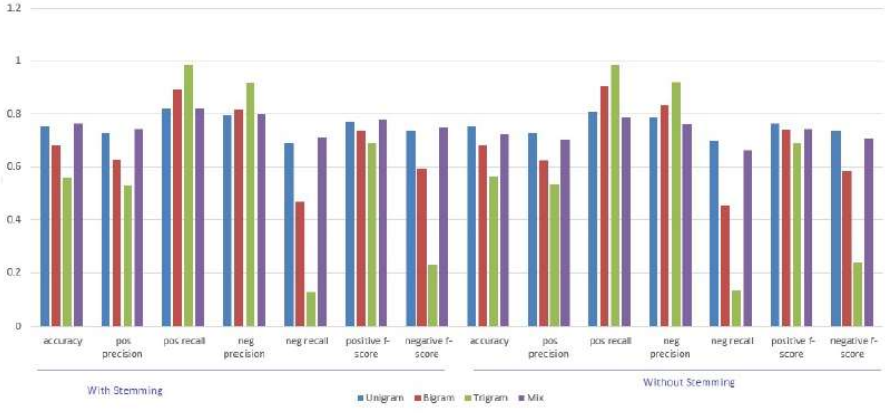

Fig. 1. Unigram, bigram, trigram, and mixgrams evaluation results using Decision Tree Classifier. A figure caption is always placed below the illustration. Short captions are centered, while long ones are justified. The macro button chooses the correct format automatically.

In this experiment, we notice that Precision and Recall using Bigram and Trigram achieve good results in terms of positive recall and negative precision but lower results for positive precision and negative recall. The accuracy and positive/negative F-score measures present better scores. We can see that stemming has a good effect on the results and the accuracy. F-scores were higher after applying the stemming method. The best result was when using a mixgram with stemming:

- Accuracy: 0.7655
- Positive f-score: 0.77736
- Negative f-score: 0.74976

4.2 Naive Bayes Classifier

This classifier is based on simple probabilistic algorithm and is based on Bayes theorem, which is widely used in text classification. It is the fastest algorithm in training our model. The results for Naive Bayes Classifier is shown in Figure 2.

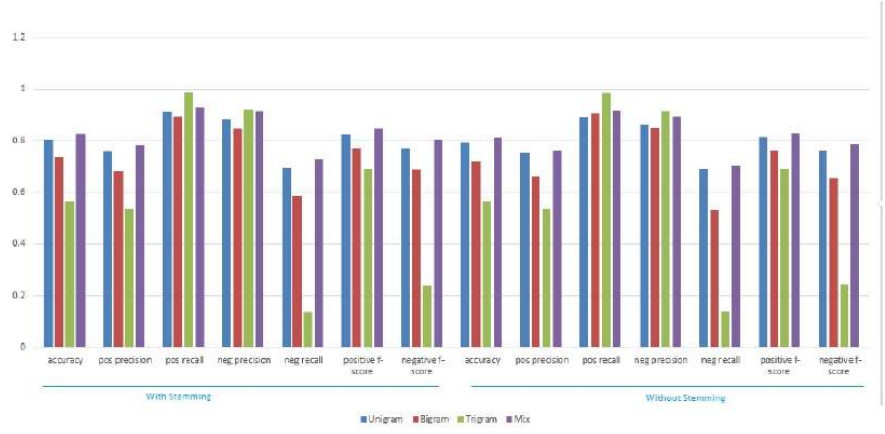

Fig. 2. Unigram, bigram, trigram, and mixgrams evaluation results using Naïve Bayes Classifier. A figure caption is always placed below the illustration. Short captions are centered, while long ones are justified. The macro button chooses the correct format automatically.

In this experiment, we noticed that Precision and Recall using Bigram and Trigram achieve good results in terms of positive recall and negative precision but lower results for positive precision and negative recall. The accuracy and positive/negative F-score measures present better scores. We can see that stemming has a good effect on the results and the accuracy. F-scores were higher after applying the stemming method. The best result was also when using a mixgram with stemming:

- Accuracy: 0.829
- Positive f-score: 0.84658
- Negative f-score: 0.80524

4.3 Maximum Entropy Classifier

Maximum Entropy is a technique related to statistical approaches, it shows longer time for training than Naive Bayes and much faster than Decision Tree training time and this affects the scaling of the training dataset. The results were the lowest in terms of Accuracy and F-scores. This might be due to the large number of features that are used as well as the different token sizes that affect both training time and model performance. The results in terms of Accuracy and Log Likelihood are:

- Accuracy: 0.570
- Log Likelihood: 0.69315

4.4 Overall Results

In this section, we compare all classifiers over all featuring and stemming to determine the classifier that obtained the best results. We excluded the Maxent classifier since the result (Accuracy: 0.570) shows big variations compared to the other two remaining classifiers. When comparing the results, we can see that Naïve Bayes outperformed other classifiers in terms of Accuracy and F-score measures and the results was the best when using a stemmer, the results for stemmed data is shown in Table 1.

Table 1. Overall results

Measure	Decision Tree	Naive Bayes
Accuracy	0.7655	0.829
Positive F-score	0.77736	0.84658
Negative F-score	0.74976	0.80524

According to Table 1, stemming has a positive effect and the best Accuracy achieved when using Naïve Bayes with stemming and Mixngram with different sized features. Figure 3 shows the comparison of performance between Decision Tree classifier and Naive Bayes classifier.

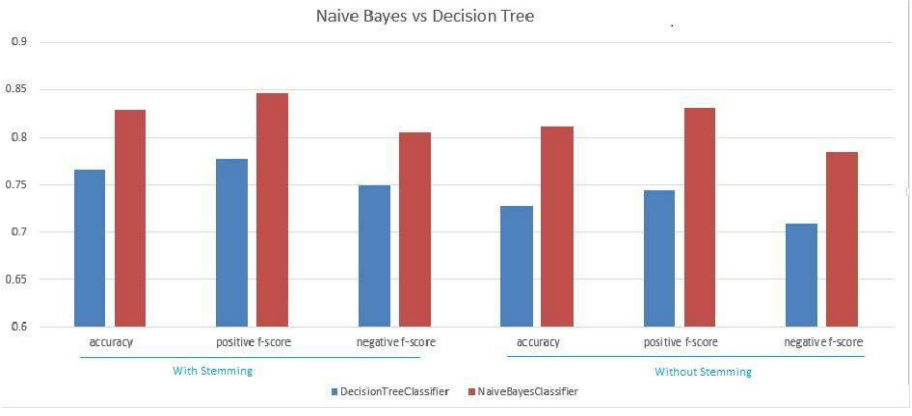

Fig. 3. Main measures of Decision Tree versus Naïve Bayes for Mixgram (best grams results)

5 Conclusion and future work

We have developed opinion mining classifier for Jordanian Dialect using three classifiers: Decision tree, Naive Bayes, and Maximum Entropy. We also studied the effect of stemming the data. We have achieved 0.829 in terms of Accuracy, and 0.84658 for Positive reviews in terms of F-score and 0.80524 for Negative reviews in terms of F-score using the Naive Bayes classifier. Larger labeled dataset should give better results since the available one has only 2730 reviews, and this size limits the features that can be extracted and used for prediction by a classifier. This needs lot of efforts, which we intend to spend in our future work. More classifiers can be explored like SVM.

References

[1] A. Soufan, "Deep Learning for Sentiment Analysis of Arabic Text," in *Proceedings of the ArabWIC 6th Annual International Conference Research Track*, 2019, p. 20.

[2] M. Alkhatib, M. El Barachi, and K. Shaalan, "An Arabic social media based framework for incidents and events monitoring in smart cities," *J. Clean. Prod.*, vol. 220, pp. 771–785, 2019.

[3] O. Al-Harbi, "Classifying Sentiment of Dialectal Arabic Reviews: A Semi-Supervised Approach," *Int. Arab J. Inf. Technol.*, vol. 16, no. 6, pp. 995–1002, 2019.

[4] F. Almuqhim, "Strategies for Sentiment Analysis and Classification of Non English Tweets," *Ph.D. thesis, Rochester Inst. Technol.*, 2016.

[5] S. A. Salloum, M. Alshurideh, A. Elnagar, and K. Shaalan, "Machine Learning and Deep Learning Techniques for Cybersecurity: A Review," in *Joint European-US Workshop on Applications of Invariance in Computer Vision*, 2020, pp. 50–57.

[6] S. A. Salloum, R. Khan, and K. Shaalan, "A Survey of Semantic Analysis Approaches," in *Joint European-US Workshop on Applications of Invariance in Computer Vision*, 2020, pp. 61–70.

[7] S. A. Salloum, M. Alshurideh, A. Elnagar, and K. Shaalan, "Mining in Educational Data: Review and Future Directions," in *Joint European-US Workshop on Applications of Invariance in Computer Vision*, 2020, pp. 92–102.

[8] K. M. Alomari, A. Q. AlHamad, and S. Salloum, "Prediction of the Digital Game Rating Systems based on the ESRB."

[9] M. S. Al-Batah, S. Mrayyen, and M. Alzaqebah, "Investigation of Naive Bayes Combined with Multilayer Perceptron for Arabic Sentiment Analysis and Opinion Mining.," *JCS*, vol. 14, no. 8, pp. 1104–1114, 2018.

[10] S. A. Salloum, A. Q. AlHamad, M. Al-Emran, and K. Shaalan, *A survey of Arabic text mining*, vol. 740. 2018.

[11] S. A. Salloum, M. Al-Emran, A. A. Monem, and K. Shaalan, *Using text mining techniques for extracting information from research articles*, vol. 740. 2018.

[12] S. A. Salloum, M. Al-Emran, and K. Shaalan, "A Survey of Lexical Functional Grammar in the Arabic Context," *Int. J. Com. Net. Tech*, vol. 4, no. 3, 2016.

[13] E. Omara, M. Mosa, and N. Ismail, "Deep Convolutional Network for Arabic Sentiment Analysis," in *2018 International Japan-Africa Conference on Electronics, Communications and Computations (JAC-ECC)*, 2018, pp. 155–159.

[14] J. O. Atoum and M. Nouman, "Sentiment analysis of Arabic jordanian dialect tweets," *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 2, pp. 256–262, 2019.

[15] I. A. Farha and W. Magdy, "Mazajak: An Online Arabic Sentiment Analyser," in *Proceedings of the Fourth Arabic Natural Language Processing Workshop*, 2019, pp. 192–198.

[16] O. Al-Harbi, "A Comparative Study of Feature Selection Methods for Dialectal Arabic Sentiment Classification Using Support Vector Machine," *arXiv Prepr. arXiv1902.06242*, 2019.

[17] M. M. Fouad, A. Mahany, N. Aljohani, R. A. Abbasi, and S.-U. Hassan, "ArWordVec: efficient word embedding models for Arabic tweets," *Soft Comput.*, pp. 1–8, 2019.

[18] H. Najadat, A. Al-Abdi, and Y. Sayaheen, "Model-based sentiment analysis of customer satisfaction for the Jordanian telecommunication companies," in *2018 9th International Conference on Information and Communication Systems (ICICS)*, 2018, pp. 233–237.

[19] A. Y. Al-Obaidi and V. W. Samawi, "Opinion mining: Analysis of comments written in arabic colloquial," in *Proceedings of the World Congress on Engineering and Computer Science*, 2016, vol. 1.

[20] M. N. Al-Kabi, H. A. Wahsheh, and I. M. Alsmadi, "Polarity classification of Arabic sentiments," *Int. J. Inf. Technol. Web Eng.*, vol. 11, no. 3, pp. 32–49, 2016.

[21] H. K. Aldayel and A. M. Azmi, "Arabic tweets sentiment analysis–a hybrid scheme," *J. Inf. Sci.*, vol. 42, no. 6, pp. 782–797, 2016.

[22] M. Daoud, "Building Arabic Polarizerd Lexicon from Rated Online Customer Reviews," in *2017 International Conference on New Trends in Computing Sciences (ICTCS)*, 2017, pp. 241–246.

[23] O. Al-Harbi, "Using objective words in the reviews to improve the colloquial arabic sentiment analysis," *arXiv Prepr. arXiv1709.08521*, 2017.

The Potential of Machine Learning Techniques for Student Performance Prediction Based on Large Dataset

Afiqah Zahirah Zakaria[a] , and Ali Selamat[a,b,d] , and Hamido Fujita[c] , and Ondrej Krejcar[d]

[a] Malaysia Japan International Institute of Technology (MJIIT), Universiti Teknologi Malaysia, Jalan Sultan Yahya Petra, Kuala Lumpur, Malaysia
[b] School of Computing, Faculty of Engineering, UTM & Media and Games Center of Excellence (MagicX), Universiti Teknologi Malaysia, Johor Bahru, Malaysia
[c] Faculty of Software and Information Science, Iwate Prefectural University, 152-52 Sugo, Takizawa, Iwate 020-0693, Japan
[d] Faculty of Informatics and Management, University of Hradec Kralove, Rokitanskeho 62, 50003 Hradec Kralove, Czech Republic

Abstract. Learning analytics is one of the growing fields in this decade. Students' performances become a hot topic to be solved all around the world to help the three main characters in the education field, which are learners, instructors, and administrators, to produce the capable leaders in the future. The leading machine learning techniques in many fields also encouraged the use of techniques in the educational area. In this paper, the approaches of three machine-learning techniques used to train the large dataset of students' information. Based on the information, the prediction of students' performance is classified based on four main classes, which are distinction, fail, pass, and withdrawn. The results are produced based on three evaluation parameters, which are accuracy, prediction speed, and training time after undergoing a feature selection process. The Bagged Tree from Ensemble Classifiers revealed as the potential machine learning technique. It proved by getting 99.6% accuracy to predict the students' performance based on the massive data of information.

Keywords: Machine Learning, Decision Tree, Ensemble Classifiers, Support Vector Machine, Student Performance, Bagging, Learning Analytics.

1 Introduction

Learning analytics is one of the current topics that are blooming in worldwide. Mainly, it involved three main characters, which are administrators, instructors, and students or learners; those are focusing on the students' database starting from the students register in the institution until the students become the alumni. All the data is a recording based on the real-time so that the responsible person-in-charge can analyze and report the result based on the data collection. In order to get that information, data mining needs to extract the information from the data [1].

Educational data mining is a field that focused on educational data [1]. It categorized by task [2], becoming an analysis or visualization of data [3, 4] or recommenda-

tion for the students [5]. The task can be in many forms, such as providing the feedback for supporting the instructors, predicting the students' performances, grouping the students, planning and scheduling the academic plans, and many more. Therefore, in this paper, the students' performance prediction is a chosen topic to be discussed further.

With the advancement of technology, one data can represent a lot of information. Due to that, the student data recorded are instrumental and valuable for the institution and instructors. It is usually related to personal information, demographic factors, family background, health, registration information, and examination or test results. Based on this information, it gives many branches of extra or new information related to the academic or education of each student. Besides, as a critical part of teaching and school routine management, the initial analysis and evaluation of student's achievement are needed. Therefore, the instructors must grasp the students' learning status accurately, enables the students to understand their learning situation and to measure the students' learning situation comprehensively [6].

Machine learning is a famous branch of Artificial Intelligence that can learn the data and come out with the new result to solve the current problems. It enables machines to perform and learn from their jobs by using intelligent software and structured algorithm [7]. Each machine learning classifier algorithm is defined by statistics that play an essential role in the machine learning itself [8]. It also requires data; hence, it produces different results on each case based on each machine-learning algorithm [8].

In this paper, the author tried to find out the best potential technique to implement in learning analytics. In this case, the author chose to predict based on the students' performances. The data recorded based on several parameters such as code module, code presentation, student ID, gender, region, highest education, number of previous attempts, studied credits, disability, and final results. Hence, the results measured based on accuracy, prediction speed, and training time as the evaluation parameters.

This paper is organized into five sections. Section 1 is about the introduction of the overall topic that the authors want to solve. The next section is an explanation regarding the previous researches related to the prediction in the education field (Section 2). Section 3 is about the steps throughout this research. It focused on the three techniques of Machine Learning, which are Fine Tree, Fine Gaussian SVM, and Bagged Tree. Section 4 is related to the results and discussions based on the techniques implemented in Section 3. Section 5 showed the summarized results of each technique based on accuracy. The last section is the conclusion regarding the whole research that can give benefits to the education field and the plan for improvisation in the future.

2 Related Research Works

With the growth of learning analytics and machine learning, the researchers came out with many ideas related to both fields. Based on the authors' observation, the researchers interested in upgrading existing techniques through the education field. Student's academic performance is a crucial factor in building their future [9]. The

classification method selected and implemented based on students' data [10]. Naïve Bayes (NB) is a classification method selected as a data mining technique [10]. Based on the academic performance of the students, NB produced an accuracy of 92.37%, as it was possible to obtain a good prediction model [10].

Classification methods used to categorize the data based on the given label and constraints [11]. The researchers chose NB, Random Forest, and J48 algorithms to classify the educational database by extracting using predictive data mining to predict the student's performance to enhance the study level of the students in the organization [11]. The NB showed the best performance accuracy as the clustering and apriori rules have a vital relationship in students' performance in the given dataset [11]. To predict the students' performance, the researchers chose machine learning as a decision support tool used by the educational agents to prevent a student from failing the course [10]. This is due to the machine learning is one of the capability algorithms to answer uncommon problems [11]. Based on this reason, the educational field is the best field to solve using this algorithm [8]. The researchers found out that kNN had the highest accuracy compared to Logistic Regression (LR), Linear Discriminant Analysis, Gaussian Naïve Bayes and Support Vector Machine (SVM), which is 96% [8].

N Lukman et al. (2019) came out with the prediction of national examination questions that will appear in the final examination using the C4.5 algorithm [12]. This is due to the score of the students for national examination was decreasing from 2015, and the researchers tried to help the students to increase the value of the score [12]. Based on the result, the C4.5 algorithm succeeds in getting an accuracy of 70% as the researchers proved that this algorithm had an excellent performance [12]. Also, research done to find out the best algorithms and features by predicting the performance of 76 second-year University students registered in the Computer Hardware course [12]. The research started in earlier weeks to predict the students' performance. The result showed that the k-Neural Network (kNN) is the best predictor classifier as it gave 89% accuracy based on the prediction of the unsuccessful students at the end of the term [13]. Based on the previous researches that summarized in Table 1, it proved that many researchers already used various techniques and methods in the education field. It shows that it is significant to have the best technique to get the best classification result for students' performance using a vast dataset.

Table 1: Comparisons of Previous Prediction Researches

No	Paper	Year Published	Technique	Result	Dataset
1.	eMineProve: Educational Data Mining for Predicting Performance Improvement Using Classification Method [10]	2019	Naïve Bayes	Accuracy: 92.37%	4250
2.	Enhancing the Quality Education using Predictive	2019	Naïve Bayes	Accuracy: 80%	25000

	and Descriptive Data Mining Model [11]				
3.	Machine Learning Technique Analysis and Applications for Predicting Student Performance [8]	2019	KNN	Accuracy: 96%	279
4.	Prediction of national examination question using C4.5 Algorithm [12]	2019	C4.5	Accuracy: 82%	268
5.	Using learning analytics to develop early warning system for at-risk students [13]	2019	KNN	Accuracy: 89%	76

3 Predicting the Students' Performances Using Machine Learning Techniques

3.1 Data Preprocessing

The data used is based on the Kaggle Online Database [14]. The database is consists of 32,593 information about the students in the United States of America. The data has 9 predictors, which are `code_module`, `code_presentation`, `id_student`, `gender`, `region`, `highest_education`, `num_of_prev_attempts`, `studied_credits`, `disability`. The data preprocessing is the process of transforming the data completely into an understandable format before the dataset continuing the learning process [14]. It is essential to remove all the incomplete, noise, and inconsistent data to produce accurate value as the result of the learning process. In this paper, the author implemented the data cleaning, which is replacing the missing value with zero.

3.2 Feature Selection (FS)

A feature selection is also known as variable selection or attribute selection. Feature selection usually can lead to better learning performance, for example, higher learning accuracy, lower computational cost, and better model interpretability [16]. The feature selection method usually includes and excludes attributes present in the data without changing them. This method is used to identify and remove unneeded, irrelevant, and redundant attributes from data that do not contribute to the accuracy of a predictive model or may decrease the accuracy of the model [16]. The fewer attributes are desirable because it reduces the complexity of the model, and a simpler model is simpler to understand and explain.

3.3 Decision Tree

Decision Tree (DT) is a graphical display like a tree of a particular decision condition that used when a complex gap occurs in a structured decision process. An action performed when each rule in a decision tree that displayed by tracking a series of paths from the root to the node to the next node continuously [17]. The decision tree has properties in the data found in the upper nodes in the tree structure, where the marginal properties are set aside [17]. Based on that, the DT is chosen to use if useful or not. In order to search for the possible DT, the algorithm developed usually uses a greedy and top-down approach for the version of basic algorithms [17].

Besides, DT is one of the popular machine learning techniques that already implemented widely in many fields in the world. Based on the standard CART algorithm, decision tree design is presented [18]. The dataset was recursively divided according to the split criterion throughout the training process until the optimal DT hierarchy of nodes was reached. In order to measure of node goodness in DT, a Gini's Diversity Index (GDI) is selected as the split optimization criterion in the DT model. The node is pure as it contains only observations of one class (ABMR + ve and ABMR - ve), the GDI of a pure node is equal to 0 [18].

Besides, DTs are one of the main techniques for discriminant analysis in knowledge discovery. DTs are, to some extent, prone to overfitting due to their non-parametric and flexible algorithm [18]. There are also known to be unstable, as small variations in the training set can result in different trees and non-repeatable predictions [19].

For this technique, the maximum number of splits is chosen the manipulation variable in this research. This is due to the decision tree is about partitioning the dataset into the subsets. The authors changed the maximum number of splits 500 per training using Gini's diversity index as the split criterion. The surrogate decision splits is on by 10 maximum surrogates per nodes.

3.4 Support Vector Machine

SVM is an algorithm used in both classification and regression [20]. On the apace, the data points are described and are categorized into groups, while the similar properties of points are located in the same group in the SVM model [20]. For linear SVM, the given dataset is considered as a p-dimensional vector that can be separated by a maximum of p-1 planes called a hyperplane [20]. For solving classification and regression problems, among the data groups, both planes separated by the data space or set the boundaries [20]. The best hyperplane selected among the number of hyperplanes based on the distance between the two classes it separates [20]. The plane that has the maximum margin between the two- classes proves the maximum-margin hyperplane [20, 21, 22].

In this research, the authors focused on using Fine Gaussian SVM, which means the kernel function is Gaussian. The box constraint level is starting from 100 and increasing 200 for each training process. The box constraint should be more significant; it can reduce the appropriate time and have a smaller number of support vector

used in SVM. This is due to this research is a hard margin that can separate positive and negative points. Besides, the kernel scale mode is manual at static scale 0.1, and the multiclass method is one to one.

3.5 Ensemble Classifier

Ensemble Classifier is the combination technique of Machine Learning techniques with other techniques to get a more efficient technique to solve many problems in several fields. It approaches to train multiple classifiers independently before joining them to create the final classifier [23]. The ensemble classifier is parallel nature makes it well suited for the problem of distributing sequential single-node classification algorithms across many computing nodes [23]. Therefore, the ensemble classifier is a good approach to have the best classification model for solving the problems.

The Bagged Tree is selected as the ensemble classifier in this research. In the ensemble classifier, the number of learners generated from training data with the help of a base learning algorithm. Therefore, in this research, the number of learners is increasing by 50 numbers for each training process with a fixed learning rate = 0.1, and the maximum number of splits is 32,592, as the dataset is 32,593.

3.6 Evaluation Parameters

Accuracy is acting as a metric to evaluate the classification technique, whether the prediction model is correct or not. Indeed, it measures how often the classifier makes the correct prediction [24].

$$\text{Overall Accuracy} = \frac{\sum_{i=1}^{N} Ci, i}{\sum_{i=1}^{N} \sum_{j=1}^{N} C_{i,j}}$$

Prediction speed is the speed recorded for complete training. The speed is measured in obs/sec. While training time is a time taken for the full training to be completed. The unit is counted in seconds (sec).

4 Results and Discussion

In this section, the results based on Decision Tree, SVM and Ensemble Classifiers are revealed. The results are based on accuracy, prediction time and training time. Based on the Section 3, these three techniques have their own algorithms and specialty. Due to that, each technique came out with the different results, as each algorithm is clearly different from one to another.

4.1 Early Prediction Results of Machine Learning

At the early stage, these three machine learning algorithms are undergoing the training process before undergoing a feature selection method. The graph below shows the early prediction results of machine learning.

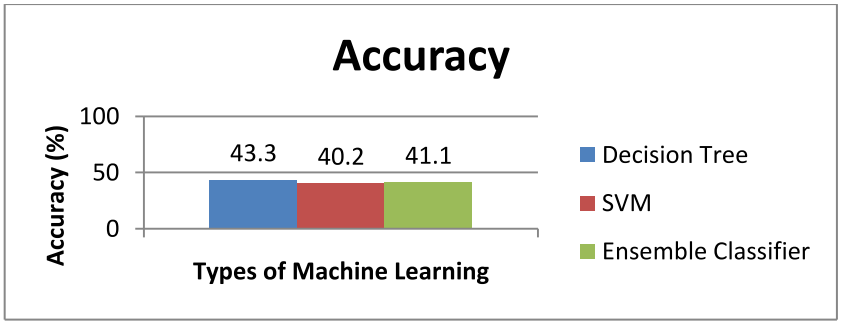

Figure 1: The Early Prediction Results of Machine Learning

Based on Figure 1, the accuracy of these three machine learning is recorded. It shows that Decision Tree shows the highest accuracy at the early prediction. The accuracy is 43.3%. The SVM shows the lowest accuracy, 40.2% at the early prediction. Lastly, the ensemble classifier has 41.1% for this early prediction training process.

4.2 Results for Decision Tree

Table 2 below shows the full results of the Decision Tree. The type of Decision Tree chosen is Fine Tree. The highest accuracy of Fine Tree is 70.7%. Even though the maximum number of splits 7000, the prediction speed is the slowest at 920000 obs/sec at 1.6514 seconds. The 100 maximum numbers of splits only gave 44.6% at the shortest training time, which is 0.79156 seconds. The prediction speed in this decision tree method is quite unpredictable, as, at 600 numbers of splits, the speed becomes the highest speed, 10000000 obs/sec. This is due to the tree have the branches that we cannot estimate roughly the speed and time accordingly based on the accuracy stated.

Table 2: Results of the Decision Tree

Classifier	Type of classifier	Maximum number of splits	Accuracy (%)	Prediction speed (obs/sec)	Training time (sec)
Decision Tree	Fine Tree	100	44.6	1500000	0.79156
		500	47.7	1200000	0.93357
		1000	50.3	1200000	0.9609
		1500	52.5	1400000	0.93069
		2000	54.7	1100000	0.98705
		2500	56.7	1300000	1.1111

		3000	58.6	1200000	1.0642
		3500	60.5	1100000	1.0942
		4000	62.2	960000	1.148
		4500	63.8	950000	1.2883
		5000	65.6	710000	1.4505
		5500	67.3	920000	1.3078
		6000	68.8	10000000	1.4196
		6500	70.1	960000	1.5536
		7000	70.7	920000	1.6514
		7500	70.7	720000	1.442
		8000	70.7	340000	1.6957

4.3 Results for Support Vector Machine

Results of Fine Gaussian SVM are summarized as below. The maximum box constraint level is 1000 that gave the highest accuracy, which is 83.2%. The least number of box constraint levels gave the least accuracy to 80.2%. The accuracy in SVM shows the accuracy is increasing in a small amount even though the box constraint has 100 differences between each training process. The fastest prediction speed is 600 obs/sec for both 600 and 1000 of the box constraint level. While the shortest time is 869.97 seconds, and the longest time is 2621.2 seconds.

Table 3: Results of Fine Gaussian SVM

Classifier	Type of classifier	Box constraint level	Accuracy (%)	Prediction speed (obs/sec)	Training time (sec)
Support Vector Machine	Fine Gaussian	100	80.2	640	869.97
		200	81.3	630	1210.4
		300	81.8	600	1464.7
		400	82.1	650	1669.7
		500	82.4	650	1934.1
		600	82.6	660	2171.8
		700	82.8	620	2390.5
		800	83.0	590	2540.0
		900	83.1	620	2647.4
		1000	83.2	660	2621.2

4.4 Results for Ensemble Classifier

Table 4 below shows the results of the Ensemble Classifier based on Bagged Tree technique. Surprisingly, the results show the accuracy nearest to 100% accuracy throughout the training process. The accuracy recorded is 99.6% starting from 10 numbers of learners until 600 numbers of learners. Even though the number of learn-

ers increases by 50 for each training, the results of accuracy still maintain. However, the prediction becomes slower as the difference between the 10 number of leaners and 600 numbers of learners is 85900 obs/sec. It is vice versa with the training time; the time keeps increasing as the more time needed as the number of learners increase. The highest training time is 429.82 sec for 600 numbers of learners.

Table 4: Results for Bagged Tree

Classifier	Ensemble Method	Number of learners	Accuracy (%)	Prediction speed (obs/sec)	Training time (sec)
Bagged Tree algorithm	AdaBoost	10	99.6	87000	8.9252
		50	99.6	16000	37.353
		100	99.6	9000	86.751
		150	99.6	6100	105.16
		200	99.6	4500	131.01
		250	99.6	3400	162.52
		300	99.6	2800	198.22
		350	99.6	2200	253.97
		400	99.6	1900	272.33
		450	99.6	1500	305.47
		500	99.6	1400	348.6
		550	99.6	1200	390.46
		600	99.6	1100	429.82

Based on the experiments above, the results show the differences from one to other techniques. It is clearly shown that the results of speed and training time cannot be predicted as the accuracy is increasing. For Decision Tree, even though the accuracy is increasing, the speed is not increased continuously, and the time is also changing unpredictably. This is due to the graphic tree-like structure that cannot handle the large dataset, and it took some time to solve the training process. Indeed, a maximum number of splits in the decision tree also affected the accuracy of training. The increasing number of splits, it will increase the accuracy as well, and however, it has the limitation that needs to be cover in the future.

SVM also produced the results just like the Decision Tree. The unpredictably results of speed and training time is due to the SVM model itself is about to categorize the data from non-linear separated function into higher dimension linearly separable functions. During this process, it might need some time, and less speed as the dataset used is more significant. This research proved that if the box constraint level is higher, the accuracy is increasing in percentage.

Lastly, the ensemble classifier, like Bagged Tree, can predict the accuracy nearest to 100% as the bagging function itself. It uses bootstrap sampling to obtain the data subsets for training the base learners [25]. The ensemble classifier proved that even though the number of learners is increasing, the accuracy is still constant at the highest accuracy, as the ensemble classifiers are well known the ability to boost the weak learner. Therefore, it helps the algorithm to work efficiently and effectively

even though it used the large dataset and can predict the students' results according to the four classes easily.

5 Accuracy of Machine Learning

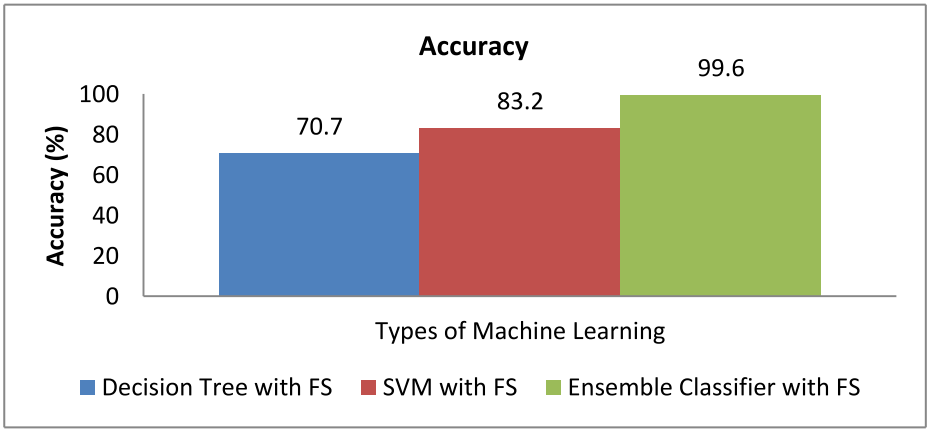

Figure 2: The Results of the Accuracy of Machine Learning

Figure 2 shows the results of the accuracy of machine learning techniques after undergoing feature selection. The Ensemble Classifier based on Bagged Tree proved as the highest accuracy as 99.6% compared to SVM and Decision Tree. The SVM by Fine Gaussian method has 83.2% of accuracy. Lastly, the least accuracy is Decision Tree using Fine Tree is only 70.7%. That shows that the Bagged Tree shows an accurate prediction of the final result based on the distinction, pass, fail, and withdrawn based on 32,593 of student information datasets.

6 Conclusion

Education is a relevant field that can determine the success of future generations. Due to this, the advancement of technology, learning analytics, and a large number of the dataset should be improved together. The prediction of students' performance is significant to develop a capable future leader at the early stage. Besides, the results are needed to the three main parties of learning analytics, which are learners, administrators, and instructors. Learners can identify aware their weaknesses at the early stage, instructors can find out the appropriate ways to enhance the learners' performance, and administrators can standby on the best equipment and tools needed by the instructors to help the learners. Besides, the parents can give extra class and motivation as an additional initiative to have a competent learner in the future. Even though most of the Machine Learning techniques are capable of being trained in massive learning data, it

still needs enhancement to produce a stable technique Even though most of the Machine Learning techniques are capable of being trained in the massive learning data; it still needs enhancement to produce a stable technique. The technique that has the highest percentage of accuracy is the most factors need to produce the most suitable technique to predict the students' performance in education fields. Therefore, in the future, the combination or improvisation of Ensemble Classifiers based on Bagged Tree with other techniques or algorithms is valuable to solve the problem regarding the learning analytics or other fields.

Acknowledgments: The authors wish to thank Universiti Teknologi Malaysia (UTM) under Research University Grant Vot-20H04, Malaysia Research University Network (MRUN) Vot 4L876 and the Fundamental Research Grant Scheme (FRGS) Vot 5F073 supported under Ministry of Education Malaysia for the completion of the research. The works were also supported by the SPEV project, University of Hradec Kralove, FIM, Czech Republic (ID: 2102-2020). We are also grateful for the support of Ph.D. student Sebastien Mambou in consultations regarding application aspects.

References

1. Jauhari, F. and Supianto, A. A. (2019) 'Building student's performance decision tree classifier using boosting algorithm', 14(3), pp. 1298–1304. doi: 10.11591/ijeecs.v14.i3.pp1298-1304.
2. C. Romero and S. Ventura, "Educational data mining: A review of state of the art," *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.*, vol. 40, no. 6, pp. 601–618, 2010.
3. A. A. Supianto, Y. Hayashi, and T. Hirashima, "Visualizations of problem-posing activity sequences toward modelling the thinking process," *Res. Pract. Technol. Enhanc. Learn.*, vol. 11, no. 1, p. 14, 2016.
4. A. A. Supianto, Y. Hayashi, and T. Hirashima, "Process-based Assignment-Setting Change for Support of Overcoming Bottlenecks in Learning by Problem-Posing in Arithmetic Word Problems," in *Journal of Physics: Conference Series*, 2017, vol. 812, no. 1, p. 012004.
5. P. Kaur, M. Singh, and G. S. Josan, "Classification and Prediction Based Data Mining Algorithms to Predict Slow Learners in Education Sector," *Procedia Comput. Sci.*, vol. 57, pp. 500–508, 2015.
6. T. Hu, and T. Song, 'Research on XGboost academic forecasting and analysis modelling'. doi: 10.1088/1742-6596/1324/1/012091, 2019.
7. Mohammed, Mohssen & Khan, Muhammad & Bashier, Eihab. (2016). Machine Learning: Algorithms and Applications. 10.1201/9781315371658.
8. Madyatmadja, E. D. and Susanto, A. J. (2019) 'Machine Learning Technique Analysis and Applications for Predicting Student Performance', (1), pp. 2133–2136. doi: 10.35940/ijeat.A9678.109119.
9. Baker, R. S., Corbett, A. T., & Koedinger, K. R. (2004). Detecting student misuse of intelligent tutoring systems. International conference on intelligent tutoring systems (pp. 531–540). Springer.
10. Rosado, J. T., Payne, A. P. and Rebong, C. B. (2019) 'eMineProve : Educational Data Mining for Predicting Performance Improvement Using Classification Method'. doi: 10.1088/1757-899X/649/1/012018.

11. Jayakumar, S., Parameswari, R. and Akila, A. (2019) 'Enhancing the Quality Education using Predictive and Descriptive Data Mining Model', (3), pp. 6843–6847. doi: 10.35940/ijrte.C5795.098319.
12. Lukman, N., Subaeki, B., Abdullah, H.N., Atmadjal, A.R. and Wildanuddin, M. (2019) 'Prediction of national examination question using C4.5 algorithm', *J. Phys.: Conf. Ser.* 1402 077016. doi:10.1088/1742-6596/1402/7/077016.
13. Akçap, G., Altun, A. and Petek, A. (2019) 'Using learning analytics to develop early-warning system for at-risk students', 16:40. https://doi.org/10.1186/s41239-019-0172-z.
14. Kuzilek, J., Hlosta, M. and Zdrahal, Z. (2017) 'Data Descriptor : Open University Learning Analytics dataset', pp. 1–8.
15. Mahdi, A., Alzubaidi, N. and Al-shamery, E. S. (2020) 'Projection pursuit Random Forest using discriminant feature analysis model for churners prediction in telecom industry', 10(2), pp. 1406–1421. doi: 10.11591/ijece.v10i2.pp1406-1421.
16. Miao, J. and Niu, L. (2016) 'A Survey on Feature Selection', *Procedia - Procedia Computer Science*. Elsevier Masson SAS, 91(Itqm), pp. 919–926. doi: 10.1016/j.procs.2016.07.111.
17. Shayan, P. and Zaanen, M. Van (2019) 'Predicting Student Performance from Their Behavior in Learning Management Systems', 9(5), pp. 337–341. doi: 10.18178/ijiet.2019.9.5.1223.
18. Shaikhina, T. *et al.* (2019) 'Biomedical Signal Processing and Control Decision tree and random forest models for outcome prediction in antibody incompatible kidney transplantation', *Biomedical Signal Processing and Control*. Elsevier Ltd, 52, pp. 456–462. doi: 10.1016/j.bspc.2017.01.012.
19. Czajkowski, M. and Kretowski, M. (2019) 'Decision tree underfitting in mining of gene expression data. An evolutionary multi-test tree approach', *Expert Systems with Applications*. Elsevier Ltd, 137, pp. 392–404. doi: 10.1016/j.eswa.2019.07.019.
20. Kaur, H. and Kumari, V. (2018) 'Applied Computing and Informatics Predictive modelling and analytics for diabetes using a machine learning approach', *Applied Computing and Informatics*. King Saud University, pp. 0–5. doi: 10.1016/j.aci.2018.12.004.
21. M.A. Hearst, S.T. Dumais, E. Osuna, J. Platt, B. Scholkopf, Support vector machines, IEEE Intell. Syst. Appl. 13 (4) (1998) 18–28.
22. G.B. Huang, Q.Y. Zhu, C.K. Siew, Extreme learning machine: theory and applications, Neurocomputing 70 (1) (2006) 489–501.
23. Khalifa, S., Martin, P. and Young, R. (2019) 'Label-Aware Distributed Ensemble Learning : A Simplified Distributed Classifier Training Model for Big Data', *Big Data Research*. Elsevier Inc., 15, pp. 1–11. doi: 10.1016/j.bdr.2018.11.001.
24. Roshan, S. E. and Asadi, S. (2020) 'Engineering Applications of Artificial Intelligence Improvement of Bagging performance for classification of imbalanced datasets using evolutionary multi-objective optimization', *Engineering Applications of Artificial Intelligence*. Elsevier Ltd, 87(November 2019), p. 103319. doi: 10.1016/j.engappai.2019.103319.
25. Deepika, K. and Sathyanarayana, N. (2019) 'Classification and Prediction of Student Academic Performance using Gray Wolf Optimization Based Relief-F Budget Random', (3), pp. 4411–4418. doi: 10.35940/ijrte.C5534.098319.

Research on Deep Reinforcement Learning-based Top-N Recommendation*

Bo Huang[1], Wenzhu Liu[1], Wei Zhang[1], Zhijun Fang[1], and Xing Wu[2,3]

[1] School of Electronic and Electrical Engineering, Shanghai University of Engineering Science, Shanghai 201620, China
[2] School of Computer Engineering and Science, Shanghai University, Shanghai, China
[3] Shanghai Institute for Advanced Communication and Data Science, Shanghai, China
huangbosues@sues.edu.cn

Abstract. Recommender system suggests personalized items and services to the user. It is frequently used in solving information overload problem. The most significant challenge in recommender systems is that the users usually have diverse interests, which tends to change dynamically. Although some researches have been carried out, these methods usually view recommendation as a static process and make recommendations according to a fixed strategy. In this paper, we propose a novel idea which considers recommendation as a dynamic sequential decisive process and solves it with deep reinforcement learning algorithm. Specifically, we model the recommendation process as a Markov Decision Process (MDP). Then Deep Deterministic Policy Gradient (DDPG) is used to learn the optimal recommended strategy. Furthermore, we proposed a reward function which consists of the precision of recommended strategy and the cosine similarity between recommended items and user's history rating records. Finally, N nearest neighbors of the recommendation produced by Agent are collected to cope the exploration problem.

Keywords: Recommender System,· Deep Reinforcement Learning,· Markov Decision Process,· Reward Function.

1 Introduction

The recommender system is the most effective way to solve the problem of information overload. Many different methods have been proposed, such as the content-based recommendation, the collaborative filter. For the continuous improvement of the learning ability of neural networks, most of the deep learning-based algorithms [1] performed well, but there are still some limitations. Firstly, the recommendation is considered as a static process, and it is measured by immediate rewards rather than future ones, which can improve recommendation performance for the long term [2]. Therefore, it is necessary to calculate future rewards and immediate rewards at the same time. Secondly, these studies only

* Supported by National Natural Science Foundation of China (No.61603242).

take clicks and skips of a user as feedbacks, which may lead to reward sparsity. Thirdly, it is difficult to make exploration since the state space and the action space are both abundant in recommendation scenarios. Traditional methods such as ϵ-greedy policy, Gaussian noise and parameter noise[3,4].They may introduce noise to make the unrelated recommendation that impacts the robustness of the recommender system. Contributions can be summarized as follows:

• To address these issues, we introduce deep reinforcement learning in our research. Firstly, we model the recommendation process as a Markov Decision Process and the recommender system and the users are consider as the Agent and the Environment respectively. Then, the Deep Deterministic Policy Gradient(DDPG) is applied. The recommender system will constantly interact with users according to DDPG and the optimal recommended strategy will be found.

• Then, a reward function is proposed which consists of the precision of recommended strategy and the cosine similarity between actions generated by the Agent and users history rating records. This reward function is different from the previous research and it could effectively deals with sparse rewards and makes the goal of the Agent clear.

• Finally, as mentioned above, that general exploration methods will be introduced with noise, which will harm the robustness of the system. In this paper, nearest neighbors of the action generated by Agent are gathered as a recommended list, which is recommended to users for further exploration to keep the stability of the system.

2 Literature Review

2.1 Recommender System

Collaborative filter, content-based, knowledge-based, and hybrid recommender are included in traditional recommender systems [5]. Collaborative filter, as the most popular one, is utilized to predict user's preferences based on the user's past behavioral data. However this method depends on historical data and it is usually not for the fickle users. The content-based method realizes recommendations by finding the similarity between user features and item features. the quality of the feature projects and finite user features will impair recommendation performance, such as recommendation accuracy.Hybrid model combines different approaches mentioned above to achieve recommendation.

2.2 Reinforcement Learning in Recommendation

Reinforcement learning, as a hot research area, which has several applications in many respects and recommendation is an appropriate one. Shani et. al.[6] suggests that, as a sequential decision problem, the process of the recommender engine generating a recommendation can be identified as a MDP. Based on their studies, researchers started to use reinforcement learning to solve the current issues in recommendation.Taghipour et. al.[7]proposes a web page recommendation based on Q-learning, but this method is not suitable for the task which

has enormous state and action space. For this issue, Choi et. al. [8] mentions biclustering-based Q-learning to cluster user and item. Hu et. al.[9] introduces RLWRec model clustering songs and recommends a kind of songs replaces recommends a song. Liebman et. al.[10]presents DJ-MC, a reinforcement-learning framework for music recommendation, based on a model of preferences for both songs and song transition. Zheng et. al.[2] thinks that the future reward of the recommender has a significant influence on the recommendation performance, and they propose a Deep Q-learning based recommendation framework, which can model reward explicitly. Moreover, Zheng further consider user return pattern as a supplement with click / no click label to capture more user feedbacks. Munemasa et. al.[11] uses Latent Dirichlet Allocation (LDA) to convert store information into distributed representation, and the click labels are adopted to calculates the reward. Finally, Munemasa use the DDPG to solve the store recommendation problem.

Related studies have shown the feasibility of the reinforcement learning algorithm in the field of recommendation. Based on the above researches, this paper proposes the deep reinforcement learning based Top-N recommendation model(DTNR) and the DDPG algorithm is utilized to realize the Top-N items recommendation.

3 Problem Description

It is necessary to summarize recommendations as a MDP for DDPG algorithm, and view the recommender system as an Agent and the user as Environment. S is the environment state space corresponding to the user,$s_{u,t} \in S$ and $s_{u,t} = item_t^1, \cdots, item_t^K$ is the history record of user u at time t, $K(K \in N^*)$ is the length of a state. A is the action space, including all items which the Agent can recommend to users, $a_{u,t} \in A$ is the item output action by Agent according to $s_{u,t}$ at time t, then we collect n nearest neighbor of $a_{u,t}$ and recommend those to users and realize Top-N recommendation. R is the reward function, $R_{u,t}(s_{u,t}, a_{u,t})$ means the reward of recommended items and it can modelling users' preferences enables Agent to roughly capture the user's interests. P refers to transition probability, $P_{s_{u,t}s_{u,t+1}}^{a_{u,t}} = P[s_{u,t+1}|s_{u,t}, a_{u,t}]$ indicates the probability of state $s_{u,t}$ and its corresponding uncertain action $a_{u,t}$ transfers to $s_{u,t+1}$ which means the probability of the next state. $\gamma(\gamma \in [0, 1])$ is the discount factor that can weigh the impact of future rewards on long-term cumulative rewards.

As mentioned above, the recommendation can be considered as a quintuple (s, a, p, r, γ). Agent learn how to generate appropriate recommendation and it is known as policy π. For the policy π , $a_{u,t} = \pi(s_{u,t})$. This paper aims to find the optimal recommended policy π^* [12] which can satisfy the user's interest to a large extent, and it can be defined as the equation (1):

$$\pi^* : max(\gamma^0 r_{u,t} + \gamma^1 r_{u,t+1} + \gamma^2 r_{u,t+2} + \cdots + \gamma^{n-t} r_{u,n}) \tag{1}$$

Where $r_{u,t}$ is the reward of recommended items. γ is the discount factor. When $\gamma = 0$, the policy only focuses on immediate rewards $\gamma_{u,t}$ and other parts of the

equation equal to 0. When $\gamma > 0$, the policy will focus on long-term cumulative rewards $\gamma_{u,t} + \gamma\gamma_{u,t+1} + \gamma^2\gamma_{u,t+2} + \ldots\ldots$.

4 The Recommendation Algorithm

This section mainly proposes the methods, such as the procedure of how the Agent makes recommendations and the design of the reward function indispensable for successful recommendation.

4.1 The Generation of Top-N Recommendation

Exploration aims to explore the action space more thoroughly and find diverse policy. The conventional exploring methods include adding random Gaussian noise or parametric noise. These method may not be suitable for recommendation problems because the addition of noise may generates recommendations which utterly unrelated to users' interests. It will reduce the recommended accuracy and stability of the recommender system. The paper collects n nearest neighbors of $a_{u,t}$, and recommends these to users to realize Top-N recommendation and mining at the same time. Agent gets $s_{u,t}$ of user u and output action $a_{u,t}$. We compute the similarity between $a_{u,t}$ and items from action space A .

$$sim\_cos(a_{u,t}, Item_i) = \frac{a_{u,t} Item_i}{\|a_{u,t}\| \|Item_i\|} \tag{2}$$

Obtain n nearest neighbors of $a_{u,t}$:

$$n\_list_{u,t} = \{nearest_1, nearest_2, \cdots, nearest_n\} \tag{3}$$

$$nearest_i = argmax_{Item_i \in Item}(sim\_cos(a_{u,t}, Item_i)) \tag{4}$$

Which $nearest_i$ is the nearest neighbor of $a_{u,t}$ according to cosine similarity.

4.2 The Reward Function

Previous studies design the reward function by users' feedback like skips and clicks or similarities between user rated items(list of items actually marked by users) and the $n\_list_{u,t}$ (recommended item list).In this paper, we not only use similarity to design reward function but also introduced recommendation precision, which is different from previous research. Therefore, the reward function contains two parts.The first part is the similarity between $n\_list_{u,t}$ and user's total history list $user\_list_u$:

$$reward_1 = similarity(n\_list_{u,t}, user\_list_u) \tag{5}$$

The second part is the recommendation precision of $n\_list_{u,t}$:

$$reward_2 = \frac{|n\_list_{u,t} \bigcap \overline{user\_list_u}|}{|n\_list_{u,t}|} \tag{6}$$

Therefore, the total reward function at time t is:

$$R_{u,t}(s_{u,t}, a_{u,t}) = \alpha * reward_1 + \beta * reward_2 \tag{7}$$

In particular, $\alpha \in (0, 1]$ and α is steadily reduced. At the beginning of the learning process, the policy tends to recommend randomly; thus $reward_2$ is so sparse that cannot guide the Agent learning the optimized policy and $reward_1$ works. As the training progresses, the Agent slowly learns better strategies than random recommendation. The similarity-based reward weight gradually decreases, and the recommendation accuracy-based reward weight gradually increase. It begins to guide the recommendation Agent to learn the optimal recommended strategy. $\beta > 1$ is an amplification factor of $reward_2$ to accelerate the convergence.

4.3 The Reinforcement Learning

DDPG algorithm is used and set $Q(s, a|\theta^\mu)$, f_{θ^ρ} to respresent the Critic and Actor network,respectively. They both have two sub-networks, namely, evaluate and target network. In addition, a Replay Memory D is needed to store interactive data (s, a, r, s') generate by Actor and Critic network.

The learning of the optimal recommendation policy can be divided into two parts, as shown in Figure 1. The first part is the interaction process between Agent and Environment. At each moment, the Environment transmits a state $s_{u,t}$ to the Agent; The Agent outputs action $a_{u,t} = f_{\theta^\rho}(s_{u,t})$, and further generates a Top-N recommendation list $n\_list_{u,t}$ as described in Section 4.1. Then, the Environment evaluates the reward $r_{u,t}$ (as shown in equation (9)) and updates the state to $s_{u,t+1}$. Store the data $(s_{u,t}, a_{u,t}, r_{u,t}, s_{u,t+1})$ in Replay Memory D .

The second part is the training process of the optimal recommendation policy. The mini-batch $(s_{u,t}, a_{u,t}, r_{u,t}, s_{u,t+1})$ are randomly obtained from D and transmits them to Actor and Critic neural network respectively. For Actor, as a policy-based network, needs to introduce J_{θ^ρ} as the object function and update the network by equation (10).Critic, as a value-based network, will update by gradient descent method according to equation (12), where $y = r_{u,t} + \gamma Q'(s_{u,t+1}, a'|\theta^{\mu'})$ and $\gamma \in [0, 1]$ is the discount factor.

$$\nabla_{\theta^\rho} J_{\theta^\rho} = \frac{1}{m} \sum_{i}^{m} \nabla Q(s_{u,t}, a|\theta^\mu) \nabla_{\theta^\rho} f_{\theta^\rho}(s_{u,t}) \tag{8}$$

$$L = \frac{1}{m} \sum_{t=1}^{m} (y - Q(s_{u,t}, a_{u,t}|\theta^\mu))^2 \tag{9}$$

$$\nabla_{\theta^\mu} L(\theta^\mu) = \frac{1}{m} [(y - Q(s_{u,t}, a_{u,t}|\theta^\mu)) \nabla_{\theta^\mu} Q(s_{u,t}, a_{u,t}|\theta^\mu)] \tag{10}$$

In addition, we just training the parameters of the evaluate network, the target network replicates the parameters of the evaluate network every once in a while.

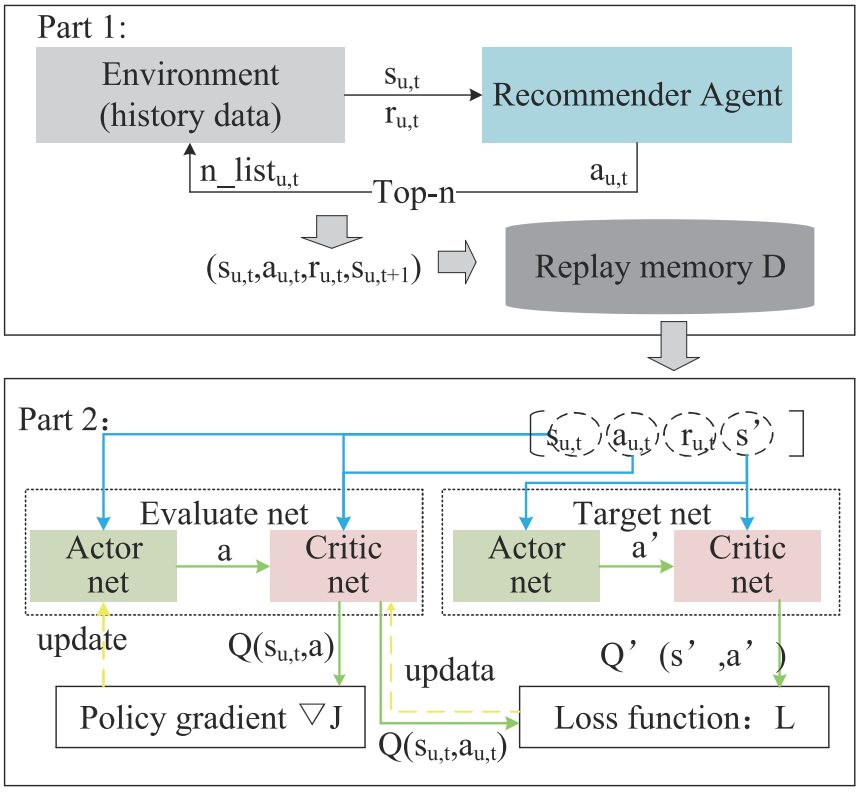

Fig. 1. Learning process of recommendation strategies

5 The Experiment

The experiment is performed on movielens-ml-20 dataset. Word2vec[13] is applied to learn the presentation of movie items and two experiments are conducted. Additionally, we choose precision and recall rate[14] as the evaluation criterion.

Here, we discuss the results of the first experiment to verify the effectiveness of exploration method and reward function described in section 4. Figure 2 shows the scores of Precision@20 for every 1000 episodes when changing the different settings. The black line is the DTNR method. The blue one stands for exploration by Gaussian Noise rather than nearest neighbor exploration. And the red one is the DTNR method without amplification factor.

As shown in Figure 2, DTNR method can lead to convergence at episode 8000, with the precision being 0.132, and then recommendation precision remains stable. The method without amplification factor β reaches convergence at episode 25000, the rate of convergence being slower than DTNR. These prove that amplification factor β does help quicken convergence. In addition, the pre-

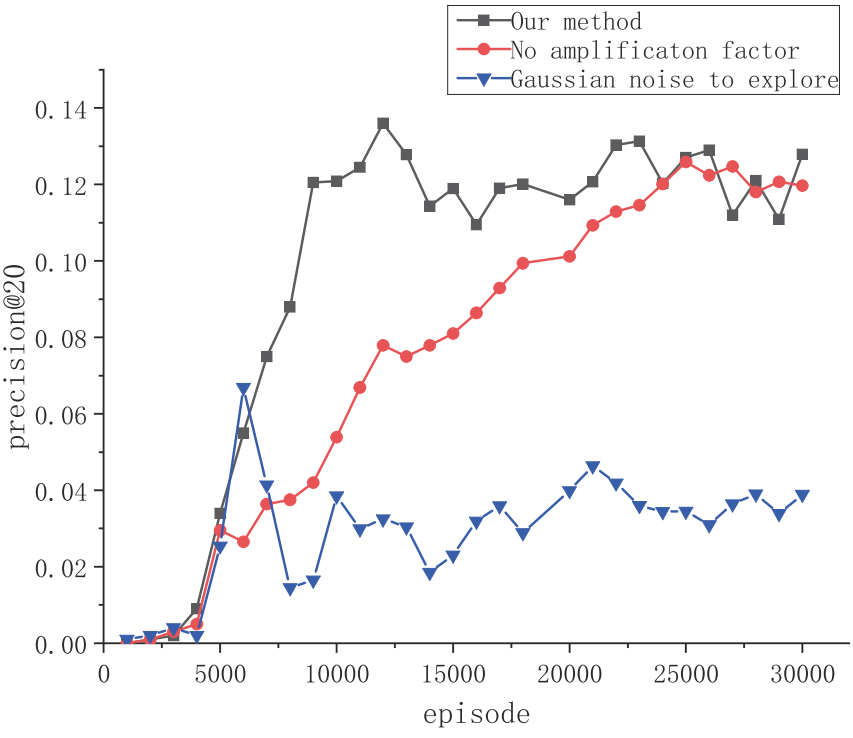

Fig. 2. The result of *Precision@20*

cision of the method with Gaussian noise exploration is much lower and more fluctuant, which is consistent with previous statements in Section 4.

The second experiment is the contrast experiment to confirm the validity of DTNR method and Item2vec and popularity-based Top-N algorithm are choosed.

Item2vec: O.Barkan[15] transfers Skip-gram with Negative Sampling model to Item-based collaborative filter in the music domain with good performance. We take this method in movie recommendation as a comparison.

Popularity-based Top-N Recommendation: This method recommends the most popular N movies to users.

Table 1. the result of the contrast experiment

Algorithm	recall@10	recall@20	precision@10	precision@20
DTNR	**0.078**	0.101	**0.118**	**0.132**
Item2vec	0.073	**0.124**	0.110	0.113
Popularity-based Top-N	0.049	0.088	0.103	0.095

The contrast results are shown in Table 1. Evidently, the method proposed in this paper is inferior to Item2vec at recall@20, but the score of precision@10, precision@20 and recall@10 is superior to others.

6 The Conclusion

This paper proposes deep reinforcement learning based on Top-N recommendation method and some positive effect is obtained. The further research will focus on representation learning on action space or the definition of recommendation problem in deep reinforcement learning.

References

1. Singhal A, Sinha P, Pant R. Use of deep learning in modern recommendation system: a summary of recent works. International Journal of Computer Applications, 180(7),17-22(2017).
2. Zheng G, Zhang F, Zheng Z, et al. DRN: A deep reinforcement learning framework for news recommendation. The 2018 World Wide Web Conference, pp.167-176. ACM, New York (2018).
3. Plappert M, Houthooft R, Dhariwal P, et al. Parameter Space Noise for Exploration. 2017.
4. Fortunato M, Azar M G, Piot B, et al. Noisy Networks for Exploration. International Conference on Learning Representations, 2017.
5. Ricci F, Rokach L, Shapira B, et al. Recommender systems handbook. 2nd. Springer Press, Boston (2011).
6. Shani G, Heckerman D, Brafman R I. An MDP-based recommender system. Journal of Machine Learning Research 6(1),1265-1295 (2005).
7. Taghipour N, Kardan A. A hybrid web recommender system based on Q-learning. ACM Symposium on Applied Computing. pp. 1164-1168. ACM, New York (2008).
8. Choi S, Ha H, Hwang U, et al. Reinforcement learning based recommender system using bi-clustering technique. 2018. arXiv:1801.05532.
9. Hu B, Shi C, Liu J.Playlist Recommendation Based on Reinforcement Learning. International Conference on Intelligence Science. pp. 172-182. Springer, Cham (2017).
10. Liebman E,Saar-tsechansk M,Stone P.DJ-MC: A reinforcement-learning agent for music playlist recommendation. In Proceedings of the 14th Int'l Conference on Autonomous Agents and Multiagent Systems, pp. 591-599. IEEE Press, New York (2015).
11. Munemasa I, Tomomatsu Y, Hayashi K, et al. Deep reinforcement learning for recommender systems. International Conference on Information & Communications Technology, pp. 226-233. IEEE Press, New York (2018).
12. Zhou Z H. Machine learning. 1nd. Tsinghua University Press, Beijing (2016).
13. Mikolov T, Chen K, Corrado G, et al. Efficient Estimation of Word Representations in Vector Space. Computer Science,2013.
14. Zhu Y X, Lu L Y. Evaluation Metrics for Recommender Systems. Journal of University of Electronic Science and Technology of China, 41(2), 163-175 (2012).
15. Barkan O, Koenigstein N. Item2Vec: Neural item embedding for collaborative filtering. IEEE 26th International Workshop on Machine Learning for Signal Processing, pp. 1-6. IEEE Press, New York (2016).

Stock Portfolio Management with Deep Reinforcement Learning Methods

Xing Wu[1,2][0000−0001−5331−022X], Ping Yang[1][0000−0002−5748−2595], Haolei Chen[1][0000−0002−0273−8063], Mingyu Zhong[1], Jianjia Wang[1,2][0000−0003−1983−1632], Bo Huang[3], and Hamido Fujita[4,5,6,⋆][0000−0001−5256−210X]

[1] School of Computer Engineering and Science, Shanghai University, Shanghai, China
[2] Shanghai Institute for Advanced Communication and Data Science, Shanghai University, Shanghai, China
[3] School of Electronic and Electrical Engineering, Shanghai University of Engineering Science, Shanghai, China
[4] Faculty of Information Technology,Ho Chi Minh City University of Technology (HUTECH),Ho Chi Minh City, Vietnam
[5] Andalusian Research Institute on Data Science and Computational Intelligence (DaSCI),University of Granada, Spain
[6] Faculty of Software and Information Science,Iwate Prefectural University (IPU),Iwate, Japan

Abstract. Portfolio management has been a popular domain in financial industry which involves large datasets with increasing complexity and dynamical property. In order to solve the challenge, the deep reinforcement learning framework is presented to provide a solution to the portfolio management problem. The proposed methods are called as CLPG (Convolutional-LSTM Policy Gradient Agent) and CLDPG (Convolutional-LSTM Deterministic Policy Gradient Agent). To verify the robustness and effectiveness of CLPG and CLDPG , they are tested in American stock market. Experimental results show that CLPG is more desirable in financial market than CLDPG, although both of them are more advanced.

Keywords: portfolio management · Convolutional Neural Network · Long Short-Term Memory.

1 Introduction

Portfolio management is the asset management of various securities (shares, bonds and other securities) in order to meet specified investment goals for the benefit of the investors. Asset allocation is an important factor in determining returns for an investment portfolio. Asset allocation aims to balance risk and

⋆ Corresponding author at:Faculty of Information Technology, Ho Chi Minh City University of Technology (HUTECH), Ho Chi Minh City, Viet Nam.

reward by apportioning a portfolio's assets according to an individual's goals, risk tolerance and investment horizon.

Researches on financial markets have increased considerably in recent years. On the one hand, researchers build neural network models to predict the price trend of asset securities. On the other hand, researchers build intelligent agents through reinforcement learning to automate the trading of a single security. Largely inspired by the above, we attempt to build an agent for portfolio management by utilizing deep learning and reinforcement learning. The deep learning part is applied to extract hierarchical financial features. Then, the reinforcement learning part interacts with the deep learning part and makes decision about asset allocation to acquire the best ultimate rewards in an unknown environment.

2 RELATED WORKS

Machine learning algorithms are used in addressing financial problems in some work. Hamido Fujita et al.[16] used adaboost SVM to succeed in predicting financial distress. And his team[17] proposed a new vision on financial class-imbalanced problem.

In recent years, deep learning and reinforcement learning have achieved tremendous development. Reinforcement learning has been successfully applied in many tasks, such as unmanned driving [20, 19], job scheduling [12, 14], and game playing [11, 6]. In some areas, reinforcement learning has even surpassed human experts. Besides, the reinforcement learning also is successfully applied to financial market. Deng et al. [2] utilize reinforcement learning to build a quantitative trading system. But this system is only suitable for a single asset. To improve financial trading decisions, deep q-learning [7] is employed to predicte the number of shares.

In addition, deep learning has achieved remarkable success in many fields, such as image processing [10, 15], speech recognition [1] and natural language processing [5, 4]. At the same time, deep learning has been proven to be a powerful tool for effectively extracting market characteristics [9]. Meanwhile, deep learning has also been successfully applied to financial markets. Troiano et al. [18] train a robot with long short-term memory machine which can replicate the underlying logic of a strategy. In these works [13, 3, 8], deep learning is mainly used to predict the trend of stock prices to help trading.

3 Problem formulation

The essence of portfolio management is that the agent constantly interacts with the environment and learns the optimal strategy to reallocate assets.

In our experiment, the portfolio consists of $m + 1$ assets, including m stocks and one risk-free asset(money). The price vector v_t consists of the close price of all assets at period t. For example, the ith element of vector $v_{i,t}$ represents the close price of the ith asset in the tth period. For the continuity of markets, the elements of price vector v_t are the close price of assets for period t as well as

the open price of assets for period $t+1$. The price fluctuating vector for the tth period is defined as y_t:

$$y_t = \frac{v_t}{v_{t-1}} = (1, \frac{v_{1,t}}{v_{1,t-1}}, \frac{v_{2,t}}{v_{2,t-1}}, ..., \frac{v_{m,t}}{v_{m,t-1}}) \tag{1}$$

The first element of y_t is 1,because the cash is without depreciation. The price fluctuating vector can indicate the change in total assets after each transaction. In portfolio management, the state is a 3-dimensional vector with the shape (t, m, f), where t is the number of time steps, m is the number of assets and f is the number of features. Features for asset i on period t are its close, high, open and low prices in the interval. There are $c_{i,t}$, $h_{i,t}$, $o_{i,t}$,$l_{i,t}$.

In the period t, action can be expressed as:

$$a_t = (a_{0,t}, a_{1,t}, ..., a_{m,t}) \tag{2}$$

Where m is the number of assets. The sum of the assigned weights should be 1.

4 Depiction of stock agent model

Reinforcement learning is a learning method, by which the agent interacts with the environment with less prior information and learns from the environment by trail-and-error while refining its strategy at the same time. In our experiment, RL transaction models are built on actor-based method and actor-critic method respectively.

4.1 Convolutional-LSTM Policy Gradient

The construction of the first trading strategy is based on actor-based algorithms of RL(Policy Gradient). A policy is a mapping from the state space to the action space:

$$\pi : S \to A \tag{3}$$

The deep policy gradient combines CNN with LSTM. First, a convolutional layer is utilized to extract the structural features of input vectors. The convolutional kernel shape is (1,2). The number of filter size is 2 and the number of convolutional layers is 2. The next layer is a fully-connected layer. To capture time dependency across the stock, the LSTM network is connected to the fully connected network. Finally, the last layer is $softmax$ which outputs the weight of each asset.

The parameters are continuously updated along the gradient direction with a learning rate α.

$$\theta < -\theta + \alpha \bigtriangledown \pi_\theta(s_t, a_t) \tag{4}$$

4.2 Convolutional-LSTM Deterministic Policy Gradient Agent

In the previous section, the RL model of CLPG can be generally categorized as actor-based (policy function) methods. In this section, a new trading agent will be introduced, and the RL part of this strategy is based on actor-critic methods. The actor part selects the trading action according to the state of the market environment.

The μ is $softmax$ function which outputs the weight of each asset allocation. The critic part will rate the trading action. Specifically, a Q-value function gives expected accumulated reward when executing action a in state s which is:

$$value = Q(s_t, a_t|\theta^{\mu}) \tag{5}$$

The s_t is the stock market state of the current time t. The a_t is the trading action at the time step t. The actor function and the critic function are represented by neural network.

The current strategy consists of an actor network and a critic network. The actor is used to update the trading strategy while the critic is used to score the action of each transaction. Based on the DQN algorithm, a network which has the same structure but different parameters is created for each of the existing actor network and the critic network. Like DQN, one is called a policy network and the other is called a target network.

5 EXPERIMENT

In this section, the above model will be tested on real-world financial data. In order to verify the robustness of the two trading agents, stock experimental data is selected from American market. The asset portfolio is constructed of six stocks. The selected stocks are listed in the table below.

Table 1. STOCKS IN THE AMERICAN STOCK MARKET

Symbol	Company	Symbol	Company
ADBE	Adobe Inc	GE	General Electric
AXP	American Express	IBM	International Business
NOK	Nokia Corporation	QCOM	Qualcomm Incorporated

In addition, the portfolio value of these algorithms is compared with the basic strategy(buy-and-hold). The portfolio value curve of these strategies are shown in the following figure.

6 DISCUSSION

From the experimental results, observations can be found as follows. In American stock market, the CLPG agent and the CLDPG agent have achieved good profitability.

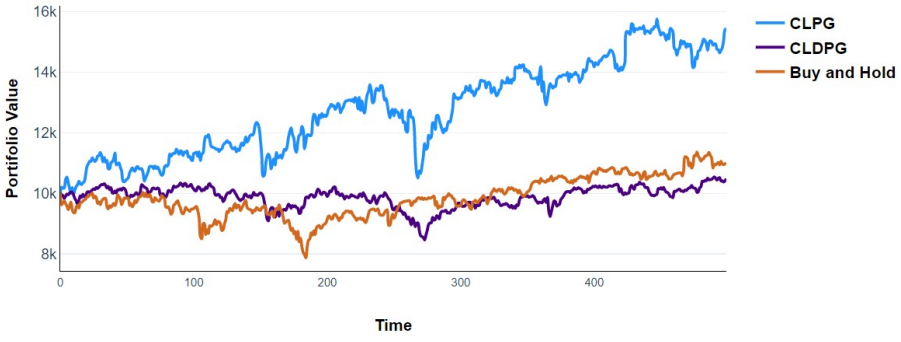

Fig. 1. Comparison of portfolio value in American stock market.

From fig. 1, in the US stock market, all agents increase the portfolio value of assets at the end of the period. Furthermore, the portfolio under the management of the CLPG agent increased by 54% at the end of the period. The portfolio under buy and hold strategy only grew by 9%. Meanwhile, the Sharpe ratio of CLPG agent is also the highest. The CLDPG agent receives the same benefits with minimal risk.

7 CONCLUSION AND FUTURE WORK

In order to profit from financial portfolio management, deep reinforcement learning algorithms are applied to asset allocation. The deep learning is utilized to summarize multi-stock features. To better illustrate the multi-stock features, the convolutional neural networks are used to extract structural information between stocks and the long short-term memory algorithms are used to extract sequence information from stock data. Meanwhile, the reinforcement learning outputs the weight of each asset through deep features. Multiple experiments are carried out to verify the effectiveness of the proposed methods. According to the experimental results, The CLPG agent outperforms other algorithms. A complete portfolio problem involves the stock selection and how to allocate the proportions of each asset. Our algorithms are applied to asset allocation. Hence, it is the stock selection that will be our future research subject.

Acknowledgements

This work is supported by the National Key R&D Program of China under Grant 2019YFE0190500, by the State Key Program of National Nature Science Foundation of China (Grant No. 61936001).

References

1. Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Cheng, Q., Chen, G., et al.: Deep speech 2: End-to-end speech recognition in english and mandarin. In: International conference on machine learning. pp. 173–182 (2016)
2. Deng, Y., Bao, F., Kong, Y., Ren, Z., Dai, Q.: Deep direct reinforcement learning for financial signal representation and trading. IEEE transactions on neural networks and learning systems **28**(3), 653–664 (2016)
3. Di Persio, L., Honchar, O.: Artificial neural networks architectures for stock price prediction: Comparisons and applications. International journal of circuits, systems and signal processing **10**, 403–413 (2016)
4. Goldberg, Y.: A primer on neural network models for natural language processing. Journal of Artificial Intelligence Research **57**, 345–420 (2016)
5. Goldberg, Y.: Neural network methods for natural language processing. Synthesis Lectures on Human Language Technologies **10**(1), 1–309 (2017)
6. Guo, X., Singh, S., Lee, H., Lewis, R.L., Wang, X.: Deep learning for real-time atari game play using offline monte-carlo tree search planning. In: Advances in neural information processing systems. pp. 3338–3346 (2014)
7. Jeong, G., Kim, H.Y.: Improving financial trading decisions using deep q-learning: Predicting the number of shares, action strategies, and transfer learning. Expert Systems with Applications **117**, 125–138 (2019)
8. Jia, H.: Investigation into the effectiveness of long short term memory networks for stock price prediction. arXiv preprint arXiv:1603.07893 (2016)
9. Li, Y., Ni, P., Chang, V.: Application of deep reinforcement learning in stock trading strategies and stock forecasting. Computing pp. 1–18 (2019)
10. Litjens, G., Kooi, T., Bejnordi, B.E., Setio, A.A.A., Ciompi, F., Ghafoorian, M., Van Der Laak, J.A., Van Ginneken, B., Sánchez, C.I.: A survey on deep learning in medical image analysis. Medical image analysis **42**, 60–88 (2017)
11. Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M.: Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602 (2013)
12. Moradi, M.: A centralized reinforcement learning method for multi-agent job scheduling in grid. In: 2016 6th International Conference on Computer and Knowledge Engineering (ICCKE). pp. 171–176. IEEE (2016)
13. Nelson, D.M., Pereira, A.C., de Oliveira, R.A.: Stock market's price movement prediction with lstm neural networks. In: 2017 International Joint Conference on Neural Networks (IJCNN). pp. 1419–1426. IEEE (2017)
14. Shahrabi, J., Adibi, M.A., Mahootchi, M.: A reinforcement learning approach to parameter estimation in dynamic job shop scheduling. Computers & Industrial Engineering **110**, 75–82 (2017)
15. Shen, D., Wu, G., Suk, H.I.: Deep learning in medical image analysis. Annual review of biomedical engineering **19**, 221–248 (2017)
16. Sun, J., Fujita, H., Chen, P., Li, H.: Dynamic financial distress prediction with concept drift based on time weighting combined with adaboost support vector machine ensemble. Knowledge-Based Systems **120**, 4 – 14 (2017)
17. Sun, J., Li, H., Fujita, H., Fu, B., Ai, W.: Class-imbalanced dynamic financial distress prediction based on adaboost-svm ensemble combined with smote and time weighting. Information Fusion **54**, 128 – 144 (2020)

18. Troiano, L., Villa, E.M., Loia, V.: Replicating a trading strategy by means of lstm for financial industry applications. IEEE transactions on industrial informatics **14**(7), 3226–3234 (2018)
19. Xu, X., Lian, C., Wang, J., He, H., Hu, D.: Actor–critic reinforcement learning for autonomous control of unmanned ground vehicles. Science Robotics **354**, 42–47 (2016)
20. Zhang, T., Li, Q., Zhang, C.s., Liang, H.w., Li, P., Wang, T.m., Li, S., Zhu, Y.l., Wu, C.: Current trends in the development of intelligent unmanned autonomous systems. Frontiers of information technology & electronic engineering **18**(1), 68–85 (2017)

The Time Efficient Centralisation Measure for Social Networks Assessment

Rafał Palak[1][0000−0002−4632−7709] and Krystian Wojtkiewicz[1][0000−0002−7851−4330]

Faculty of Computer Science and Management,
Wrocław University of Science and Technology,
Wybrzeże Stanisława Wyspiańskiego 27, 50-370 Wrocław, Poland
{rafal.palak, krystian.wojtkiewicz}@pwr.edu.pl

Abstract. The paper presents a new centrality measure. The presented measure was compared with state of the art measures and proved to combines two features, low computational complexity and high precision of calculations. The measure was mathematical analyzed, then experiments on real-life graphs were conducted. The mathematical analysis showed many essential properties of the proposed measure. Statistical analysis of the conducted experiment showed that measure has precision the same as state of the art measures with much lower computational complexity.

Keywords: centralisation, collective assessment, social networks, collective measures

1 Introduction

The scope of this paper covers the emerging research area of collective intelligence [19, 18] with a particular interest in assessing how the properties of the crowd affect its performance. There are many definitions of collective intelligence, e.g. 'the capacity of human collectives to engage in intellectual cooperation in order to create, innovate and invent' [14] or 'groups of individuals acting collectively in ways that seem intelligent' [16]. There are two common critical aspects of both definitions. First of them collaborative action of a group that we name collective, while another is a shared objective. In this paper, we want to focus on the group aspect of the collective. Authors in [12] used graph theory to describe the complexity of collective structure, as well as its properties. One of them is centralisation, which, however, has not been deeply analysed in [12].

The centrality is a notion introduced by Bavelas in 1948 [2] as one of the main factors influencing group efficiency [13, 3]. The importance of centrality has grown over the years in a wide range of problems, e.g. to analyse the performance of companies, to asses efficiency of information propagation or as well as a factor affecting collective prediction. According to [24, 23, 22], highly centralised collectives tend to be less effective than a decentralised one. At the same time, authors in [15] show that centralisation of companies positively relates to their

performance, as well as centralised child welfare systems exhibit greater success at achieving desired outcomes [7].

The main contribution of this paper is an introduction of a centralisation measure for the collective model presented in [12]. Authors' solution characterises in low computational complexity and similar accuracy as the state of the art measures. Authors define the measure in this paper, providing a discussion of its properties in comparison to other known approaches. Last but not least, experimental results allow a better understanding of the measure mechanics.

The remaining of the paper starts with the related work that presents the state of the art in the field of graph theory and centralisation measures. In the next section, authors formally define the collectives' structure and the definition of the proposed measure with its properties. Section 4 presents the experimental results. The last section concludes the paper.

2 Related works

Social network analysis is becoming one of the most potent tools in such areas as sociology, psychology, and political sciences. Researchers use techniques adopted from graph theory [10] to analyze the structure of relationships among social actors, i.e. individuals or groups. In most scenarios, the nodes of the graph represent actors, while the edges typically relate to relationships identified among them. Although the graph theory is a relatively new research field in mathematics, it already introduces many methods and tools particularly useful for exploring structural properties of networks [5, 6, 1], thus making it a perfect tool for social networks analysis. There is numerous research that shows how the assessment of graph structure properties, can provide vital insights on, e.g. decision-making process in the group [4], sports team performance [20] or trust evaluation [11].

One of the properties of the graphs becomes particularly interesting for research, namely, the scope of centralisation. Many practical collective intelligence applications prove the enormous role of decentralisation in obtaining highly rated collective wisdom. In 2005 Surowiecki distinguished decentralisation as one of the crucial features of the wise crowd stating that "Decentralization's great strength is that it encourages independence and specialization on the one hand while still allowing people to coordinate their activities and solve difficult problems on the other" [24]. Another example could be the decentralised quality control mechanism of Wikipedia, that help to identify and to correct many quality threats [23]. The research in this area do not concentrate only on the virtual world but also considers crowd behaviour in real-life scenarios. Natural disasters like tsunamis or hurricanes, force people to make decisions under high-stress conditions. The decision centres are naturally decentralised; nevertheless, the overall outcome of the collective is highly efficient [22].

Sabidussi in 1966 [21] proposed one of the first and quite intuitive centralisation measures:

$$C(p_k) = \sum_{i=1}^{n} d(p_i, p_k) \tag{1}$$

where,

p_k is the node for which centralisation is calculated,
p_i is the i-th node in graph,
$d(p_i, p_k)$ is the number of edges in the shortest path connecting p_i and p_k.

This measure is quite informative for connected graphs, where we can always find the shortest path. However, for unconnected ones, finding the shortest path for every node in a graph is impossible. Considering the class of problems covered by the studies on collectives, we cannot rely on the measure that may or may not produce results. A separate issue is a fact that results obtained from this measure are not normalised, thus does not allow comparison between different graphs.

In 1974 [17], Nieminen proposed another quite simple measure to assess graph centralisation:

$$C(p_k) = \sum_{i=1}^{n} a(p_i, p_k) \tag{2}$$

where,

p_k is the node for which centralisation is calculated,
p_i is the i-th node in graph,
$a(p_i, p_k)$ is the function evaluating the connection of edges defined as

$$a(p_i, p_k) = \begin{cases} 1, & \text{if } p_i \text{ and } p_k \text{ are connected,} \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

The simplicity is the highest value of this measure since it allows rather fast calculation for big graph, i.e. having billions of edges and nodes. It is also free of usage restriction of the previous measure. However, the downside of such a simple and straightforward approach is its accuracy and interpretability.

Authors in [9] propose a more sophisticated measure of centralisation that takes under consideration the weights associated with edges that represent the communication channel bandwidth. This formula is defined as:

$$C(x_i) = \frac{\sum^{n} \sum_{j<k}^{n} m_{jk}(x_i)}{\sum_{j<k}^{n} \sum^{n} m_{jk}} \tag{4}$$

where,

m_{jk} is maximum flow[1] between x_j to x_k, that is calculated based on edges weights,
$m_{jk}(x_i)$ is maximum flow from x_j to x_k that passes through point x_i.

For graph where no values are assigned to the edges, we can use following formula

$$C(x_i) = \frac{x \sum^{n} \sum_{j<k}^{n} b_{jk}(x_i)}{n^2 - 3n + 2} \tag{5}$$

[1] Maximum flow should be understood as the maximum amount of flow that the network would allow to flow from source (x_j) to sink (x_k) through any feasible path.

with $b_{jk}(x_i)$ calculated as

$$b_{jk}(x_i) = \frac{g_{jk}(x_i)}{g_{jk}} \tag{6}$$

where,

g_{jk}	is the total number of paths that connects x_j and x_k,
$g_{jk}(x_i)$	is the total number of paths that connects x_j and x_k that contain x_i, and $i \neq j \neq k$
n	is the number of nodes in the graph.

Centralisation of the graph as a whole should be calculated using:

$$C_T = \frac{\sum_{i=1}^{n}[C(p^*) - C^{'}(p_i)]}{n-1} \tag{7}$$

where,

$C(p^*)$	is normalised centrality of the most central point in the graph.

Measures 4 and 5 use the same approach, which proves accurate in measuring graph centralisation. However, both measures are very complicated and have a heavy computational load. The greediness of the methods originates in a requirement to search all possible shortest paths between each pair of nodes. Algorithms that perform that task, unfortunately, need to traverse the graph multiple times. Due to high complexity, these measures are hard to use for large structures like those generated based on social networks.

The measures mentioned above provide different approaches towards assessing the centrality of graphs. They have different properties, requirements and shortcomings that make that hard to compare systematically. Without any doubt, the possibility to evaluate various properties of graphs is vital for research on collective intelligence. In their previous work [12], authors proposed the formal definition of the collective, along with the taxonomy of graph measure types that describe its properties. Based on that, in this paper, the new centrality measure will be introduced that aims in low computational complexity and high accuracy.

3 Centralization measure for social networks

3.1 Prerequirements

The measure is a system that we use to assign a value to an object property. The design of such a system is a complex task, and we will begin by providing several definitions. Let's start with the graph defined as:

$$G = (V, E) \tag{8}$$

where:

V is a set of nodes (vertices). Each vertex represents one person in a social network or a collective.
E is a set of edges. Each edge represents a direct connection between two members of a social network or a collective.

There is a vast amount of simple measures already defined within the scope of graph theory. Let's consider those that characterise the structure of V and E.

$$n = |V| \tag{9}$$

where,

n is a cardinality of V understood as a number of its elements.

$$S = s_1, s_2, ..., s_n \tag{10}$$

where,

S is a set where each element s_i represents a number of edges connected to node v_i.

Following, S_{max} and S_{min} are accordingly maximum and minimum values within the set S, thus the maximum and the minimum number of edges connected to any $v \in V$ of G. Similarly, we can define S_{avg} as

$$S_{avg} = \frac{\sum_{i=1}^{n} s_i}{n} \tag{11}$$

It is an average number of edges connected to all nodes in the set V, calculated as average from S.

The measures mentioned above are intuitive, thus easy to comprehend and understand. This paper aims to introduce the new standard for graph centrality assessment, which has to respond to two main issues. First of them is the proper identification of phenomena that it is supposed to characterise. For this, we have identified following constraints for proposed measure C:

$C = 0$ for G_1 where $|E| = 0$ *(graph has no edges)*,
$C = 0$ for G_2 where $|E| = n^2 - n$ *(graph is complete)*,
$C = 1$ for G_3 where $|E| = 2n - 2$ and $S_{max} = n - 1$ and $S_{min} = 1$ *(graph has a star shape)*,
$C_4 = C_5$ for G_4 and G_5, if graphs G_4 and G_5 are isomorphic.

Additionally, if we consider any change to graph in terms of its structure, the measures' value should reflect this change.

The second identified issue relates to the properties of the measure as a mathematical construct, i.e. we need to ensure that it is:

objective does not rely on subjective assessment of properties,
repeatable produces same value for the same input,
interpretable value of measure can be interpreted as is, without the need of knowing any other properties of the object.

3.2 Formal definition of centralisation measure

Taking into consideration requirements from the previous subsection, authors have decided to construct the measure using basic properties describing graphs, namely:

S_{max}	is the maximum number of edges connected to any of nodes in G,
S_{avg}	is the average number of edges connected to nodes in G,
n	is the number of nodes in G.

For the computation of the measure, we use the formula 12.

$$\frac{S_{max} - S_{avg}}{S_{max}} * \frac{1}{C_{max}} \tag{12}$$

where,

C_{max} is the maximum possible centralisation factor computed according to following formula

$$\frac{n-2}{n} \tag{13}$$

In the next step, we normalise the measure using the graph filling factor 14, that scale the output of the measure to values within $< 0, 1 >$.

$$\frac{S_{max}}{n-1} \tag{14}$$

Finally, the proposed centralisation measure follows the definition 1.

Definition 1. *Centralisation measure*

$$C = \begin{cases} n > 2 & \frac{n(S_{max} - S_{avg})}{(n-1)(n-2)} \\ otherwise & 0 \end{cases} \tag{15}$$

3.3 Theoretical proof of measure properties

In this section, the authors will provide a discussion on how the proposed measure fulfils the pre-requirements defined in subsection 3.1.

Question 1. $C = 0$ for G_1 where $|E| = 0$

Solution 1. Let's check if, for any graph that has an empty set of nodes, the measure C takes the value of 0.

$$\begin{gathered} n = 0 \Rightarrow S_{avg} = 0 \wedge S_{max} = 0 \\ C = \frac{n(0-0)}{(n-1)(n-2)} = \frac{0}{(n-1)(n-2)} = 0 \end{gathered} \tag{16}$$

Question 2. $C = 0$ for G_2 where $|E| = n^2 - n$

Solution 2. Let's check if for any graph that is complete, i.e., each node connects with all other nodes, the measure C takes the value of 0.

$$\begin{gathered} n \neq 0 \wedge |E| = n^2 - n \Rightarrow S_{avg} = n - 1 \wedge S_{max} = n - 1 \\ C = \frac{n((n-1)-(n-1))}{(n-1)(n-2)} = \frac{0}{(n-1)(n-2)} = 0 \end{gathered} \tag{17}$$

Question 3. $C = 1$ for G_3 where $|E| = 2n-2$ and $max(S) = n-1$ and $min(S) = 1$

Solution 3. Let's check if for any graph that has star shape, i.e., one node a connects with all other nodes in graph g, while all other nodes connect only with node a, the measure C takes the value of 1 (maximum for the measure).

$$\begin{gathered} n \neq 0 \Rightarrow S_{avg} = \frac{n-1+n-1}{n} = \frac{2n-2}{n} \wedge S_{max} = n - 1 \\ C = \frac{n(n-1-\frac{2n-2}{n})}{(n-1)(n-2)} = \frac{n(\frac{n(n-1)}{n} - \frac{2(n-1)}{n})}{(n-1)(n-2)} = \frac{(n-1)(n-2)}{(n-1)(n-2)} = 1 \end{gathered} \tag{18}$$

Question 4. $C_4 = C_5$ for G_4 and G_5 where both graphs are isomorphic graphs.

Solution 4. We consider two graphs to be isomorphic when they have the number of nodes connected in the same manner. According to that, the simple answer would be that they have the same value of centralisation measure. However, to make it even more clear let's consider following:

$$\begin{gathered} G_4, G_5 \text{ are isomorphic} \Rightarrow S_{avg_4} = S_{avg_5} \wedge S_{max_4} = S_{max_5} \\ C_4 = \frac{S_{max_4} - S_{avg_4}}{(n-1)(n-2)} = \frac{S_{max_5} - S_{avg_5}}{(n-1)(n-2)} = C_5 \end{gathered} \tag{19}$$

The proposed measure satisfies all pre-requirements in the theoretical discussion. However, further analysis of C, including experimental considerations, will be conducted in the next section.

4 Experiment

Table 1: Comparison of different centralisation measures for the 5 nodes graph class structures

No.	Graph	Measure 1 [s]	Measure 2 [s]	**Measure 3** [s]
1		0.0000	0.0000	**0.0000**
2		0.0000	0.1875	**0.2500**

3	0.1667	0.3750	**0.5000**
4	0.0000	0.0625	**0.0833**
5	0.5000	0.5625	**0.7500**
6	0.1667	0.2500	**0.3333**
7	0.0000	0.2500	**0.3333**
8	0.2500	0.2500	**0.3333**
9	1.0000	0.7500	**1.0000**
10	0.7083	0.4375	**0.5833**
11	0.4167	0.1250	**0.1667**
12	0.3333	0.4375	**0.5833**
13	0.0000	0.1250	**0.1667**
14	0.0208	0.1250	**0.1667**
15	0.8333	0.6250	**0.8333**

16	0.3750	0.3125	**0.4167**
17	0.5417	0.3125	**0.5000**
18	0.4792	0.3125	**0.4167**
19	0.0625	0.3125	**0.4167**
20	0.0000	0.0000	**0.0000**
21	0.6667	0.5000	**0.6667**
22	0.1463	0.1875	**0.2500**
23	0.5625	0.5000	**0.6667**
24	0.1458	0.1875	**0.2500**
25	0.4167	0.1875	**0.2500**
26	0.0000	0.1875	**0.2500**
27	0.2917	0.3750	**0.5000**
28	0.1875	0.3750	**0.5000**

29	0.0838	0.0625	**0.0833**
30	0.5000	0.3750	**0.5000**
31	0.0550	0.2500	**0.3333**
32	0.1250	0.2500	**0.3333**
33	0.0275	0.1250	**0.1667**
34	0.0000	0.0000	**0.0000**

In this section, the authors focus on the comparison of the proposed measure with other states of the art solutions. We use two approaches toward centralisation assessment, namely measure 5 and measure 2 from section 2. The comparison focuses on the family of 5 nodes graphs, that include all graphs having 5 nodes but a varying number of edges and their alignment. This approach relates to the study of Freeman [8], thus allows a proper quality of comparison. Table 1 presents the computed values for all three centralisation measures for all identified 5 nodes graphs.

The obtained results indicate high similarity between measure 1 and measure 3. However, there are a few cases in which results differ. Let's discuss them in detail.

graph 3 measure's 3 value of 0.5 suggests a little bit centralised graph, while measure's 1 value of 0,1667 indicates it to be decentralised. The proposed measure considers the node that has a connection with two other nodes as a central point. If we take into consideration that no other link is present in the graph, such a result is appropriate.

graph 11 measure 1 has a higher value than the proposed measure. In this graph, all of the nodes have at least one edge with three nodes having two edges. One of the nodes might be a central point. However, its centrality is rather faint; thus, the result of the 3rd measure is acceptable.

graph 19 There is a difference between results of measure 1 and 3 in favour of the later. While the careful analysis of the graph one can find that 1

node has no edges attached, 2 nodes have two edges each, and the last two have three edges attached. Even though there is no single node that might be the centre of the graph, we cannot assume that this particular graph is almost uncentralised as suggested by the value of measure 1. The value of the proposed measure seems to characterise the centrality of this graph better.

graph 31

graph 32

graph 33 those graphs have a high number of edges, making them very close to the complete graph. Values of measure 1 indicate that those graphs are highly decentralised, while measure‘s 3 results are not so strict. However, in the author‘s opinion, the results of proposed measures might be closer to intuition. The direct interpretation for both measures is the same, but they only differ in the degree.

Table 1 contains also results for measure 2, but they are hard to interpret. Most of the values are rather small, with just a few going over 0.5. Graph 9, the most centralized one, obtained the highest value for the measure equal to 0.75. That sets the highest possible value for the graph. For all other in the experiment, the values show little variability. However, in most cases, they fall between values returned by other measures.

Table 2: Comparison of computation time for 3 measures based on graphs built based on ResearchGate dataset

No.	Nodes	Edges	Measure 1 [s]	Measure 2 [s]	**Measure 3** [s]
1	19	50	2.110000	0.000011	**0.000010**
2	21	79	3.680000	0.000005	**0.000006**
3	17	17	9.820000	0.000005	**0.000004**
4	20	73	2.320000	0.000005	**0.000005**
5	20	22	1.700000	0.000007	**0.000005**
6	20	33	1.670000	0.000005	**0.000005**
7	25	31	4.060000	0.000006	**0.000006**
8	30	87	10.230000	0.000007	**0.000008**
9	18	26	1.410000	0.000004	**0.000005**
10	183	508	39129.800000	0.000049	**0.000046**
11	37	65	22.410000	0.000009	**0.000009**
12	20	19	1.590000	0.000009	**0.000010**
13	243	619	166302.550000	0.000067	**0.000066**
14	19	20	1.390000	0.000005	**0.000005**
15	31	91	11.660000	0.000008	**0.000008**
16	41	131	43.180000	0.000010	**0.000010**
17	21	20	2.020000	0.000005	**0.000005**
18	19	45	1.770000	0.000005	**0.000005**
19	47	116	86.930000	0.000012	**0.000011**
20	116	389	7756.450000	0.000029	**0.000028**
21	20	41	2.460000	0.000005	**0.000005**

22	20	24	1.730000	0.000005	**0.000005**
23	43	149	54.380000	0.000010	**0.000011**
24	26	66	4.940000	0.000006	**0.000006**
25	38	121	30.060000	0.000009	**0.000010**
26	17	26	8.450000	0.000004	**0.000005**
27	25	26	3.740000	0.000006	**0.000007**
28	19	65	1.540000	0.000005	**0.000005**
29	18	21	1.190000	0.000004	**0.000004**
30	17	46	1.280000	0.000004	**0.000005**
31	20	55	1.990000	0.000005	**0.000005**

The second part of the experiment focuses on the computational cost of algorithms for each measure. We have built a dataset with data on followers and following of 4055 researchers present in Researchgate. Since all measures do not use the direction of a link as a crucial element, we uniformed the set and removed the information regarding the connection direction. Later, we have prepared 31 different graphs. For each sample, we have randomly chosen one researcher as a starting node, and then we added all nodes (researchers) connected to this researcher. According to this method, most of the samples should be highly centralised. We stored the sample sets in the MongoDB database. For each measure, we have written algorithms in Java 8. We used a computer with 16 GB RAM, Intel i7 processor and Windows 8.1 operating system. The experiment results present table 2.

It is visible that the time needed to obtain results for measure 1 is significantly higher than that for other methods, that run almost in the comparable time. Due to high computational time, the calculations for the first method were conducted only once. Values for measure 2 and 3 are average execution time, calculated for 1000 runs. The results were statistically validated. Shapiro-Wilk with alpha= 0.05 tests rejected H0 about the normality of the populations‘ distribution. Therefore, in further analysis, the Mann–Whitney U test was used to compare samples. Results of U test with alpha 0.05 for the pair of measure 1 and 3 reject H0. Results of U test with alpha 0.05 for pair of measure 2 and 3 do not allow to reject H0 with p-value=0.927.

It is worth noting that measure 1 proved to be very accurate in its measurement, however it has a very high computational cost. Contrary, the second measure is computationally reasonable, but the results are less reliable. The proposed method for assessing graph centralisation share the best from both opposing measures, i.e., it is cost-efficient and accurate.

5 Summary

The paper presents a novel measure for graph centralisation assessment that can be used by collective models like the one defined in [12]. Apart from its

theoretical background, the authors presented its results in comparison with two other, state of the art measures. The experiment consisted of firstly, the family of 5 node graphs, secondly artificial set of graphs representing collectives of ResearchGate researchers. The proposed method proved to be time-efficient while producing accurate results. Authors plan to evaluate the method further using a wider variety of graphs. It will also be tested if the technique might be extended to other types graphs, e.g. directed graphs.

References

1. Barnes, J.A., Harary, F.: Graph theory in network analysis. Social Networks **5**, 235–244 (1983)
2. Bavelas, A.: A mathematical model for group structures. Applied anthropology **7**(3), 16–30 (1948)
3. Bavelas, A.: Communication patterns in task-oriented groups. The journal of the acoustical society of America **22**(6), 725–730 (1950)
4. Blythe, J., McGrath, C., Krackhardt, D.: The effect of graph layout on inference from social network data. In: International symposium on graph drawing. pp. 40–51. Springer (1995)
5. Bollobás, B.: Modern graph theory, vol. 184. Springer Science & Business Media (1998)
6. Dehmer, M.: Structural analysis of complex networks. Springer Science & Business Media (2010)
7. Elgin, D.J., Carter, D.P.: Administrative (de) centralization, performance equity, and outcome achievement in rural contexts: An empirical study of us child welfare systems. Governance **32**(1), 23–43 (2019)
8. Freeman, L.C.: Centrality in social networks conceptual clarification. Social networks **1**(3), 215–239 (1978)
9. Freeman, L.C., Borgatti, S.P., White, D.R.: Centrality in valued graphs: A measure of betweenness based on network flow. Social networks **13**(2), 141–154 (1991)
10. Harary, F.: Graph theory. addison. Reading, MA (1969)
11. Jiang, W., Wang, G., Bhuiyan, M.Z.A., Wu, J.: Understanding graph-based trust evaluation in online social networks: Methodologies and challenges. ACM Computing Surveys (CSUR) **49**(1), 1–35 (2016)
12. Jodłowiec, M., Krótkiewicz, M., Palak, R., Wojtkiewicz, K.: Graph-based crowd definition for assessing wise crowd measures. In: International Conference on Computational Collective Intelligence. pp. 66–78. Springer (2019)
13. Leavitt, H.J.: Some effects of certain communication patterns on group performance. The Journal of Abnormal and Social Psychology **46**(1), 38 (1951)
14. Lévy, P.: From social computing to reflexive collective intelligence: The ieml research program. Information Sciences **180**(1), 71–94 (2010)
15. Mahmood, I.P., Zhu, H., Zaheer, A.: Centralization of intragroup equity ties and performance of business group affiliates. Strategic Management Journal **38**(5), 1082–1100 (2017)
16. Malone, T.W., Bernstein, M.S.: Handbook of collective intelligence. MIT Press (2015)
17. Nieminen, J.: On the centrality in a graph. Scandinavian journal of psychology **15**(1), 332–336 (1974)

18. Palak, R., Nguyen, N.T.: Prediction markets as a vital part of collective intelligence. In: 2017 IEEE International Conference on INnovations in Intelligent SysTems and Applications (INISTA). pp. 137–142. IEEE (2017)
19. Palak, R., Nguyen, N.T.: Independency aspect as a vital feature of intelligent collectives. In: 2019 IEEE International Conference on Systems, Man and Cybernetics (SMC). pp. 82–87. IEEE (2019)
20. Ribeiro, J., Silva, P., Duarte, R., Davids, K., Garganta, J.: Team sports performance analysed through the lens of social network theory: implications for research and practice. Sports medicine **47**(9), 1689–1696 (2017)
21. Sabidussi, G.: The centrality index of a graph. Psychometrika **31**(4), 581–603 (1966)
22. Stephenson, W.D., Bonabeau, E.: Expecting the unexpected: The need for a networked terrorism and disaster response strategy. Homeland Security Affairs **3**(1), 1–9 (2007)
23. Stvilia, B., Twidale, M.B., Gasser, L., Smith, L.C.: Information quality discussions in wikipedia. In: Proceedings of the 2005 international conference on knowledge management. pp. 101–113. Citeseer (2005)
24. Surowiecki, J.: The wisdom of crowds. Anchor (2005)

Empirical Study of K-means Clustering on Usability Performance Metrics in MAR-learning

Lim Kok Cheng[1,3], Ali Selamat[2,3,4], Mohd Hazli Mohamed Zabil[1], Md Hafiz Selamat[3], Rose Alinda Alias[3], Fatimah Puteh[3], Farhan Mohamed[3] and Ondrej Krejcar[4]

[1] *Universiti Tenaga Nasional, Selangor, Malaysia*
[2]*Malaysia-Japan International Institute of Technology, UTM KL, Malaysia*
[3] *Faculty of Computing, Universiti Teknologi Malaysia, Johor Bahru, Malaysia*
[4]*Center for Basic and Applied Research, Faculty of Informatics and Management, University of Hradec Kralove, Rokitanskeho 62, 500 03 Hradec Kralove, Czech Republic*
kokcheng@uniten.edu.my

Abstract. This paper presents empirical findings on machine learning clustering results from a bigger project relating to several related works embarking on performing machine learning algorithms on usability data. In this paper, results from applying K-means clustering on usability performance data is presented. This paper will highlight several related works followed the methodology formulated in this research. After that, the results and discussion were presented showing the feasibility and accuracy of K-means clustering on usability performance data relating to Mobile Augmented Reality (MAR) learning application. This paper contributes in proposing a model confirming the feasibility of K-means algorithm in prioritizing usability issues in MAR-learning.

Keywords: Usability, Mobile Augmented Reality Learning, English Language Teaching, Hierarchical Agglomerative Clustering, Performance Metrics.

1 Introduction

Performance metrics in usability engineering has been a common neglected measure in human computer interaction related studies. Furthermore, usability analysis often times relies on conventional comparative studies utilizing only descriptive statistical analysis without the ability of measuring usability factors independently. Due to that particular reason, the application of machine learning algorithms, specifically the clustering methods on usability data has been in its infancy. While researchers like [1] analyses the fundamentals of clustering algorithms, just very few researchers like [2] implemented machine learning analysis on usability improvements. It has been an area of interest on why has usability and machine learning have not been studied together. This project

[1] Lim Kok Cheng, Department of Computer Science and Information Technology, Universiti Tenaga Nasional, Jalan IKRAM-UNITEN, 43000, Selangor, Malaysia; E-mail: kokcheng@uniten.edu.my

therefore has embark on a quest tackling this very issues. Several works has been done in this project coupling with an emerging technology of Mobile Augmented Reality learning (MAR-learning) on English language. The several works can be referred from the content of [3] prior to this paper. The content of this article is to explore the validity of applying K-means algorithm on usability data collected through performance and self-reported metrics [4]. The focus is to re-replicate non significantly different results on 2 different datasets that goes though similar methodological procedures. The effort in this research is to strengthen confirmation on the feasibility of applying K-means on usability performance (quantitative based on users ability) and self-reported data (users qualitative and subjective satisfaction)

2 Related Works

2.1 Usability Issues in Augmented Reality Research

A study done by [5] shows that in Augmented Reality Learning Environment (ARLE) systems, usability evaluation emphasizes in producing better ease of use, satisfaction, immersion, motivation and performance. Santos et al. did an extensive systematic reviews on 43 different ARLEs with the concluded findings [5]. Santos et. al. in [5] reported that common tools used among their findings consists of observation, interviews and expert reviews among others. The methods suggested by the findings in [5] agrees with several techniques and methods earlier coined by Albert and Tullis in [4]. According to the research done by [4], most usability studies uses more self-reported metrics as compared to performance metrics. As a continuation of works done by [5], this research has further conducted studies as reported in [3]. The findings shows that in MAR-learning more than 80% of surveyed works preferred self-reported metrics over performance metrics. Despite for the fact that, there has been many arguments on the lesser reliable techniques of self-reported metrics, many researchers still cling to the comfort usage of self-reported metrics due to acceptable conveniences [6]–[8]. Both usability metrics of performance and self-reported can be combined is usability studies [9]. However, performance metrics has long been underrated in usability measures due to its technicality in implementation, even though it is much reliable. Many arguments has taken place in this matter by many researchers like [2], [4], [5], [10], [11], however it has been a never ending debate. Since the work involving performance and self-reported data has been reported in [3], [9], [12], this paper will present results on performance data only without possible interference of self-reported data.

2.2 K-means Clustering

Several clustering algorithms are used across many research areas in recent times. While K-means are generally used, partitioning based algorithms work in different fashion to obtained data analysis [13], [14]. K-means is one of the simplest unsupervised machine learning algorithms used to partition data based on locations and distance between data points [15]. K-means algorithm is reported to have worked better for large

dataset, while hierarchical clustering works better for smaller datasets [7], [16]. While K-means has been used in many research domains, the feasibility of K-means to be performed on usability performance data is still within infancy. The one reference of such research has been conducted is based on our prior work presented in [3]. In [3] however, only HAC has been performed only on combination of usability performance and self-reported data, without comparing with the performance of K-means and clustering performance on respective datasets comparatively. While works in [9] shows comparative result of HAC and K-means on usability data, individual clustering performance of either method has yet to be explored on separate usability features based on the collection method of either performance or self-reported metrics.

3 Methodology

As far as the methodology is concern, the generation of methods and techniques for this paper is to support the results presented in earlier publication of the same research team in [3], [12]. The works in [17] has layout the preliminary studies of this research project as a pre-operational framework clarification of the MAR-learning application called "InterviewME". This is succeeded by the works done in [9], deciding on the suitable clustering algorithm, where Hierarchical Agglomerative Clustering (HAC) is chosen as compared to K-means Clustering (K-means) based on the size of the datasets and several primary variables. Then, real time datasets were collected in [3], where usability prioritization were carried out for analysis of performance data based on three Mobile Augmented Reality (MAR) interaction parameters namely, Object Tracking (OT), Selecting (S) and Navigating (N). In [3], two performance metrics were used (Time-on-tasks and Error registration), while a self-reported metrics was used (5 Likert Scale) to measure participants' satisfaction level. The research work presented in this paper is objectified to explore the results attained from [3] with a different clustering algorithm, mainly K-means. The proposed methodology (Figure 1) with 3 phases is the entire process where the finding of this paper is extracted from the research activities done in Phase 3 of the project. The initial methodology has been designed to achieve these 2 objectives for phase 3:

- To evaluate the performance of K-means algorithm on usability performance data obtained through MAR object tracking.
- To get insight on the quality of K-means algorithm in clustering usability performance data obtained through MAR learning application.

The work of phase 1 has actually been presented in [17] where a repetitive pilot study has been carried out validating the usability and interactivity of the testable application called InterviewME, an MAR-learning application helping Malaysian tertiary educationist to master interview skills in English language. The development of the application has been guided by suggestions and recommendation discussed in [17]. The works of Phase 2 has generally been presented in [3], [12]. In [12], 168 students with equal gender representation has been selected, and after profiling 102 students were selected for the evaluation protocols. The selected samples will then go through a usability experimentation by carrying out object tracking activity (adapted from Real World Annotation Interaction in [5]). Usability metrics used is the standard ISO 9241-11, incorporating effectiveness, efficiency and satisfaction. Effectiveness were measured through time-on-

tasks, effectiveness were measured through error(s) registration and satisfaction were measured through engagement time (subsequent play)[12]. All students were selected using similar profiling techniques and requirements presented in [3]. Similar criteria were also used namely, years of smartphone usage to equalize device familiarity, frequency in engaging in social media applications, hourly usage of smartphone per day and has taken a specific English language course in their institution. Similar Android operating device is given to all 102 students, eliminating any possibilities of device familiarity and operational biases.

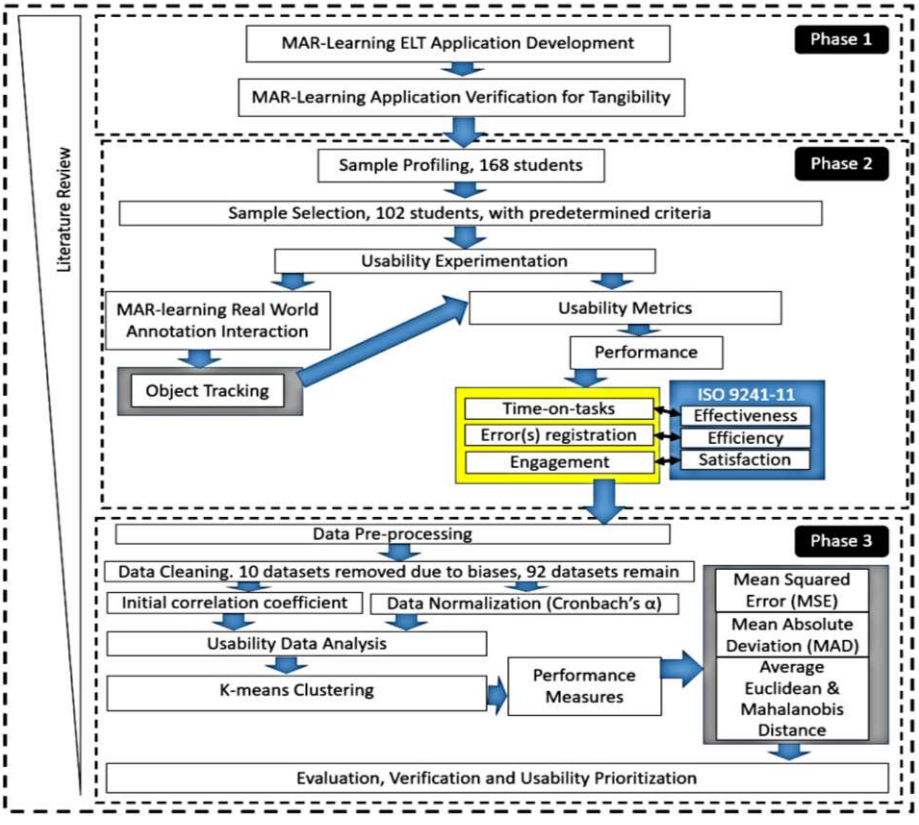

Fig. 1. Proposed methodology

The third phase involves data pre-processing. Since the initial correlation coefficient and data normalization [1] has been presented in [3], the focus of this paper is to compare only the performance measure of Correlation Coefficient (CC) [2]. Both formula of normalization and CC were explained in [1] and [2] respectively. Data cleaning were performed to prepare a low noise dataset where ambiguous rows and incomplete columns were eliminated. In this phase 10 set of results were removed due to anomalies not complying with expected criteria. Data normalization will then be performed using rescaling and evaluated through Cronbach's Alpha, where results were expected to be more than 0.70 (where reliabilities were deemed to be acceptable) [18], [19]. Correla-

tional values will also be measured through the usability metrics using correlation coefficient where results should be more than 0.30 (well-corelated) in behavioral sciences [2]. After the normalization process, Mean Squared Error (MSE), Mean Absolute Deviation (MAD) and Average Euclidean Distances (AED) (which will also be triangulated with Mahalanobis Distance Measure (MDM) [20]) were applied.

3.1 Hypothesis

Considering different discussions of how comparative results can be significantly different discussed by [1], [2], [21], this research believe that through the properly executed procedures from phase 1 to phase 3, these two hypotheses can be obtained.

- H1 – K-means clustering will achieve acceptable accuracy with low MSE, MAD, AED and MDM
- H2 – K-means clustering is proven feasible for usability data clustering by highlighting usability areas to be re-engineered.

4 Results and Discussion

Normalization is first performed on the original data set before the clustering process. With Cronbach's Alpha (α), before normalization, the α value is 0.640206 and after normalization the α value improved to 0.814953 showing better reliabilities between the 3 data arrays (Table 1). The α value for the original dataset also does not surpassed the benchmark suggested by [18] which is 0.70. After normalization, each combination of metrics were measured using correlation coefficient. All three metrics' data were more than 0.30 as proposed by [2]. For better reading comprehension, this paper will represent effectiveness metric with A, efficiency with B and engagement with C. The metric combination of AB scores a correlation coefficient value of 0.818583, AC scores a correlation coefficient value of 0.541752 and BC scores a correlation coefficient value of 0.451739. Even though the correlation value of BC is the lowest among the 3 combination, it is still above 0.30 which qualifies the combination to be analyzed through K-means clustering in the next processes. Next, the dataset combinations were clustered using Scikitlearn runnned by Python [22]. The K value of K-means needs to be determined before applying the Machine Learning techniques on them. The elbow method is applied here [15] to find the right amount of clusters for respective datasets. All usability metrics combination yields the value of optimal K to be 5, representing 5 clusters for each datasets respectively (Figure 4)

Table 1 summarizes the number of clusters and total population in each cluster within each combination. Cluster 1 shows the best performance while cluster 5 shows the worse performance in terms of metric combination.

Table 2 and table 3 respectively measure the MSE and MAD of each metric combination. For AB metric combination, the obtained MSE for vector x is 0.00827 and vector y is 0.00504, while the MAD for vector x is 0.07103 and vector y is 0.05512. While in AC metric combination, the obtained MSE for vector x is 0.08037 and vector y is 0.06025, while the MAD for vector x is 0.21219 and vector y is 0.15255. For BC metric combination, the obtained MSE for vector x is 0.01426 and vector y is 0.00949, while

the MAD for vector X is 0.09090 and vector y is 0.07954. Finally, for ABC metric combination, the obtained MSE for vector x is 0.01367, vector y is 0.00949 and vector z is 0.01598, while the MAD for vector X is 0.09385, vector y is 0.08400 and vector z is 0.10324. The results shown that the squared errors and absolute deviation from respective centroids vectors are generally low with all 4 K-means clustering indicating good performance accuracy in a nutshell. While the best performing clusters is the effectiveness/ efficiency metric combination (AB), higher errors were seen in the effectiveness and engagement metric combination (AC). Similar conclusion can be made from measuring the average Euclidean and Mahalanobis distances (Table 4), where the lowest (smaller distance means better clustering accuracy) average Euclidean and Mahalanobis scores were obtained by the effectiveness/ efficiency metric combination (AB), stating averages at 0.098179 and 0.096772 respectively. On the other hand the highest distances were obtained through the effectiveness and engagement metric combination (AC) scoring 0.288787 and 0.307562 respectively. The efficiency/ engagement (BC) scores 0.139947 and 0.139856 on both average distance measures, while the effectiveness, efficiency and engagement (ABC) metric combination scores 0.182280 and 0.139856 respectively. While AC scores the biggest distance measures, the overall distance measures were still acceptable in conforming the accuracy of the K-means clustering performance in this study. The results therefore accept the first hypothesis indicating the performance of K-means on usability performance data obtained through MAR learning application. On the other hand, referring back to Table 3, showing different percentages of clusters population, we can summarize that after the clustering process, data set from Group AC reveals the biggest population in the weakest usability cluster (Cluster 5), which stands at 25 samples (25.2%). Although the cluster differences are less significant as compared to Hierarchical Agglomerative Clustering reported in [3], [12], prioritization is still possible indicating that usability problems existed in the effectiveness and engagement metric combination, which can provide insights to design engineer to re-engineer the highlighted problems. With the ability of highlighting weak usability areas, the second hypothesis can be confirmed.

Table 1: Clusters distribution for usability metric combination

Datasets	**Number of Distribution/ Centroids Coordinates**				
	Cluster 1	**Cluster 2**	**Cluster 3**	**Cluster 4**	**Cluster 5**
Group AB	8 (8.6%)	28 (30.4%)	15 (16.3%)	21 (22.8%)	20 (21.7%)
Group AC	21 (22.8%)	12 (13.0%)	12 (13.0%)	22 (23.9%)	25 (27.2%)
Group BC	27 (29.3%)	12 (13.0%)	23 (25.0%)	19 (20.7%)	11 (12.0%)
Group ABC	10 (10.9%)	24 (26.1%)	21 (22.8%)	20 (21.7%)	17 (18.5%)

Table 2: Mean Squared Error for Metric Combinations

Metric Combination	**Mean Squared Error (MSE)**		
Vector	**x**	**y**	**z**
Effectiveness/ Efficiency (AB)	0.00827	0.00504	-
Effectiveness/ Engagement (AC)	0.08037	0.06025	-
Efficiency/ Engagement (BC)	0.01426	0.00949	-
Effectiveness/ Efficiency/ Engagement (ABC)	0.01367	0.01133	0.01598

Table 3: Mean Absolute Deviation for Metric Combinations

Metric Combination	Mean Absolute Deviation (MAD)		
Vector	x	y	z
Effectiveness/ Efficiency (AB)	0.07103	0.05512	-
Effectiveness/ Engagement (AC)	0.21219	0.15255	-
Efficiency/ Engagement (BC)	0.09090	0.07954	-
Effectiveness/ Efficiency/ Engagement (ABC)	0.09385	0.08400	0.10324

Table 4: Average Euclidean and Mahalanobis Distances

Metric Combination	Euclidean Distance Mean	Mahalabonis Distance Mean
Effectiveness/ Efficiency (AB)	0.098179	0.096772
Effectiveness/ Engagement (AC)	0.288787	0.307562
Efficiency/ Engagement (BC)	0.139947	0.139856
Effectiveness/ Efficiency/ Engagement (ABC)	0.182280	0.139856

5 Conclusion and Future Works

The results and discussion in the previous section re-affirms the implementation, feasibility and performance of K-means clustering technique on usability performance data obtained through MAR learning application. The presented outcome also shows that K-means is an acceptable algorithm to be implemented on usability data through empirical measures. Moving forward, this research will look into more clustering techniques to be performed on similar dataset as a comparison searching for the most and moderately suitable clustering techniques specifically used for usability performance data within the MAR learning domain.

References

[1] A. Bouguettaya, Q. Yu, X. Liu, X. Zhou, and A. Song, "Efficient agglomerative hierarchical clustering," *Expert Syst. Appl.*, vol. 42, no. 5, pp. 2785–2797, Apr. 2015, doi: 10.1016/j.eswa.2014.09.054.

[2] A. Oztekin, D. Delen, A. Turkyilmaz, and S. Zaim, "A machine learning-based usability evaluation method for eLearning systems," *Decis. Support Syst.*, vol. 56, pp. 63–73, Dec. 2013, doi: 10.1016/j.dss.2013.05.003.

[3] L. K. Cheng *et al.*, "Usability Prioritization Using Performance Metrics and Hierarchical Agglomerative Clustering in MAR-Learning Application.," in *SoMeT*, 2017, vol. 297, pp. 731–744, [Online]. Available: http://dblp.uni-trier.de/db/conf/somet/somet2017.html#ChengSZSAPMK17.

[4] W. Albert and T. Tullis, *Measuring the user experience: collecting, analyzing, and presenting usability metrics*. Newnes, 2013.

[5] M. E. C. Santos, A. Chen, T. Taketomi, G. Yamamoto, J. Miyazaki, and H. Kato, "Augmented reality learning experiences: Survey of prototype design and evaluation," *IEEE Trans. Learn. Technol.*, vol. 7, no. 1, pp. 38–56, 2014.

[6] J. Sandberg, M. Maris, and K. de Geus, "Mobile English learning: An evidence-based study with fifth graders," *Comput. Educ.*, vol. 57, no. 1, pp. 1334–1347, 2011.
[7] O. Tanaseichuk *et al.*, "An efficient hierarchical clustering algorithm for large datasets," *Austin J. Proteomics Bioinforma. Genomics*, vol. 2, no. 1, 2015.
[8] J. Boase and R. Ling, "Measuring Mobile Phone Use: Self-Report Versus Log Data," *J. Comput.-Mediat. Commun.*, vol. 18, no. 4, pp. 508–519, Jul. 2013, doi: 10.1111/jcc4.12021.
[9] K. C. Lim, A. Selamat, R. A. Alias, M. H. M. Zabil, F. Puteh, and F. Mohamed, "Measuring the Feasibility of Clustering Techniques on Usability Performance Data," *Indian J. Sci. Technol.*, vol. 11, no. 4, Jan. 2018, doi: 10.17485/ijst/2018/v11i4/121089.
[10] A. Dünser and M. Billinghurst, "Evaluating augmented reality systems," in *Handbook of augmented reality*, Springer, 2011, pp. 289–307.
[11] A. Dix, *Human-computer interaction*. Springer, 2009.
[12] L. K. Cheng *et al.*, "Feasibility Comparison of HAC Algorithm on Usability Performance and Self-Reported Metric Features for MAR Learning," in *SoMeT*, 2018.
[13] M. E. Celebi, H. A. Kingravi, and P. A. Vela, "A comparative study of efficient initialization methods for the k-means clustering algorithm," *Expert Syst. Appl.*, vol. 40, no. 1, pp. 200–210, Jan. 2013, doi: 10.1016/j.eswa.2012.07.021.
[14] A. Fahad *et al.*, "A Survey of Clustering Algorithms for Big Data: Taxonomy and Empirical Analysis," *IEEE Trans. Emerg. Top. Comput.*, vol. 2, no. 3, pp. 267–279, Sep. 2014, doi: 10.1109/TETC.2014.2330519.
[15] V. T., "Performance based analysis between k-Means and Fuzzy C-Means clustering algorithms for connection oriented telecommunication data," *Appl. Soft Comput.*, vol. 19, pp. 134–146, Jun. 2014, doi: 10.1016/j.asoc.2014.02.011.
[16] M. Kaur, "Comparison between k-means and hierarchical algorithm using query redirection," *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol. 3, no. 7, pp. 1454–1459, Jul. 2013.
[17] L. K. Cheng, A. Selamat, R. A. Alias, F. Puteh, and F. bin Mohamed, "InterviewME: A Comparative Pilot Study on M-Learning and MAR-Learning Prototypes in Malaysian English Language Teaching," in *Computational Intelligence in Information Systems*, 2016, pp. 219–234, doi: 10.1007/978-3-319-48517-1_20.
[18] S. Martin *et al.*, "Compensating for Memory Losses throughout aging: Validation and Normalization of the Memory Compensation Questionnaire for Non-Clinical French Populations," *Arch. Gerontol. Geriatr.*, vol. 60, no. 1, pp. 28–38, 2015, doi: 10.1016/j.archger.2014.10.013.
[19] A. Bouguettaya, Q. Yu, X. Liu, X. Zhou, and A. Song, "Efficient agglomerative hierarchical clustering," *Expert Syst. Appl.*, vol. 42, no. 5, pp. 2785–2797, Apr. 2015, doi: 10.1016/j.eswa.2014.09.054.
[20] A. Chokniwal and M. Singh, "Faster Mahalanobis K-means clustering for Gaussian distributions," in *2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2016, pp. 947–952, doi: 10.1109/ICACCI.2016.7732167.
[21] A. Joshi and R. Kaur, "A Review: Comparative Study of Various Clustering Techniques in Data Mining," *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol. 3, no. 3, pp. 55–57, Mar. 2013.
[22] M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter, "Efficient and Robust Automated Machine Learning," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 2962–2970.